IMS
Version 11

System Definition

GC19-2444-05

IMS
Version 11

System Definition

GC19-2444-05

Note
Before using this information and the product that it supports, be sure to read the general information under "Notices" on page 823.

This edition applies to IMS Version 11 (program number 5635-A02) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 1974, 2015.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this information ix
Prerequisite knowledge ix
IMS function names used in this information . . . ix
How new and changed information is identified . . ix
How to read syntax diagrams x
Accessibility features for IMS Version 11 xi
How to send your comments xii

Changes to the IMS library xiii

Chapter 1. Overview of the IMS system definition process 1
IMS system definition 1
How system definition is related to installation . . . 1
Coding IMS macros that define the system 2
Choosing a system definition 4
Selecting the appropriate macros to define your system 5
System definition preprocessor overview 15
IMS stage 1 system definition 21
Verifying the stage 1 input 22
Determining when system definition is required . . 23
Defining a large system 24
Building ETO descriptors in large system definitions 25
Including Fast Path in a DCCTL or DB/DC system definition 25
Including Fast Path in a DBCTL system definition 27
IMS stage 2 system definition 27
JCLIN processing 28
Applying maintenance using SMP/E 28
Adding entries to the z/OS Program Properties Table 28
Configuring IMS support for the IMS Universal drivers 32

Chapter 2. Dynamic resource definition 35
Overview of dynamic resource definition 35
Managing resource and descriptor definitions . . 36
Commands that support dynamic resource definition 38
Dynamic resource definition and system definition 40
Recovery of changed runtime resource and descriptor definitions 41
Requirements for dynamic resource definition . . . 41
Restrictions for dynamic resource definition . . . 42
Considerations for using dynamic resource definition 43
Enabling dynamic definition for IMS resource groups 44
Enabling dynamic resource definition for MODBLKS resources 45
Creating runtime resource and descriptor definitions 46
Creating runtime database resource and descriptor definitions with the CREATE command 47
Creating runtime application program resource and descriptor definitions with the CREATE command 48
Creating runtime Fast Path routing code resource and descriptor definitions with the CREATE command 48
Creating runtime transaction resource and descriptor definitions with the CREATE command 49
Creating runtime resource and descriptor definitions in an IMSplex with the CREATE command 50
Updating runtime resource and descriptor definitions 50
Updating runtime database resource and descriptor definitions with the UPDATE command 51
Updating runtime application program resource and descriptor definitions with the UPDATE command 51
Updating runtime Fast Path routing code resource and descriptor definitions with the UPDATE command. 52
Updating runtime transaction resource and descriptor definitions with the UPDATE command 53
Updating runtime resource and descriptor definitions in an IMSplex with the UPDATE command 54
Deleting runtime resource and descriptor definitions 54
Deleting runtime database resource and descriptor definitions with the DELETE command 55
Deleting runtime application program resource and descriptor definitions with the DELETE command 57
Deleting runtime Fast Path routing code resource and descriptor definitions with the DELETE command 58
Deleting runtime transaction resource and descriptor definitions with the DELETE command 58
Deleting runtime resource and descriptor definitions in an IMSplex with the DELETE command 59
Exporting resource and descriptor definitions . . . 59
Exporting resource and descriptor definitions to an RDDS 60
Importing resource and descriptor definitions . . . 61
Importing resource and descriptor definitions by using the automatic import function 61
Importing resource and descriptor definitions from an RDDS by using the IMPORT command . 62

© Copyright IBM Corp. 1974, 2015

Backing out runtime resources imported with the IMPORT command 63
Maintaining your dynamic resource definition environment 64
Converting resource definitions into IMS stage 1 macro statements or IMS type-2 CREATE commands 65
Disabling dynamic resource definition 66
Disabling dynamic resource definition for MODBLKS resources 66

Chapter 3. Designing the IMS system 69
Choosing the number of regions 69
Setting a checkpoint frequency 69
Selecting an IMS lock manager 70
Using a DL/I separate address space 71
Starting the DL/I separate address space 73
Allocating Message Format Service buffer pool space 74
Use of the message format buffer pool 75
Declaring online databases 76
Declaring online application programs 76
Defining online applications 79
Declaring Fast Path application programs 80
Defining IMS transactions 81
Defining Fast Path transactions 83
Planning a scheduling algorithm 83
Grouping application transactions in DB/DC and DCCTL environments 84
Stopping transactions and PSBs because of unavailable data 85
Assigning message class and initializing a region in DB/DC and DCCTL 85
Message priorities within message classes in DB/DC and DCCTL 86
Selection priorities for transactions in DB/DC and DCCTL environments 86
Processing limits for messages in DB/DC and DCCTL environments 87
Quick reschedule for regions in DB/DC and DCCTL environments 88
Pseudo WFI option for MPP regions in DB/DC and DCCTL environments 88
Processing transactions against unavailable data in DB/DC and DBCTL environments. 89
Scheduling transactions using the suspend queue . 90
Parallel scheduling of applications and transactions 92
Alternative scheduling options for messages in DB/DC and DCCTL environments 93
Scheduling for BMP processing 94
Assigning priorities for programs with exclusive intent 94
Scheduling for CPI-Communications-driven programs 94
Defining terminals with data communication macros 95
Defining VTAM terminals 95
Defining non-VTAM devices. 97
Choosing the IMS master terminal. 98
Specifying macros for Multiple Systems Coupling 101
Using multiple IMS systems at the same release level and environment 101
Using multiple IMS systems at different release levels 102
Including IMS ETO Support in the IMS system . . 103
Defining an FDBR region 104
Configuration options for FDBR regions . . . 105
Enabling an IMS subsystem for FDBR 106
IMSplex requirements for FDBR 107
Specifying enqueue and dequeue requirements . . 107
Specifying security options 108
Including RSR in the IMS system 108

Chapter 4. Allocating and cataloging IMS system resources. 111
IMS system data sets for online change 111
Initializing system data sets when not using online change 114
IMS online data sets 115
Allocating ACBLIB data sets 116
Dynamically allocating the ACB staging library for ACBLIB member online change 118
Dynamically allocating the IMSACBA and IMSACBB library data sets 118
Allocating log data sets 119
Allocating online log data sets (OLDSs) . . . 120
Allocating the write-ahead data set (WADS) . . 123
Allocating the system log data set (SLDS) . . . 124
Allocating the restart data set 125
Setting the TOD clock during IPL 125
Message queue data set allocation in DB/DC and DCCTL environments 126
Allocating OSAM data sets 128
Allocating VSAM data sets 129
Allocating data sets in extended address volumes 130
Resource definition data sets 130
Changing RDDS block sizes 132
Allocating IMS spool data sets. 133
Allocating direct output lines 135
Initializing the RECON data set for DBRC. . . . 136
Allocating XRF data sets. 137
Defining a HALDB indirect list data set 140
Defining large sequential data sets 141
Falling back from large sequential data sets . . 142
Defining Message Formatting Service 142
Implementing Message Formatting Service . . . 145

Chapter 5. Specifying IMS execution parameters 147
System control and performance EXEC parameters for the IMS control region 147
Dynamic resource definition EXEC parameters . . 149
Database and PSB exec parameters for the control region 150
Database buffer requirements 150
Buffer sizes for databases that use OSAM or VSAM. 150
Performance options for data management and program specification blocks 151
Data communication EXEC parameters for the control region 151
Fast Path EXEC parameters in DCCTL or DB/DC 153

Fast Path EXEC parameters in DBCTL 155
Fast Path dependent region parameters in DCCTL or DB/DC 156
Fast Path parameters in BMP and CCTL regions in DBCTL 157
Online DEDB utility region parameters in DCCTL, DBCTL, or DB/DC 158
Recovery-related EXEC parameters for the control region 159
Security-related EXEC parameters for the control region 161
EXEC parameters for IMS message processing regions 162
EXEC parameters for IMS batch message processing regions 165

Chapter 6. Tailoring the IMS system to your environment 171
Tailoring the IMS PROCLIB data set 171
Controlling the IMS PROCLIB data set 174
IMS buffer pools 179
VSAM subpool definition 180
OSAM subpool definition 180
OSAM buffer pool compatibility definition . . 181
Specifying VSAM and OSAM subpools 182
Specifications for OSAM sequential buffering 182
Creating and sizing the 64-bit storage pool . . 183
Making IMS and IMSRDR procedures accessible to z/OS 183
Organizing PL/I modules for use with the PL/I Optimizer 184
Specifying sequential buffering control statements 184
Using high-speed sequential processing control statements 185
Supporting CCTL users with DBCTL databases . . 185
Preparing a CCTL 186
Enabling IMS ETO Support for ACF/VTAM terminals 186
ETO descriptors 186
Logon descriptors 188
MFS device descriptors 189
MSC descriptors 189
User descriptors 190
Specifying the number of hash table slots . . . 191
HALDB selective partition processing 192

Chapter 7. CQS definition and tailoring 197
CQS's support of multiple clients 197
Defining structure size for CQS connections . . . 197
Calculating resource structure entry and element values to maximize storage 197
Preparing to start the CQS address space 202

Chapter 8. CSL definition and tailoring 205
Defining the Common Service Layer using IMS PROCLIB data set members 205
Sequence for starting CSL address spaces 208

Chapter 9. IMS Connect definition and tailoring 209
Sample JCL to start IMS Connect 209
Configuring IMS Connect 210
Defining IMS Connect security 212
Setting up AT-TLS SSL for IMS Connect . . . 213
Configuring XML conversion support for IMS Connect clients 215
Configuring the IMS Base Primitive Environment for IMS Connect 218
Allocating IMS Connect log record data sets . . . 220
JCL to print IMS Connect RECORDER output . . 220

Chapter 10. Accessing external subsystem data 221
Defining your external subsystems to IMS 221
Specifying the SSM= EXEC parameter 221
Defining the language interface module 223
Specifying trace options 224
Specifying DB2 for z/OS groups for IMS online regions 224
Specifying DB2 for z/OS for IMS batch regions . . 226
Preparing DB2 for z/OS for use with IMS 227
Providing appropriate tables in your ESS . . . 227
Placing DB2 for z/OS modules and tables in the appropriate library 227
Configuring JMP and JBP regions for DB2 for z/OS data access 228
Attaching a DB2 for z/OS subsystem to IMS using RRSAF 229

Chapter 11. Installing the Transport Manager Subsystem 231
Defining the Transport Manager Subsystem to IMS 231
Defining the Transport Manager Subsystem to VTAM 231

Chapter 12. Setting up IMS for diagnostics 235
Setting the size of the z/OS trace tables 235
Common Storage Tracker 235
CHNGDUMP MAXSPACE 236
Automatic dump data set allocation 236
Diagnostic setup recommendations for IMS . . . 237
CQS trace setup recommendations 239
Trace environment - conservative 239
Trace environment - more aggressive 239
Installing the IMS Dump Formatter 239
Making IMS.SDFSRESL available 240
Setting up the external trace environment 241
Controlling the volume of traces 241
Activating Fast Path traces 242
Writing trace tables externally 242
Creating output data sets with correct attributes 243
Setting up tracing for BPE-managed address spaces 244
Setting up the IMS abend search and notification function 245
Customizing the IMS abend search and notification email 248

Chapter 13. IMS system definition examples 253

IMS DB/DC environment 253
Data communication characteristics 253
System configuration macro statements. . . . 253
Database and application macro statements . . 254
DL/I application programs 255
Data Communication macro statements 258
Local SYSOUT line groups 258
Macro statements to define VTAM communication devices 259
System configuration macro statement 270
IMS Multiple Systems Coupling 270
Data-sharing system configuration 275
Data sharing at the database-level with update access 275
Data sharing at the database level with read access 277
Intra-CPC block-level data sharing 278
Inter-CPC block-level data sharing 279
IMS DBCTL environment 280

Chapter 14. IMS Syntax Checker . . . 283

Function keys and the Syntax Checker 284
Starting the Syntax Checker 285
Syntax Checker online help. 285
Using the Syntax Checker 285
IMS Release and Control Region panel 286
IMS Release panel. 287
Keyword Display panel 288
Save Member prompt panel 295
SAVE AS prompt panel 296
IMS Parameter Syntax Checker panel 297

Chapter 15. Macros used in IMS environments 299

Reference information for using IMS macros . . . 299
APPLCTN macro 304
BUFPOOLS macro. 311
COMM macro 315
DATABASE macro. 323
DFSMDA macro 325
FPCTRL macro 336
IMSCTF macro 337
IMSCTRL macro 340
IMSGEN macro 351
LINE macro 363
LINEGRP macro 365
MSGQUEUE macro 367
MSLINK macro. 372
MSNAME macro 375
MSPLINK macro 376
NAME macro 380
RTCODE macro 384
SECURITY macro 386
SUBPOOL macro 390
TERMINAL macro 391
TRANSACT macro 420
TYPE macro 441
VTAMPOOL macro 443

Chapter 16. Procedures used in IMS environments 445

The specification of IMS procedures 445
Parameter descriptions for IMS procedures . . . 448
ABND= through AUTO= procedure parameters 448
BKO= through CSLG= procedure parameters 453
DBBF= through DIRCA= procedure parameters 459
DLINM= through EXVR= procedure parameters 464
FBP= through HSBMBR= procedure parameters 468
IMS= through LHTS= procedure parameters . . 472
LOCKMAX= through MNPSPW= procedure parameters 476
MOD= through ORSMBR= procedure parameters 479
OTHR= through PIMAX= procedure parameters 482
PRDR= through QTU= procedure parameters 486
RC= through RVFY= procedure parameters . . 490
SAV= through SUF= procedure parameters . . 493
SVC2= through TMINAME= procedure parameters 497
TRACE= through YEAR4= procedure parameters 500
DD statements for IMS procedures 502
CQS startup procedure 510
CSLODBM procedure 515
CSLOM procedure 517
CSLRM procedure. 518
CSLSCI procedure. 519
DBBBATCH procedure 520
DBC procedure. 523
DBRC procedure 529
DCC procedure. 532
DFSIASN0 procedure. 541
DFSJBP procedure. 543
DFSJMP procedure 545
DFSMPR procedure 547
DLIBATCH procedure 548
DLISAS procedure. 551
DXRJPROC procedure 554
FDR procedure 555
FPUTIL procedure. 559
IMS procedure 560
IMSBATCH procedure 571
IMSCOBGO procedure 572
IMSCOBOL procedure 575
IMSDALOC procedure 576
IMSFP procedure 578
IMSMSG procedure 579
IMSPLI procedure. 580
IMSPLIGO procedure 581
IMSRDR procedure 584
RDIBATCH procedure 584

Chapter 17. Members of the IMS PROCLIB data set 587

BPE configuration parameter member of the IMS PROCLIB data set 587
BPE exit list members of the IMS PROCLIB data set 605
CQSIPxxx member of the IMS PROCLIB data set 614

CQSSLxxx member of the IMS PROCLIB data set 617
CQSSGxxx member of the IMS PROCLIB data set 619
CSLDCxxx member of the IMS PROCLIB data set 628
CSLDIxxx member of the IMS PROCLIB data set 633
CSLOIxxx member of the IMS PROCLIB data set 636
CSLRIxxx member of the IMS PROCLIB data set 640
CSLSIxxx member of the IMS PROCLIB data set 643
DBFMSDBx member of the IMS PROCLIB data set 646
DFS62DTx member of the IMS PROCLIB data set 647
DFSCGxxx member of the IMS PROCLIB data set 649
DFSDCxxx member of the IMS PROCLIB data set 656
DFSDFxxx member of the IMS PROCLIB data set 672
COMMON_SERVICE_LAYER section of the DFSDFxxx member . . . 673
DATABASE section of the DFSDFxxx member 679
DIAGNOSTICS_STATISTICS section of the DFSDFxxx member . . . 680
DYNAMIC_RESOURCES section of the DFSDFxxx member . . . 682
FASTPATH section of the DFSDFxxx member 685
SHARED_QUEUES section of the DFSDFxxx member . . . 686
USER_EXITS section of the DFSDFxxx member 687
DFSDRFnn member of the IMS PROCLIB data set 688
DFSDSCMx member of the IMS PROCLIB data set 689
Common format of ETO descriptor statements 690
Logon descriptor format and parameters . . . 691
MFS device descriptor format and parameters 695
MSC descriptor format and parameters. . . . 696
User descriptor syntax and parameters 697
DFSDSCTy member of the IMS PROCLIB data set 701
DFSFDRxx member of the IMS PROCLIB data set 702
DFSFIXnn member of the IMS PROCLIB data set 704
DFSHSBxx member of the IMS PROCLIB data set 709
DFSINTxx member of the IMS PROCLIB data set 714
DFSJVMAP member of the IMS PROCLIB data set 714
DFSJVMMS member of the IMS PROCLIB data set 715
DFSMPLxx member of the IMS PROCLIB data set 716
DFSORSxx member of the IMS PROCLIB data set 718
DFSPBxxx member of the IMS PROCLIB data set 718
DFSRSRxx member of the IMS PROCLIB data set 720
DFSSPMxx member of the IMS PROCLIB data set 731
DFSSQxxx member of the IMS PROCLIB data set 736
DFSVSMxx member of the IMS PROCLIB data set 738
Defining Fast Path DEDB buffer pools for single-area structures . . . 739
Defining Fast Path DEDB buffer pools for multi-area structures . . . 740
Defining VSAM buffer pools . . . 741
Defining VSAM subpools . . . 743
Defining VSAM performance options . . . 746
Defining OSAM subpools . . . 748
Requesting that z/OS dynamic allocation use extended storage . . . 751
Specifying sequential buffering for an online system . . . 751
Defining serviceability and trace options . . . 752
Defining DL/I call image trace . . . 760
Defining DASD logging initialization parameters . . . 761
Defining IMS batch without dynamic allocation 764
Disabling the re-opening of databases after an IMS restart . . . 764
Defining coupling facility structure names for sysplex data sharing . . . 764
Using the coupling facility for OSAM data caching . . . 766
Enabling the long busy handling function . . . 767
Enabling the IRLM lock timeout function . . . 768
Preventing transactions from being terminated when HALDB partitions are unavailable . . . 769
Preventing DBRC calls for HALDB version verification . . . 769
Resuming an online reorganization for HALDBs during IMS warm or emergency restart. . . . 770
Discarding preallocated SDEP control intervals 770
Preventing a /DBRECOVERY command for a database that has INDOUBT EEQEs 771
DFSYDTx member of the IMS PROCLIB data set 771
OTMA client descriptor syntax and parameters 771
OTMA destination descriptor syntax and parameters . . . 775
DFSOTMA descriptor syntax and parameters 778
DSPBIxxx member of the IMS PROCLIB data set 780
HWSCFGxx member of the IMS PROCLIB data set 782
ADAPTER statement . . . 783
DATASTORE statement . . . 784
HWS statement. . . . 787
IMSPLEX statement . . . 788
ODACCESS statement . . . 789
RUNOPTS statement . . . 792
TCPIP statement . . . 793
IMS Connect configuration examples . . . 799

Chapter 18. Other control statements used in IMS environments . . . 805
Sequential buffering control statements . . . 805
High-speed sequential processing control statements . . . 812
Set Index Maintenance Off (SETI) control statement 817
Database resource adapter startup table for CCTL regions . . . 817
DFSDFSRT . . . 820

Notices . . . 823
Programming interface information . . . 825
Privacy policy considerations . . . 826

Bibliography . . . 827

Index . . . 829

About this information

These topics provide guidance information for designing an IMS™ system, including information about defining and tailoring the IMS system, IMS Common Queue Server (CQS), IMS Common Service Layer (CSL), integrated IMS Connect, IMS Transport Manager Subsystem (TMS), and dynamic resource definition (DRD). The topics also include descriptions of the IMS macros, procedures, and members of the IMS PROCLIB data set.

This information is available in IBM® Knowledge Center at www.ibm.com/support/knowledgecenter.

Prerequisite knowledge

Before using this information, you should have knowledge of either IMS Database Manager (DB) or IMS Transaction Manager (TM). You should also understand basic z/OS® and IMS concepts, your installation's IMS system, and have general knowledge of the tasks involved in project planning.

You can learn more about z/OS by visiting the "z/OS basic skills education" topics in IBM Knowledge Center.

You can gain an understanding of basic IMS concepts by reading *An Introduction to IMS*, an IBM Press publication. An excerpt from this publication is available in the IBM Knowledge Center.

IBM offers a wide variety of classroom and self-study courses to help you learn IMS. For a complete list of courses available, go to the IMS home page at ibm.com/ims and link to the Training and Certification page.

IMS function names used in this information

In this information, the term HALDB Online Reorganization refers to the integrated HALDB Online Reorganization function that is part of IMS Version 11, unless otherwise indicated.

IMS 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 11, unless otherwise indicated.

How new and changed information is identified

New and changed information in most IMS library PDF publications is denoted by a character (revision marker) in the left margin. The first edition (-00) of *Release Planning*, as well as the *Program Directory* and *Licensed Program Specifications*, do not include revision markers.

Revision markers follow these general conventions:

- Only technical changes are marked; style and grammatical changes are not marked.

© Copyright IBM Corp. 1974, 2015

- If part of an element, such as a paragraph, syntax diagram, list item, task step, or figure is changed, the entire element is marked with revision markers, even though only part of the element might have changed.
- If a topic is changed by more than 50%, the entire topic is marked with revision markers (so it might seem to be a new topic, even though it is not).

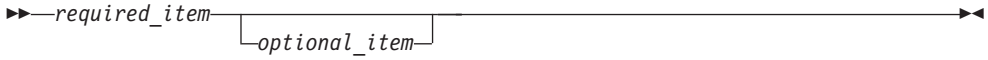
Revision markers do not necessarily indicate all the changes made to the information because deleted text and graphics cannot be marked with revision markers.

How to read syntax diagrams

The following rules apply to the syntax diagrams that are used in this information:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol indicates that a syntax diagram is continued from the previous line.
  - The --->< symbol indicates the end of a syntax diagram.
- Required items appear on the horizontal line (the main path).

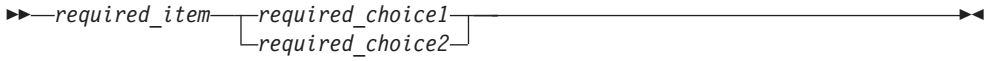
- Optional items appear below the main path.



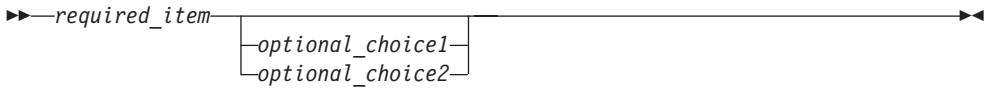
If an optional item appears above the main path, that item has no effect on the execution of the syntax element and is used only for readability.

- If you can choose from two or more items, they appear vertically, in a stack.

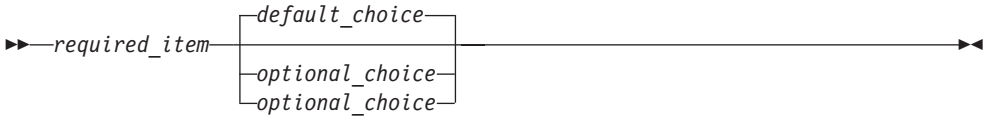
  If you *must* choose one of the items, one item of the stack appears on the main path.



If choosing one of the items is optional, the entire stack appears below the main path.



If one of the items is the default, it appears above the main path, and the remaining choices are shown below.



- An arrow returning to the left, above the main line, indicates an item that can be repeated.

If the repeat arrow contains a comma, you must separate repeated items with a comma.

A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.

fragment-name:

- In IMS, a b symbol indicates one blank position.
- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown. Variables appear in all lowercase italic letters (for example, *column-name*). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols, exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses, for example (1).

Accessibility features for IMS Version 11

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in z/OS products, including IMS Version 11. These features support:

- Keyboard-only operation.

- Interfaces that are commonly used by screen readers and screen magnifiers.
- Customization of display attributes such as color, contrast, and font size.

Keyboard navigation

You can access IMS Version 11 ISPF panel functions by using a keyboard or keyboard shortcut keys.

For information about navigating the IMS Version 11 ISPF panels using TSO/E or ISPF, refer to the *z/OS TSO/E Primer*, the *z/OS TSO/E User's Guide*, and the *z/OS 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.

Related accessibility information

Online documentation for IMS Version 11 is available in IBM Knowledge Center.

IBM and accessibility

See the *IBM Human Ability and Accessibility Center* at www.ibm.com/able for more information about the commitment that IBM has to accessibility.

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 take one of the following actions:

- Click the **Feedback** link at the bottom of any IBM Knowledge Center topic.
- Send an email to imspubs@us.ibm.com. Be sure to include the book title and the publication number.

Changes to the IMS library

Changes to the IMS library for IMS Version 11 include minor title changes to all the publications, merging four sets of publications, a change to how *IMS Messages and Codes* are delivered, and a reorganization of *IMS Version 11 Exit Routines*. The library is no longer provided in BookManager format. In addition, as of the May, 2012, a master index is no longer provided.

The titles of all PDF publications in the IMS Version 11 library have been changed so that the publication titles match the contents view (navigation tree) titles in IBM Knowledge Center. For example, the *IMS Diagnosis Guide* has been renamed to *IMS Diagnosis*. In addition, the titles for the three *IMS Command Reference* PDF volumes have been renamed so that their titles better reflect the contents of each publication, as follows:

- *IMS Version 11 Commands, Volume 1: IMS Commands A-M*
- *IMS Version 11 Commands, Volume 2: IMS Commands N-V*
- *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands*

Four sets of publications have been streamlined:

- To eliminate redundancy and group relevant application programming planning information with associated application programming tasks, the *IMS Application Programming Planning Guide* has been merged with the *IMS Application Programming Guide*, in a new title: *IMS Version 11 Application Programming*.
- To group diagnosis reference information with associated diagnosis tasks, the *IMS Diagnosis Reference* has been merged with the *IMS Diagnosis Guide*, in a new title: *IMS Version 11 Diagnosis*.
- To group all system administration tasks, regardless of whether the tasks relate to using a single IMS or an IMSplex, the *IMS IMSplex Administration Guide* has been merged with the *IMS System Administration Guide*, in a new title: *IMS Version 11 System Administration*.
- To group system definition reference information with associated system definition tasks, the *IMS System Definition Reference* has been merged with the *IMS System Definition Guide*, in a new title: *IMS Version 11 System Definition*.

IMS Version 11 Exit Routines is now organized by the call interface and available services for each group of exit routines, as follows:

- IMS system exits, database manager exits, and transaction manager exit routines are grouped under “IMS control region exit routines.”These exit routines use IMS interfaces and callable services, as described in the topic “Overview of exit routines for IMS.”
- Exit routines that run in an address space with BPE support, including those for CQS, BPE-based DBRC, and most CSL components, are grouped under “Base Primitive Environment-based exit routines.” BPE-based exit routines use the interfaces and services provided by the Base Primitive Environment, as described in the topic “BPE user-supplied exit routine interfaces and services.”
- The CQS exit routines are now divided into two groups: those that run in the CQS address space and have access to BPE services are grouped under “BPE-based CQS user-supplied exit routines,” and those that execute in a CQS client address space are grouped under “CQS client exit routines.”

© Copyright IBM Corp. 1974, 2015

- Standard DBRC exit routines are grouped under “IMS system exit routines,” while the variants that run in BPE are group under “BPE-based DBRC exit routines.”
- CSL SCI exit routines are now divided into two categories: those that run in the CSL SCI address space are grouped under “BPE-based SCL SCI user exit routines,” and those that run in the address space of a registered IMSplex member are grouped under “CSL SCI IMSPlex member exit routines.”
- Exit routines that have specific guidelines, including those for IMS Connect, CQS clients, and CSL SCI IMSPlex members, are described in the sections for those routines.

Chapter 1. Overview of the IMS system definition process

The IMS system definition process describes the system resources (databases, applications, devices, terminals, and links) to IMS when you first install IMS, you apply maintenance to your IMS system, or definition changes are required.

IMS system definition

IMS system definition at a high-level view comprises several supertasks.

You should have a working knowledge of the z/OS system modification program/extended (SMP/E). SMP/E is required for installation and is used as part of the IMS system definition process.

IMS system definition is evolving to be more dynamic rather than static.

1. Modify or tailor IMS-supplied macro statements and procedures (JCL) to define the IMS system that your business requires. These macro statements and procedures are the building blocks for your IMS system.
2. Run the IMS preprocessor to verify the correctness of the macros and procedures.
3. Execute Stage 1 assembly, in which you run the JCL that you modified in step 1 through the z/OS High Level Assembler program to assemble your program into the JCL that is required as input to Stage 2.
4. Execute Stage 2 assembly, in which you:
   a. Build the IMS executable load modules.
   b. Optionally build Message Format Service (MFS) default formats.
   c. Update the member of the IMS PROCLIB data set.
5. Run the JCLIN, an SMP/E process that tells SMP/E how to assemble and bind modules.
6. Use the SMP/E `APPLY` command to process any maintenance that has not been processed using the SMP/E `ACCEPT` command.

How system definition is related to installation

System definition and JCL preparation are only a part of the total installation process. You should understand the relationship between installation and system definition before defining your system.

Related reading: For step-by-step instructions on installing an IMS system, see *IMS Version 11 Installation*.

A full installation process includes:

1. Building IMS system libraries
2. Allocating and cataloging IMS data sets
3. Defining the IMS system
4. Preparing the operating system for IMS, including:
   - VTAM®
   - RACF®
   - APPC/MVS

© Copyright IBM Corp. 1974, 2015

5. Installing IMS exit routines
6. Tailoring IMS buffers and certain performance options
7. Defining terminals (VTAM and non-VTAM)
8. Updating MFS device characteristics table and MFS default formats
9. Defining LU 6.2 descriptors
10. Defining ETO descriptors
11. Building IMS.DBDLIB
12. Building IMS.PSBLIB
13. Building IMS.ACBLIB
14. Preparing for dynamic allocation of databases and related system data sets
15. Compiling message format descriptions
16. Loading application programs
17. Initial loading of databases
18. Establishing IMS security
19. Initializing IMS.MODSTAT
20. Copying staging libraries to active libraries

Coding IMS macros that define the system

Use IMS macros to define and tailor a new IMS system or to modify an existing IMS system.

The first step in defining an IMS system is to modify or tailor the macros that IMS supplies. The composite of all IMS macro statements becomes *stage 1 input*; during stage 1, the specifications you make on the macros are checked, and a series of z/OS job steps is generated as input to Stage 2. Within each set of macro statements, individual macros specify data that is specific to a required function or to a part of the total physical online configuration of your IMS system.

There are seven types of IMS system definition that can be performed. The IMSCTRL macro provides the basic IMS control program options and the z/OS system configuration under which IMS is to execute:

Table 1. Seven types of IMS System Definitions

System definition option used on IMSCTRL macro	Components that are generated	When to use this option
ALL	Builds most IMS modules and procedures. Includes BATCH and ON-LINE types too.	Use ALL during a typical initial system definition. This option is also often required for maintenance.
BATCH	Moves required modules from IMS distribution libraries to your libraries; generates system procedures and a database system.	Use BATCH only to define an IMS batch environment.
CTLBLKS	Generates modules for all IMS control blocks. A CTLBLKS system definition type also includes the MODBLKS and MSVERIFY types.	Use CTLBLKS to rebuild the existing IMS nucleus and to create communications definitions.

Table 1. Seven types of IMS System Definitions (continued)

System definition option used on IMSCTRL macro	Components that are generated	When to use this option
MSVERIFY	Builds control blocks for the Multiple Systems Verification utility (DFSUMSV0).	Use MSVERIFY only for Multiple Systems Coupling (MSC).
MODBLKS	Generates control block members for resources to be added online (for example, APPLCTN, DATABASE, TRANSACT, and RTCODE macros).	Use MODBLKS when online changes are required to the IMS system, such as changes to programs, transactions, and database definitions.
NUCLEUS	Generates an IMS nucleus for the control region. A NUCLEUS system definition type also includes the CTLBLKS type.	Use NUCLEUS when performing major maintenance that affects the contents of the IMS nucleus, or when a new nucleus with a new suffix is required.
ON-LINE	Builds all the modules and procedures needed for the online IMS environment. An ON-LINE system definition type also includes the NUCLEUS type.	Use ON-LINE during initial system definition or to perform a major update. This option is also often required for maintenance.

You can think of these groupings of macros as a type of hierarchical structure, as shown in Figure 1 on page 4. The group of required system configuration macros (IMSCTRL, IMSCTF, BUFPOOLS, MSGQUEUE, SECURITY, and COMM) is shown as a root segment.

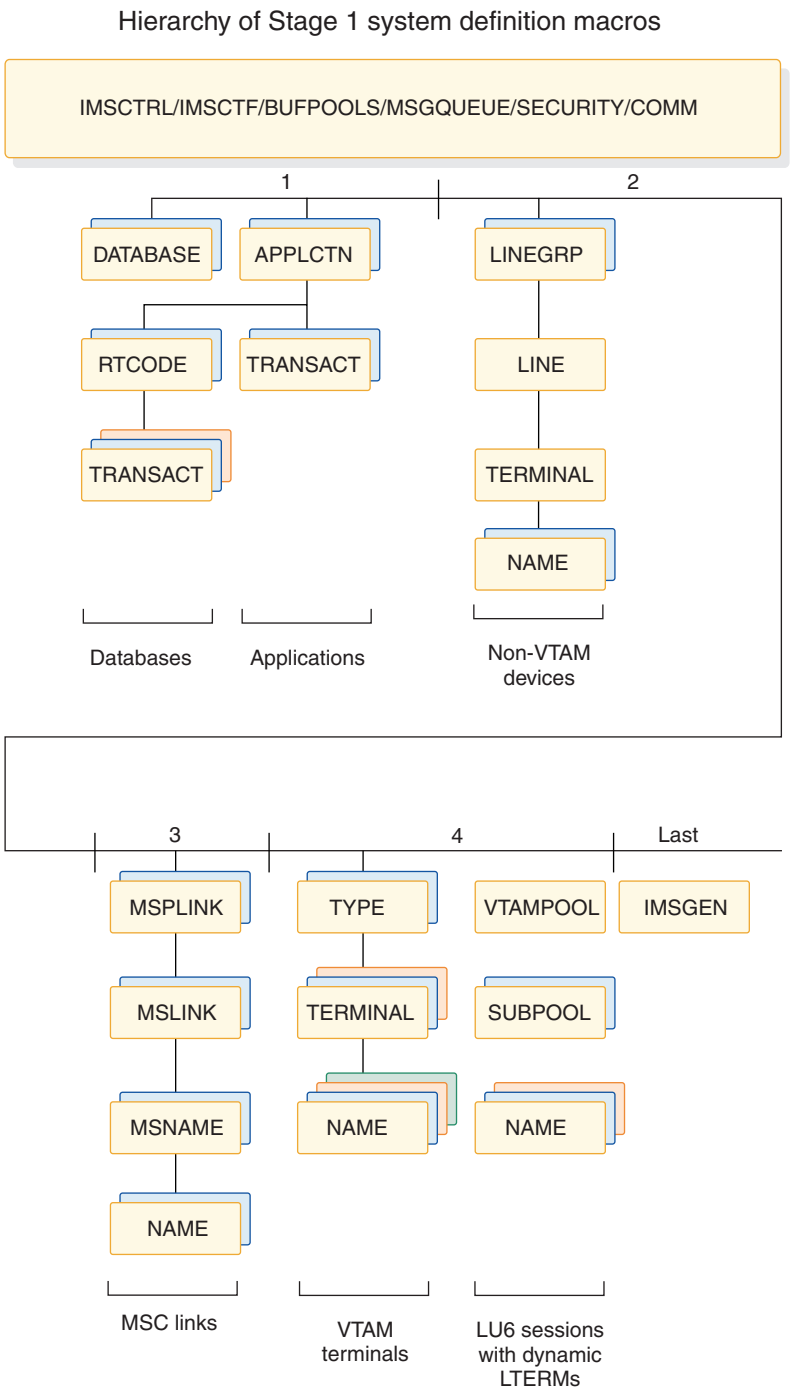


Figure 1. Hierarchy of stage 1 system definition macros

Restriction: The IMSGEN macro must be the last macro defined.

Choosing a system definition

Use the SYSTEM= parameter on the IMSCTRL macro to choose the type of system you want to define.

Use the IMSCTRL SYSTEM= parameter to identify the system you want to define:

DB/DC
Builds an IMS system that includes both the Database Manager (DB) and Transaction Manager (DC)

DBCTL
Builds an IMS system that includes only the Database Manager

DCCTL
Builds an IMS system that includes only the Transaction Manager

After you initially define your IMS system, you can later use the ON-LINE, CTLBLKS, and NUCLEUS options to implement changes. The ON-LINE, CTLBLKS, and NUCLEUS options require a cold start of the IMS online system to take effect.

For certain changes to your IMS system, you can take advantage of the online change method using the MODBLKS type of system definition. With the MODBLKS type of system definition, the changes are made active during the execution of the online system and do not require a restart operation.

Selecting the appropriate macros to define your system

When you choose to define or modify an IMS system, the system definition parameters that you can define or modify vary by the type of system definition. Certain macro statements, keywords, and parameters require additional consideration if you are newly adding or entirely deleting them from the system.

The following table of macros shows, in order of complexity, which IMS system definition parameters can be modified by type of system definition. You can override some keywords for some macros by specifying certain keyword parameters on the EXEC statement for the IMS, DBC, or DCC procedure, or by modifying the JCL. Other keywords can also be overridden through IMS commands. For further information, see Chapter 6, “Tailoring the IMS system to your environment,” on page 171 or the notes that follow the table.

To use the following table, find the macro-associated keyword and parameter that you want to change. Then scan across the table to find the first column with an “X” under the system definition types. This identifies the minimum system definition required to change the parameter. If multiple changes are to be made, scan each macro statement on the figure to determine the minimum system definition. When changing operands, make certain the minimum system definition required to update the operand is performed. If you specify a lower level than that indicated in the following table the results are unpredictable.

The JCL column in the table identifies the EXEC parameter, DFSPBIMS keyword, DFSPBDBC keyword, DFSPBDCC keyword, or, IMS command to override the operand in the macro.

Keywords and parameters changed during a MODBLKS system definition can be brought online through a series of `/MODIFY` operator commands, except as described in the notes that follow the table below.

The CTLBLKS system definition option on the IMSCTRL macro statement can be used only to replace the control blocks of an existing nucleus—that is, a nucleus having the same suffix.

Adding new device support features or options might require a NUCLEUS system definition, because additional modules (for example, the VTAM terminal COPY option and module DFSCVEQ0) might need to be bound into the nucleus.

Some of the values can be displayed using the /DISPLAY command shown in table notes. The value in the /DISPLAY column is used to qualify the /DISPLAY command as necessary.

Table 2. Macro Table - Selecting the appropriate IMS system definition

Macro	Operand	Value	JCL or Command	/DISPLAY	MODBLKS	CTLBLKS	NUCLEUS	ON-LINE	ALL	MSVERIFY	DB BATCH	TM BATCH	DBCTL	DB/DC	DCCTL	Notes
APPLCTN	DOPT				X	X	X	X	X				X	X	X	35 on page 14
	PGMTYPE	class	/ASSIGN		X	X	X	X	X					X	X	22 on page 14
		All others			X	X	X	X	X				X	X	X	
	RESIDENT				X	X	X	X	X				X	X	X	
	FPATH				X	X	X	X	X					X	X	
	SYSID				X	X	X	X	X					X	X	1 on page 13
	All Others				X	X	X	X	X				X	X	X	
BUFPOOLS	DMB		DMB=	DMBP		X	X	X	X				X	X		2 on page 13, 23 on page 14
	EPCB		EPCB=	EPCB		X	X	X	X				X	X	X	2 on page 13, 23 on page 14
	FORMAT	size 1	FBP=	MFP		X	X	X	X					X	X	2 on page 13, 23 on page 14
	FRE		FRE=			X	X	X	X					X	X	2 on page 13
	PSB		PSB=	PSBP		X	X	X	X				X	X	X	2 on page 13, 23 on page 14
	PSBW		PSBW=	PSBW		X	X	X	X				X	X	X	2 on page 13, 23 on page 14
	SASPSB	size 1	CSAPSB=			X	X	X	X				X	X		2 on page 13
		size 2	DLIPSB=			X	X	X	X				X	X		2 on page 13
COMM	COPYLOG		/SMCOPY			X	X	X	X					X	X	
	OPTIONS	NOFMAST FMTMAST					X	X	X					X	X	
		NOMFSTEST MFSTEST					X	X	X					X	X	2 on page 13
		NOPAGE PAGING					X	X	X					X	X	

Table 2. Macro Table - Selecting the appropriate IMS system definition (continued)

Macro	Operand	Value	JCL or Command	/DISPLAY	MODBLKS	CTLBLKS	NUCLEUS	ON-LINE	ALL	MSVERIFY	DB BATCH	TM BATCH	DBCTL	DB/DC	DCCTL	Notes
		NOUSEMSG USERMSGS				X	X	X	X					X	X	5 on page 13
		All Others				X	X	X	X					X	X	
	RECANY					X	X	X	X					X	X	3 on page 13, 4 on page 13
	All Others					X	X	X	X					X	X	
DATABASE	ACCESS		DD		X	X	X	X	X				X	X		1 on page 13, 6 on page 13, 7 on page 13, 33 on page 14, 34 on page 14, 36 on page 14
			/START		X	X	X	X	X				X	X		1 on page 13, 24 on page 14
	RESIDENT				X	X	X	X	X				X	X		
	All Others				X	X	X		X					X		
IDLIST						X	X	X	X					X	X	
IMSCTF	APNDG							X	X				X	X	X	
	CPLOG		CPLOG= /CHANGE	CPLOG		X	X	X	X				X	X	X	
	LOG					X	X	X	X			X	X	X	X	
	PRDR		PRDR=			X	X	X	X				X	X	X	2 on page 13
	RDS					X	X	X	X				X	X	X	
	SVCNO	type 2						X	X		X	X	X	X	X	8 on page 13
		type 4						X	X				X	X	X	
IMSCTRL	CMDCHAR		CRC=			X	X	X	X				X	X	X	
	DBRC	parm 1	DBRC=						X		X	X	X	X	X	9 on page 13

Table 2. Macro Table - Selecting the appropriate IMS system definition (continued)

Macro	Operand	Value	JCL or Command	/DISPLAY	MODBLKS	CTLBLKS	NUCLEUS	ON-LINE	ALL	MSVERIFY	DB BATCH	TM BATCH	DBCTL	DB/DC	DCCTL	Notes
		parm 2	DBRC=						X		X	X	X	X	X	9 on page 13
	DBRCNM		DBRCNM=			X	X	X	X				X	X	X	2 on page 13, 25 on page 14
	DCLWA					X	X	X	X					X	X	
	DESC					X	X	X	X			X	X	X	X	
	DLINM		DLINM=			X	X	X	X			X	X	X		2 on page 13, 25 on page 14
	ETOFEAT							X	X					X	X	
	GSGNAME		GSGNAME=					X	X		X	X	X	X	X	31 on page 14
	HSB							X	X					X	X	11 on page 13
	IMSID		IMSID=			X	X	X	X			X	X	X	X	2 on page 13
	IRLM		IRLM=			X	X	X	X			X	X	X		2 on page 13, 10 on page 13, 32 on page 14
	IRLMNM		IRLMNM=			X	X	X	X			X	X	X		2 on page 13, 10 on page 13, 32 on page 14
	MAXCLAS					X	X	X	X			X		X	X	
	MAXREGN		PST=			X	X	X	X				X	X	X	2 on page 13
	MCS					X	X	X	X			X	X	X	X	
	MSVID					X	X	X	X	X			X	X	X	
	NAMECHK					X	X	X	X			X	X	X	X	
	RSRFEAT							X	X		X	X	X	X	X	
	SYSTEM							X	X		X	X	X	X	X	
	TMINAME		TMINAME=			X		X	X		X	X	X	X	X	31 on page 14
IMSGEN	MFSDFMT					X	X	X	X					X	X	
	MFSTEST						X	X	X				X	X	X	2 on page 13

Table 2. Macro Table - Selecting the appropriate IMS system definition (continued)

Macro	Operand	Value	JCL or Command	/DISPLAY	MODBLKS	CTLBLKS	NUCLEUS	ON-LINE	ALL	MSVERIFY	DB BATCH	TM BATCH	DBCTL	DB/DC	DCCTL	Notes
	PSWDSEC					X	X	X	X				X	X	X	
	SECCNT						X	X	X				X	X	X	
	SUFFIX						X	X	X				X	X	X	
	SURVEY								X				X	X		12 on page 13
	SYSMSG					X	X	X	X					X	X	
	All Others				X	X	X	X	X	X	X	X	X	X	X	
LINE	ADDR		DD			X	X	X	X					X	X	13 on page 13, 26 on page 14
	All Others					X	X	X	X					X	X	
LINEGRP						X	X	X	X					X	X	14 on page 13
MSGQUEUE	BUFFERS	NBR	QBUF=			X	X	X	X					X	X	2 on page 13
		SIZE				X	X	X	X					X	X	
	All Others					X	X	X	X					X	X	
MSLINK						X	X	X	X					X	X	
MSNAME						X	X	X	X					X	X	
MSPLINK	ADDR		DD			X	X	X	X					X	X	13 on page 13, 15 on page 14, 27 on page 14
	MODETBL		/CHANGE	link# MODE		X	X	X	X					X	X	28 on page 14
	OPTIONS	asr	/CHANGE			X	X	X	X					X	X	27 on page 14
	TYPE					X	X	X	X					X	X	15 on page 14, 16 on page 14
	All Others					X	X	X	X					X	X	
NAME						X	X	X	X					X	X	17 on page 14
RTCODE					X	X	X	X	X					X	X	37 on page 15
SECURITY	TYPE	RACFCOM	RCF=				X	X	X					X	X	2 on page 13, 39 on page 15

Table 2. Macro Table - Selecting the appropriate IMS system definition (continued)

Macro	Operand	Value	JCL or Command	/DISPLAY	MODBLKS	CTLBLKS	NUCLEUS	ON-LINE	ALL	MSVERIFY	DB BATCH	TM BATCH	DBCTL	DB/DC	DCCTL	Notes
		RACFTERM	RCF=				X	X	X					X	X	2 on page 13
		RAS	ISIS=				X	X	X				X	X	X	
		RASEXIT	ISIS=				X	X	X				X	X	X	
		RASRACF	ISIS=				X	X	X				X	X	X	
		SIGNEXIT				X	X	X	X					X	X	19 on page 14
		TRANEXIT				X	X	X	X					X	X	19 on page 14
	SECCNT		SECCNT			X	X	X	X					X	X	
	SECLVL	SIGNON	SGN /NRESTART			X	X	X	X					X	X	2 on page 13, 18 on page 14
		TRANAUTH	TRN /NRESTART			X	X	X	X					X	X	2 on page 13, 18 on page 14
	RCLASS		RCLASS			X	X	X	X					X	X	
	All Others					X	X	X	X				X	X	X	
SUBPOOL						X	X	X	X					X	X	
TERMINAL	MODETBL		/CHANGE	nodename NAME		X	X	X	X					X	X	29 on page 14
	OPTIONS	asr	/CHANGE			X	X	X	X					X	X	30 on page 14
	All Others					X	X	X	X					X	X	
TRANSACT	AOI	yes/tran/cmd			X	X	X	X	X					X	X	38 on page 15
	DCLWA				X	X	X	X	X					X	X	
	EDIT	name				X	X	X						X	X	20 on page 14, 21 on page 14
		All Others			X	X	X	X	X					X	X	
	MAXRGN		/CHANGE		X	X	X	X	X					X	X	22 on page 14
	MSGTYPE	class	/ASSIGN		X	X	X	X	X					X	X	1 on page 13, 22 on page 14

Table 2. Macro Table - Selecting the appropriate IMS system definition (continued)

Macro	Operand	Value	JCL or Command	/DISPLAY	MODBLKS	CTLBLKS	NUCLEUS	ON-LINE	ALL	MSVERIFY	DB BATCH	TM BATCH	DBCTL	DB/DC	DCCTL	Notes
		All Others			X	X	X	X	X					X	X	
	PARLIM		/ASSIGN		X	X	X	X	X					X	X	1 on page 13, 22 on page 14
	PROCLIM	count	/ASSIGN		X	X	X	X	X					X	X	1 on page 13, 22 on page 14
		All Others			X	X	X	X	X					X	X	
	PRTY		/ASSIGN		X	X	X	X	X					X	X	1 on page 13, 22 on page 14
	SEGNO		/ASSIGN		X	X	X	X	X					X	X	1 on page 13, 22 on page 14
	SEGSIZE		/ASSIGN		X	X	X	X	X					X	X	1 on page 13, 22 on page 14
	SYSID				X	X	X	X	X					X	X	1 on page 13
	All Others				X	X	X	X	X					X	X	
TYPE	OPTIONS		SIGNON=			X	X	X	X					X	X	4 on page 13, 14 on page 13
VTAMPOOL						X	X	X	X					X	X	

Notes to the preceding table:

1. Changes to existing transactions cannot be introduced through a MODBLKS system definition if those attributes are changeable by online commands.
2. Refer to "IMS procedure" on page 560, "DBBBATCH procedure" on page 520, "DBC procedure" on page 523, "DCC procedure" on page 532, and "DLIBATCH procedure" on page 548.
3. This keyword can be changed with any type of system definition except MODBLKS.
4. Adding or deleting VTAM support requires the minimum of an ON-LINE system definition.
5. Adding a user-supplied exit routine, user message table, or the shared printer message router exit routine requires the minimum of a CTLBLKS system definition. The exceptions to this requirement are the ETO, message greeting, and command security exit routines. To remove the use of these exit routines requires a minimum of NUCLEUS system definition. These exit routines are then loaded or deleted during initialization if they are in IMS.SDFSRESL.
6. The ACCESS attribute is used in data sharing and can be dynamically changed with the /START command.
7. ACCESS can be forced to exclusive by specifying DISP=OLD in the database DD statement or in the DFSMDA definition.
8. Although the value for the Type 2 SVC number is changed in the necessary control blocks for all system definition types, only the ON-LINE and ALL types of system definition bind the SVC module into IMS.SDFSRESL with the proper name required for subsequent bind into the operating system libraries.
9. The DBRC= parameter on the IMSCTRL macro is ignored by IMS.
10. For an online system, modifying the IRLM= or IRLMNM= parameters requires a minimum of a CTLBLKS type system definition. A batch system requires a minimum of either a BATCH or an ALL system definition.
11. Generating a system for use in an IMS backup configuration requires a minimum of an ON-LINE system definition.
12. Adding Database Surveyor support to an IMS system initially generated without it can be done in two ways. If done through system definition, either a BATCH system definition (for batch) or an ALL system definition (for both batch and online) must be specified to include Surveyor. Alternatively, in either a batch or online system, the following JCL can be run:

```
//LINKSUR  JOB
//        EXEC PGM=IEWL,REGION=128K,
//             PARM='NCAL,LET,XREF'
//SYSPRINT  DD SYSOUT=A
//LOAD      DD DSNAME=IMS.ADFSLOAD,DISP=SHR
//SYSLMOD   DD DSNAME=IMS.SDFSRESL,DISP=SHR
//SYSUT1    DD SPACE=(CYL,(1,1)),UNIT=SYSDA
//SYSLIN    DD *
 INCLUDE LOAD(DFSPRSUR)
 INCLUDE LOAD(DFSLI000)
 NAME DFSPRSUR(R)           DATABASE SURVEYOR
/*
```

13. Change the DD statement.
14. Adding or deleting support for a specific non-ETO terminal type requires a minimum of a NUCLEUS system definition. A minimum of a NUCLEUS system definition is required if the SIZE= or FEAT= parameter is specified for the first time in an existing system or if the values of these parameters are changed. You can use any type of system definition except MODBLKS to add, delete, or change this macro statement.

15. Adding MSC support for the first time or deleting MSC support requires a minimum of an ON-LINE system definition. A physical link change requires a DD statement change.
16. Adding or deleting channel-to-channel support requires a minimum of an ON-LINE system definition. Adding MSC VTAM or MTM support requires a minimum of an ON-LINE system definition. Deleting MSC VTAM or MTM support requires a minimum of a NUCLEUS system definition.
17. Removing remote LTERM support from an MSC system requires a minimum of a NUCLEUS system definition.
18. JCL can be used to override SIGNON and TRANAUTH only if SIGNON is required or TRANSACT authorization is specified. The signon and transaction authorization exit routines can still be used if you sign on.
19. To remove exit routine support requires a minimum of a NUCLEUS system definition defined for static terminals.
20. MODBLKS system definition cannot be used to add a transaction that is also adding a new user edit routine. MODBLKS system definition cannot be used to change the order of the transactions if user edit routines are specified. The minimum system definition for either of these occurrences is a CTLBLKS system definition.
21. MODBLKS system definition cannot be used to delete a transaction that has a user edit routine if this is the first or only transaction that is using this edit routine. MODBLKS system definition cannot be used to change the order of the transactions if user edit routines are specified. The minimum system definition for either of these occurrences is a CTLBLKS system definition.
22. The command is `/DISPLAY TRANSACTION`.
23. The command is `/DISPLAY POOL xxxx`, where `xxxx` is the indicated parameter.
24. The command is `/DISPLAY DATABASE`.
25. The command is `/DISPLAY ACTIVE`.
26. The command is `/DISPLAY LINE`.
27. The command is `/DISPLAY LINK`.
28. The command is `/DISPLAY LINK xxxx`, where `xxxx` is the indicated parameter.
29. The command is `/DISPLAY NODE`.
30. The command is `/DISPLAY MODE xxxx`, where `xxxx` is the indicated parameter.
31. For batch systems, specify the parameter in the JCL; for online systems, specify the parameter in the DFSRSRxx member.
32. A BATCH or ALL system definition is required for the change to take effect in a batch environment.
33. A database that is defined in an ALL or MODBLKS system definition cannot be converted into a HALDB partition without a cold start of IMS. A cold start is required even if the database is deleted online. After the cold start, the database must be redefined as a HALDB partition.
34. A database that is defined as a HALDB partition to DBRC and IMS cannot be redefined in an ALL or MODBLKS system definition without a cold start of IMS. A cold start is required even if the database is deleted online.
35. The APPLCTN macro is optional in DBCTL, DB/DC, and DCCTL environments. No warning message is issued if the macro is not included in the Stage 1 system definition. If you exclude the APPLCTN macro from your Stage 1 system definition, application programs must then be defined through the dynamic resource definition (DRD) process.
36. The DATABASE macro is optional in DBCTL and DB/DC environments. No warning message is issued if the macro is not included in the Stage 1 system

definition. If you exclude the DATABASE macro from your Stage 1 system definition, databases must then be defined through the dynamic resource definition (DRD) process.

37. The RTCODE macro is optional in DB/DC and DCCTL environments. No warning message is issued if the macro is not included in the Stage 1 system definition. If you exclude the RTCODE macro from your Stage 1 system definition, routing codes must then be defined through the dynamic resource definition (DRD) process.
38. The TRANSACT macro is optional in DB/DC and DCCTL environments. No warning message is issued if the macro is not included in the Stage 1 system definition. If you exclude the TRANSACT macro from your Stage 1 system definition, transactions must then be defined through the dynamic resource definition (DRD) process.
39. IMS Version 12 is the last version to support the SECURITY macro. You can use initialization parameters to specify all of the SECURITY macro keyword values.

Related reference:

Chapter 15, “Macros used in IMS environments,” on page 299

System definition preprocessor overview

Use the IMS system definition preprocessor to ensure that stage 1 input does not include duplicate resource names, such as transaction codes, LTERMs, and system names.

IMS provides a preprocessor that you can use to check the stage 1 input for duplicate names among the names you have defined. The preprocessor also ensures that the names are of the correct length and format. Assigned names are checked across resource types, too, so that transaction codes, LTERMs, and IMS system names (used for multiple systems coupling) do not contain duplications. The preprocessor helps maintain the integrity of the stage 1 input stream.

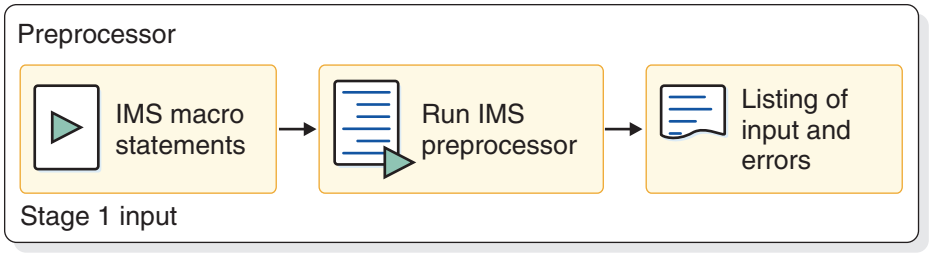


Figure 2. The preprocessor stage of the system definition process

The system definition preprocessor is optional unless the LGEN configuration has been specified in the IMSCTRL SYSTEM keyword. Use the LGEN configuration when more than 4 MB of private storage are needed in stage 1 or in any stage 2 assembly job step. See “LGEN stage 1 processing” on page 16 for information about running the preprocessor when defining a large IMS system.

If the IMSCTRL NAMECHK=NO option is selected for IMS system definition, run the system definition preprocessor.

Stage 1 source statements are used as input and can include copy statements. However, they cannot include inline copies of IMS macros from the stage 1 input, inline user macros, user macro calls, or conditional assembler statements.

The preprocessor locates, reads, and processes copy statements by using the copy members as input. The order of search, which conforms to z/OS standards, depends upon the order in which you have concatenated the libraries. Copy members are retrieved from the data sets specified by SYSLIB DD statements. The preprocessor scans the input for the IMS macros listed in Table 3.

Table 3. IMS macros scanned by the preprocessor

APPLCTN	NAME
DATABASE	RTCODE
MSLINK	SUBPOOL
MSNAME	TERMINAL
MSPLINK	TRANSACT

The preprocessor does not recognize keywords that are split across continuation statements. The related parameters associated with these keywords generate error messages from the preprocessor.

After scanning for macros, the preprocessor builds resource name tables for each resource name type. It then performs the following verification steps:

- Within each resource type, the preprocessor verifies that no resources of that type have duplicate names. The following types of resource names are verified:
  - DBD names
  - PSB names
  - VTAM node names
  - MS link names
  - Logical terminal names
  - Transaction codes
  - Routing codes
  - Subpool names
  - MSLINK partner IDs
  - MSPLINK physical link names
  - Remote system VTAM node names
- These resource names are checked to ensure that they are of the appropriate length and are alphanumeric.
- The names specified for transactions, logical terminals, and multiple systems are cross-checked to ensure that no duplicate names exist across these three resource types.

Also see the following sections of this overview of the systems definition preprocessor:

- "LGEN stage 1 processing"
- "LGEN additional processing" on page 17
- "Estimating storage requirements for the preprocessor" on page 18
- "Preprocessor exit routines" on page 19
- "Sample JCL to execute the preprocessor" on page 20

LGEN stage 1 processing

Use the LGEN configuration when more than 4 MB of private storage are needed in stage 1 or in any stage 2 assembly job step.

Stage 1 processing for LGEN system generation differs from a standard system definition. You must execute the preprocessor to assemble an LGEN generation. A stand-alone assembly of an LGEN generation causes an error. Stage 1 source statements are used as input and can include copy statements. However, the stage 1 input source statements cannot include inline copies of IMS macros, inline user macros, or user macro calls.

In turn, the preprocessor checks the resource name. The preprocessor uses storage above the 16 MB line and calls the assembler multiple times, called "cycles", to assemble all the input. This ensures that the stage 1 execution requires only 4 MB of private storage. You receive two reports from stage 1 processing: a summary of the return codes for each assembly and a summary of error messages.

When running LGEN, do not include assembler listing control instructions (such as PRINT OFF) in the stage 1 source statements. If you do so, you may have unpredictable results, because LGEN processes the output from the assembler listing.

The results of the assembly cycles are:

- A job stream that creates members of a PDS for the IMS control blocks
- The job steps that are dictated by the system definition type

The PDS members are held in the LGENIN data set and are used as input to the Sort/Split utility. The Sort/Split utility sorts all the PDS members for a given resource type and then splits them into parts that are to be assembled in a 4 MB region. Each part becomes a member of a PDS stored in the LGENOUT data set. The Sort/Split utility supplies a report about its processing.

LGEN additional processing

When the preprocessor executes with LGEN specified, the following additional processing occurs:

- Stage 1 source statements are saved in storage above the 16 MB line according to resource type.
- The specifications for the system identification (SYSID) keyword on APPLCTN, TRANSACT, and MSNAME stage 1 source statements are checked for syntax errors. They are also cross-checked to ensure that no conflicts exist between the **local** system identification keywords and the **remote** system identification keywords.
- If no errors are detected by the preprocessor, the assembler is called multiple times to process the stage 1 source statements that were saved. Input to each assembly consists of all system macro statements and up to 10000 of either DB, MSC, or VTAM resource. All non-VTAM resources (such as spool terminals) are processed during a single execution of the assembler.

  The output of the preprocessor is a listing of the input records, a listing of diagnostic and error messages, and a return code.

  You can optionally write two exit routines, DFSPRE60 and DFSPRE70, to perform additional processing.

  Related Reading: Information about the register and coding requirements for the DFSPRE60 and DFSPRE70 exit routines can be found in *IMS Version 11 Exit Routines*.

  If an LGEN configuration is active, the output includes the following:

  – Two summary reports that are printed before any other preprocessor output. The first report contains the return code for each assembly cycle and the total

number of resources processed by the preprocessor. The second report lists each error message in an assembly cycle and the five records that preceded the error message.

- Information from the listing of each assembly is appended to the preprocessor listing to form a consolidated listing data set. A list of the cycle macros is included when any assembly fails with a return code greater than 4. This list aids problem determination.
- The object data set from each assembly is appended to the object data sets of all previous assemblies to form a consolidated object data set. Execution of the job stream in this data set is functionally equivalent to stage 2 processing of a standard system definition for fewer resources.

In the LGEN configuration, the IMSCTRL NAMECHK=NO option is ignored and need not be specified. Under LGEN, no sorting is done in the stage 1 or stage 2 assemblies. Sorting is done in the preprocessor and by the Sort/Split utility.

Estimating storage requirements for the preprocessor

For both the standard and LGEN system definitions, the preprocessor uses the default values to reserve an initial amount of storage for each resource name table (RNT). If this storage is insufficient, the preprocessor dynamically expands the table until all extended private storage is exhausted. Resource names not added to the appropriate RNT are ignored and excluded from further processing.

Additional space is required if the number of errors for any given resource name exceeds 50, or if the number of nested copy statements exceeds 50. For most cases, this additional storage is accounted for in the base system storage number.

The storage size is the minimum region size that can be specified on preprocessor invocation JCL.

The preprocessor also requires 1 MB of private storage below the 16 MB line.

Standard system definition:

The storage estimate is for extended private storage. The storage requirement is the sum of:

1. 1000 KB for the base system.
2. Space needed for exit routines DFSPRE60 and DFSPRE70, if they are loaded.
3. Table space in excess of the default. For each of the following resource types that exceeds 5120, determine the amount of storage from Table 4.
   - APPLCTN
   - DATABASE
   - NAME
   - RTCODE
   - SUBPOOL
   - TERMINAL (VTAM)
   - TRANSACT

Table 4. Resource name table storage

Resource range	Storage amount in K bytes
5120 to 10,240	80

Table 4. Resource name table storage (continued)

Resource range	Storage amount in K bytes
10,240 to 20,480	240
20,480 to 41,060	560
41,060 to 82,120	1200
82,120 to 164,240	2480
164,240 to 328,480	5040
328,480 to 656,960	10,160
656,960 to 1,313,920	20,400

LGEN system generation:

If the total number of resources is less than 200000, allow 32 MB for extended private storage.

When the total number of resources exceeds 200000 resources, determine the amount of extended private storage that is required.

The sum of the following items, numbered 1 and 2, is the total extended private storage (in kilobytes) required to run the preprocessor with an LGEN system generation. The preprocessor also requires 4 MB of private storage below the 16-MB line.

LGEN limits the amount of private storage required to 4 MB for DB, MSC, and VTAM resources. All non-VTAM resources (such as spool terminals) are assembled in a single assembly cycle. Many non-VTAM resources can require more than 4 MB of private storage for the stage 1 and stage 2 assemblies.

1. Storage for system definition input is the sum of:
   - 3000 for the base system ((Number of system definition input records/8192) x 584)
   - 584 if any non-VTAM resources are defined
   - 584 if any MSC resources are defined
   - 584 if any VTAM resources are defined
2. Table space in excess of default. For each of the following resource types that exceeds 5120, determine the amount of storage from Table 4 on page 18.
   - APPLCTN (total)
   - APPLCTN (remote)
   - DATABASE
   - NAME
   - RTCODE
   - SUBPOOL
   - TERMINAL (VTAM)
   - TRANSACT (total)
   - TRANSACT (remote)

Preprocessor exit routines

You can develop exit routines that gain control during the execution of the preprocessor. You can use one or both of the following exit routines:

- Exit DFSPRE60

  This routine gains control after each record in the stage 1 input is read, but before any other processing takes place. It can modify the contents of the record and can even submit further statements to the preprocessor for checking. Any changes made by this routine are not permanent, nor are these changes automatically passed to stage 1.
- Exit DFSPRE70

  This routine gains control when all cross-checking has been completed. It has access to all the tables of resource names. The routine can then format these tables as part of a documentation effort.

Sample JCL to execute the preprocessor

To execute the preprocessor, you must provide JCL as shown in the following example.

```
//         JOB
//         EXEC PGM=DFSPRE00,REGION=32M,PARM='xxx' [1]
//STEPLIB  DD   DSN=IMS.SDFSRESL,DISP=SHR [2]
//         DD   DSN=USER.EXITLIB,DISP=SHR [3]
//SYSLIB   DD   DSN=USER.MACLIB,DISP=SHR [4]
//         DD   DSN=IMS.SDFSMAC,DISP=SHR [4]
//SYSIN80  DD   DSN=&&SYS80, [5]
//  UNIT=SYSDA,SPACE=(CYL,(5,5)), [5]
//  DCB=(BLKSIZE=11440,LRECL=80,RECFM=FB,DSORG=PS) [5]
//SYSUT1   DD   DSN=&SYSUT1, [5]
//  UNIT=SYSDA,SPACE=(CYL,(10,5)) [5]
//SYSLIN   DD   DSN=&&SYSLN, [5]
//  UNIT=SYSDA,SPACE=(CYL,(1,1)), [5]
//  DCB=(BLKSIZE=11440,LRECL=80,RECFM=FB,DSORG=PS) [5]
//SYSPRT80 DD   DSN=&&SYSPRT, [5]
//  UNIT=SYSDA,SPACE=(CYL,(5,5)), [5]
//  DCB=(BLKSIZE=3146,LRECL=121,RECFM=FBM,DSORG=PS) [5]
//SYSSUMPR DD   SYSOUT=A [6]
//SYSCYLPR DD   SYSOUT=A [6]
//SYSPRINT DD   SYSOUT=A
//SYSCOBJ  DD   DSN=IMS.SYSCOBJ,DISP=(NEW,KEEP), [7]
//  UNIT=SYSDA,SPACE=(CYL,(5,5)), [7]
//  DCB=(BLKSIZE=11440,LRECL=80,RECFM=FB,DSORG=PS) [7]
//SYSIN    DD   * [8]
//SYSTERM  DD   SYSOUT=* [9]
```

Notes:

1. The region size indicated is the size required to execute the preprocessor on a z/OS system. The LGEN subparameter is specified in the SYSTEM= keyword on the IMSCTRL source statement. Guidelines for determining the region sizes without LGEN are in "Estimating storage requirements for the preprocessor" on page 18.

   The PARM field is specified only if the default exit routine indicators should be overridden.
2. This DD statement should specify the library containing the preprocessor version that you want to execute. Different versions can exist because of SMP maintenance.
3. This concatenated DD statement is required only if exit routines are requested.
4. The SYSLIB DD statement should point to a library that contains the copy members, but with an LGEN definition, the statement should point to desired system definition macro instruction libraries. This DD statement is required if stage 1 source COPY statements or an LGEN configuration exists.
5. Only the LGEN configuration requires these DD statements.

6. Only the LGEN configuration requires these DD statements. These DD statements must appear before the SYSPRINT DD statement to allow the summary reports to precede the consolidated assembly listing.
7. Only the LGEN configuration requires this data set; it contains the consolidated object module from all the LGEN assembler cycles. The input/output unit assigned to this data set can be a card punch or an intermediate storage device capable of sequential access.
8. You can also use the statement:

   ```
   //SYSIN DD  DSN=....,DISP=SHR
   ```
9. To see the assembler error out messages, the assembler parameter TERM needs to be used and a DD statement for SYSTERM needs to be present in the JCL.

The PARM field on the EXEC statement is specified as follows:`PARM='a,b,(asmopt1,asmopt2,...,asmoptn)'`

The parameters specified are positional and are specified as follows:

a Indicates whether (Y) or not (N) exit routine DFSPRE60 is to be used during this invocation of the preprocessor. If DFSPRE60 is to be used, it must reside on the libraries pointed to by the STEPLIB DD statement. The default is N.

b Indicates whether (Y) or not (N) exit routine DFSPRE70 is to be used during this invocation of the preprocessor. If DFSPRE70 is to be used, it must reside on the libraries pointed to by the STEPLIB DD statement. The default is N.

asmopt1, ..., asmoptn
Assembler options you want for stage 1 assemblies in the LGEN configuration. Enclose the options in parentheses, and use the comma as a delimiter when using more than one option. The preprocessor forces the LIST and OBJECT options.

Related Reading: For more information about the LGEN configuration, refer to *IMS Version 11 System Administration*.

IMS stage 1 system definition

Stage 1 of IMS system definition takes the IMS macros that you coded and assembles them into Stage 2 JCL.

Stage 1 of the system definition process uses the z/OS High Level Assembler program and uses the IMS macros as input. Other references are to the IMS distribution macro libraries (IMS.ADFSMAC).

The output from stage 1 of the IMS system definition process includes:

- Standard assembler listing output with any appropriate error messages.
- Stage 2 system definition input JCL, which is also used for the JCLIN process.

Depending on what is specified in the IMSGEN macro for stage 1, stage 2 of the system definition process can be divided up into a single job with multiple steps, or into many jobs with fewer steps.

You specify the assembler and binder data sets and options, and the system definition output options and features in the IMSGEN macro.

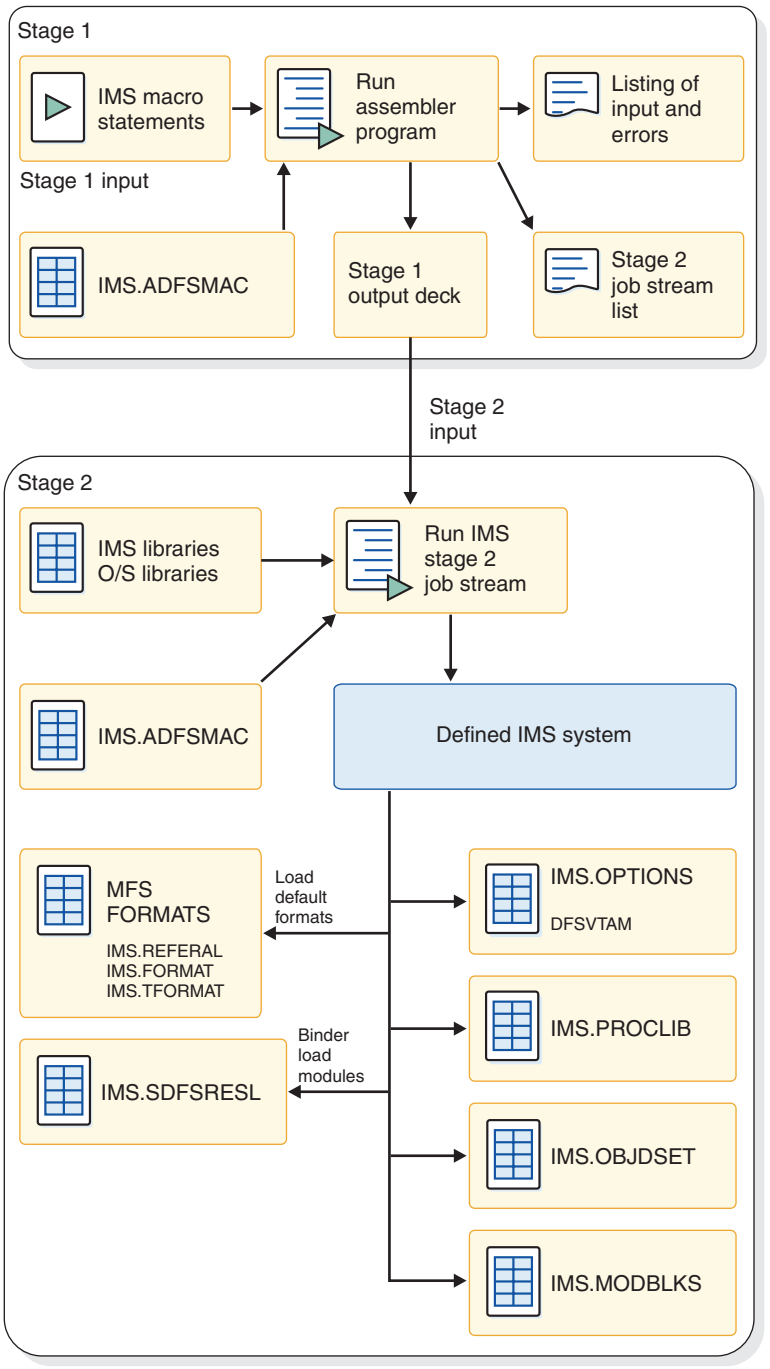


Figure 3. Stage 1 and Stage 2 of the system definition process

Verifying the stage 1 input

After stage 1 input is built, resource names are checked for validity uniqueness. You can bypass name checking to reduce overall processing.

Stage 1 input contains a structured definition of many resources to be used by the IMS online control program. It is important that you verify the content of the stage 1 input and the accuracy of the macro statement coding. The two kinds of verification are:

- Resource name checks
- Macro statement checks

Resource name checks

The names of resources are important not only from a documentation standpoint (conformance to naming conventions), but also from an operational standpoint. An LTERM name could be explicitly used by the MTO or it can be coded in an application program. As such, the resource names must be unique. Also, IMS reserves the use of certain resource names.

Stage 1 verifies that resource names are valid and appropriately unique. A benefit of this checking is to ensure that conflicts between resource definitions are detected before the control block building that takes place in stage 2. Some installations regularly execute stage 1 to perform this checking.

Macro statement checks

Because definite sequence requirements and dependencies exist between the parameter specifications, the macro statement checking performed in stage 1 is also valuable.

An option is available that reduces the processing performed by stage 1. The NAMECHK parameter on the IMSCTRL macro lets you bypass name checking—assuming you have made sure that no invalid or duplicate resource names exist. If your installation has made use of the optional preprocessor and you are checking a sizeable stage 1 input, specifying NAMECHK=NO can reduce the processing performed by stage 1. You can also specify that the sort is not performed in stage 1. (Specify S2 as the second value for the NAMECHK parameter.) When you perform the system definition and execute both stage 1 and stage 2, the default of NAMECHK=(YES,S1) is recommended so that full checking is part of the definition process.

You should track the progress of the stage 1 runs and, if necessary, investigate any detected problems.

Determining when system definition is required

You do not have to perform a complete system definition each time a parameter value changes.

For example, you can use JCL parameters to implement some system definition changes. However, if you add application programs (excluding APPC/IMS), databases, or physical changes (excluding ETO) to the network, redefinition of the IMS system is required. Use the SYSTEM keyword of the IMSCTRL macro to define different types of system definition.

Specifying alternative versions of an online system

You can define different configurations of the IMS online system. The control modules in the nucleus and control blocks carry a one-character suffix. You specify

this suffix with the SUFFIX= keyword on the IMSGEN macro. The default value is 0 (zero). The value specified for the control region parameter SUF= controls which IMS configuration is to be executed.

System definition stage 1 output

A successful stage 1 execution generates the following output:

- An extensive series of self-contained jobs.
- Warning messages, which are embedded within a listing of your input source.
- A trailer to this listing containing instructions on the content and execution of the stage 2 job stream.

Because the amount of stage 2 processing time is significant, you should review the stage 1 messages carefully. A misplaced NAME macro or misspelled LTERM name can cause service to be unavailable to an end user and require an additional stage 2 execution.

Stage 2 processing builds a control program, often called a *nucleus*, which is tailored according to the specifications you made in stage 1. The control program, along with most of the internal control blocks, is placed in the IMS.SDFSRESL data set.

The SYSTEM keyword on the IMSCTRL macro determines the extent of stage 2 processing.

Defining a large system

An IMS system that requires more than 9 MB of private storage requires a large system definition. A large system definition provides more storage for IMS system resources and requires two additional data sets.

If you need to define an IMS system with resources that require more than 9 MB of private storage during stage 1 processing, specify a *large system definition* using the LGEN parameter on the SYSTEM= keyword in the IMSCTRL system definition macro. A large system definition provides more storage for IMS system resources and requires two additional data sets.

Two additional steps are required if you are defining a large system:

1. Specify LGEN as the fourth subparameter of the SYSTEM keyword in the IMSCTRL macro. Define all other system definition macros as you would for any other type of system definition.
2. Allocate and catalog two additional data sets for the stage 1 processing of the large system definition. By default, these data sets are named LGENIN and LGENOUT.

Stage 1 and Stage 2 processing for large system definitions

The stage 1 processing for a large system definition is different from a normal system definition. You must execute the preprocessor to assemble a large system definition. A stand-alone assembly of a large system definition causes an error.

In turn, the preprocessor performs the necessary resource name checking. The preprocessor uses storage above the 16 MB line, assembling the generation in cycles. This ensures that the application program requires only 4 MB of private

storage. You receive two reports from the stage 1 processing: a summary of return codes for each assembly and a summary of error messages.

The stage 1 processing for a large system definition creates altered job steps for stage 2 processing. But the effect of this is transparent to you and to IMS function.

The output from the assembly cycles is the job stream that creates members of a partitioned data set (PDS). These members are held in the LGENIN data set and are used as input to the Large System Definition Sort/Split Input exit routine (DFSSS050). The Large System Definition Sort/Split Input exit routine sorts all the PDS members for a given resource type and then splits them into parts that assemble in a 4 MB region. Each part becomes a member of a PDS, stored in the LGENOUT data set. The Large System Definition Sort/Split Input exit routine then supplies you with a report about its processing.

The Large System Definition Sort/Split Input exit routine processes each resource type individually and includes both standard and customized resources. Customized, or user-defined, resources are normally created by modifying stage 1 macros. A control record is required for each user-defined resource in the Resource Information file member. This record tells the Large System Definition Sort/Split Input exit routine to process the resource.

Related reading: For more information about customized resources and the Large System Definition Sort/Split Input exit routine, see *IMS Version 11 Exit Routines*.

Using online change with large IMS systems

You can use online change with large system definitions if you include the online change specifications during large system definition. You can use online change to incorporate the LGENIN and LGENOUT data sets required by large system definitions. If you do not specify online change during system definition, you must shut down and restart IMS to incorporate any changes.

Building ETO descriptors in large system definitions

If you want to build ETO descriptors as part of a large IMS system, you must run a standard CTLBLKS system definition after running an LGEN system definition.

If you need to build ETO descriptors in a large system:

1. Run system definition with the LGEN parameter specified first (in both stage 1 and stage 2).
2. Run a standard CTLBLKS system definition and specify ETOFEAT=(,,ONLY) in the IMSCTRL macro.

Large systems defined using the LGEN parameter do not support the building of ETO descriptors.

Including Fast Path in a DCCTL or DB/DC system definition

During a DCCTL or DB/DC system definition process, you can specify which Fast Path application programs and resources should be included in the IMS system.

Fast Path drives relatively simple transaction and database processing through the IMS system at high transaction rates. You must assess whether the combination of

Fast Path regions and other regions, as well as the increased requirement for control program storage, can be supported in your system definition.

To include the Fast Path control program facilities and to reserve certain resources for Fast Path in an IMS online system, specify FP=Y in the DFSPBxxx member in the IMS PROCLIB data set. To include Fast Path processing and transactions in a DCCTL system, specify FP=Y in the DFSPBxxx member in the IMS PROCLIB data set.

If you specify FP=N, and you attempt to use a Fast Path resource or command, the results are unpredictable.

To specify Fast Path options and buffer sizes for an IMS online system, use the Fast Path parameters in the IMS procedure.

You do not need to include the Fast Path parameters for a DCCTL system because Fast Path parameters apply only to Fast Path databases, and the DCCTL environment does not support Fast Path databases. Although the DCCTL environment does not support Fast Path databases, it does support Fast Path processing and transactions.

Use the following macros to describe the Fast Path application programs and associated resources:

APPLCTN
Declares the application programs

RTCODE
Directs input messages to Fast Path programs

TRANSACT
Declares the transaction code and processing characteristics

In an IMS DB/DC system, you can specify the following keyword parameters:

OTHR
Keyword specifying how many output threads are to be used for the asynchronous DEDB update processing.

The OTHR= parameter can have any value from 1 to 32,767. If a value is not specified, the default is 255. If `0` or a value greater than `32767` is specified, the default is 2. If a value from `1` to `32767` is specified, the specified value is used.

DBFX Keyword specifying the number of buffers to be set aside for DEDB updates used by sync point processing. These buffers are part of the total number given for DBBF.

DBBF Keyword specifying the maximum number of buffers available to the online system for MSDB and DEDB processing. The range for both numbers is 1 - 4 294 967 295. The default values are 4 for MSDBs and 10 for DEDBs.

BSIZ Keyword specifying the actual size of an individual buffer. The control interval size is 512 bytes, 1 KB, 2 KB, 4 KB, or a multiple of 4 KB (up to 28 KB), depending on the size of the largest control interval (CI) used for DEDB processing.

FPBP64
Enables (Y) or not (N) the Fast Path 64-bit buffer manager. This keyword is

specified in the <SECTION=FASTPATH> section of the DFSDFxxx PROCLIB member. If you choose to use the Fast Path 64-bit buffer manager (FPBP64=Y), the DBBF, DBFX, and BSIZ parameters that define Fast Path buffers are ignored. For more information about the FPBP64 keyword, see DFSDFxxx member of the IMS PROCLIB data set (System Definition).

Including Fast Path in a DBCTL system definition

During the DBCTL system definition process, you can specify which Fast Path application programs and resources should be included in the IMS system.

Describe Fast Path application programs as part of system definition input with:

- The FP=Y keyword in the DFSPBxxx member of the IMS.PROCLIB data set, which causes Fast Path facilities to be included in the IMS online system and reserves certain resources for Fast Path.

  If you specify FP=N, and you attempt to use a Fast Path resource or command, the results are unpredictable.
- The APPLCTN macro, which creates the application program definitions.

You use a system environment macro to specify Fast Path control program facilities in an IMS DBCTL environment in the same way that you use macros to specify options and buffer sizes.

Use the OTHR parameter of the IMS procedure to specify how many output threads are to be used for the asynchronous DEDB update processing.

The OTHR= parameter can have any value from 1 to 32,767. If a value is not specified, the default is 255. If `0` or a value greater than `32767` is specified, the default is 2. If a value from `1` to `32767` is specified, the specified value is used.

Use the DBFX parameter of the IMS procedure to specify the number of buffers to be set aside for DEDB updates used by sync point processing. These buffers are part of the total number given for the next parameter, DBBF. The DBBR parameter is used to specify the maximum number of buffers available to the online system for DEDB processing. Use the BSIZ parameter to specify the actual size of an individual buffer.

Related reference:

"APPLCTN macro" on page 304

"IMS procedure" on page 560

"DFSPBxxx member of the IMS PROCLIB data set" on page 718

IMS stage 2 system definition

IMS stage 2 system definition assembles and binds all the modules that are required to build the necessary load modules, depending on what type of system is being defined.

The steps involved in stage 2 refer to the IMS distribution macro library (IMS.ADFSMAC) at assembly time, and the distribution load library (IMS.ADFSLOAD) at bind time.

The output of stage 2 of the system definition process includes:

- Executable load modules in data sets IMS.SDFSRESL and IMS.MODBLKS.

- IMS options definitions in data set IMS.OPTIONS.
- Assembled object code for use in later system definition steps in data sets IMS.OBJDSET.
- Optionally, the runtime IMS. PROCLIB data set.

  The PROCLIB= parameter in the IMSGEN stage 1 macro determines whether the IMS PROCLIB data set is to be populated by this system definition. The IMS PROCLIB data set contains IMS started tasks and JCL procedures, as well as the IMS PROCLIB data set members required by IMS and IMS utilities to provide options.
- Optionally, the runtime IMS default MFS screens in data sets IMS.FORMAT, IMS.TFORMAT, and IMS.REFERAL.

  The MFSDFMT= parameter in the IMSGEN stage 1 macro determines whether the default message format screens are built as part of stage 2 of the system definition process.

JCLIN processing

The JCLIN process ensures that SMP/E knows how to manage any maintenance that is added to the system following an IMS system definition.

Because the stage 2 system definition process actually assembles and binds the IMS modules based on the definitions for that particular system and is run outside of SMP/E control, the input JCL for system definition stage 2 must be used as input to the JCLIN process. This input JCL ensures that SMP/E knows how to manage any maintenance that is added to the system following this IMS system definition.

Run the JCLIN process following any IMS system definition, to ensure that SMP/E is always synchronized with the updated IMS.

Applying maintenance using SMP/E

Use SMP/E to apply and accept IMS maintenance before an IMS system definition.

All IMS system definitions use the IMS SMP/E distribution libraries and the IMS stage 1 macros as input. As a result, an IMS system definition might reverse the changes from any SMP/E maintenance (SYSMODs - PTFs, APARs, or USERMODs) that was processed using the SMP/E `APPLY` command, but not processed using the SMP/E `ACCEPT` command. This depends on the type of IMS system definition, and the impact of the SYSMOD.

Related reading: For more information about performing SMP/E maintenance on IMS, see IMS service considerations (System Administration).

Adding entries to the z/OS Program Properties Table

Use the z/OS Program Properties Table (PPT) to identify a list of programs that require special attributes, or to change the attributes of the IBM-supplied default entries.

In an IMS environment, you must update the PPT for the following IMS components or address spaces:

- IMS control region
- Base Primitive Environment
- Common Queue Server

- Common Service Layer
- IMS Connect
- IRLM

For more information about updating the program properties table, see the topic on the SCHEDxx SYS1.PARMLIB member in *z/OS MVS™ Initialization and Tuning Reference*

1. Edit the SCHEDxx member of the SYS1.PARMLIB data set.
2. Add the entry for the component or address space to the SCHEDxx member. Examples are shown below.
3. To make the SCHEDxx changes effective, you can do one of the following:
   a. IPL the z/OS system again.
   b. Issue the z/OS SET SCH= command.

Updating the PPT for the IMS control region

An IMS online environment (DB/DC, DBCTL, DCCTL) requires this PPT entry.

Although DFSMVRC0 is predefined in the default PPT that is shipped with z/OS V1R4 and later, you must ensure that there is an entry for module DFSMVRC0 in the z/OS Program Properties Table. If you did not modify the default z/OS PPT, no further action is required. If you removed the default entry for DFSMVRC0, you must reinstate this entry using the procedure above.

If you are only using IMS BATCH, this entry is not needed.

A sample of the required entry is shown below and can be found in the IMS.INSTALIB data set. See "IVP jobs and tasks" in *IMS Version 11 Installation* for the correct entry titled "Update SCHEDxx -- PPT Entries".

```
                        /* IMS ONLINE CONTROL REGION           */
PPT PGMNAME(DFSMVRC0)   /* PROGRAM NAME = DFSMVRC0             */
         CANCEL         /* PROGRAM CAN BE CANCELLED            */
         KEY(7)         /* PROTECT KEY ASSIGNED IS 7           */
         NOSWAP         /* PROGRAM IS NOT-SWAPPABLE            */
         NOPRIV         /* PROGRAM IS NOT PRIVILEGED           */
         SYST           /* PROGRAM IS A SYSTEM TASK            */
         DSI            /* DOES REQUIRE DATA SET INTEGRITY     */
         PASS           /* PASSWORD PROTECTION ACTIVE          */
         AFF(NONE)      /* NO CPU AFFINITY                     */
         NOPREF         /* NO PREFERRED STORAGE FRAMES         */
```

The PPT Entry for program DFSMVRC0 must specify NOSWAP as shown.

Updating the PPT for BPE

Although BPEINI00 is predefined in the default PPT that is shipped with z/OS, you must ensure that there is an entry for module BPEINI00 in the z/OS Program Properties Table. If you did not modify the default z/OS PPT, no further action is required. If you removed the default entry for BPEINI00, you must reinstate this entry using the procedure above.

Separate PPT entries for BPE-based DBRC and for IMS Connect are not required.

```
    PPT PGMNAME(BPEINI00)   /* PROGRAM NAME = BPEINI00       */
        CANCEL              /* PROGRAM CAN BE CANCELED       */
        KEY(7)              /* PROTECT KEY ASSIGNED IS 7     */
        NOSWAP              /* PROGRAM IS NON-SWAPPABLE      */
```

```
         NOPRIV             /* PROGRAM IS NOT PRIVILEGED     */
         DSI                /* REQUIRES DATA SET INTEGRITY   */
         PASS             /* CANNOT BYPASS PASSWORD PROTECTION */
         SYST               /* PROGRAM IS A SYSTEM TASK      */
         AFF(NONE)          /* NO CPU AFFINITY               */
         NOPREF             /* NO PREFERRED STORAGE FRAMES   */
```

Updating the PPT for CQS

If you are using CQS, either the CQSINIT0 or the BPEINI00 z/OS PPT entry is required.

Although CQSINIT0 and BPEINI00 are predefined in the default PPT that is shipped with z/OS V1R4 and later, you must ensure that there is an entry for module CQSINIT0 or BPEINI00 in the z/OS Program Properties Table. If you did not modify the default z/OS PPT, no further action is required. If you removed the default entries for CQSINIT0 and BPEINI00, you must reinstate at least one of those entries using the procedure above.

A sample of the CQSINIT0 entry is shown below and can be found in the IMS.INSTALIB data set.

```
                        /* CQS - COMMON QUEUE SERVER           */
PPT PGMNAME(CQSINIT0)   /* PROGRAM NAME = CQSINIT0             */
         CANCEL         /* PROGRAM CAN BE CANCELLED            */
         KEY(7)         /* PROTECT KEY ASSIGNED IS 7           */
         NOSWAP         /* PROGRAM IS NOT-SWAPPABLE            */
         NOPRIV         /* PROGRAM IS NOT PRIVILEGED           */
         SYST           /* PROGRAM IS A SYSTEM TASK            */
         DSI            /* DOES REQUIRE DATA SET INTEGRITY     */
         PASS           /* PASSWORD PROTECTION ACTIVE          */
         AFF(NONE)      /* NO CPU AFFINITY                     */
         NOPREF         /* NO PREFERRED STORAGE FRAMES         */
```

The PPT Entry for program CQSINIT0 must specify NOSWAP as shown.

Updating the PPT for CSL

The Common Service Layer (CSL), which consists of address spaces operations manager (OM), resource manager (RM), and structured call interface (SCI), requires an entry in the PPT. Only one entry is necessary for the CSL.

Although BPEINI00 is predefined in the default PPT that is shipped with z/OS V1R4 and later, you must ensure that there is an entry for module BPEINI00 in the z/OS Program Properties Table. If you did not modify the default z/OS PPT, no further action is required. If you removed the default entry for BPEINI00, you must reinstate this entry using the procedure above.

```
                        /* CSL - COMMON SERVICE LAYER           */
PPT PGMNAME(BPEINI00)   /* PROGRAM NAME = BPEINI00             */
         CANCEL         /* PROGRAM CAN BE CANCELLED            */
         KEY(7)         /* PROTECT KEY ASSIGNED IS 7           */
         NOSWAP         /* PROGRAM IS NOT-SWAPPABLE            */
         NOPRIV         /* PROGRAM IS NOT PRIVILEGED           */
         DSI            /* REQUIRES DATA SET INTEGRITY         */
         PASS           /* CANNOT BYPASS PASSWORD PROTECTION   */
         SYST           /* PROGRAM IS A SYSTEM TASK            */
         AFF(NONE)      /* NO CPU AFFINITY                     */
         NOPREF         /* NO PREFERRED STORAGE FRAMES         */
```

Updating the PPT for IMS Connect

IMS Connect uses the predefined z/OS Program Properties Table entry BPEINI00, but HWSHWS00 is also supported.

If using BPEINI00, you must ensure that there is an entry for module BPEINI00 in the z/OS Program Properties Table. If you did not modify the default z/OS PPT that was predefined in the default PPT that is shipped with z/OS V1R4 and later, no further action is required. If you removed the default entry for BPEINI00, you must reinstate this entry using the procedure above.

The examples shown below use HWSHWS00 instead of BPEINI00.

You can include both entries in the PPT, for example, in an environment that runs both IMS Connect Version 10 and IMS Connect Version 11.

For TCP/IP communications only, add the following entry in the z/OS PPT:

```
PPT PGMNAME(HWSHWS00)     /* PROGRAM NAME = HWSHWS00           */
         CANCEL           /* PROGRAM CAN BE CANCELED           */
         KEY(7)           /* PROTECT KEY ASSIGNED IS 7         */
         SWAP             /* PROGRAM IS SWAPPABLE              */
         NOPRIV           /* PROGRAM IS NOT PRIVILEGED         */
         DSI              /* REQUIRES DATA SET INTEGRITY       */
         PASS             /* CANNOT BYPASS PASSWORD PROTECTION */
         SYST             /* PROGRAM IS A SYSTEM TASK          */
         AFF(NONE)        /* NO CPU AFFINITY                   */
         NOPREF           /* NO PREFERRED STORAGE FRAMES       */
```

If you are using local option for client communications, either by itself or with TCP/IP communications, add the following entry in the z/OS PPT:

```
PPT PGMNAME(HWSHWS00)     /* PROGRAM NAME = HWSHWS00           */
         CANCEL           /* PROGRAM CAN BE CANCELED           */
         KEY(7)           /* PROTECT KEY ASSIGNED IS 7         */
         NOSWAP           /* PROGRAM IS NOT SWAPPABLE          */
         NOPRIV           /* PROGRAM IS NOT PRIVILEGED         */
         DSI              /* REQUIRES DATA SET INTEGRITY       */
         PASS             /* CANNOT BYPASS PASSWORD PROTECTION */
         SYST             /* PROGRAM IS A SYSTEM TASK          */
         AFF(NONE)        /* NO CPU AFFINITY                   */
         NOPREF           /* NO PREFERRED STORAGE FRAMES       */
```

Updating the PPT for IRLM

If you are using IRLM, the following z/OS PPT entry is required.

Although DXRRLM00 is predefined in the default PPT that is shipped with z/OS V1R4 and later, you must ensure that there is an entry for module DXRRLM00 in the z/OS Program Properties Table. If you did not modify the default z/OS PPT, no further action is required. If you removed the default entry for DXRRLM00, you must reinstate this entry using the procedure above.

A sample of the required entry is shown below and can be found in the IMS.INSTALIB data set.

```
                          /* IRLM - RESOURCE LOCK MANAGER        */
PPT PGMNAME(DXRRLM00)     /* PROGRAM NAME = DXRRLM00             */
           CANCEL         /* PROGRAM CAN BE CANCELLED            */
           KEY(7)         /* PROTECT KEY ASSIGNED IS 7           */
           NOSWAP         /* PROGRAM IS NOT-SWAPPABLE            */
           NOPRIV         /* PROGRAM IS NOT PRIVILEGED           */
           SYST           /* PROGRAM IS A SYSTEM TASK            */
           DSI            /* DOES REQUIRE DATA SET INTEGRITY     */
```

```
PASS            /* PASSWORD PROTECTION ACTIVE          */
AFF(NONE)       /* NO CPU AFFINITY                     */
NOPREF          /* NO PREFERRED STORAGE FRAMES         */
```

The PPT Entry for program DXRRLM00 must specify NOSWAP as shown.

Configuring IMS support for the IMS Universal drivers

To set up IMS support for the IMS Universal drivers, configure the CSL Open Database Manager (ODBM), IMS Connect, and any required ODBM, IMS Connect, or Base Primitive Environment (BPE) user exits, and then start ODBM and IMS Connect.

The following high-level summary describes the steps for configuring IMS support for the IMS Universal drivers:

1. Configure ODBM:
   a. Use the ODBM configuration member of the IMS PROCLIB data set (CSLDCxxx) to define the connections between ODBM and one or more data stores.
   b. Use the ODBM initialization member of the IMS PROCLIB data set (CSLDIxxx) to specify parameters that initialize the ODBM address space.
   c. Use the BPE configuration member of the IMS PROCLIB data set to define the BPE execution environment settings for ODBM.
2. Optional: Write CSL ODBM user exit routines to customize and monitor the ODBM environment, and then define these user exit routines to BPE:
   a. Write CSL ODBM user exits to customize and monitor the ODBM environment.
   b. Specify the ODBM user exit routine in the BPE exit list member of the IMS PROCLIB data set.
   c. Specify the BPE exit list member in the BPE configuration parameter member of the IMS PROCLIB data set.
3. Configure IMS Connect to communicate with ODBM. Use the ODACCESS statement of the IMS Connect configuration member of the IMS PROCLIB data set (HWSCFGxx) to define characteristics of the communication between ODBM and IMS Connect.
4. Use the BPE configuration member of the IMS PROCLIB data set to define the BPE execution environment settings for IMS Connect.
5. Optional: Write IMS Connect user exit routines to configure the connection to ODBM, and then define these user exit routines to BPE:
   a. Use the routing exit HWSROUT0 to override the IMS alias or select an ODBM.
   a. Use the IMS Connect DB security user exit routine, HWSAUTH0, to authenticate the input user ID and password specified by IMS Connect clients.
   b. Specify the IMS Connect user exit routines in the BPE exit list member of the IMS PROCLIB data set.
   c. Specify the BPE exit list member in the BPE configuration parameter member of the IMS PROCLIB data set.
6. Optional: Set up tracing for the BPE-managed ODBM address space.
7. Set up security for ODBM.
8. Set up security for IMS Connect.

9. Develop and deploy applications by using the IMS Universal drivers. You can also set up and deploy from external environments.
10. Start the following components:
    a. Start the Structured Call Interface (SCI).
    b. Start the Operations Manager (OM).
    c. Start the IMS control region.
    d. Start ODBM.
    e. Start IMS Connect.

Related concepts:

"Defining IMS Connect security" on page 212

How to start the IMS control region (Operations and Automation)

"Sample JCL to start IMS Connect" on page 209

Related tasks:

"Setting up tracing for BPE-managed address spaces" on page 244

IMS Universal drivers: WebSphere® Application Server type-4 connections (Communications and Connections)

Starting the CSL SCI (Operations and Automation)

Starting the CSL OM (Operations and Automation)

Starting the CSL ODBM (Operations and Automation)

Related reference:

"CSLDCxxx member of the IMS PROCLIB data set" on page 628

"CSLDIxxx member of the IMS PROCLIB data set" on page 633

"CSLODBM procedure" on page 515

"BPE configuration parameter member of the IMS PROCLIB data set" on page 587

"BPE exit list members of the IMS PROCLIB data set" on page 605

"HWSCFGxx member of the IMS PROCLIB data set" on page 782

CSL ODBM user exit routines (Exit Routines)

IMS Connect DB Routing user exit routine (HWSROUT0) (Exit Routines)

IMS Connect DB security user exit routine (HWSAUTH0) (Exit Routines)

Chapter 2. Dynamic resource definition

You can create, update, or delete certain IMS resources and add them to your IMS system dynamically, thereby eliminating the need to use the batch system definition or online change processes. This process is called the dynamic resource definition (DRD) process.

Only certain IMS resources, which are organized by groups of resource types, support dynamic definition. The following resource groups support dynamic definition:

- MODBLKS resources, which include definitions for runtime attributes of application programs, databases, Fast Path routing codes, and transactions. MODBLKS resources are so named because when dynamic resource definition is not used, these resources are generated into the MODBLKS data set by the IMS system definition process.

Overview of dynamic resource definition

With dynamic resource definition (DRD), you can use type-2 commands to define resources such as application programs, databases, routing codes, and transactions, along with resource descriptors.

MODBLKS resources

For IMS systems in which DRD for MODBLKS resources is not enabled or is unavailable, you must use the APPLCTN, DATABASE, RTCODE, and TRANSACT macros, along with the batch system definition process, to create the resource definitions for these resources and store them in the IMS.MODBLKS data set. The following figure shows the control block generation process:

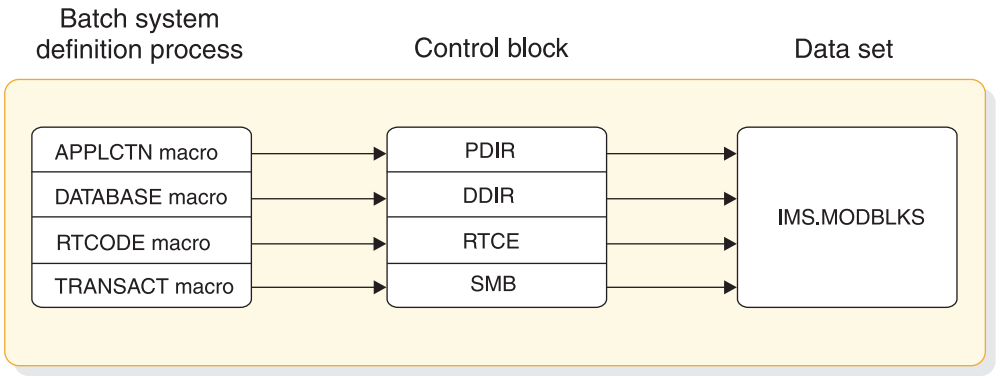


Figure 4. Control block generation process for MODBLKS resources

With DRD enabled, the APPLCTN, DATABASE, RTCODE, and TRANSACT macros are optional. If you do not code these macros as part of the system definition process, you can either import resource definitions into IMS from a resource definition data set (RDDS), or you can use type-2 commands to define resources to IMS dynamically (instead of using online change).

© Copyright IBM Corp. 1974, 2015

For IMS systems in which DRD is not enabled or unavailable, after IMS initialization, you use the online change process to add, change, and delete resource definitions dynamically. The online change process requires that you:

1. Generate the resource definitions and store them in an IMS.MODBLKS staging library data set
2. Run the Online Change Copy utility (DFSUOCU0) to copy the staging library into an inactive library
3. Run a series of online change commands to cause the change to take effect (see the figure below)

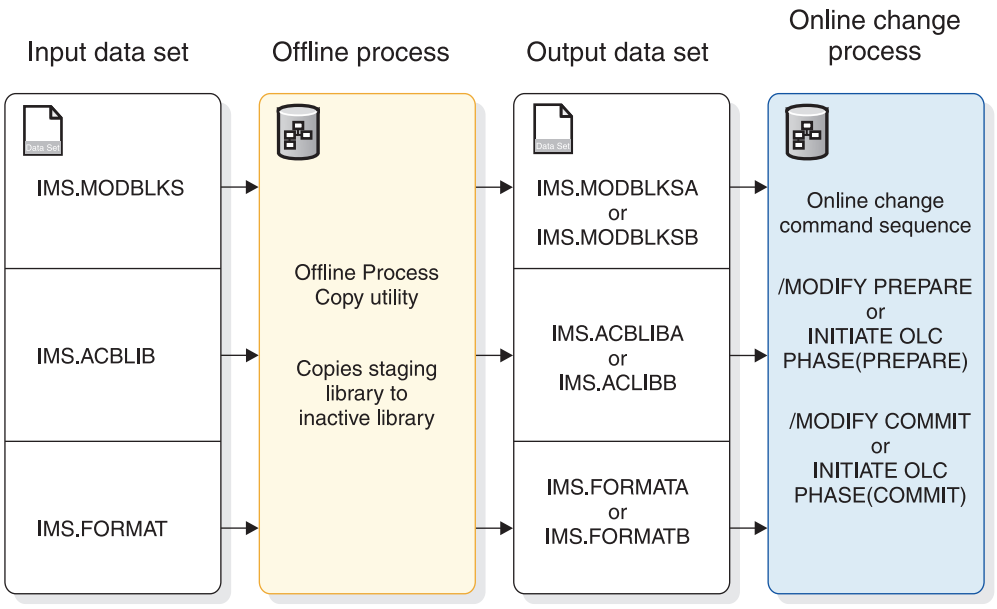


Figure 5. Online change process for IMS resources

Another aspect of the online change process is that, during the COMMIT phase, all the resources associated with the definitions in the IMS.MODBLKS data set are quiesced, which affects IMS availability.

Recommendation: Unless you have a simple system, and online change meets your requirements, use DRD with RDDSs rather than the IMS.MODBLKS data set.

To change some resources online (for example, resources in IMS.ACBLIB), you cannot use DRD and must use the online change process.

Related concepts:

Modifying system resources online (Operations and Automation)

Managing resource and descriptor definitions

With dynamic resource definition (DRD) enabled, you can perform several tasks to manage the resource and descriptor definitions for your IMS system.

With DRD enabled, you can perform the following tasks from a single point of control (SPOC) application, which can be the IMS Control Center, TSO SPOC, vendor-supplied software, or an application program that you write:

- Create resources or descriptors
- Update resources or descriptors

- Delete resources or descriptors
- Query definitional attributes of resources or descriptors (by using the QUERY command)
- Export resource and descriptor definitions from IMS to a resource definition data set (RDDS)
- Import resource and descriptor definitions from an RDDS into IMS, which creates resources at IMS initialization without using the system definition process

With DRD, you can use the CREATE, IMPORT, UPDATE, and DELETE type-2 commands to dynamically create, update, and delete application program, database, routing code, and transaction resource and descriptor definitions.

Issue the type-2 commands from a single point of control (SPOC) application (for example, the TSO SPOC shipped with IMS). You can also use the IMS Manage Resources application that is available from the IMS Application Menu (option 2).

Resource descriptors are templates that can be used to define new resources and descriptors. IMS supplies four resource descriptors, one for each resource type. The descriptors contain IMS-system default values for each resource attribute. These IMS-supplied descriptors cannot be deleted or modified. IMS-supplied descriptors are not exported to RDDS. The four IMS-supplied resource descriptors are:

- DFSDSDB1 (database descriptor)
- DFSDSPG1 (application program descriptor)
- DBFDSRT1 (Fast Path routing code descriptor)
- DFSDSTR1 (transaction descriptor)

You can use the IMS-supplied descriptors as models for creating resources or additional descriptors. You can also create descriptors without modeling them after existing descriptors. Initially, the IMS-supplied descriptors are set as the default descriptors, but you can designate one of your own descriptors as the default by using the DEFAULT(Y) keyword on the CREATE or UPDATE command.

When you create a resource without specifying a model (that is, you do not specify the LIKE keyword), any attribute values that are not specified on the CREATE command are inherited from the default descriptor. When you create a resource that is modeled from a descriptor (using the LIKE keyword), any attribute values that are not specified on the CREATE command are inherited from the descriptor.

Similarly, when you create a resource using an existing resource as a model (using the LIKE keyword), any attribute values that are not specified on the CREATE command are inherited from the existing resource.

Tip: To ease the transition from using the online change process to using DRD, you can create runtime resource definitions using the batch system definition process and then enable DRD so that you can use the DRD commands.

IMS systems can export resources that are defined by the batch system definition process and resource definitions that have been created or updated dynamically, to an RDDS, using the EXPORT DEFN TARGET(RDDS) command or by setting up your system for automatic export.

These resource definitions can then be imported from the RDDS into an IMS system during cold start processing. To ensure that your changes are recovered

across a cold start, set up your system to automatically export your resource and descriptor definitions at checkpoint time and to automatically import the definitions from the RDDS during cold start processing.

In situations when IMS receives a message for an unknown destination, you can also use the Destination Creation exit routine (DFSINSX0) to define a new transaction to IMS for that message and, if necessary, define the application program that is associated with the transaction.

Attention: DRD changes are not necessarily made across all IMS systems in an IMSplex. Changes might be successful on some IMS systems but fail on others. You must verify that changes have been made across all systems.

The figure that follows shows the ISPF panel for managing resources.

```
  Help
------------------------------------------------------------------------------
SPOC                          IMS Manage Resources

Command ===>

Select an action and press Enter.


* Action . . . . . .    1. Create new resources
                        2. Delete resources
                        3. Query resources
                        4. Update resources
                        5. Import resources
                        6. Export resources
                        7. Manage RDDS
```

Figure 6. IMS Manage Resources main panel

Commands that support dynamic resource definition

With dynamic resource definition (DRD), you can use type-1 and type-2 commands to define resource and descriptor definitions.

The following IMS type-2 commands support DRD.

CREATE DB
CREATE DBDESC
CREATE PGM
CREATE PGMDESC
CREATE RTC
CREATE RTCDESC
CREATE TRAN
CREATE TRANDESC
DELETE DB
DELETE DBDESC
DELETE PGM
DELETE PGMDESC
DELETE RTC
DELETE RTCDESC
DELETE TRAN
DELETE TRANDESC
EXPORT DEFN
IMPORT DEFN
QUERY DB
QUERY DBDESC

QUERY PGM
QUERY PGMDESC
QUERY RTC
QUERY RTCDESC
QUERY TRAN
QUERY TRANDESC
UPDATE DB
UPDATE DBDESC
UPDATE PGM
UPDATE PGMDESC
UPDATE RTC
UPDATE RTCDESC
UPDATE TRAN
UPDATE TRANDESC

The following IMS type-1 commands support DRD:

- /CHECKPOINT

  If automatic export is enabled, and one or more resource or descriptor definitions have been created, updated, or deleted since the last checkpoint, the /CHECKPOINT command causes changed resource and descriptor definitions with a status of EXPORTNEEDED to be written to the system RDDS that contains the oldest data.
- /CHECKPOINT DUMPQ|FREEZE|PURGE

  If automatic export is enabled, and one or more resource or descriptor definitions have been created, updated, or deleted since the last checkpoint, the /CHECKPOINT DUMPQ|FREEZE|PURGE command causes all resource and descriptor definitions to be written to the system RDDS that contains the oldest data.
- /MODIFY

  If DRD is enabled and MODBLKS is specified on the /MODIFY command, the command fails.

The X'22' log record represents an action that is generated by a type-2 command. It is written for most type-2 commands that are recoverable and for some type-2 commands, such as QUEUE and UPDATE, for diagnostic information. Refer to DFSLOG22 for mapping of the X'22' log record. For most commands, the X'22' log records indicate that the type-2 command is recoverable, and it is used to reprocess the command during a warm or emergency restart. The X'22' log record is also used to communicate the command changes to the XRF alternate system, the DBCTL warm standby system, or the FDBR system. For some commands, the X'22' log records are for diagnostics only and are not used to recover or reprocess the command.

For the transaction, program, routing code, and database resources and descriptors, the X'22' log record tracks the changes made to the runtime resource and descriptor definitions that result from the CREATE, UPDATE, DELETE, and IMPORT commands, and the MODBLKS online change procedure.

The EXPORT command does not generate a X'22' log record.

In addition to being able to dynamically create, update, query, and delete runtime resource and descriptor definitions, with DRD you have other benefits, including:

- Type-2 command benefits, including the ability to sort, scroll, and use wildcards in the command.

- Returned command completion code text is displayed with a brief description of the completion code.
- All create, update, and delete activities are logged in the log record X'22', to aid in recovery after a warm or emergency restart.
- Create, update, access time stamps are maintained.

If DRD is not enabled for MODBLKS resources, you can use the following type-2 commands:

- QUERY DB
- QUERY PGM
- QUERY RTC
- QUERY TRAN
- UPDATE DB START()
- UPDATE DB STOP()
- UPDATE PGM START()
- UPDATE PGM STOP()
- UPDATE RTC START()
- UPDATE RTC STOP()
- UPDATE TRAN SET(LOCK())
- UPDATE TRAN START()
- UPDATE TRAN STOP()

The EXPORT command can be issued in a non-DRD enabled IMS system. However, in a non-DRD enabled IMS system, the only RDDS that can be exported to is a non-system RDDS.

Related reference:

IMS commands (Commands)

Dynamic resource definition and system definition

Specify parameters related to dynamic resource definition (DRD) in the DFSDFxxx member of the IMS PROCLIB data set.

The DFSDFxxx member of the IMS PROCLIB data set contains parameters for the IMS Common Service Layer (CSL), shared queues, databases, restart exit routines, dynamic resource definition (DRD), the Fast Path 64-bit buffer manager, and the IMS abend search and notification procedure.

The DFSDFxxx member includes some parameters that also exist in the DFSSQxxx member (for defining parameters related to shared queues) and the DFSCGxxx member (for defining parameters related to the CSL). If you specify these common parameters in the DFSDFxxx member, you do not have to specify them in the DFSSQxxx and DFSCGxxx members. Any attributes explicitly defined in the DFSSQxxx and DFSCGxxx members override the attributes defined in the DFSDFxxx member.

The MODBLKS= keyword enables either DRD (MODBLKS=DYN) or the online change process (MODBLKS=OLC). The MODBLKS= keyword can only be changed as part of a cold start. MODBLKS=OLC and MODBLKS=DYN are mutually exclusive. You can specify the MODBLKS= keyword in either the DFSCGxxx member, or in the CSL section of the DFSDFxxx member. If you specify a value for

the MODBLKS= keyword in the DFSCGxxx member, that value overrides the value specified for the MODBLKS= keyword in the DFSDFxxx member.

If the online change process is disabled (DRD is enabled), the IMS, DBC, and DCC procedures no longer require the DD statements for the IMS.MODBLKS data sets, IMS.MODBLKSA and IMS.MODBLKSB.

Recovery of changed runtime resource and descriptor definitions

Changes that are made to resource and descriptor definitions by using type-2 commands are logged and recoverable across a warm or emergency restart.

The external data source in which resource and descriptor definitions are stored is a BSAM data set called a *resource definition data set* (RDDS). The definitions in an RDDS are in a binary format. RDDSs are not supported in IMS FDBR regions or on IMS RSR tracking systems.

The runtime resource and descriptor definitions are restored from the logs during restart processing.

For changes to be recoverable across a cold start, the changed resource and descriptor definitions must be exported to, or stored in, a repository or a resource definition data set (RDDS) before IMS terminates, and imported into IMS either:

For changes to be recoverable across a cold start, the changed resource and descriptor definitions must be exported to, or stored in, a resource definition data set (RDDS) before IMS terminates, and imported into IMS either:

- During cold start with the automatic import function.
- After IMS is up and running with the IMPORT command.

Alternatively, you can do the following tasks:

1. Update your IMS system definition macros.
2. Perform a system generation.
3. Cold start IMS.
4. Import the resource definitions from the IMS MODBLKS data set.

Requirements for dynamic resource definition

Dynamic resource definition (DRD) has the same hardware and software requirements as the base IMS product, certain configuration and system definition requirements exist.

To use dynamic resource definition (DRD), you must define a Common Service Layer (CSL) with at least Structured Call Interface (SCI) and Operations Manager (OM).

You must also ensure that your system has a type-2 command interface, such as the TSO SPOC application, the Manage Resources application, or another application that interacts with the OM API.

It is also suggested that you define one or more resource definition data sets (RDDSs) in which to save your current resource definitions.

DRD requires:

- A CSL that contains SCI and OM.
- One or more RDDSs.
- A type-2 command interface such as the TSO SPOC or other OM interface, for DRD commands.

The INITMOD procedure requires that you define either MODBLKSA or MODBLKSB to initialize the MODSTAT data set, even for an IMS that does not define the MODBLKS data set.

In a DRD for MODBLKS resources environment, the MODBLKS staging, active, and inactive data sets are no longer required. However, the Global Online Change utility (DFSUOLC0) requires that you define either MDBS=A or MDBS=B to initialize the OLCSTAT data set, even for an IMS that does not define the MODBLKS data set.

Restrictions for dynamic resource definition

If you plan to use dynamic resource definition (DRD) in your IMS system, some restrictions apply to your use of DRD.

The following restrictions apply to DRD:

- When DRD for MODBLKS resources is enabled, you cannot perform online change for the resources that are typically defined in the IMS.MODBLKS data set. If you have a simple system, such as a single IMS, and online change meets your requirements, consider continuing to use online change and not enabling DRD.
- In an IMSplex with only one IMS and DRD enabled, the /MODIFY PREPARE MODBLKS and INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS) commands are rejected. The commands /MODIFY PREPARE ALL and INITIATE OLC PHASE(PREPARE) ALL do not apply to the IMS.MODBLKS data set.
- When DRD for MODBLKS resources is disabled, CREATE, DELETE, IMPORT, and most UPDATE commands that change the definitional attributes of a resource are rejected.
- The following parameters for the UPDATE TRAN command are permitted whether or not DRD is enabled:
  - CLASS(class)
  - CPRI(value)
  - LCT(value)
  - LPRI(value)
  - MAXRGN(number)
  - MSNAME(name)
  - NPRI(value)
  - PARLIM(value)
  - PLCT(value)
  - SEGNO(number)
  - SEGSZ(size)
  - TRANSTAT(Y | N)
- You cannot delete an IMS-supplied descriptor. The only attribute you can update on an IMS-supplied descriptor is the DEFAULT attribute.
- Because of the default Operations Manager (OM) routing in a sysplex, DRD commands are routed to all the IMS systems, unless you specify otherwise. However, the commands are not coordinated across the sysplex, so a command can succeed on some IMS systems and fail on others.

- The Multiple Systems Verification utility (DFSUMSV0) can verify resources defined only by the batch system definition process; it cannot verify resources that were created using DRD. Use the /MSVERIFY command to verify resources that are created dynamically.
- You can update a resource or a descriptor definition, but the changes to that resource or descriptor definition are not propagated to the resource or descriptor definitions that were built from the updated resource or descriptor definition.
- You can import a resource or descriptor from a resource definition data set (RDDS) and it affects any resource or descriptor explicitly specified in the IMPORT DEFN command, but it does not affect any resources or descriptors built from the specified resource or descriptor.
- DRD commands can be issued only through the OM API.
- DRD commands that work on database resources and descriptors cannot be issued in a DCCTL environment.
- DRD commands that work on transactions and routing code resources and descriptors cannot be issued in a DBCTL environment.
- None of the commands that support DRD can be issued on XRF alternate or RSR tracker systems, or FDBR regions.
- DELETE DB and UPDATE DB commands are rejected for MSDBs.

Related reference:

IMS commands (Commands)

Chapter 17, “Members of the IMS PROCLIB data set,” on page 587

Considerations for using dynamic resource definition

To ensure that your use of dynamic resource definition (DRD) is successful, be aware of these usage considerations.

In addition to the requirements and restrictions that are associated with using DRD, consider the following issues:

- Before enabling dynamic resource definition or shared queues, evaluate any existing DFSINSX0 exit routines. The DFSINSX0 exit might need to be changed so that it checks whether LTERM creation is allowed before it accesses the USEQDATA parameter list that is related to LTERM processing. If LTERM creation is not allowed, the USEQDATA buffer address (INSXAUSQ) is zero.
- In an IMSplex with multiple IMS systems, some having DRD enabled, the command INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS) is allowed so that the IMS systems that do not have DRD enabled can perform online change processes.
- The online change process is enabled and DRD is not enabled when:
  - MODBLKS=OLC is specified in the DFSCGxxx member of the IMS PROCLIB data set
  - MODBLKS=OLC is specified in the COMMON_SERVICE_LAYER section of the DFSDFxxx member of the IMS PROCLIB data set
  - A value is not specified for the MODBLKS= keyword in either member.
- In a shared EMH queues environment, a CREATE PGM command creates the application program, even if there are messages on the shared EMH queues for the application program. If the messages are placed on the shared EMH queues because the program was defined as Fast Path on another IMS, but this program is being created as non-Fast-Path on this IMS, this IMS is unable to access the messages on the shared EMH queues.

- All the DRD commands that work on application program resources and descriptors can be issued in DB/DC, DBCTL, and DCCTL environments.
- The MODBLKS DD statement is not required for MODBLKS=OLC or MODBLKS=DRD in an FDBR system.
- If you are not running with DRD enabled, any changes you make to the MODBLKS resources (database, transactions, programs, and routing codes) by using the type-1 commands persist across a warm or emergency restart, but they do not persist across a cold start. In a non-DRD environment, the control blocks that are used to manage the resources (DDIRs, PDIRs, SMBs, and RCTEs) are loaded from the MODBLKS data set at cold start. If a type-1 command is issued to change the attribute of a resource (such as the database access type or the transaction class), the internal control blocks are updated and the changes are recovered across a warm or emergency restart. If you perform a cold start, however, the control blocks are reloaded from the MODBLKS data set, and, unless you have updated your MODBLKS data set, the updated attributes revert to the original values.

  If you are running with DRD enabled, any changes you make by using the type-1 or type-2 commands persist across a warm or emergency restart. They also persist across a cold start if the updated resource definitions are exported to an RDDS and then imported from the RDDS during cold start. When you export the resource definitions to an RDDS, you export all the current attribute values. If you have changed the value of one of the attributes by using a type-1 or type-2 command (such as the database access type or the transaction class), the updated attribute is exported. The updated attribute values are then imported during cold start if you have automatic import enabled.
- During a cold start, the partition status or access state is copied from the HALDB master. If you export the HALDB master status to the RDDS, the status of the master and its partitions is obtained from the RDDS.

Enabling dynamic definition for IMS resource groups

Before you can dynamically add, change, or delete a particular type of IMS resource, you must first enable dynamic resource definition for the resource group that includes the resource type.

You can enable dynamic resource definition for only MODBLKS resources, which include definitions of certain runtime attributes of application programs, databases, Fast Path routing codes, and transactions.

Requirement: The dynamic definition of IMS resources requires the Common Service Layer (CSL) with at least the Structured Call Interface (SCI) and the Operations Manager (OM).

The CSL can be enabled by parameters in either of the following members of the IMS PROCLIB data set:

- The DFSDFxxx member, in the COMMON_SERVICE_LAYER section.
- The DFSCGxxx member. The CSL values in the DFSCGxxx member override the values in the COMMON_SERVICE_LAYER section of the DFSDFxxx member.

Related concepts:

Common Service Layer overview (System Administration)

Related reference:

COMMON_SERVICE_LAYER section of the DFSDFxxx member (System Definition)

"COMMON_SERVICE_LAYER section of the DFSDFxxx member" on page 673

"DFSCGxxx member of the IMS PROCLIB data set" on page 649

Enabling dynamic resource definition for MODBLKS resources

You can enable dynamic resource definition for MODBLKS resources with a resource definition data set (RDDS).

To ensure that the procedure for enabling dynamic resource definition for MODBLKS goes smoothly, you should validate that your IMS environment is ready for the dynamic definition of MODBLKS resources.

Validating the IMS environment before enabling dynamic definition for MODBLKS resources

Before you enable dynamic resource definition for MODBLKS resources with a resource definition data set (RDDS), validate the IMS environment by performing the following steps.

1. Verify DRD is not yet enabled, meaning either:
   - The MODBLKS keyword is not specified anywhere.
   - The MODBLKS=OLC keyword is specified in either the DFSCGxxx member or the COMMON_SERVICE_LAYER section of the DFSDFxxx member of the IMS PROCLIB data set.
2. Verify that IMS is stabilized in your test environment and the IMS.MODBLKS data set is complete and working properly.
3. Verify that the IMS in which you are enabling DRD has a Common Service Layer (CSL) defined. Specify the CSL parameters in either:
   - The COMMON_SERVICE_LAYER section of the DFSDFxxx member in the IMS PROCLIB data set
   - The DFSCGxxx member in the IMS PROCLIB data set

   If you specify values in both the COMMON_SERVICE_LAYER section of the DFSDFxxx member and in the DFSCGxxx member, the CSL values in the DFSCGxxx member override the values in the COMMON_SERVICE_LAYER section of the DFSDFxxx member.
4. Verify that your in-house procedures that use the online change process for the resources defined in the IMS.MODBLKS data set have been copied and stored, and you have replacement procedures that use DRD commands.

Enabling dynamic definition for MODBLKS resources with a resource definition data set

After performing the initial verification of your IMS environment, you can enable dynamic resource definition (DRD) for MODBLKS resources with a resource definition data set (RDDS).

To enable dynamic resource definition (DRD) for MODBLKS resources with a resource definition data set (RDDS):

1. Shut down IMS normally.
2. Change MODBLKS=OLC to MODBLKS=DYN in either the COMMON_SERVICE_LAYER section of the DFSDFxxx member or in the DFSCGxxx member in the IMS PROCLIB data set.

3. Specify the following keywords in the DYNAMIC_RESOURCES section of the DFSDFxxx member of the IMS PROCLIB data set:
   - AUTOEXPORT=AUTO or AUTOEXPORT=RDDS
   - AUTOIMPORT=AUTO or AUTOIMPORT=RDDS
   - RDDSDSN=(*rdds1*,*rdds2*,*rdds3*)
4. Use either the IEBGENER program or the ALLOCATE function of ISPF utilities to allocate the RDDS data sets.
5. Cold start IMS, specifying DFSDF=*xxx* in the control region execution parameters, where *xxx* identifies the suffix of the DFSDF*xxx* member in the IMS PROCLIB data set that IMS uses for this startup. IMS loads the resource definitions from the IMS.MODBLKS data set because the defined RDDSs are empty. The IMS.MODBLKS data set is used only the first time you implement DRD, except for fallback purposes. IMS uses the resource definitions to create the internal control blocks required to manage the resources. When the initial checkpoint is taken, the resource definitions are automatically written out to one of the defined RDDSs because AUTOEXPORT=AUTO is specified. Also, because AUTOIMPORT=AUTO is specified and the RDDSs are no longer empty, IMS loads its resource and descriptor definitions from the RDDS with the most current data the next time that IMS coldstarts.

   Restriction: After IMS starts with DRD enabled, you can no longer use the online change process to add, change, or delete database, application program, route code, or transaction resource definitions.

Creating runtime resource and descriptor definitions

When dynamic resource definition (DRD) is enabled, you can define runtime resources and descriptor definitions to IMS dynamically, eliminating the requirement to use the online change process or to cold start IMS.

When a resource or descriptor is defined to IMS, IMS stores information about the resource or descriptor, such as the attributes and the status, in control blocks. The information that is kept in the control blocks is referred to as the runtime resource or descriptor definition. Although the overall process for creating runtime database, application program, transaction, and routing code definitions is the same, there are minor variations in the steps you complete.

When DRD is enabled for MODBLKS resources, runtime resources can be created by:

- Importing the stored resource definitions from the IMS.MODBLKS data set during an IMS cold start.
- Importing the stored resource definitions from the resource definition data set (RDDS) with the automatic import function or the IMPORT DEFN SOURCE(RDDS) command.
- Importing the stored resource definitions from the IMSRSC repository with the automatic import function or the IMPORT DEFN SOURCE(REPO) command.
- Using the appropriate CREATE command (according to the type of resource definition you want to create).

When DRD is enabled for MODBLKS resources, runtime descriptor definitions can be created by:

- Importing the stored descriptor definitions from the RDDS with the automatic import function or the IMPORT DEFN SOURCE(RDDS) command.

- Using the appropriate CREATE command (according to the type of descriptor definition you want to create).

The resource and descriptor definitions that you create dynamically using either the CREATE command or the IMPORT DEFN command exist only during the current execution of the IMS system, and they are recoverable across a warm or emergency restart.

To preserve the resource and descriptor definitions across a cold start, export the definitions to an RDDS before IMS terminates, and then import the stored definitions from the RDDS back into IMS either during cold start processing using the automatic import function, or when IMS is up and running using the IMPORT DEFN command.

Each resource or descriptor definition can be created individually, unlike the online change process where either all resource definitions must be created or no resource definitions are created.

Creating runtime database resource and descriptor definitions with the CREATE command

With dynamic resource definition (DRD) enabled, you can create runtime resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Create runtime database resource and descriptor definitions using the CREATE DB and CREATE DBDESC commands.

1. Optional: Issue a QUERY DBDESC DEFAULT(Y) SHOW (ALL) command to view the current database default settings and evaluate whether they meet your requirements for the new database resource or descriptor definition. If the default settings do not meet your requirements, you have the following options:
   - Override the default settings by issuing the CREATE command with parameter values that meet your requirements.
   - Create the resource or descriptor definition based on an existing resource or descriptor definition by using either the LIKE (RSC(*resource_name*)) keyword or the LIKE (DESC(*descriptor_name*)) keyword on the CREATE command.
   - Create a default descriptor by issuing a CREATE DBDESC NAME() SET (DEFAULT(Y)) command with the settings that meet your requirements.

   The next two steps can be performed in any order.
2. Issue a CREATE DB or CREATE DBDESC command to create a database resource definition or database descriptor definition, respectively.
3. Define the database, as well as its relationships to other resources, by running the Database Description Generation (DBDGEN) utility.

After dynamically defining a database resource, before it can be used, a database management block (DMB) must reside in the IMS.ACBLIB data set. If the database that was dynamically defined is a main storage database (MSDB):

- The database segments must be in the MSDBINIT data set.
- The MSDBs must be defined in the DBFMSDBx procedure.
- The DBFMSDBx procedure must be in the IMS.PROCLIB data set.

For information about the CREATE command, see *IMS Version 11 Commands, Volume 1: IMS Commands A-M*.

For information about loading MSDB segments into the MSDBINIT data set, see MSDB Maintenance utility (DBFDBMA0) (Database Utilities).

For information about the DBFMSDBx procedure, see "DBFMSDBx member of the IMS PROCLIB data set" on page 646.

For general information about MSDBs, see *IMS Version 11 Database Administration*.

Creating runtime application program resource and descriptor definitions with the CREATE command

With dynamic resource definition (DRD) enabled, you can create runtime application program resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Create runtime application program resource and descriptor definitions using the CREATE PGM and CREATE PGMDESC commands.

1. Optional: Issue a QUERY PGMDESC DEFAULT(Y) SHOW (ALL) command to view the current application program default settings and evaluate whether they meet your requirements for the new application program resource or descriptor definition. If the default settings do not meet your requirements, you have the following options:
   - Override the default settings by issuing the CREATE command with parameter values that meet your requirements.
   - Create the resource or descriptor definition based on an existing resource or descriptor definition by using either the LIKE (RSC(*resource_name*)) keyword or the LIKE (DESC(*descriptor_name*)) keyword.
   - Create a default descriptor by issuing a CREATE PGMDESC NAME() SET (DEFAULT(Y)) command with the settings that meet your requirements.

   The next two steps can be performed in any order.
2. Issue a CREATE PGM or CREATE PGMDESC command to create application program resources or descriptors definitions, respectively.
3. Define the application program, as well as its relationships to other resources, by running the Program Specification Block Generation (PSBGEN) utility.

For the details about parameters for the CREATE command, see *IMS Version 11 Commands, Volume 1: IMS Commands A-M*.

Creating runtime Fast Path routing code resource and descriptor definitions with the CREATE command

With dynamic resource definition (DRD) enabled, you can create runtime resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Create runtime Fast Path routing code resources and descriptor definitions using the CREATE RTC and CREATE RTCDESC commands.

1. Optional: Issue a QUERY RTCDESC DEFAULT(Y) SHOW (ALL) command to view the current Fast Path routing code default settings and evaluate whether they meet your requirements for the new Fast Path routing code resource or descriptor definition. If the default settings do not meet your requirements, you have the following options:
   - Override the default settings by issuing the CREATE command with parameter values that meet your requirements.
   - Create the resource or descriptor definition based on an existing resource or descriptor definition by using either the LIKE (RSC(*resource_name*)) keyword or the LIKE (DESC(*descriptor_name*)) keyword
   - Create a default descriptor by issuing a CREATE RTCDESC NAME() SET (DEFAULT(Y)) command with the settings that meet your requirements.
2. Issue a CREATE RTC to create Fast Path routing code resources or a CREATE RTCDESC command to create Fast Path routing code descriptor definitions.

For the details about parameters for the CREATE command, see *IMS Version 11 Commands, Volume 1: IMS Commands A-M*.

Creating runtime transaction resource and descriptor definitions with the CREATE command

With dynamic resource definition (DRD) enabled, you can create runtime resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Create runtime transaction resource and descriptor definitions using the CREATE TRAN and CREATE TRANDESC commands.

In addition to using the CREATE TRAN command to create transaction resources, you can use the Destination Creation exit routine (DFSINSX0).

To create runtime transaction resource and descriptor definitions using the CREATE TRAN and CREATE TRANDESC commands, perform the following procedure:

1. Optional: Issue a QUERY TRANDESC DEFAULT(Y) SHOW (ALL) command to view the current transaction default settings and evaluate whether they meet your requirements for the new transaction resource or descriptor definition. If the default settings do not meet your requirements, you have the following options:
   - Override the default settings by issuing the CREATE command with parameter values that meet your requirements.
   - Create the resource or descriptor definitions based on an existing resource or descriptor definition by using either the LIKE (RSC(*resource_name*)) keyword or the LIKE (DESC(*descriptor_name*)) keyword.
   - Create a default descriptor by issuing a CREATE TRANDESC NAME() SET (DEFAULT(Y)) command with the settings that meet your requirements.
2. Issue a CREATE TRAN or CREATE TRANDESC command to create transaction resources or descriptor definitions, respectively.

For the details about parameters for the CREATE command, see *IMS Version 11 Commands, Volume 1: IMS Commands A-M*.

Creating runtime resource and descriptor definitions in an IMSplex with the CREATE command

If you have a mixture of DRD-enabled and DRD-disabled IMS systems in an IMSplex, you can create resource or descriptor definitions for this environment.

Perform the following procedure:

1. Perform an online change process for the IMS.MODBLKS data set on the IMS systems that have not migrated to DRD.
2. Issue CREATE commands for the resource or descriptor definitions you want to create on the IMS systems that have DRD enabled.

Updating runtime resource and descriptor definitions

When dynamic resource definition (DRD) is enabled, you can update runtime resource and descriptor definitions dynamically, eliminating the need to use the online change process or to cold start IMS. Although the overall process for updating runtime database, application program, transaction, and routing code definitions is the same, there are minor variations in the steps you complete.

When DRD is enabled for MODBLKS resources, runtime resources can be updated by:

- Importing the stored resource definitions from the resource definition data set (RDDS) by using the IMPORT DEFN SOURCE(RDDS) command.
- Using the appropriate UPDATE command (according to the type of resource definition you want to update).

When DRD is enabled for MODBLKS resources, runtime descriptor definitions can be updated by:

- Importing the stored descriptor definitions from the RDDS by using the IMPORT DEFN SOURCE(RDDS) command.
- Using the appropriate UPDATE command (according to the type of descriptor definition you want to update).

The changes that you make to resource and descriptor definitions by using the UPDATE and IMPORT commands exist during the current execution of the IMS system and are recoverable across a warm or emergency restart. To preserve the changes across a cold start, export the updated definitions to an RDDS before IMS terminates, and then import the stored definitions from the RDDS back into IMS either during cold start processing (by using the automatic import function), or when IMS is up and running (by using the IMPORT DEFN command).

If the attributes specified on the UPDATE command are already defined for the resource:

- No update is made
- No resources are quiesced
- No log record is created
- A completion code of zero is returned

By taking no action, IMS avoids unnecessary processor usage.

Unlike the online change process where either all resource definitions are updated or no resource definitions are updated, each resource or descriptor definition can be updated individually.

Updating runtime database resource and descriptor definitions with the UPDATE command

With dynamic resource definition (DRD) enabled, you can update runtime resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

You cannot update database resource definitions if there is work in progress for the database, or if the database is currently in use.

Update runtime database resource and descriptor definitions using the UPDATE DB and UPDATE DBDESC commands.

1. Check for work in progress using a QRY DB SHOW(WORK) command. If there is work in progress, either:
   a. Wait for the work to finish.
   b. Address the work in progress. Examples of work in progress include a command is in progress for the database, or the database is in use.
2. Stop the database using either an UPDATE DB STOP(ACCESS) or /DBR DB command
3. Issue an UPDATE DB or UPDATE DBDESC command specifying the values you want to change for the database or descriptor definitions, respectively.

Updating runtime application program resource and descriptor definitions with the UPDATE command

With dynamic resource definition (DRD) enabled, you can update runtime application program resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

You cannot update application program resource definitions if there is work in progress for the application program. In a local queues environment, if there are queued messages for transactions associated with the program being updated, the update fails. In a shared queues environment, if there are queued messages for a transaction that is associated with the program being updated, the update succeeds. However, the update might succeed on some IMS systems and fail on others.

Update runtime application program resource and descriptor definitions using the UPDATE PGM and UPDATE PGMDESC commands.

1. Check for work in progress using a QRY PGM SHOW(WORK) command. If there is work in progress, either:
   a. Wait for the work to finish.
   b. Address the work in progress. Examples of work in progress include a command being in progress for the program or the program being scheduled.
2. Check whether any transactions reference the program using a QRY PGM SHOW(TRAN) command. If any of the program attributes you plan to update (such as FP or BMPTYPE) conflict with any transactions that reference the program, before updating the program definition, you must either:
   a. Delete the conflicting transaction definitions that reference the program.

b. Update the transaction definitions to reference a different program before updating the program.

To delete a transaction, issue a DELETE TRAN command. To update a transaction, issue an UPDATE TRAN command.

3. If the FP attribute is being changed from FP(E) to FP(N), determine whether any routing codes reference the program by using a QRY PGM SHOW(RTC) command. If there are routing codes that reference the program, before updating the program definition, you must either:
   - Delete all the routing codes that reference the program.
   - Update the routing code definitions to reference a different program before updating the program.

   To delete a routing code associated with an FP(E) transaction, issue a DELETE TRAN command. To delete a routing code associated with an FP(P) transaction, issue a DELETE RTC command.
4. The next step varies depending on where the application program is running.
   - If the program is running in an IFP region, issue a /STOP REGION command to terminate the program.
   - If the program is running in a BMP or JBP region, either wait for the program to finish, or issue a /STOP REGION command to terminate the program.
   - If the program is a BMP that is processing a WFI transaction, issue a /STOP REGION command to terminate the program.
   - If the program is running in an MPP or JMP region, issue a /STOP PGM or UPDATE PGM STOP(SCHD) command to terminate the program.
5. Issue the UPDATE PGM or UPDATE PGMDESC command specifying the values you want to change for the application program or descriptor definitions, respectively.

If all the attributes specified by the UPDATE command are already defined for the resource, no update is made, no resources are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

For the details about parameters for the UPDATE and QUERY commands, see *IMS Version 11 Commands, Volume 2: IMS Commands N-V*.

Updating runtime Fast Path routing code resource and descriptor definitions with the UPDATE command

With dynamic resource definition (DRD) enabled, you can update runtime Fast Path routing code resource and descriptor definitions dynamically, eliminating the requirement to use the online change process or the batch system definition process with an IMS cold start.

If the program is scheduled, the routing code cannot be updated. If the program is scheduled, you must stop the region before you issue the UPDATE RTC command.

Update runtime Fast Path routing code resources and descriptor definitions using the UPDATE RTC and UPDATE RTCDESC commands.

1. Check for work in progress by using a QRY RTC SHOW(WORK) command. If there is work in progress, either:
   - Wait for the work to finish.

- Address the work in progress.

  Examples of work in progress include a command in progress for the routing code or an active routing code.

2. If the application program is running in an IFP region, terminate the program by using a /STOP REGION command.
3. Issue an UPDATE RTC or UPDATE RTCDESC command, specifying values you want to change for the Fast Path routing code resources or descriptor definitions. To update a routing code associated with a Fast Path exclusive FP(E) transaction, you must issue an UPDATE TRAN command. The UPDATE TRAN command updates the transaction definition and the associated routing code definition.

For the details about parameters for the UPDATE and QUERY commands, see *IMS Version 11 Commands, Volume 2: IMS Commands N-V*.

If you are using a set of system resource definition data sets (RDDSs) to store your Fast Path routing code resource and descriptor definitions, the routing codes associated with both Fast Path potential FP(P) and FP(E) transactions are maintained in the RDDS.

Updating runtime transaction resource and descriptor definitions with the UPDATE command

With dynamic resource definition (DRD) enabled, you can update runtime resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

You cannot update a transaction if it is in use. If a transaction is in use, the update fails. In a local queues environment, if there are queued messages for the transaction, the update fails. In a shared queues environment, if there are queued messages for the transaction, the update succeeds.

Update runtime transaction resource and descriptor definitions using the UPDATE TRAN and UPDATE TRANDESC commands.

1. Check for work in progress by using a QRY TRAN SHOW(WORK) command and either wait for the work to finish or address the work in progress. Examples of work in progress are a command in progress for the transaction, a scheduled transaction, a transaction in conversation, or a suspended transaction.
   - To wait for work in progress to complete, issue either a /PURGE TRAN or UPDATE TRAN STOP(Q) command to stop the transaction from being queued, but to allow existing queued messages to be processed.
   - If you do not want to wait for work in progress, or if the transaction is a WFI transaction, or if the transaction is running in a PWFI=Y MPP region, stop the transaction by issuing a /STOP TRAN or UPDATE TRAN STOP(SCHD,Q) command. If you are stopping the transaction in a non-shared queues environment, no messages can be queued for the transaction. To dequeue messages for the transaction, issue the QUEUE TRAN OPTION(DEQALL) command.
2. The next step varies depending on where the transaction is running.
   - If the transaction is running in an IFP region, terminate the program by issuing a /STOP REGION.

- If the transaction is a non-WFI transaction running in a message-driven BMP, either wait for the program to finish, or terminate the program by issuing a /STOP REGION command.
- If the transaction is a WFI transaction running in a message-driven BMP, terminate the program by issuing a /STOP REGION command.

3. If the transaction has messages on the suspend queue, and you are running in a non-shared queues environment, you must either:
   - Process the messages on the suspend queue.
   - Delete the messages. To delete the messages on the suspend queue, stop the transaction by issuing a /STOP TRAN or UPDATE TRAN STOP(SCHD) command.
     a. Move the messages to the ready queue by issuing either a /DEQ SUSPEND or UPDATE TRAN START(SUSPEND) command.
     b. After the messages are on the ready queue, you can delete them by issuing a QUEUE TRAN OPTION(DEQALL) command.
4. Issue the UPDATE TRAN or UPDATE TRANDESC command for the resource or descriptor definition you want to update, respectively.

For the details about parameters for the UPDATE and QUERY commands, see *IMS Version 11 Commands, Volume 2: IMS Commands N-V*.

Updating runtime resource and descriptor definitions in an IMSplex with the UPDATE command

If you have a mixture of DRD-enabled and DRD-disabled IMS systems in an IMSplex, you can update resources or descriptor definitions for this environment.

Perform the following procedure:

1. Perform an online change process for the IMS.MODBLKS data set on the IMS systems that have not migrated to DRD.
2. Issue UPDATE commands for the resource or descriptor definitions you want to update on the IMS systems that have DRD enabled.

Deleting runtime resource and descriptor definitions

When dynamic resource definition (DRD) is enabled, you can use the DELETE command to delete runtime resource and descriptor definitions dynamically, eliminating the requirement to use the online change process or to cold start IMS. Although the overall process for deleting runtime database, application program, transaction, and routing code definitions is the same, there are minor variations in the steps you complete.

The resource and descriptor definitions that you delete dynamically using the DELETE command remain deleted across a warm or emergency restart.

Each resource or descriptor definition is deleted individually, unlike the online change process, where either all resources or descriptor definitions are deleted or no resources or descriptor definitions are deleted.

Deleting runtime resource and descriptor definitions when using an RDDS

To delete runtime resource and descriptor definitions from IMS and from an RDDS, perform the following procedure:

1. Issue the DELETE command locally at each IMS that contains the runtime definitions.

   If definitions of more than one type are being deleted, issue multiple different DELETE commands (according to the type of resource or descriptor definitions you want to delete).

   If you have set up automatic export to an RDDS, automatic export occurs at system checkpoint and removes the deleted definitions from the system RDDS.
2. If you have not set up automatic export to an RDDS, issue the EXPORT command at each IMS from which the definitions were deleted to ensure the removal of deleted definitions from the system RDDS (or non-system RDDS).

You can then import the resource and descriptor definitions from the RDDS back into IMS in one of the following ways:

- During cold start processing by using the automatic import function.
- After IMS is up and running by using the IMPORT DEFN command.

Restrictions

You cannot delete a runtime resource definition if the resource is in use. Examples of resources in use include:

- A database that is being accessed by an application program
- A database that is referenced by the program specification block (PSB) for an existing program
- A /STOP or UPDATE command that is in progress for the resource
- An application program that is scheduled to run
- An active routing code
- A transaction with messages queued for it
- A transaction that has any work in progress

If you attempt to delete a resource that is in use, the delete attempt fails. In a sysplex environment, the delete attempt might succeed on some IMS systems and fail on others.

You cannot delete the following resources that are supplied by IMS: DBFDSRT1, DFSDSDB1, DFSDSPG1, DFSDSTR1, and DBF#FPU0.

You cannot delete a HALDB master database resource unless it has no partitions defined and you have issued a /DBRECOVERY command for the database.

Related reference:

IMPORT command (Commands)

DELETE commands (Commands)

Deleting runtime database resource and descriptor definitions with the DELETE command

With dynamic resource definition (DRD) enabled, you can delete runtime resource and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Delete runtime database resource and descriptor definitions using the DELETE DB and DELETE DBDESC commands.

1. Check for work in progress with a QRY DB SHOW(WORK) command. If there is work in progress, either:
   - Wait for the work to finish.
   - Address the work in progress.
2. Determine whether any application programs reference the database by issuing a QRY DB SHOW(PGM) command. The DELETE DB command fails if any programs reference the database. If a program references the database, before deleting the database, you must either:
   - Delete the programs that reference the database. If you decide to delete the programs that reference the database, and if there are transactions or routing codes that reference the programs, they must be deleted first.
   - Perform an online change to delete the PSB from ACBLIB.
   - Update the PSB so that it does not reference the database.
3. The next step varies depending on whether the resource to be deleted is a database with secondary or logical relationships, an MSDB, or a HALDB master database.
   - If the resource to be deleted is a database with secondary indexes or logical relationships, remove the relationships by performing these steps:
     a. Update the database description (DBD) to remove the relationships.
     b. Run the Database Description Generation (DBDGEN) utility.
     c. Run the Application Control Block (ACB) Maintenance utility.
     d. Bring the updated DBDs and ACBs online using an online change process.
   - If the resource to be deleted is a main storage database (MSDB), perform these steps:
     a. Use the MSDB Maintenance utility (DBFDBMA0) to remove segments from the MSDBINIT data set.
     b. Delete the MSDB from any PSB that references it using the DELETE DBD= action control statement of DBFDBMA0.
     c. Perform a PSBGEN for the PSB that was modified.
     d. Perform an ACBGEN for the PSB that was modified.
     e. Perform an ACBGEN for the DBD of the MSDB that was deleted from the PSB.
     f. Either change or remove the appropriate DBFMSDBx procedure that is in the IMS.PROCLIB data set.
     g. Shut down IMS normally using the /CHE FREEZE command.
     h. Copy the above ACBLIB to both the active and inactive ACBLIB.
     i. Normally restart IMS using the /NRE command specified with the MSDBLOAD keyword to refresh the MSDBs.
   - If the resource to be deleted is a HALDB master database, perform these steps:
     a. Issue a /DBRECOVERY command.

        The /DBRECOVERY command removes this IMS system's knowledge of the HALDB partitions and stops the HALDB master database so that it can be deleted.
4. Stop the database using either a /DBR DB or an UPDATE DB STOP(ACCESS) command. A database must be successfully stopped by /DBR command before

it can be deleted. Because the MSDB is no longer in the ACBLIB, IMS does not know that the database was previously an MSDB and allows the /DBR command.

5. Issue the DELETE DB or DELETE DBDESC command to delete the database resource or descriptor definition, respectively.

Deleting runtime application program resource and descriptor definitions with the DELETE command

With dynamic resource definition (DRD) enabled, you can delete runtime application program resources and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Delete runtime application program resource and descriptor definitions using the DELETE PGM and DELETE PGMDESC commands.

1. Check for work in progress with a QRY PGM SHOW(WORK) command. If there is work in progress, either:
   a. Wait for the work to finish.
   b. Address the work in progress.
2. Determine whether any transactions reference the program by using a QRY PGM SHOW(TRAN) command. The DELETE PGM command fails if any transactions reference the program. If a transaction references the program, before deleting the program you must either:
   - Delete the transactions that reference the program by issuing a DELETE TRAN command.
   - Update the transactions to reference a different program by using the UPDATE PGM command.
3. Determine whether any routing codes reference the program by using a QRY PGM SHOW(RTC) command. The DELETE PGM command fails if any routing codes reference the program. If a routing code references the program, before deleting the program you must either:
   - Delete the routing codes that reference the program.

     To delete a routing code associated with a Fast Path exclusive FP(E) transaction, issue a DELETE TRAN command.

     To delete a routing code associated with a Fast Path potential FP(P) transaction, issue a DELETE RTC command.
   - Update the routing code to reference a different program using the UPDATE RTC command.

     To update a routing code associated with an FP(E) transaction, issue an UPDATE TRAN command.

     To update a routing code associated with an FP(P) transaction, issue an UPDATE RTC command.
4. The next step varies depending on where the application program is running.
   - If the program is running in an IFP region or is a BMP processing a WFI transaction, issue a /STOP REGION command to terminate the program.
   - If the program is running in a BMP or JBP region, either:
     a. Wait for the program to finish.
     b. Issue a /STOP REGION command to terminate the program.

- If the program is running in an MPP or JMP region, issue a /STOP PGM or UPDATE PGM STOP(SCHD) command to terminate the program.

5. Issue the DELETE PGM or DELETE PGMDESC command to delete the application program resource or descriptor definition, respectively.

Deleting runtime Fast Path routing code resource and descriptor definitions with the DELETE command

With dynamic resource definition (DRD) enabled, you can delete runtime Fast Path routing code resource and descriptor definitions dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Delete runtime Fast Path routing code resources and descriptor definitions using the DELETE RTC and DELETE RTCDESC commands.

1. Check for work in progress by issuing a QRY RTC SHOW(WORK) command. If there is work in progress, either:
   - Wait for the work to finish.
   - Address the work in progress.
2. The next step can vary based on whether the transaction is associated with a Fast Path potential FP(P) or Fast Path exclusive FP(E) transaction.
   - Delete a routing code that is associated with a Fast Path potential transaction using the DELETE RTC command. If the program is scheduled, the routing code cannot be deleted and you must stop the region before you issue the DELETE RTC command.
   - Delete a routing code that is associated with a Fast Path exclusive transaction using the DELETE TRAN command. The DELETE RTC command fails.
   - Delete a runtime routing code descriptor definition by issuing the DELETE RTCDESC command.

Deleting runtime transaction resource and descriptor definitions with the DELETE command

With dynamic resource definition (DRD) enabled, you can delete runtime resource and resource descriptors dynamically, eliminating the need to use the online change process or the batch system definition process with an IMS cold start.

Delete runtime transaction resource and descriptor definitions using the DELETE TRAN and DELETE TRANDESC commands.

1. Check for work in progress with a QRY TRAN SHOW(WORK) command and either:
   - Wait for the work to finish.
   - Address the work in progress.

   Examples of work include a command in progress for the transaction, or the transaction is either scheduled, in conversation, or suspended.
2. To wait for the work in progress to complete, issue either a /PURGE TRAN or UPDATE TRAN STOP(Q) command to stop the transaction from being queued, but to allow existing queued messages to be processed.
3. If you do not want to wait for the work in progress to complete, or if the transaction is a WFI transaction, stop the transaction by using a /STOP TRAN or UPDATE TRAN STOP(SCHD,Q) command. If you are stopping the

transaction in a non-shared queues environment, no messages can be queued for the transaction. To dequeue messages for the transaction, issue the QUEUE TRAN OPTION(DEQALL) command.

4. The next step can vary depending on where the transaction is running.
   - If the transaction is running in a PWFI=Y MPP region, stop the transaction by using a /STOP TRAN or UPDATE TRAN STOP(SCHD,Q) command. If you are stopping the transaction in a non-shared queues environment, no messages can be queued for the transaction. To dequeue messages for the transaction, issue the QUEUE TRAN OPTION(DEQALL) command.
   - If the transaction is running in an IFP region, terminate the program by issuing a /STOP REGION command.
   - If the transaction is a non-WFI transaction running in a message-driven BMP, either wait for the program to finish, or terminate the program by issuing a /STOP REGION command.
   - If the transaction is a WFI transaction running in a message-driven BMP, terminate the program by issuing a /STOP REGION command.
5. If the transaction has messages on the suspend queue and you are running in a non-shared queues environment, you must either process the messages on the suspend queue or delete the messages. To delete the messages on the suspend queue:
   a. Stop the transaction by issuing a /STOP TRAN or UPDATE TRAN STOP(SCHD) command.
   b. Move the messages to the ready queue by issuing a /DEQ SUSPEND or UPDATE TRAN START(SUSPEND) command.
   c. After the messages are on the ready queue, delete them by issuing a QUEUE TRAN OPTION(DEQALL) command.
6. Issue the DELETE TRAN or DELETE TRANDESC command to delete the transaction resource or descriptor definition, respectively.

Deleting runtime resource and descriptor definitions in an IMSplex with the DELETE command

If you have a mixture of DRD-enabled and DRD-disabled IMS systems in an IMSplex, you can delete resource or descriptor definitions for this environment.

Perform the following procedure:

1. Perform an online change process for the IMS.MODBLKS data set on the IMS systems that have not migrated to DRD.
2. Issue DELETE commands for the resource or descriptor definitions you want to delete on the IMS systems that have DRD enabled.

Exporting resource and descriptor definitions

When resource and descriptor definitions are ready to be created in a resource definition data set (RDDS) as the stored resource and descriptor definitions, you can export them to the RDDS.

Use the automatic export function or the EXPORT DEFN TARGET(RDDS) command to save your current resource and descriptor definition to the RDDS.

Exporting resource and descriptor definitions to an RDDS

Use the automatic export function or the EXPORT DEFN TARGET(RDDS) command to save your current resource and descriptor definition to a resource definition data set (RDDS).

- When exporting to a system RDDS, all the IMS resource and descriptor definitions must be exported. All definitions in the system RDDS are overwritten with the definitions being exported.
- When exporting to a non-system RDDS, all or some of the IMS resource and descriptor definitions can be exported. You can either overwrite the existing definitions with the new definitions or append the new definitions to the end of the data set.
- Definitions for IMS resources such as the Fast Path utility (DBF#FPU0) and the IMS defined descriptors (DBFDSRT1, DFSDSDB1, DFSDSPG1, and DFSDSTR1) cannot be exported.

Exporting resource and descriptor definitions with automatic export

Perform the following procedure to set up automatic export of resource and descriptor definitions to an RDDS:

1. Define two or more system resource definition data sets to IMS by using the RDDSDSN() parameter in the DYNAMIC_RESOURCES section of the DFSDFxxx member in the IMS PROCLIB data set.

   If you are using automatic import or automatic export, each IMS system must have its own set of system RDDSs.
2. Specify AUTOEXPORT=AUTO or AUTOEXPORT=RDDS in the DYNAMIC_RESOURCES section in the DFSDFxxx member of the IMS PROCLIB data set.

Resource and descriptor definitions are exported to the oldest RDDS at checkpoint time if the definitional attributes of one or more resources has changed since the last checkpoint.

Exporting resource and descriptor definitions with the EXPORT command

Use the EXPORT DEFN TARGET(RDDS) command to export resource and descriptor definitions to an RDDS.

Resource and descriptor definitions are either written to the data set specified by the RDDSDSN() command keyword, or to the system RDDS containing the oldest data. A system RDDS is one of the RDDS data sets that is defined with the RDDSDSN= parameter in the DYNAMIC_RESOURCES section of the DFSDFxxx member of the IMS PROCLIB data set. A system RDDS contains all the resource and descriptor definitions for a single IMS system.

Related reference:

"DYNAMIC_RESOURCES section of the DFSDFxxx member" on page 682

EXPORT command (Commands)

Importing resource and descriptor definitions

With dynamic resource definition (DRD) enabled, you can import resource and descriptor definitions from a resource definition data set (RDDS) by using the automatic import function or the IMPORT command.

You can also use the IMS Manage Resources application that is available from the IMS Application Menu (option 2) to import resource and descriptor definitions into IMS.

Recommendation: Anticipating that you might eventually have to back out the changes that were made by using the IMPORT DEFN command, perform these tasks:

- Establish a baseline backout recovery point by issuing an EXPORT DEFN TARGET(RDDS) command to export all existing resources and descriptors to a non-system RDDS. If you must back out system definitions afterwards, you can use this set of resources and descriptors with an IMPORT DEFN command to restore the resources and descriptors.
- Specify OPTION(ABORT) on the IMPORT DEFN command so that IMS processes the resource definitions as a group. Either all or none of the definitions are imported.
- If you specify NAME(*) on the IMPORT DEFN command, command responses are returned only for the resource and descriptor names that resulted in an error. Therefore, also specify OPTION(ABORT, ALLRSP) to obtain the command responses for all the resource and descriptor names that are processed.
- Save the output from the IMPORT DEFN OPTION(ABORT, ALLRSP) command, which lists all the resources and descriptor names, in case you must back out the changes made by the IMPORT command.

Related concepts:

Managing IMS resources by using TSO SPOC (Operations and Automation)

Modifying system resources online (Operations and Automation)

Related reference:

IMPORT command (Commands)

Importing resource and descriptor definitions by using the automatic import function

With dynamic resource definition (DRD) enabled, you can enable the automatic import function so that resource and descriptor definitions are imported to IMS during IMS cold start processing.

For IMS to automatically import definitions from a system RDDS, IMS must be enabled with DRD.

For IMS to automatically import definitions from a system RDDS, AUTOIMPORT=RDDS or AUTOIMPORT=AUTO must be specified in the DYNAMIC_RESOURCES section of the DFSDFxxx member and all of the conditions listed in the description of the parameter must be true.

With DRD and the automatic import function enabled, resource definitions are imported from the IMS.MODBLKS data set, or resource and descriptor definitions are imported from a system RDDS. When a resource or descriptor definition is imported, IMS creates the control blocks that are used to manage the resource.

Information, such as the attributes of the resource or descriptor and the status of the resource, is saved in internal control blocks. The information kept in the control blocks is referred to as the *runtime resource or descriptor definition*. When IMS is up and running, you can use the IMPORT DEFN SOURCE(RDDS) command to import resource and descriptor definitions from a system or non-system RDDS.

- To enable automatic import from an RDDS:
  1. Make sure the conditions for automatic import from an RDDS are true.

     If all of the following conditions are true, automatic import from an RDDS is enabled:
     - Two or more system RDDSs are defined in the DFSDFxxx member of the IMS PROCLIB data set.
     - All the defined RDDSs can be allocated and read.
     - At least one of the RDDSs contains valid resource and descriptor definitions.
  2. Specify AUTOIMPORT=RDDS or AUTOIMPORT=AUTO in the DYNAMIC_RESOURCES section of the DFSDFxxx member of the IMS PROCLIB data set.
- To enable automatic import from the IMS.MODBLKS data set:
  1. Make sure the conditions for automatic import from the MODBLKS data set are true.

     If all of the following conditions are true, automatic import from the MODBLKS data set is enabled:
     - The MODBLKS data set exists.
     - No RDDSs are defined in the DFSDFxxx member of the IMS PROCLIB data set.
     - The MODBLKS data set exists, and all the defined system RDDSs are empty.
  2. Specify AUTOIMPORT=MODBLKS in the DYNAMIC_RESOURCES section of the DFSDFxxx member of the IMS PROCLIB data set.

Related reference:

COMMON_SERVICE_LAYER section of the DFSDFxxx member (System Definition)

Importing resource and descriptor definitions from an RDDS by using the IMPORT command

After IMS is up and running, use the IMPORT DEFN command to import resource and descriptor definitions from a resource definition data set (RDDS).

If the definition is for a resource or descriptor that is unknown to IMS, IMS creates the internal control blocks needed to manage the resource or descriptor. If the definition is for a resource or descriptor that exists in IMS, the import of that definition fails.

Resource and descriptor definitions can be imported from either the system RDDS with the most current resource and descriptor definitions, or an RDDS that is specified using the RDDSDSN() keyword on the IMPORT DEFN SOURCE(RDDS) command. The RDDS that is specified with the RDDSDSN() command keyword can be a system RDDS or a non-system RDDS. A system RDDS is one of the RDDS data sets that is defined with the RDDSDSN() parameter in the DYNAMIC_RESOURCES section of the DFSDFxxx member of the IMS PROCLIB data set. A system RDDS contains all the resource and descriptor definitions for a

single IMS system. If you are using the automatic import or automatic export function, each IMS system must have its own set of system RDDS data sets.

If an RDDS contains multiple definitions for the same resource or descriptor, the last definition imported is the definition that is used to create the runtime resource or descriptor definition.

Resources defined by IMS such as the Fast Path utility (DBF#FPU0) and the IMS defined descriptors (DBFDSRT1, DFSDSDB1, DFSDSPG1, and DFSDSTR1) are restricted and therefore not allowed to be specified on the IMPORT DEFN command.

Backing out runtime resources imported with the IMPORT command

If you created runtime resources definitions by importing definitions from a resource definition data set (RDDS) by using the IMPORT DEFN SOURCE(RDDS) command, you can back out those changes to your online system both with and without a cold start of IMS.

Related reference:

IMPORT command (Commands)

DELETE commands (Commands)

EXPORT command (Commands)

Backing out runtime definitions imported by the IMPORT command with a cold start of IMS

If you created runtime resources definitions by importing definitions from a resource definition data set (RDDS) by using the IMPORT DEFN command, you can back out the changes to your online system with a cold start of IMS.

Prerequisite: This option requires that all the resource and descriptor definitions were exported to a non-system RDDS before the original IMPORT DEFN command was issued.

For resources imported from an RDDS, perform the following steps:

1. Cold start IMS with no MODBLKS resources defined (by specifying AUTOIMPORT=N in the Dynamic Resources section of the DFSDFxxx member of the IMS PROCLIB data set).
2. Issue the IMPORT DEFN SOURCE(RDDS) command to import all the definitions from the non-system RDDS that was created for backout recovery before the original IMPORT DEFN command was issued.

After the backout is complete, if you are running with system RDDSs defined, issue the EXPORT DEFN TARGET(RDDS) command to export the current definitions to a system RDDS.

Backing out runtime definitions imported by the IMPORT command without a cold start of IMS

If you created runtime resources definitions by importing definitions from a resource definition data set (RDDS) by using the IMPORT DEFN command, you can back out the changes to your online system without a cold start of IMS.

Prerequisite: This option requires that all the resource and descriptor definitions were exported to a non-system RDDS before the original IMPORT DEFN command was issued. It also requires that you saved the output generated from the IMPORT DEFN command, which shows the resource and descriptor definitions that were created or updated as a result of the IMPORT DEFN command.

To back out the changes:

Issue DELETE commands for any resource and descriptor definitions that were created by the IMPORT DEFN command.

After the backout is complete, if you are running with system RDDSs defined, issue the EXPORT DEFN TARGET(RDDS) command to export the current definitions to a system RDDS.

Maintaining your dynamic resource definition environment

After you have set up your IMS systems to support dynamic resource definition (DRD), it is important to establish processes to maintain your DRD environment.

Here are some steps to follow to establish processes to maintain your DRD environment.

Enabling automatic export to an RDDS

Enable the automatic export function to an RDDS by:

- Defining two or more system resource definition data sets.
- Specifying AUTOEXPORT=AUTO in the DYNAMIC_RESOURCES section of the DFSDFxxx member in the IMS PROCLIB data set.

When automatic export is enabled, your resource and descriptor definitions are automatically exported to an RDDS at IMS checkpoint if the attribute values of one or more of your resources or descriptors have changed since the last IMS checkpoint. If you must perform a cold start later, IMS imports the resource and descriptor definitions from the RDDS and creates the runtime resource and descriptor definitions that it must run.

Exporting definitions to an RDDS with the EXPORT command

If the attribute values of one or more of your resources or descriptors have changed since the last IMS checkpoint, save the definitions to an RDDS.

As an alternative to enabling the automatic export function to an RDDS, use the EXPORT DEFN TARGET(RDDS) command to save your current resource and descriptor definition to an RDDS.

Enabling automatic import

Enable the automatic import function by specifying AUTOIMPORT=AUTO in the DYNAMIC_RESOURCES section of the DFSDFxxx member of the IMS PROCLIB data set. When AUTOIMPORT=AUTO is specified, IMS determines whether to enable automatic import processing. If IMS enables automatic import, it also determines the data source from which to import the definitions.

Whether the conditions for automatic import that are listed in the description of the AUTOIMPORT parameter are met determines if automatic import from a MODBLKS data set or an RDDS takes place.

If you delete runtime resource definitions using a DELETE command and then cold start IMS, the deleted resource definitions reappear if the resource definitions are automatically imported from the original IMS.MODBLKS data set instead of the RDDS that contains the most current definitions.

Removing IMS.MODBLKS data sets

When DRD is enabled, the IMS.MODBLKS data sets are no longer required. After you are satisfied with the setup of your DRD environment and it is running successfully, you can remove the IMS.MODBLKS data sets. If automatic import is enabled, resource and descriptor definitions are imported during IMS cold start from the RDDS that contains the most current data. However, you can continue to use the IMS.MODBLKS data set as the source for all of your resource definitions, instead of an RDDS. If you choose to continue using the IMS.MODBLKS data set, keep your system definition macros synchronized with the changes you make dynamically using DRD commands.

Synchronizing the contents of IMS.MODBLKS

As you migrate to DRD, keep resource definitions that are in the IMS.MODBLKS data set synchronized with the resource definitions that are in the RDDS. This synchronization enables you to maintain a viable IMS.MODBLKS data set if you must disable DRD and fall back to using the online change process for the resources in the IMS.MODBLKS data set. To keep your IMS.MODBLKS data set synchronized with your online definitions, update your static macro definitions with the changes that you make dynamically using type-2 commands. When changes are made dynamically, perform a MODBLKS system definition to add, change, or delete resources from the IMS.MODBLKS data set.

Converting resource definitions into IMS stage 1 macro statements or IMS type-2 CREATE commands

After you create resource definitions and export them to a resource definition data set (RDDS), you can use the RDDS Extraction utility (DFSURDD0) to convert the resource definitions that are in the RDDS into either IMS stage 1 macro statements, or IMS type-2 CREATE commands.

You can convert the resource definitions into IMS stage 1 APPLCTN, DATABASE, RTCODE, and TRANSACT macro statements.

The type-2 CREATE commands into which you can convert the resource definitions include:

- CREATE DB
- CREATE DBDESC
- CREATE PGM
- CREATE PGMDESC
- CREATE RTC
- CREATE RTCDESC
- CREATE TRAN

- CREATE TRANDESC

For additional information about the Resource Definition Data Set (RDDS) Extraction utility, see *IMS Version 11 System Utilities*.

Disabling dynamic resource definition

You might need to disable the dynamic resource definition (DRD) process for MODBLKS resources and fall back to using the online change process.

Disabling dynamic resource definition for MODBLKS resources

You might need to disable the dynamic resource definition (DRD) process for MODBLKS resources and fall back to using the online change process.

With DRD enabled, ensure that the IMS.MODBLKS data set has a complete set of the current resource definitions. This complete set includes resource definitions that were originally in the IMS.MODBLKS data set or in the RDDS at cold start, and that were not changed or deleted dynamically. It also includes any resource definitions that were added or changed dynamically since cold start. This check enables you to fall back to using online change for the IMS.MODBLKS data set with all the resources that were either defined originally by system definition or defined dynamically.

Perform the following procedure to disable the DRD process and fall back to using the online change process:

1. Use one of the following methods to synchronize your IMS.MODBLKS data set with your online definitions:
   - Update your static macro definitions with the changes that you make dynamically using type-2 commands. When changes are made dynamically, perform a MODBLKS system definition to update the IMS.MODBLKS data set.
   - Use the Resource Definition Data Set (RDDS) Extraction utility (DFSURDD0) to extract the resource definitions to create Stage-1 macro statements from the stored resource definitions in an RDDS. Then perform a MODBLKS system definition to update the IMS.MODBLKS data set.
2. Shut down IMS normally.
3. Enable online change for the IMS.MODBLKS data set:
   a. Perform one of the following steps:
      - Remove the MODBLKS keyword from the DFSCGxxx member of the IMS PROCLIB data set. Optionally, also remove the MODBLKS keyword from the COMMON_SERVICE_LAYER section of the DFSDFxxx member of the IMS PROCLIB data set.

        If both members are defined, any values specified in DFSCGxxx override those values that are in DFSDFxxx.
      - Change the value of the MODBLKS keyword from DYN to OLC in the DFSCGxxx member of the IMS PROCLIB data set. Optionally, also change the value of the MODBLKS keyword from DYN to OLC in the COMMON_SERVICE_LAYER section of the DFSDFxxx member of the IMS PROCLIB data set.

        If both members are defined, any values specified in DFSCGxxx override those values that are in DFSDFxxx.

b. Specify AUTOIMPORT=MODBLKS or AUTOIMPORT=AUTO in the DFSDFxxx member of the IMS PROCLIB data set.
c. Remove all RDDS statements from the DFSDFxxx member of the IMS PROCLIB data set.

4. Ensure that the IMS JCL includes the MODBLKS DD statement.
5. Cold start IMS. An IMS cold start creates runtime resource definitions from the stored resource definitions in the IMS.MODBLKS data set. The online change process for the IMS.MODBLKS data set is enabled again (and DRD is disabled). No variations of the CREATE and DELETE commands are allowed in a non-DRD environment. Some variations of the UPDATE command are allowed. You can issue an EXPORT DEFN TARGET(RDDS) command to export to a non-system RDDS.
6. Reestablish your in-house procedures that use the online change process for the IMS.MODBLKS data set, and disable the in-house procedures that use DRD commands.

Chapter 3. Designing the IMS system

Before you can use the Information Management System Transaction Manager (IMS TM) and the Information Management System Database Manager (IMS DB), you must define the elements and functions that make up an IMS system. These elements and functions include databases, application programs, terminals, buffer pool sizes, security options, and more.

The parameters that are associated with the IMS system configuration macros create the integrity of the IMS online system. These parameters specify resources that are available for system checkpoints and program isolation. The allocation of system data sets and control program storage depends upon the expected processing load and the strategy for recovery of the online IMS system.

Choosing the number of regions

One system definition task is to specify how many BMP and message regions comprise your IMS system. The number of regions can vary depending on what your operations require.

Use the MAXREGN keyword on the IMSCTRL macro to specify a working set of BMP and message regions.

Your operations procedure stipulates how many regions are active and when they are active. The scheduling algorithm controls the actual programs that execute in the regions in response to enqueued messages. Overall throughput is conditionally based on the number of active dependent regions and the availability of shared resources for database processing. A processing program must have an available message region before it can enter the scheduling selection logic.

Up to 999 other dependent regions can be dynamically allocated by the MTO using the /START command. For DBCTL, the maximum sum of BMPs and DRA threads is 999.

Setting a checkpoint frequency

IMS records information for restarting interrupted operations by using checkpoints. When an operation is interrupted, reprocessing occurs from the point of system interruption to a forward continuation point. Using the status information captured during checkpoint, IMS can restore the content of the message queues and database changes.

Use the CPLOG keyword on the IMSCTF macro to control the frequency of IMS internal checkpoints. The value you choose for this keyword represents the number of system log records that are created before a new system checkpoint is invoked. When the increment exceeds the CPLOG value, checkpoint processing begins. The default is 500 000. You can also use the /CHECKPOINT command to initiate a checkpoint. Application programs that are authorized to enter the /CHECKPOINT command can also initiate a checkpoint. The /CHECKPOINT command, used with certain parameters, shuts down your IMS system. Make sure that you understand the consequences of shutting down the system before you issue this command. After the checkpoint, use the /NRESTART command to restart the system.

© Copyright IBM Corp. 1974, 2015

Take frequent checkpoints to minimize the amount of time it takes to do a restart. The trade-off for taking frequent checkpoints (and efficient restart) is an increase in processing overhead.

You can change the CPLOG value by using the /CHANGE CPLOG command. To calculate a revised CPLOG value, use the transaction profiles and estimate, for a small peak processing interval, the number of system log events. The number of log records exceeds the sum of the following values:

- (Number of input transactions) x 2
- (Number of output messages) x 3
- (Number of secondary transactions) x 2
- (Number of database update calls for REPL,DLET)
- (Number of database inserts) x 2
- (Number of related index pointer maintenance calls) x 2
- (Number of programs scheduled and terminated)
- (Instances of dial lines connected and disconnected)
- (Number of checkpoint calls from programs) x 3

To ensure an accurate estimate, your checkpoint frequency should be large enough to extend beyond the estimated response time for at least one of the longest-running transactions.

You can adjust the checkpoint frequency using /CHANGE CPLOG after observing the stability of the online IMS system. Use IMS Monitor reports to assess the processing overhead. The Region Summary report shows the total elapsed time and average elapsed time taken for checkpoints in the trace interval.

Related reference:

/CHECKPOINT command (Commands)

Selecting an IMS lock manager

To protect database integrity, an IMS system serializes requests for database resources to ensure that no two application programs are allowed to update a database segment concurrently. Lock management is the process of controlling concurrent requests.

For lock management, you can use either or both of the following methods:

- A program isolation lock manager
- The services of the Internal Resource Lock Manager (IRLM) component

The program isolation lock manager controls only lock requests for a single IMS system (local locking); the IRLM controls requests for multiple IMS systems (global locking), as well as for a single system.

You can monitor program isolation activity using the IMS Monitor, which is invoked by the /TRACE command. A separate program isolation trace can also be used to analyze performance. The advantage of using program isolation to manage locks is that you do not need any special operating and recovery procedures, as is necessary for the IRLM.

The IRLM component is used as a part of data sharing. With program isolation, all activity (modifying the database and creating messages) of an application program

that is active in the DB/DC environment is isolated from any other application programs that are active in the system. The isolation persists until that application program confirms, by reaching a synchronization point, that the data it has modified or created is valid.

Note: A limitation of the PI lock manager is that it can only manage up to 63 applications that are waiting to own a lock. If one additional application asks the PI lock manager for a lock (this application would be waiter number 64), the application terminates with an abend 2478, is backed out by IMS, and its locks are released. If the requestor is an IMS TM transaction, it is returned to the queue and is reprocessed.

To indicate that an IRLM is to manage locking, specify the IRLM's z/OS subsystem name on the IRLMNM parameter of the IMSCTRL macro. The IRLM is required if the IMS online system is to take part in block-level data sharing. Because the IRLM executes in a separate region, you can plan for a known amount of common storage area (ECSA) to be allocated for use by the IRLM component, or you can retain most of its control blocks in local storage rather than in the common storage area.

Your choice of lock manager is not necessarily permanent. On the IMS procedure, specify IRLM=Y, and give the IRLM z/OS subsystem name on the IRLMNM parameter, to indicate this change. This allows the IMS online system to take part in block-level data sharing.

Using a DL/I separate address space

The DL/I separate address space (DLISAS) is used by the online IMS control program to contain most of the DL/I code and control blocks. For some system configurations, using a DLISAS is required.

You can use a separate address space to contain DL/I code, control blocks, and buffers for full-function databases. To do this, use the local storage option parameter (LSO) on the IMS procedure. After specifying LSO=S on the IMS procedure, you can use the DLINM parameter of the IMSCTRL macro to specify the name of the DLISAS procedure to start.

For the DBCTL environment, a DLISAS is required and generated using the DBC procedure. The LSO parameter, therefore, is not supported. However, in a DB/DC environment, if a DBCTL function is used, you must specify that the DLISAS be used by specifying LSO=S.

If you have an IMS system that is defined as an RSR tracking subsystem that does Database Level Tracking (DLT) for full-function databases, you must use DLISAS.

DLISAS storage considerations

When you select the LSO=S option, storage is moved to a separate address space. Additionally, most of the PSB pools and the DMB pools are moved to the DLISAS. These pools are not moved if you choose local storage (LSO=Y).

The IMS control program automatically initiates the DLISAS. If either the IMS control program or DLISAS terminates, the other is automatically terminated.

IMS restart procedures do not maintain LSO specifications. For example, an IMS system using a DLISAS can be terminated, and IMS restart procedures can be performed that specify a local storage option.

When you use a DLISAS, the following storage elements are moved into the DLISAS:

DL/I code

Database buffers

DMB pool, both resident and nonresident

DMB work pool

Most of the PSB pool, both resident and nonresident

PI ENQ and DEQ tables (for non-Fast Path systems)

When you use a DLISAS, the following storage elements are in the z/OS common storage area:

OSAM code

Log buffers

Resident intent lists

Defining PSB pools in an IMS system that uses a DLISAS

In an IMS system that uses a DLISAS, two PSB pools exist: one in the z/OS common area (DUMP pool), and one in the DLISAS (DEPTHS pool). The DUMP pool contains the communications PCBs, Fast Path PCBs, and the key feedback area of the full-function PCBs. The DEPTHS pool contains the DL/I control blocks that are associated with a full-function database PCB (the JCB, for example).

PSBs are slightly larger in IMS Version 10 and later, due to control block size increases. The larger PSBs require more space in the PSB pool. Specify the sizes of these pools with the SASPSB parameter on the BUFPOOLS macro. You can override these sizes with the CSAPSB and DLIPSB parameters on the IMS procedure. The command /DISPLAY POOL PSBP displays usage information for the CSA pool and the DLISAS pool in an IMS system that uses a DLISAS.

For a given PSB, the output of the Application Control Blocks maintenance utility indicates the amount of space required in each pool. The IMS Monitor output reports indicate an out-of-space condition in the DEPTHS pool or a wait-for-storage condition if they occur. In these reports, 'DUMP' and 'DEPTHS' represent the CSA PSB pool and the DLS PSB pool, respectively.

The Application Control Blocks maintenance utility organizes the PSB so that all data for the DUMP pool precedes all data for the DEPTHS pool, regardless of the IMS configurations.

If local storage is used instead, a single PSB pool resides in the z/OS common area. In this case, the PSB parameter on the BUFPOOLS macro and on the IMS start procedure specifies the size of this pool.

To page fix the PSB pool (keep the pool in storage for the long term) specify POOLS=DLMP and POOLS=DPSB in the DFSFIXxx member in the IMS.PROCLIB data set. To page fix the resident PSB pool, include the following module names in the DFSFIXxx member: DFSPSBRS and DFSDLIRS.

Other planning considerations

Accounting procedures with a system that uses local storage option: Accounting procedures that are based on either the IMS system log, the z/OS Resource Measurement Facility™ (RMF™), or the z/OS System Management Facility (SMF) can be affected depending on which storage option is specified. This is because the data reflects the existence of multiple address spaces.

The CPUTIME value that is recorded in the IMS system log is one example of this:

- If the local storage option is used (LSO=Y), only application program processing time is recorded in the type X'07' log record; most DL/I processing time is not included. For some applications, DL/I processing time can represent a large percentage of the total processing time.
- If a DLISAS is used (LSO=S), DL/I processing time is included in CPUTIME.

Because the IMS system log does not indicate which LSO option was specified, do not use CPUTIME values for accounting or comparison purposes when switching options.

For SMF, full-function databases are accounted for in the DLISAS. Fast Path databases and the IMS system data sets are accounted for in the control region.

DLISAS security considerations: If the full-function databases are protected by RACF, the DLISAS procedure must be authorized to access these resources.

Tuning considerations: When tuning your system, consider that the IMS control program consists of more than one address space. You might want to include the control region and the DLISAS in the same RMF performance group so that RMF statistics can be compared with systems that do not use a DLISAS.

Starting the DL/I separate address space

If you choose to use a DL/I separate address space, use the DLINM parameter of the IMSCTRL macro to identify the name of the DLISAS procedure that IMS is to start. Some situations require that the DLISAS procedure by modified, and exit routines that run in the DLISAS have restrictions.

After opting to use a DLISAS by specifying LSO=S on the IMS procedure, use the DLINM parameter of the IMSCTRL macro to specify the name of the DLISAS procedure to start. Unless you change that name, this is the name of the DLISAS procedure that IMS tries to start. You can override this name using DLINM= in the DFSPBxxx IMS PROCLIB data set member that is used to start IMS.

IMS system definition produces the skeletal procedure DLISAS and places it in the IMS PROCLIB data set. Modify this procedure as necessary and copy it to the SYS1.PROCLIB data set.

Modifying the DLISAS procedure

The following modifications to the DLISAS procedure might be required:

- The JCL DD statements for the full-function databases must be in the DLISAS procedure, not in the IMS procedure. JCL DD statements for Fast Path databases and the IMS system data sets remain in the IMS procedure.
- The specification of the active and inactive ACBLIBs must be identical in both the IMS procedure and the DLISAS procedure. Both the control region and the

DLISAS read the ACBLIB. The data sets used must have shared disposition (DISP=SHR), and the concatenation order must be identical.

These ACBLIB data sets can be dynamically allocated by using the DFSMDA member.

- The IMS PROCLIB data set must be defined in both the IMS and DLISAS procedures.

The only parameters passed on the DLISAS procedure are the region type (DLS) and, optionally, the IMSID of the control program to be connected. Pool sizes and database buffering options are specified on the IMS start procedure.

No changes to the dynamic allocation parameter lists in IMS.SDFSRESL are required.

Exit routines running in the DLISAS

Exit routines that run in the DLISAS are entered in cross-memory mode. These exit routines have the following restrictions:

- The DL/I exit routines cannot address storage in the control address space.
- If the DL/I exit routines use the IMS ISWITCH service, they must be changed. A macro specification of TO=DLI, not TO=CTL, must be specified on ISWITCH. The TO=DLI specification performs correctly in all IMS configurations.

Related reference:

"DLISAS procedure" on page 551

"IMS procedure" on page 560

"DFSMDA macro" on page 325

Allocating Message Format Service buffer pool space

The BUFPOOLS macro allows you to reserve storage for efficient processing of communications activity. The amount of storage that is required depends primarily on how much concurrent use of message format blocks is expected, and how many communication lines are active.

You can control the size of the message format buffer pool and the number of fetch request elements (FREs) using the BUFPOOLS macro. One FRE is required to control each active block. Without a FRE, space cannot be assigned from the message format buffer pool. When estimating the number of FREs, allow at least 10 additional FREs to handle system activity. This eliminates the possibility of delayed response to other terminals. If preallocated FREs are not available, a dynamic FRE is allocated from the general area of the pool. However, dynamic FREs tend to fragment the pool, so avoid using them.

You can override the amount of storage for the message format buffer pool and the number of FREs using the FBP and FRE parameters in the EXEC statement for the control region.

To determine an adequate size for the message format buffer pool:

1. Compile a list of the Message Format Service (MFS) blocks, arranging them in DIF/MID and DOF/MOD pairs.
2. Record their sizes and some indication of their frequency (for example, whether they are used by priority or by quick response transactions).

3. Allow space for input/output pairs to be in the pool. Watch for large messages or those with multiple segments, because these might pre-empt small, frequent messages from finding space for their format blocks in the pool, and this would directly affect their response time.

If you use the MFS Service utility to build an index of the DASD addresses that reside in the message format buffer pool during online execution, the index entries also reside in extended private storage.

Use of the message format buffer pool

Message format buffer pools contain MFS control blocks. The control blocks can be loaded using one of two pool manager routines, depending on whether the control block is needed immediately.

MFS control blocks are loaded into the message format buffer pool by one of two MFS pool manager routines: an immediate fetch routine or a prefetch routine. The immediate fetch routine loads a control block that is needed immediately and not already in storage. The prefetch routine is used only if the prefetch option is specified during initialization of the online IMS control region. It is invoked when IMS anticipates the need for a control block; if the block is already in storage, use of prefetch lowers the likelihood that the block is forced out to make room for another block. The use of prefetch requests can improve response time.

When a requested control block is not already in the pool, and its name resides in the resident directory, $$IMSDIR, a single read operation loads the control block into storage. If the name is not in the resident directory, a read of one data set directory block followed by a read of the control block loads the control block into storage. Multiple reads of the data set occur when the size of the control block is greater than either the track size or the z/OS-specified BLKSIZE.

Fetch request elements

Fetch request elements (FREs) are used to represent control blocks that are either already in storage or have been requested for loading into storage. The number of FREs to be defined depends on the number of control blocks that can reside in the pool (a function of pool size) and the number of concurrent requests to load control blocks. FREs defined during IMS system definition or during initialization of the online IMS control region are allocated contiguously in the buffer pool.

Pool space management

The main objectives of message format buffer pool space management are:

- To keep as many control blocks as possible available in the pool.
- To minimize pool fragmentation.

The MFS pool manager assigns a priority to each fetch request. FREs represent prefetch requests and immediate fetch requests. When required, additional FREs are assigned from available pool space to represent immediate fetch requests. If an FRE is not available when a fetch request is made, a control block of lesser importance is discarded so that its FRE can be used to represent the new request.

When there is not enough pool space to hold a requested control block, a space-freeing algorithm is used to select less important control blocks (based on their frequency of use) for discarding.

Pool statistics

Use the /DISPLAY POOL command to display the contents of the message format buffer pool counters. The statistics displayed for directory and fetch input/output operations and pool space use should help in determining the appropriate pool size and number of FREs.

Declaring online databases

Before a database can be used in the IMS system, it must be declared. Declare online databases to the IMS control region using the DATABASE macro statement or the CREATE DB command.

The IMS control region allows the application program to access the database and provides DL/I call services. To declare online databases:

1. Define a list of all the physical databases that can be used by the programs by including DATABASE macro statements in the system definition stage 1 input).
2. Include separate statements for the index and data portions of a HIDAM or PHIDAM database.
3. Include one statement for each secondary index database that refers to any other database that is defined to the online system.

The name that you use on the DATABASE macro is the same as the name used on the DBD statement, or the member name in IMS.DBDLIB, for the corresponding physical or index database definition. Do not specify logical databases.

The DATABASE macro allows you to specify a performance option for an individual database. Specify RESIDENT to load the control data management block (DMB), which is needed to access the database, from the active IMS.ACBLIBA/B data set into a separate DMB pool at system initialization. The advantage to this option is that these databases do not incur OPEN or CLOSE penalties caused by competition for space in the DMB pool.

The default value of the DATABASE macro ACCESS keyword is exclusive (EX), which specifies that the database is for exclusive use by this online system.

Declaring online application programs

Before application programs can be used in an IMS system, they must be declared. Use the APPLCTN macro or the CREATE PGM command to specify a unique Program Specification Block (PSB) name, with which the online system identifies an application.

Declaring application program characteristics

Use the APPLCTN macro or the CREATE PGM command to specify characteristics for application programs that run under the control of DB/DC, DBCTL, and DCCTL systems; examples of these characteristics include:

- The type of application program is being defined, either batch message or message processing.
- The transaction class for the messages that the program is to receive.
- Whether the application program executes exclusively in Fast Path.

The APPLCTN macro requires that you specify whether the program can be concurrently scheduled into more than one message region or batch message region. Parallel scheduling (SCHDTYP=PARALLEL) permits the program to schedule into more than one region; however, this option requires that processing is truly independent. Each schedule requires its own section of any shared pools. Program isolation activity can increase if the processing has the potential to perform updates in the same database record. You cannot control the ultimate processing sequence for transactions of the same type.

Use the GPSB keyword of the APPLCTN macro or the CREATE PGM command to cause the scheduling process of all environments to generate a PSB that contains an I/O PCB and an alternate modifiable PCB. This generated PSB can be used, for example, by an application that makes only SQL calls. PSBGEN and ACB generation are not required when a GPSB is used.

Specifying PSB performance options

The IMS online system uses a Program Specification Block (PSB) name to identify application programs. The message processing program uses this same name for identification. The PSB accesses the message input queue and declares any alternative destinations, other than the input source, for messages sent by the program. The PSB defines a complete list of database access intentions down to the segment level, or to the field within the segment. Before access, the PSB is prepared as a member of IMS.ACBLIB. The online system expects this control block to be predefined.

Use the RESIDENT option of the APPLCTN macro or the CREATE PGM command to specify whether the PSB is to be made resident during system initialization. Using the RESIDENT option eliminates I/O to ACBLIB when the PSB is scheduled. No storage fragmentation occurs in the resident PSB space as it does in the PSB pool, so more efficient use of storage is the result.

If the PSB is resident, the PSB is managed as follows:

- During system initialization, the total space for all resident PSBs is computed. This amount of space is obtained, and all resident PSBs are read from the active Application Control Block Library (ACBLIB) into this space.
- If parallel scheduling is in effect, when the program is scheduled, the resident PSB is allocated for that scheduling if it is not currently allocated to an executing program. If the resident PSB is being used, the PSB pool is searched for an inactive copy of the PSB. If one is found, that copy is allocated and used. If none is found, space for a copy is obtained in the PSB pool, and the resident PSB is copied into that space. The addresses within the copied PSB are then updated based on its new location, and its status related to the former copy is reset.

  If serial scheduling is in effect, when the program is scheduled, the resident copy of the PSB is allocated for that scheduling. Multiple copies of the PSB are not required; therefore, copies of the PSB are not required in the PSB pool.

  Whether you specify serial or parallel scheduling, the PSB is read only once during initialization.

If the PSB is not resident, the PSB is managed as follows:

- When the program is scheduled, the PSB pool is searched for an inactive copy of the PSB and, if one is found, it is allocated and used. If none is found, space for a copy is obtained in the PSB pool.

If an active copy exists in the pool, it is copied, its addresses are updated, and its status is reset to make it available for use. If no active copy exists, the PSB is read from the active ACBLIB.

When it is necessary to release a PSB in the pool, preference is given to retain only copies of PSBs to minimize ACBLIB I/O.

There is an option to put non-resident ACBs into 64-bit storage. Scheduling with non-resident ACB resources with the 64-bit ACB storage pool works as follows: At the first scheduling of a program, its PSB and any related DMBs are loaded into the 31-bit non-resident pools and are also loaded into the 64-bit ACB storage pool. At subsequent schedulings of this program, ACB members not found in the 31-bit non-resident pools are copied from the 64-bit ACB storage pool to the 31-bit non-resident pools, avoiding an I/O to ACBLIB. If the 64-bit ACB storage pool is full, an LRU algorithm is used to remove old members to make room for new members.

Base your use of the RESIDENT option on the frequency with which the PSB is used. If the frequency would cause at least one copy of the PSB to normally be in the pool even if it were not defined as resident, it should be declared resident. If the PSB is used only occasionally, it should not be declared resident, so that its space in the PSB pool can be released when the PSB is not being used.

Specifying a dynamic PSB

Use the dynamic option (DOPT) parameter on the APPLCTN macro or the CREATE PGM command to have the PSB read from the active ACBLIB each time the program is scheduled. This specification allows an ACB generation for a PSB to be performed while the online IMS system is running. The result of that ACB generation is reflected in the next scheduling of the program using that PSB.

If you specify the dynamic option, the PSB is managed as follows:

- The PSB is not initialized at system initialization. The PSB does not need to be in the active ACBLIB during system initialization.
- When the program is scheduled, the PSB is read from the ACBLIB into the PSB pool.
- The space in the PSB pool is released when the program terminates.

You can use the dynamic option in a test configuration. The dynamic option can also provide a PSB to be used with the Online Database Image Copy utility.

Restrictions for dynamic PSBs:

- A dynamic PSB cannot be made resident.
- A dynamic PSB cannot use the 64-bit storage pool.
- A program that uses a dynamic PSB cannot be scheduled in parallel.
- An MPP that is scheduled against a dynamic PSB cannot go through quick reschedule. Quick reschedule allows application programs to process more than the processing limit of messages for each physical schedule.
- An MPP that is scheduled against a dynamic PSB cannot become a pseudo WFI (pseudo wait-for-input). The pseudo WFI option allows an MPP region to remain scheduled until another input message appears.

All databases that are referenced by the dynamic PSB must have been defined to the system and must be present in the ACBLIB at system initialization or after the

last online change was performed. This is because a corresponding option does not exist for DMBs, even though the PSB is dynamic.

The current ACBLIB must consist of two or more concatenated data sets, and the dynamic PSB must reside in any data set in the concatenation other than the first. The concatenated data sets must be of the same format and contain the output from an ACB generation.

Declaring Batch Message Processing programs

Use the PGMTYPE parameter of the APPLCTN macro or the BMPTYPE parameter of the CREATE PGM command to describe a BMP as a batch message program. Include each BMP program as a separate application on its own APPLCTN macro. You declare the BMP program by its use of a PSB, even though the batch JCL allows you to specify a program name that is different from the PSB name.

If a generalized BMP program can execute with different PSBs, you must include APPLCTN macros for all of them. A choice of PSB typically involves several queues that can be processed by the one program. At execution, use the IN parameter to specify an input transaction code.

Defining online applications

You can use either IMS commands with dynamic resource definition (DRD) enabled or the IMS system definition macros to define online applications.

IMS online applications consist of individual programs that are scheduled to process transactions. Scheduling must be invoked by JCL for batch message programs. You must identify to the control program both the available programs and the transactions that they process. You must also identify the entire set of databases that the applications can refer to.

If you want data sharing to be supported, be sure that you understand the interaction between the IMSCTRL and DATABASE macro statements and parameters.

The table that follows lists database and application macros, how many are coded, and the purpose of the macro. Although these macros are optional (because of DRD), you must ensure that the appropriate database and application control blocks are available to your IMS system. Otherwise, your IMS system might not work as you want it to.

Table 5. IMS macros used to define online applications

Macro	Number of macros coded	Purpose of macro
APPLCTN	One per PSB	Names the application program that processes the transaction codes specified on the TRANSACT statements. The TRANSACT macro follows the APPLCTN macro in the stage 1 system definition input stream.

Table 5. IMS macros used to define online applications (continued)

Macro	Number of macros coded	Purpose of macro
DATABASE	One per HSAM, HISAM, HDAM, or PHDAM database; 2 per HIDAM or PHIDAM database; 1 per secondary index database that refers to another database; 1 per MSDB, DEDB	Defines all databases that IMS online control program (and DBCTL environment) manages.
RTCODE	One or more per APPLCTN macro	Specifies the routing codes that the Fast Path Input Edit/Routing exit routine (DBFHAGU0) uses to select a Fast Path application program.
TRANSACT	One or more per APPLCTN macro; 0 for non-message BMP regions	Names the transaction codes to be processed by the application program that is specified on the APPLCTN macro. The APPLCTN macro precedes the TRANSACT macro in the stage 1 system definition input stream.

For a DBCTL environment, you define application programs only for BMP regions by using the APPLCTN macro. For an application program that runs in a CCTL or uses the ODBA interface, use the APPLCTN macro to define the PSB names that the appropriate applications require. The TRANSACT macro is not used in defining a DBCTL system.

Declaring Fast Path application programs

You use the APPLCTN macro or the CREATE PGM command to declare the Fast Path application program name and whether it is a message-driven program. The PSB keyword is used to specify the PSB name, which is the same name as that of the message-driven program.

Declaring program processing

When you specify the PGMTYPE keyword value as TP, you designate the application program as message-driven. At the same time, you can specify FPATH=YES or FPATH=SIZE if you are certain the application program is always to be executed under Fast Path control. FPATH=SIZE establishes an Expedited Message Handler Buffer (EMHB) size for the application program. Specify FPATH=SIZE for the application program if its input or output message requirements are larger than the system default.

One special effect of explicitly declaring the application program to be a message-driven Fast Path program is that the subsequent TRANSACT macro automatically specifies the transaction to be Fast Path exclusive. The sequence of APPLCTN and TRANSACT macros also causes a routing code, with the same value as the transaction code, to be automatically generated in a routing code table; the routing code indicates that this is for a Fast Path-exclusive transaction.

If you do not explicitly declare the application program to be Fast Path, the sequence of APPLCTN and TRANSACT macros (when FPATH=SIZE or FPATH=YES is specified on the TRANSACT macro) identifies a Fast Path-potential transaction.

Most of the other APPLCTN keywords have the same use as for DL/I application programs. However, for the PGMTYPE keyword, the value of BATCH is not used for Fast Path programs. Also, the OVLY parameter and the option to declare a message class are not valid for Fast Path application programs. Parallel scheduling is specified with the keyword value SCHDTYP=PARALLEL if the message-driven program is to occupy more than one Fast Path dependent region for a balancing group.

Adding routing codes

You use the RTCODE macro or the CREATE RTC command to declare any routing codes that are to be associated with the program specified in the preceding APPLCTN macro statement. You need one RTCODE statement for each routing code. The name can be from 1 to 8 alphanumeric characters. It can be a duplicate of a transaction code or an LTERM name, but each routing code must be unique. These routing codes are in addition to those automatically generated by the APPLCTN-TRANSACT macro sequence.

You use the INQUIRY keyword to specify that, when this routing code is used to send a transaction to a balancing group, the program must use inquiry-only processing. INQUIRY=YES should be specified only for those transactions that do not cause a change to any database.

Defining IMS transactions

An IMS transaction is a message destined for an application program. In defining a transaction to IMS, you identify several characteristics, including scheduling rules, exceptions to those rules, transaction codes, output limits, and more.

Use the TRANSACT macro (or the CREATE TRAN command) to define the overall online IMS system response to incoming messages.

With the TRANSACT macro, you can relate one or more transaction codes to the PSB that was named in the preceding APPLCTN macro.

The TRANSACT macro is supported in the DB/DC and DCCTL environments, but not the DBCTL environment. The following table summarizes transaction characteristics that are declared in the TRANSACT macro:

Table 6. Summary of transaction characteristics defined with the TRANSACT macro.

Transaction characteristic	TRANSACT parameter	Parameter description
What are the transaction identifiers?	CODE	Identifies unique transaction codes.
	SYSID	For Fast Path, also declares a possible routing code. Identifies remote system and local system.
What rules govern program scheduling for this transaction?	DCLWA	Log write-ahead option.
	PARLIM	Queue count that triggers parallel scheduling.
	PRTY	Normal priority and alternate priority when queues build up.
What exceptions are there to the scheduling rules?	MAXRGN	Limit for number of regions that can be scheduled per transaction.
	PROCLIM	Limit for number of input messages processed or the processing time.
	SCHD	Alternative options when the transaction is unavailable.

Table 6. Summary of transaction characteristics defined with the TRANSACT macro (continued).

Transaction characteristic	TRANSACT parameter	Parameter description
What type of processing?	INQ	Update database or not. Recovery required or not.
	FPATH	Fast Path potential transaction.
	MODE	Whether or not transactions are autonomous.
	MSGTYPE	Multisegment code or not. Response before more input or not.
	ROUTING	Whether or not program is aware of originating system.
	SERIAL	Serial processing of transaction.
	SPA	Conversational transaction.
	WFI	Wait for input transaction.
What edit operations?	EDIT	Whether or not to convert input data to uppercase. Name of an input edit routine.
What limits on output?	SEGNO	Maximum segments per transaction.
	SEGSIZE	Maximum segment size.

Configuring IMS transactions

From the application, you need to determine how a transaction is to be processed and how it looks to the user. The following table helps you specify the correct parameters on the TRANSACT macro.

Table 7. Transaction configurations and recommended parameters

Transaction configuration	Recommended keyword parameter
Declares that the program is continuously available and not reloaded each time it is scheduled.	WFI
Specifies that the transaction is an inquiry, and its database processing does not cause any updates. No recovery at system restart is required.	INQ=(YES,NORECOV)
Declares that the transaction is an inquiry, but update processing might occur. Recovery at system restart is required	INQ=(NO,RECOV)
Declares that a response to an input message is the next event.	MSGTYPE=(RESPONSE)
Declares that the response to an input message allows multiple message input actions.	MSGTYPE=(NONRESPONSE)
Specifies that database buffers are written to direct access upon each request for a new message. SNGL is forced for WFI applications.	MODE=SNGL
Specifies a group of transactions that are processed together.	MODE=MULT
Declares that the transaction is conversational.	SPA=
Declares the scratchpad area size in bytes.	SPA=(*size in bytes*)
Declares that an input message edit routine is present. The member name of the bound module in IMS.USERLB (or equivalent library) is *name* parameter. UC specifies that input is translated to uppercase before presentation to program.	EDIT=(UC,*name)*
Protects output message queues by testing the size of an output segment.	SEGSIZE=*size*
Protects output message queues by testing the number of segments issued per transaction processed.	SEGNO=*number*

Defining Fast Path transactions

The TRANSACT macro (or the CREATE TRAN command) is used to identify the transaction code and processing characteristics for a Fast Path transaction. The code is specified with the CODE keyword, and, even though the scheduling of Fast Path programs uses the routing code for its queue selection, the code must be a unique name among the set of transactions, LTERMs, and link names.

Use the FPATH keyword on the TRANSACT macro to signify that the transaction is a *Fast Path-potential* transaction (meaning it can run in either an IFP or a full-function region such as MPP). Specify a parameter value of FPATH=YES. You only need to do this when the APPLCTN macro that precedes the TRANSACT macro does not use an FPATH=SIZE parameter value. The FPATH operand on the TRANSACT macro is ignored if your preceding APPLCTN macro explicitly specifies a message-driven application program. This parameter generates a routing code in a table identical to the transaction code and marks it as Fast Path potential.

The following keywords for the TRANSACT macro differ slightly depending on whether they are used to define DL/I transactions or Fast Path transactions:

MSGTYPE
Specifies a single- or multiple-segment message and its processing mode. Use parameter values SNGLSEG and RESPONSE, because Fast Path transactions must be single segment and response mode.

INQUIRY
Specifies an inquiry transaction. You must use the default parameter value RECOVER, because these transactions are processed in the same way as other Fast Path transactions.

EDIT Specifies translation to uppercase and the presence of an input message edit routine. The latter option is not valid for Fast Path-exclusive transactions. For Fast Path-potential transactions, the edit routine is only invoked if the transaction is routed to IMS.

PROCLIM
Specifies the maximum processing time. For message-driven programs, PROCLIM is the limit for one transaction and uses the real elapsed time, because the terminal is in response mode. The count parameter is ignored.

FPBUF
Specifies the Expedited Message Handler Buffer (EMHB) size for transactions.

Planning a scheduling algorithm

A scheduling algorithm determines how application programs are scheduled to handle the load on the queues.

The basic approach to controlling an online IMS system with loaded queues is to let the demand control the scheduling of programs into a reasonable number of message regions.

You specify a working set of application-dependent regions that can execute concurrently with the first parameter of the MAXREGN keyword on the IMSCTRL macro. The MTO can use the /START command to dynamically allocate other dependent regions. Up to 999 dependent regions (the maximum allowable number

permitted by IMS) can be allocated. For a DBCTL environment, the MAXREGN keyword defines a working set of BMPs, JBPs, and CCTL region threads, or application programs, that can run concurrently. Other BMP and JBP regions can be dynamically started, up to the maximum allowable number, using the /START command. The DBCTL environment does not have any transactions, so information about transactions does not apply to DBCTL.

Scheduling algorithms in an IMSplex

IMS application programs are run either serially or in parallel and must be scheduled accordingly in an IMSplex.

- Serial application programs can only run in one message region or batch message region at a time.
- Parallel application programs can run concurrently in more than one message region or batch message region.

IMS users with an IMSplex with shared queues might want to prevent the concurrent scheduling of serial application programs. The IMS Resource Manager (RM) creates a serial program resource the first time a serial application program runs. If a second message later requires the same serial application program, RM attempts to create a second serial program resource structure. Because the program resource exists, the second recreation fails, and the second IMS cannot schedule the serial application program to run.

Users with an IMSplex and shared queues can prevent serial application programs from being scheduled concurrently.

Grouping application transactions in DB/DC and DCCTL environments

When you group transactions and associate them with a particular message class and region, the criteria you use to group them can include virtual storage considerations, PSB characteristics, and priority of service.

You must associate different groups of transactions with a particular message class that can be assigned to a region. (Batch message programs are allocated in their own region.) Criteria might be the virtual storage the message processing programs require, the PSB characteristics they share, or the priority of service for the end user. Lay out the transactions in a summary matrix. An example of this information summary for a vehicle routing application is shown in Table 8.

Table 8. Example of transaction grouping

Transaction Group	Transaction Code	Transaction Rate	Program	Preload	REGN	Processing Mode
Driver log	TRLOG	1000 /day	PGMA	-	520 KB	SINGLE
Driver changes	TRCHG	300/hr	PGMB	PRLD	200 KB	MULTIPLE
Enter load or job	TRLOAD	50/hr	PGMC	-	520 KB	CONVERSATION
Status of job	TRSTAT	20/hr	PGMD	-	520 KB	SINGLE
Optimize route	TROPT	10/day	PGME	-	400 KB	(SINGLE) BMP
Get driver route	TRROUT	50/hr	PGMF	-	100 KB	SINGLE

One solution for this group of transactions is to define a message class for TRLOG, TRLOAD, and TRSTAT, which seem to be frequently used, and assign them to a message region. The BMP has its own region. The driver control transactions,

TRCHG and TRROUT, can be assigned to another message class, because they have similar virtual storage requirements and one of them needs the program preload function specified on the region JCL.

The process of grouping transactions becomes more complex when you have competing application programs. An additional factor to consider is the database processing intent. You should attempt to combine compatible application programs (such as application programs that are inquiry only) into groups before assigning them a message class and individual priorities.

Stopping transactions and PSBs because of unavailable data

In a DB/DC environment, IMS stops the transaction type if most messages being processed are failing because they are encountering unavailable data.

In a DB/DC environment, IMS stops the transaction type if most messages being processed are failing because they are encountering unavailable data. One abort is counted each time the program aborts due to abend U3303 and the message in process has not previously been placed on the suspend queue. Two aborts are subtracted each time a program goes through commit processing. However, if this results in a negative number, two aborts are not subtracted. If the total number of aborts exceeds 10, the transaction is stopped with a USTOPPED condition.

The transaction is stopped by USTOPPED if abend U3303 occurs while processing the message and SERIAL=YES is specified for the TRANSACT macro.

When the transaction is stopped by USTOPPED, messages from this queue are not scheduled for processing. Incoming messages continue to be queued on the normal queue.

In a DBCTL environment, after ten U3303 abends IMS stops the PSB instead of the transaction, preventing further rescheduling of the application program.

Assigning message class and initializing a region in DB/DC and DCCTL

The message class that is assigned to a transaction determines into which message region an application program is loaded.

Each message (transaction code) is assigned a class using the third parameter of the MSGTYPE keyword on the TRANSACT macro. If the class is not specified this way, the value on the PGMTYPE keyword of the APPLCTN macro is applied. This class assignment determines into which message region an application program is loaded. When the IMS message regions are started, they are assigned from one to four message classes. When a message region is assigned more than one class, the scheduling algorithm treats the first class specified as the highest priority class, and each succeeding class is treated as a lower-priority class.

If more than one class is specified, message selection is processed as follows. The first class specified is scanned, in transaction-priority sequence, for waiting messages. If no messages are waiting for the first class, the second and following classes are also scanned in priority sequence. If messages are waiting in the first class, the highest-priority message is selected for scheduling.

Message priorities within message classes in DB/DC and DCCTL

When developing your scheduling algorithm, you must know the priority and transaction code for each message class. This information helps you set up test levels of queue loading.

To develop your scheduling algorithm, draw a matrix that lists each priority and transaction code. Group all those transactions that belong to a class. You can then test the contents by setting up test levels of queue loading and apply the class and priority algorithms. Such a matrix is illustrated in Table 9.

Table 9. Matrix for message classes and priorities.

Class	Priority	Transaction Code	PSB Name
001	5	TRANX	PGMX
	3	TRAND	PGMD
	2	TRANA	PGMA
003	10	TRANY	PGMY
005	10	TRANC	PGMC
	5	TRANB	PGMB

If an available region specifies a message class priority of 1, 5, 3 and if two transactions are in each queue, PGMX is scheduled first and both TRANXs are processed. If no other transactions are received, the order of processing is TRAND, TRANA, TRANC, TRANB, and TRANY.

Notice how the order of message class prioritizing (specified in the region JCL) causes class 3 to be processed last.

If another message region that specifies message classes 1, 5, 3 is started during the processing of TRAND, the region begins processing with PGMA. Whichever region completes message class 1 transactions first schedules PGMC.

Selection priorities for transactions in DB/DC and DCCTL environments

When more than one transaction of a given type is waiting to be scheduled, the specified transaction scheduling priority determines which transaction code is selected. It does not determine which transaction is scheduled. Only the tests of the transaction's readiness for scheduling, which occur after selection, determine if the transaction queue is allocated to an application program.

Selection priorities are useful for influencing the response time to input transactions and for load balancing. Two priorities can be specified:

- Normal priority
- Limit priority

Related to the normal and limit priorities is the *limit count*. When the number of input messages of a specific transaction type waiting to be scheduled is equal to or greater than the limit count, the normal priority is reset to the limit priority value.

The priority of a transaction code causes it to be selected either before or after other transaction codes. You specify the numeric priority with the PRTY keyword

of the TRANSACT macro. Values can be selected in the range 0 - 14; a value of 0 specifies the transaction is not eligible for automatic scheduling. If multiple transaction codes are at the same priority, they are selected on a first-in/first-out basis. So, if multiple transaction codes are at the same priority and class, with many messages already enqueued for each transaction code, the first scheduled transaction code processes all of its messages before the next, equal priority, and class transaction code are scheduled.

You can raise the priority normally used for a transaction after a certain level of the queue is reached. In this way, you can give the transaction an increased chance of being scheduled. Another case occurs when a program requires significant program loading time or initialization and is then followed by a batch-like processing of a group of transactions.

Suppose the transaction TRANB in Table 9 on page 86 is assigned a limit priority of 14 if the number of queued transactions rises to 10. When message class 5 is available for scheduling and the queue counts for TRANC and TRANB are 18 and 10, respectively, the first program scheduled is PGMB. The processing of TRANB stays at priority 14 until all 10 transactions, and any others added to the queue, are processed. Then TRANB reverts to a normal priority of 5.

It is possible that more messages are added to the queue while the transaction is waiting or in process at the limit priority. The normal priority is not restored until all messages enqueued on the transaction code are processed. The priorities are selection priorities, not execution priorities. After a transaction has been selected for scheduling, the selection priorities have no influence until it is again recognized to be waiting for scheduling.

Limit priority can be in the range 0 - 14. Limit count has a default of 65535 and a valid range of 1 to 65535. You specify limit priority and the queue count as the second and third parameters of the PRTY keyword on the TRANSACT macro. If you do not require this priority override technique, code the limit priority equal to the normal priority, and code the limit count as 65535.

Another way to use the selection priorities is to declare a normal priority value of zero. Zero priority is a null or “not eligible for scheduling” level. Messages accumulate until the limit count is reached; limit priority takes effect and the message is eligible for scheduling. This technique is called *batching messages*.

The effectiveness of the selection priority assignments is related to how frequently the selection process occurs.

Processing limits for messages in DB/DC and DCCTL environments

To influence the frequency with which scheduling selection occurs, you can set processing limits for messages. These limits set the length of time that application programs wait.

By setting processing limits, you can influence the frequency with which scheduling selection occurs. During the time between each scheduling, processing continues in the message regions. Meanwhile, messages are accumulating in the message queues. As messages accumulate, the interactive effects introduced by new message types and the changing of selection priorities are rearranging the order of waiting transaction codes. Conceivably, while a large queue of messages is being processed, important activity assigned to a high-priority transaction code is waiting.

When the program processes a large queue of messages and updates database segments, other application programs trying to access an updated segment are placed into a wait state. The length of time that the other application programs must wait depends on whether the updating program is processing its queue in multiple- or single-message mode.

To allow controlled reentry to the message scheduling selection process, specify a processing limit count for each transaction code. Each time a scheduled (processing) program requests a new message, the limit count is checked. When the number of requests exceeds the limit count, IMS determines if the region is eligible for quick reschedule.

- If the region is not eligible for quick reschedule, the application program completes its processing. If the current transaction code is at the same priority level as other queued transaction codes, the current code is placed last in the list.
- If the region is eligible for quick reschedule, the application program remains active and the next message is returned to the application program for processing.

Quick reschedule for regions in DB/DC and DCCTL environments

Quick reschedule allows application programs to process more than the processing limit of messages for each physical schedule. Quick reschedule eliminates processing overhead caused by unnecessary rescheduling and reloading of application programs.

When a region undergoes quick reschedule, the message count that is compared with the processing limit count is reset, the accounting (X'07') and scheduling (X'08') log records are written, and the next message is returned to the application program for processing.

A region can undergo quick reschedule only when:

- No other work of equal or higher priority exists for the region to process
- The same transaction would be scheduled if the application program terminated and the dependent region went through rescheduling.
- The region is an MPP processing a MODE=SNGL transaction
- The processing limit count is greater than zero
- The PSB is not allocated with the dynamic PSB option (DOPT)

Flags in the accounting and scheduling records written during a quick reschedule indicate that the records do not include actual program termination and scheduling times. These records are written for accounting purposes only. Restart and backout do not use these records.

Pseudo WFI option for MPP regions in DB/DC and DCCTL environments

The pseudo WFI (pseudo wait-for-input) option allows an MPP region to remain scheduled until another input message appears. With pseudo WFI, unnecessary application program termination and rescheduling can be eliminated.

Normally, if an MPP region is scheduled for a transaction and no more messages for that transaction exist, the application program terminates. Frequently, another

message appears for the same transaction after the program is terminated. Processor usage is increased because of unnecessary termination and rescheduling of that application program.

Pseudo WFI is specified with the PWFI= parameter on the MPP region startup procedure. When PWFI=Y is specified, the processing limit count is greater than 0, and no more messages are queued for the current MODE=SNGL transaction, IMS checks for other work for the region to process. If no other work is available, the region waits until another input message appears. This is *wait-for-input mode*.

When the next input message is for the currently scheduled transaction, the message is returned to the application program with a status code of "blank-blank".

When the next message is not for the currently scheduled transaction, termination and rescheduling occur.

Certain circumstances cause regions that are in wait-for-input mode to be posted and a QC status code to be returned to the application program. These circumstances include commands that involve stopping, starting, locking, unlocking, purging, and assigning a database, region, transaction, or class.

The following circumstances also cause similar problems:

- An intent conflict scheduling failure might occur as a consequence of either a local or global load balancing algorithm decision. This causes all PWFI regions to be posted and QC status codes to be returned to the application programs.

  The global load balancing algorithm is a feature of Sysplex Serial Program Manager (SSPM).
- A resource enters the OLC PREPARE phase. This causes all PWFI regions to be posted and a QC status code to be returned to the application programs.
- A program or transaction is stopped as a consequence of a dependent region abend. This causes all PWFI regions to be posted and QC status codes to be returned to the application programs that belong to regions with stopped programs or transactions.

Regions that cannot be scheduled because of a lack of pool space can also post regions currently in pseudo WFI in an attempt to terminate them. This frees pool space so that the failing region can schedule.

Processing transactions against unavailable data in DB/DC and DBCTL environments

When an application program attempts to access unavailable data, IMS abends the application program. Use the INIT call to prevent application programs from being scheduled when a database is unavailable.

IMS schedules an application program even if that application program might try to access an unavailable database. The application program can be *sensitive* or *insensitive* to unavailable data. To be sensitive, it must issue the INIT call. This requests that a status code is returned in the PSB if a subsequent call requires access to data that is unavailable. If the application program has not issued the INIT call and a call requires access to unavailable data, IMS abends the application program with U3303, and backs out any updates it has made.

After ten U3303 abends, IMS takes the following action:

- In a DB/DC environment, IMS stops the transaction with a USTOPPED condition.
- In a DBCTL environment, IMS stops the PSB, preventing further scheduling of the application program.

The disposition of the transaction depends on whether it is serial or not. Serial transactions are those that must be processed in the order of arrival. If it is serial, the processing of any transaction of its type is stopped. If it is not serial, only the processing of this particular transaction is stopped.

Data can be unavailable for these reasons:

- The database is stopped, locked, or unavailable for update.
- A lock cannot be obtained, because it is held by a failing component in a data sharing environment.
- In an XRF environment with block-level data sharing, the takeover system initiated new work before the data sharing configuration is revalidated.

IMS tries to resume transaction processing when any of these events occurs:

- A /DEQ SUSPEND command is issued.
- A /START TRAN command is issued for a transaction type that is stopped or that has stopped messages.
- A /START DATABASE command is issued for a database that is unavailable and in the intent list of the program requesting the transaction.
- A failing IRLM is reconnected.
- An emergency restart completes.
- An XRF takeover completes.
- A sharing IMS system completes a batch backout.

Scheduling transactions using the suspend queue

Messages can be placed on or removed from the suspend queue based on conditions that you describe. An example of these conditions can be user abend U3303, which occurs when a message attempts to access unavailable data, and USTOPPED transactions, which is when a transaction is stopped because most messages being processed are failing because they are encountering unavailable data.

When scheduling transactions using the suspend queue, you must describe the following:

- The conditions that result in messages being placed in or removed from the suspend queue
- The conditions that result in setting or resetting the transaction stopped because of unavailable data (USTOPPED)

When messages are placed in the suspend queue

If the program processing the message attempts to access data in a database that is unavailable, and the program has not issued the INIT call indicating that it can accept a status code that data is not available, the program is pseudoabended with abend U3303. The disposition of the message in process at the time of abend U3303 depends on whether the transaction type being processed requires serial

processing. If the transaction type does not require serial processing, the failed message is placed in the suspend queue. If the transaction requires serial processing, the message is returned to the normal queue and the transaction is USTOPPED.

Data might be unavailable for any of the following reasons:

- The database is stopped, locked, or not available for update.

  Programs are scheduled even when full-function databases are not available, or when they are available as read only. If a program issues a DL/I call that requires access to one of these databases, the program encounters unavailable data.
- A lock on the data cannot be obtained because it is held in retained state.

  In a block-level data sharing environment, it might not be possible to communicate with the sharing system because the sharing IMS has failed, the sharing IRLM has failed, or communication with the sharing IRLM has failed. The IRLM being used by the surviving IMS system retains knowledge of the locks that were held by the IMS system with which communication is temporarily unavailable. These locks are held in retained state. A similar condition can exist in a DBCTL environment when a thread failure occurs.
- In an XRF and block-level data sharing environment, the takeover system initiates new work before the data sharing configuration is revalidated.

  At the time of an XRF takeover, databases that can be shared at the block level are temporarily made unavailable until the data sharing configuration has been revalidated. If programs attempt to access these databases before the revalidation completes, they encounter unavailable data.

When messages are removed from the suspend queue

A separate suspend queue exists for each transaction type. Messages are never scheduled for processing from the suspend queue. To be scheduled, the message must be transferred to its normal queue. Some conditions cause the messages on suspend queues for all transaction types to be transferred to their normal queues. Other conditions cause the messages for specific transaction types to be transferred to their normal queue.

The conditions that trigger the transfer of all messages from the suspend queues, and the rationale for transferring the messages when that condition occurs, are:

- The /DEQ SUSPEND command is issued.

  The operator requested it.
- IMS emergency restart is completed.

  While the IMS system is down, a sharing IMS might notify the system to drain its suspend queues.
- A sharing IMS system notifies the system that the sharing IMS system has completed an emergency restart or a batch backout.

  Messages are transferred for the same reason as when these conditions occur on the local system.
- IRLM is reconnected.

  When the IRLM failed, messages that were in process at the time of failure were abended with abend U3303. Attempts to access data that the failing IRLM locked also result in abend U3303. When the IRLM is reconnected, these messages are scheduled again.
- XRF takeover has completed.

While the takeover is in process, a notification from a sharing IMS might have been missed, and databases that can be shared at the block level are temporarily unavailable until the reverification to DBRC has completed.

The following conditions trigger the transfer of messages for specific transactions to the normal queue:

- A /START TRAN command is issued. This causes the messages for the started transaction to be transferred to the normal queue from the suspend queue.
- A /START DATABASE command is issued. This causes the transfer of messages for transactions in which the program processing the transaction has access to the started database.

Parallel scheduling of applications and transactions

IMS can schedule the same application program and the same transaction in multiple message regions. Designate the application program and the transaction for parallel scheduling using the SCHDTYP keyword on the APPLCTN macro or the CREATE PGM command.

By designating an application program as a parallel-scheduled application program, any transaction processed by that program can be scheduled in multiple regions.

When a transaction is available for scheduling but is already scheduled in another region, IMS checks whether the transaction can be scheduled in parallel. The PARLIM value of the TRANSACT macro specifies the number of messages that should be enqueued before another region is scheduled. This value is multiplied by the number of regions already scheduled for this transaction. If the result is less than the number of messages enqueued, another region is scheduled for the transaction unless MAXRGN is exceeded. If the region cannot be scheduled for internal reasons (database intent), the next transaction within the class is scheduled.

If scheduling fails for intent conflicts for an SCHD=1 or SCHD=2 transaction, the next logical transaction is selected based on the SCHD= parameter of the transaction that failed. If the scheduler fails to schedule any transaction for intent five times, the next class of transactions is selected.

This method of processing can cause messages in the current class to be delayed. For example, a long running BMP has update intents on a database. Several transactions that have update intent on the same database with SCHD=1 are entered into the IMS system. More transactions that do not reference the database but have the same class are also entered into the system. These transactions might not be scheduled until the BMP terminates, if the first group of transactions fails for intent. When transactions from the first group fail for intent a total of five times, scheduling is attempted for the next class. This bypasses the group of transactions that do not reference the database.

To avoid such delays, the second group of transactions should be placed into a separate class, or the BMP job should be run at a different time.

If the PARLIM value is zero and more messages are in the queue, another region can be scheduled. To prevent one transaction from monopolizing all available regions, use the MAXRGN= parameter on the TRANSACT macro. A non-zero value for MAXRGN specifies the number of MPP regions that can be scheduled. In

addition, you can use the SERIAL option of the TRANSACT macro to process transactions in the order they arrive. IMS limits the processing to this time sequence. If data required by the transaction is unavailable, this causes IMS to stop scheduling this transaction type.

For a DBCTL environment, BMP regions and CCTL threads can schedule a PSB simultaneously when the APPLCTN macro or the CREATE PGM command's SCHDTYP keyword is defined as PARALLEL. If a PSB is not defined as PARALLEL and is already scheduled by a BMP or CCTL thread, new schedule requests for that PSB fail.

Alternative scheduling options for messages in DB/DC and DCCTL environments

Messages sometimes cannot be scheduled due to internal or external reasons. In these situations, the next message of equal or lower priority in the same message class, or the highest-priority message in a lower message class, is selected for scheduling.

If a message cannot be scheduled for external reasons (for example, the master terminal operator stopped a program or transaction), the next message of equal or lower priority in that class, or the highest-priority message in a lower class, is selected for scheduling. If the highest-priority message in the first class cannot be scheduled for internal reasons (database intent, no more space in PSB pool, or DMB pool to bring in needed blocks), the scheduling option of the transaction specifies the criteria to be used to select the next transaction to be scheduled. The SCHD keyword of the option is specified at system definition by the TRANSACT macro. The options are:

- Schedule only transactions of equal or higher priority in the selected class. This is the default option.
- Schedule higher-priority transactions in the selected class.
- Schedule any transaction in the selected class.
- Skip to the next class and attempt to schedule the highest-priority transaction in that class.

These scheduling options are specified for each transaction; therefore, if the algorithms are different for transactions within the same class, each attempt to schedule a different transaction might change the algorithm.

Message region class assignments and transaction class assignments are assigned at region initialization through the EXEC JCL statement. The assignments can be modified at execution time by the operator.

If multiple message regions process the same message class and a conflict in database processing intent occurs, the highest-priority transactions scheduled against a database are not necessarily processed before lower-priority transactions scheduled against the same database. If you want to process all higher-priority transactions before processing any lower-priority transactions, specify no processing limit for the higher-priority transactions. Using only one message region to process that message class achieves the same result.

Scheduling for BMP processing

Input transactions to batch message programs do not need to be assigned a competitive priority. BMP regions are manually scheduled.

Because BMP regions are scheduled manually, the input transactions to a batch message program do not need to be assigned a competitive priority. Specify a priority value of zero for both normal and limit priority, and coordinate a message class designation with the BMP region JCL.

Assigning priorities for programs with exclusive intent

Application programs that have exclusive use of a segment type are not concurrently scheduled with other application programs that are sensitive to the same segment type. Conflicts in scheduling can occur if the same segment type is declared exclusive by more than one application program.

When a program's PSB includes exclusive use of a segment type, the program is not scheduled concurrently with any other program that is sensitive to the same segment type. Similarly, when a program executes with exclusive use of segment types, other programs that include sensitivity to any of those segments are not scheduled concurrently. Exclusive intent does not use enqueue/dequeue serialization. Programs have exclusive intent declared by PROCOPT=E on a SENSEG or program communication block, or PCB, statement within the PSB generation—if the option K, for key sensitivity, is not appended. Conflicting actions occur only if the same segment type is declared by at least one of two programs intending to reference a segment exclusively.

One case in which programs require exclusive intent occurs when a program that uses HSAM in its PSB is scheduled.

Use care when assigning priorities for programs with exclusive intent. Even if a program is selected for execution, a conflict with its processing intent and that of an already-executing program causes the transaction to drop out of the selection process until the next program termination or region start event.

Exceptions to the use of program isolation are programs that use the PROCOPT=GO option. These programs can retrieve segments that have been altered or modified by programs that are still active. Those changes might be subject to backout. The programs might not update the segments, and there is no enqueue on the segments when the programs retrieve them.

Scheduling for CPI-Communications-driven programs

Scheduling information for LU 6.2 CPI-Communications-driven application programs is defined in a TP_Profile entry that is managed by APPC/MVS.

When APPC/IMS recognizes a CPI-Communications-driven application program for the first time after restart, it dynamically builds an IMS transaction. IMS dynamically builds the definition for CPI-Communications-driven application programs when a transaction is presented for scheduling by APPC/MVS, based on the APPC/MVS TP_Profile definition after IMS restart. A dynamically built transaction is not checkpointed unless SYNCLVL=SYNCPT and IMS is participating in a protected conversation.

Defining terminals with data communication macros

By using different combinations of the 16 macro statements used for the IMS data communications facilities, you can define terminals to be attached to the IMS online system.

Restriction: This topic does not apply to DBCTL.

Most of the system definition stage 1 input is comprised of declarations that define terminals to be attached to the IMS online system. Sixteen macro statements describe all of your IMS system's data communication facilities. By using different combinations of these statements, you can create macro sets that support:

Non-VTAM devices

Multiple Systems Coupling

VTAM

If you are using IMS Extended Terminal Option Support (IMS ETO Support) of IMS TM to define terminals, you can reduce the number of macro statements that are required for system-defined VTAM terminals. For more information about IMS ETO Support, refer to "Including IMS ETO Support in the IMS system" on page 103.

Table 11 on page 98 provides an overview of each macro set and a detailed description of the macro statements in each macro set. Use this information and the accompanying configuration diagrams to define your terminals. Each of the terminal types and hardware options has equivalent parameters in one or more of the terminal-related macros in the stage 1 system definition input.

In some instances, you must prepare the same macro statements for more than one of the macro sets. Refer to the section "Coding IMS macros that define the system" on page 2 for information about the order of entering the macro sets into the assembler input stream. IMS issues stage 1 system definition output warning messages and does not complete system definition if the macro sets are not entered in this order.

Defining VTAM terminals

If your IMS system uses VTAM, use the COMM, NAME, SUBPOOL, TERMINAL, TYPE, and VTAMPOOL macros to describe the VTAM data communication facilities.

If your IMS system uses VTAM, prepare a set of VTAM macro statements to describe the VTAM data communication facilities. The number of sets you must prepare depends on the hardware configuration of your IMS system.

To add VTAM support, specify an ONLINE or ALL system definition on the IMSCTRL macro statement.

You must enter the macro sets in your IMS system stage 1 input deck in the order specified in "Coding IMS macros that define the system" on page 2. IMS issues stage 1 output warning messages and does not complete system definition if the macro sets are not entered in this order.

All non-VTAM data communication specifications must precede the VTAM macro set in your IMS system definition stage 1 input deck; a stage 1 output warning message occurs if the VTAM macro set is not the last physical set. If an MSC

macro set is part of your system definition, it must precede the VTAM macro set, or your system definition does not complete. The following table lists macros for defining VTAM terminals.

Table 10. VTAM data communication macros.

Macro	Number of macros coded	Purpose
COMM[1]	One only per IMS system definition	Specifies general communication options not associated with any particular terminal type.
NAME	One or more for each physical terminal or component	Specifies the logical terminal name for physical terminals of the type specified by the TERMINAL or SUBPOOL statement.
SUBPOOL	One for each dynamically allocated session	Defines logical unit type 6 (LU 6.1) subpools that can be dynamically allocated within the VTAMPOOL.
TERMINAL	One for each terminal or component; must follow the TYPE macro	Provides the characteristics of the physical terminal type that is specified in the TYPE statement.
TYPE	One for each terminal type	Specifies a group of VTAM terminals of the same type.
VTAMPOOL	One or more per SUBPOOL; the only purpose for multiple VTAMPOOL statements is to facilitate the documentation of the intended use of the logical terminal group	Defines subpools of logical terminals to be dynamically allocated to LU 6.1 sessions.

Note: [1] You can specify only one COMM macro in an IMS system definition. The COMM macro is required for the VTAM macro set.

The following figure is an example of coding for a group of 3270 VTAM terminals.

```
TYPE UNITYPE=3270,MODEL=2,PTRSIZE=132,OPTIONS=COPY
TERMINAL NAME=CT3275,UNIT=3275,COMPT=PTR1,MODEL=1
NAME VT3275
NAME VT3275P,COMPT=PTR1
TERMINAL NAME=CT3277A
NAME VT3270A
TERMINAL NAME=CT3277B
NAME VT3270B
TERMINAL NAME=CT3277C,MODEL=1
NAME VT3270C
TERMINAL NAME=CT3277D,UNIT=3284
NAME VT3270P1
TERMINAL NAME=CT3277E,UNIT=3286
NAME VT3270P2
```

The following figure shows the configuration for 3270 VTAM terminals using the VTAM macro set.

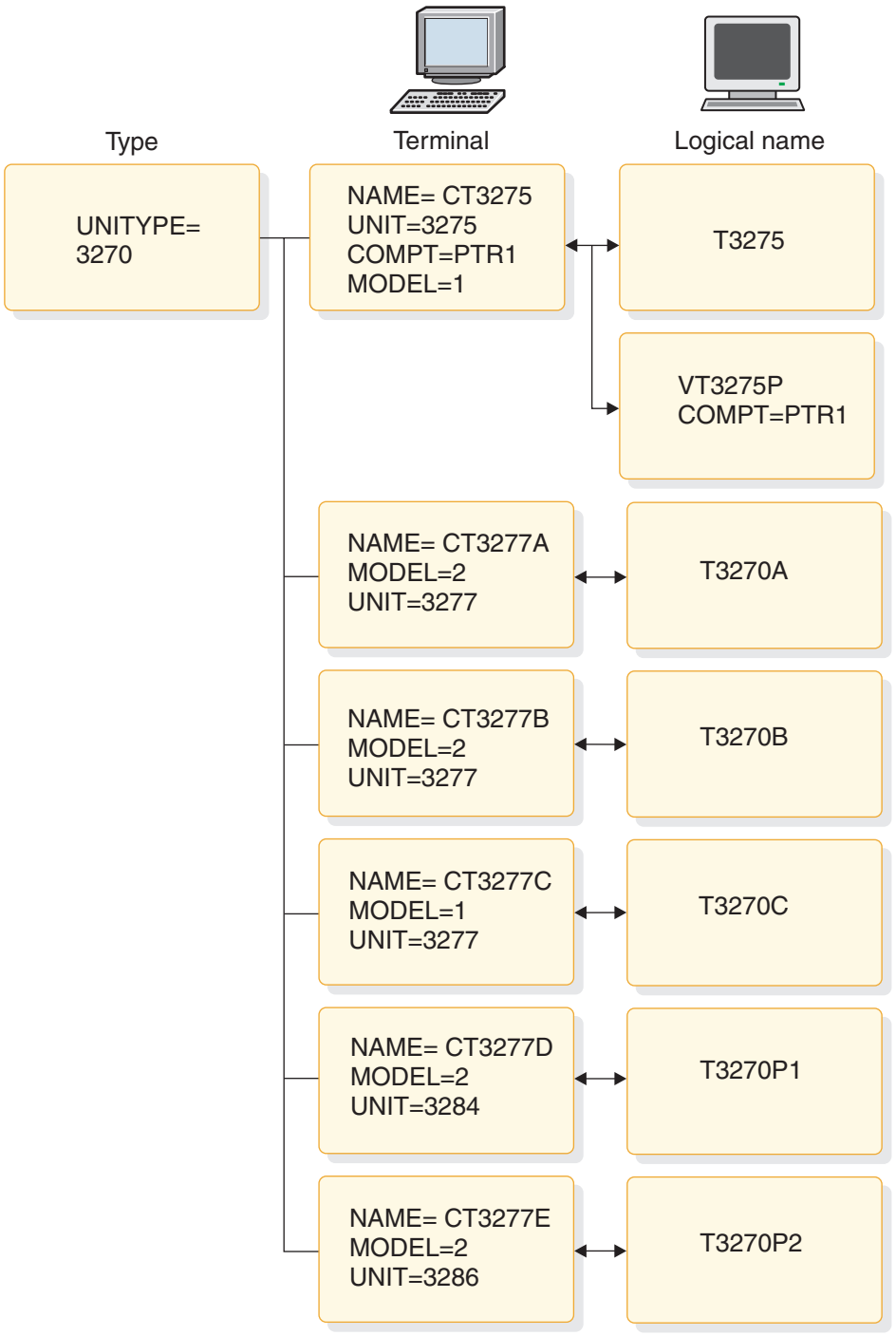


Figure 7. Configuration for 3270 VTAM terminals

Defining non-VTAM devices

You can use the COMM, LINE, LINEGRP, NAME, and TERMINAL macros to define non-VTAM devices to your IMS system.

If your IMS system uses non-VTAM devices, use the macro statements from the set shown in Table 11 to describe your data communication facilities. The number of sets you must prepare depends on the hardware configuration of your IMS system.

All non-VTAM data communication specifications must precede the VTAM macro set in your IMS system definition stage 1 input deck. You receive a stage 1 output warning message if the VTAM macro set is not the last physical set. If an MSC macro set is part of your system definition, it should precede your VTAM macro set; otherwise, your system definition does not complete. To add VTAM support, you must specify either an ONLINE or ALL system definition on the IMSCTRL macro statement. Non-VTAM data communication macros, the number that is coded, and their purpose are listed in the following table.

Table 11. Non-VTAM data communication macros.

Macro	Number of macros coded	Purpose of macro
COMM[1]	One only per IMS system definition	Specifies general communication options that are not associated with any particular terminal type.
LINE	One for each line of the line group following the LINEGRP data	Assigns address and characteristics of one line in a line group.
LINEGRP	One for each line type	Assigns DD names for a group of lines and terminals with like attributes.
NAME	One or more for each logical terminal or component following each TERMINAL macro	Specifies the logical terminal name for the physical terminal that is specified by the TERMINAL macro.
TERMINAL[2]	One for each physical terminal attached to the line specified by the LINE statement	Specifies physical terminal characteristics.

Note:

1. You can specify only one COMM macro in an IMS system definition. The COMM macro is required for VTAM.
2. When the TERMINAL statement describes a switched physical terminal, the NAME statement cannot be used to specify logical terminal names for it.

Choosing the IMS master terminal

The IMS master terminal is the key control point for IMS online operations. Choosing a device as the master terminal provides the advantage of convenient data entry and output response.

The IMS master terminal acts as the control point for IMS online operations, data entry, and output response. Keep in mind that you might need a printed copy of many of the responses to commands, as well as a record of the system messages that are sent to the master terminal.

To choose the master terminal, specify MASTER after the logical terminal name on the NAME macro. The NAME macro must follow the TERMINAL macro that describes the terminal chosen.

To choose a secondary master terminal, specify SECONDARY after the logical terminal name on the NAME macro. Logging at the secondary master terminal can be turned on and off using the MSG keyword of the /SMCOPY command.

The /ASSIGN command can be used to switch the secondary master terminal to another destination such as a spool SYSOUT line group.

Choosing master terminal devices

Some restrictions apply for choosing the device for master and secondary terminals, as shown in the table that follows. Your choice for secondary master terminal depends on the expected amount of output and the user's requirement for promptness of printing.

Table 12. Device choices for master terminals.

Master terminal type	Choice of device type
Primary	3270[1]
	SLUTYPE1 (console)
	SLUTYPE2 (console)[2]
Secondary	328x[1]
	SLUTYPE1 (first component)
	SPOOL

Notes:

[1] If a 3270 device is specified as the master terminal, a 328x or SPOOL device must be specified as the secondary terminal.

[2] SLUTYPE2 requires a secondary master terminal. The NAME macro for the secondary terminal must be placed before the primary terminal.

Specifying the master terminal configuration

The following figure shows an example of coding for a master terminal on a 3270 local line.

```
TYPE  UNITYPE=(3270,LOCAL),MODEL=2
TERMINAL NAME=L3270A
NAME  (VT3270L1,MASTER)
TERMINAL NAME=L3270B,MODEL=1,OPTIONS=FORCRESP
NAME  VT3270L2
TERMINAL NAME=L3270C,TYPE=3270-A2,SIZE=(24,80)
NAME  VT3270L3
TERMINAL NAME=L3270D,TYPE=3270-A3,SIZE=(32,80)
NAME  VT3270L4
TERMINAL NAME=L3270E,TYPE=3270-A4,SIZE=(43,80)
NAME  VT3270L5
TERMINAL NAME=L3284A,UNIT=3284,PTRSIZE=132
NAME  (VT3270P3,SECONDARY)
```

The configuration for the master terminal is shown in the figure below:

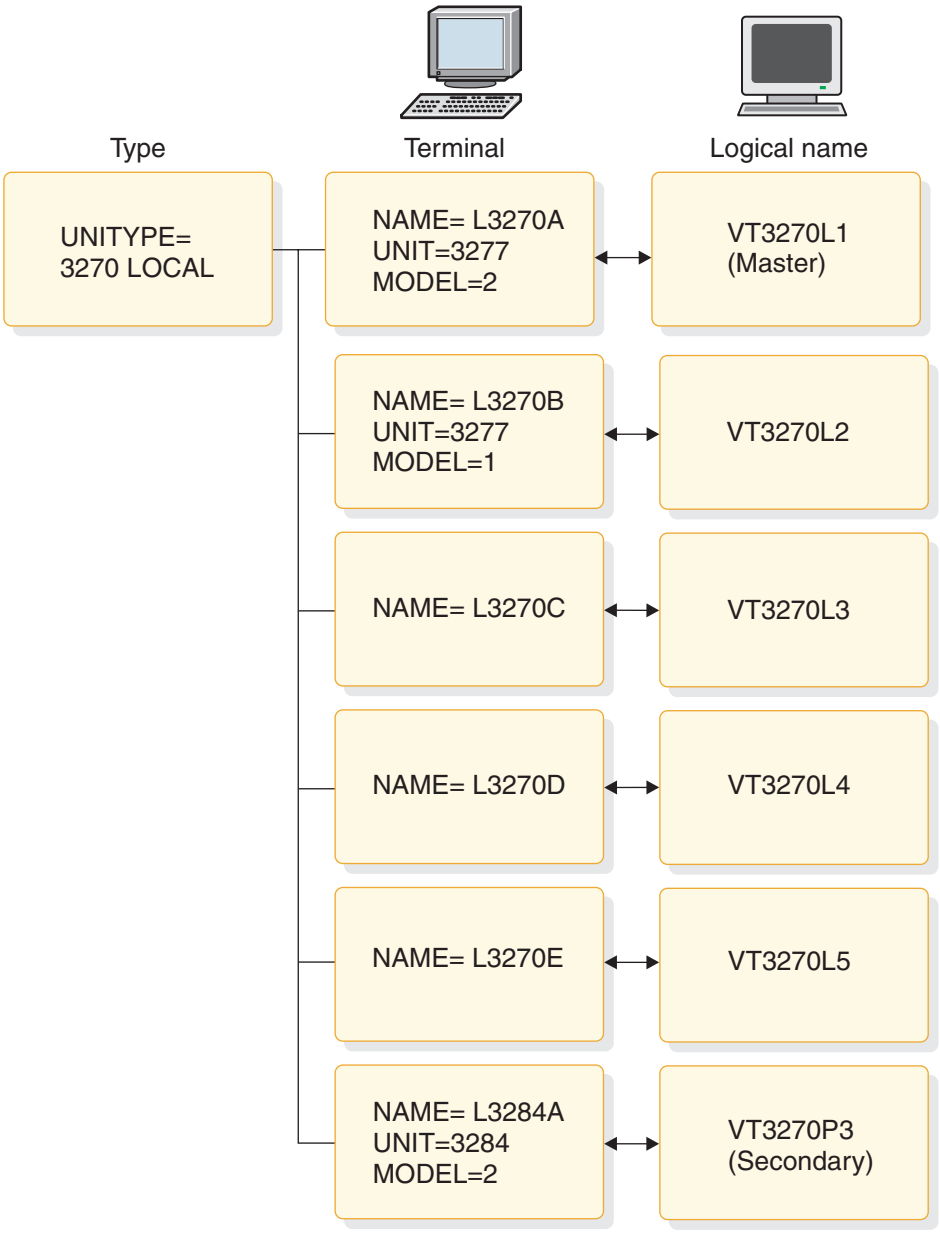


Figure 8. Configuration for a master terminal

Choosing the extent of secondary master logging

To automatically copy the entry and response of key system control commands, use the COPYLOG keyword on the COMM macro. The COPYLOG values that you specify cause all command activity that is issued by any terminal to be copied.

Unless terminals other than the master terminal are likely to issue many commands, specify COPYLOG=ALL. The values of NONE or NOMASTER are not recommended because the printed log of activity provides a valuable audit mechanism. Your choice of automatic copying does not affect the logging of IMS system messages to the secondary master terminal.

Specifying macros for Multiple Systems Coupling

You can use the MSLINK, MSNAME, MSPLINK, and NAME macros to define Multiple Systems Coupling (MSC).

If you plan to use MSC, you must prepare one set of MSC macro statements to describe the additional data communication facilities.

Although Intersystem Communication (ISC) is part of MSC, the macro statements that are used to define ISC are included in the VTAM macro set and not in the MSC macro set.

You must enter the macro sets in the order specified in your IMS system stage 1 system definition input deck. IMS issues stage 1 output warning messages and does not complete system definition if the macro sets are not entered in this order.

When you use MSC in your IMS system, the non-VTAM device macro sets (if used) must precede the MSC macro set in your IMS system definition stage 1 input deck. The VTAM macro set (if used) must occur after the MSC macro set in the stage 1 input deck. The following table lists macros for Multiple Systems Coupling (MSC).

Table 13. MSC data communication macros

Macro	Number of macros coded	Purpose
MSLINK	One for each logical system-to-system link	Defines a logical link to another system.
MSNAME	One for each remote and local system identification	Defines the link name block that relates to the MSLINK statement and the corresponding remote and local system identifications.
MSPLINK	One for each physical link	Defines a physical link to another system in an IMS MSC configuration.
NAME	One or more for each logical terminal that is referenced by your IMS system	Provides the logical terminal name for a physical terminal that is part of the remote system specified in the MSNAME statement.

You can use the UPDATE commands to alter any of the MSC values that have been defined in the macros.

Using multiple IMS systems at the same release level and environment

You can run multiple copies of IMS, with or without multiple systems coupling (MSC), in the same z/OS system and execute them concurrently. However, adding MSC allows communication and sharing of work between IMS systems. If your environment includes multiple IMS systems at the same release level, several considerations related to system definition apply.

When using multiple copies of IMS at the same release level and environment, the following requirements and conditions apply (regardless of the operating system):

- A unique subsystem identifier is required for each IMS DB/DC, DBCTL, or DCCTL control region. Specify this parameter (IMSID) in the IMS procedure for IMS, or in the DBC procedure for DBCTL, and in the dependent address space

procedures (IFP, BMP, and MPP) that override the value specified during system definition. The Parm Block member DFSPBxxx can also override the IMSID value specified during system definition. This value must not conflict with any subsystem identifier defined in the system, including other DB or DB/DC systems.

- Type 2 and Type 4 SVCs and the channel-end appendages can be shared.
- When using multiple copies of IMS systems at the same release level in the same z/OS system, you only need one copy of the Type 2 and Type 4 SVCs.
- All suffixed modules must be unique.

  Related reading: See the description of the SUFFIX= keyword of the IMSGEN macro in "IMSGEN macro" on page 351.
- DFSVNUCx modules are required to run different IMS control regions.
- You can store unique copies of module DFSVC000, and module DFSVNUCx for each IMS system in a partitioned data set (PDS), concatenated with and in front of IMS.SDFSRESL. Alternatively, you can have unique copies of DFSVC000 in a PDS as described, and separate other modules within IMS.SDFSRESL through the SUFFIX= parameter of the IMSGEN macro during system definition.
- Under the z/OS authorized program facility, authorize all libraries from which modules are to be loaded for the control region.
- IMS systems can use the same IMS.SDFSRESL and IMS.OPTIONS data sets regardless of the resources that are defined for each system with one exception. That exception is systems define with RSR and without RSR can not share the same IMS.SDFSRESL data set.
- If systems share IMS.SDFSRESL, you can store DFSMDA definitions in separate, authorized PDSs concatenated with IMS.SDFSRESL or use the IMSDALIB feature.
- The following IMS data sets must be unique and separately allocated to each IMS control region:

  IMS.QBLKS

  IMS.SHMSGx

  IMS.LGMSGx

  IMS.IMSMON (IMS Monitor) if used

  IMS.MSDBCP1 if used

  IMS.MSDBCP2 if used

  IMS.MSDBDUMP if used

  IMS.MSDBINIT if used

  IMS.RDS

  Online log data sets (minimum of 3)

  Write-ahead data sets (minimum of 1)

  To make these data sets unique for each IMS control region, you can use the NODE= keyword of the IMSGEN macro.
- Each IMS system must have its own terminal network and MSC network (if MSC is included).

Using multiple IMS systems at different release levels

If your environment includes multiple IMS systems at different release levels, several considerations related to system definition apply.

If you are running multiple copies of IMS at different release levels under the same operating system, the operating system must be at a version and release level that is required for the most recent release of IMS.

If you are installing different release levels of IMS in the same z/OS system, remember that running a system using the SVC from a lower-level system is not supported. For example, running an IMS Version 11 system using the SVC from IMS Version 10 is not supported. Similarly, running an IMS Version 11 system using the SVC from IMS Version 9 is not supported.

As of IMS Version 11, IMS uses a dynamic abend dump formatting module (DFSAFMX0). If you are running only versions of IMS that are Version 11 or later, you do not need to install the static abend dump formatting module (DFSAFMD0) on the host z/OS system. If you want to have IMS online dump formatting, and your z/OS system is running any jobs (either online or batch) for IMS Version 10 or earlier, the DFSAFMD0 module must be installed on the z/OS system.

For example, if you are running IMS Version 11 and Version 10, you must install the version of DFSAFMD0 that shipped with either IMS Version 10 or Version 11. For information about installing DFSAFMD0, see *IMS Version 11 System Administration*.

IMS uses a dynamic resource cleanup module (DFSMRC20). You do not need to install the static resource cleanup module (DFSMRCL0) on the host z/OS system. However, the most recent version of the DFSMRCL0 module must be installed on any z/OS system where you run any pre-IMS Version 9 jobs (either online or batch).

Including IMS ETO Support in the IMS system

IMS Extended Terminal Option Support (IMS ETO Support) enables VTAM terminals to log on to IMS TM, regardless of whether they were defined during the system definition process.

IMS Extended Terminal Option Support (IMS ETO Support) of IMS TM enables VTAM terminals to log on to IMS TM, even if they are not defined during the IMS system definition process. IMS TM dynamically builds the required control blocks and queues based on VTAM information and IMS skeleton definitions called descriptors. Without IMS ETO Support, adding, deleting, or changing terminals that are defined to IMS TM requires that your online system is terminated to incorporate the changes. Additionally, the stage 1 system definition input can become large if you are defining many terminals. IMS ETO Support is an optional feature; its installation is verified during IMS TM initialization.

Definitions: Terminals that are defined using system definition are *static* terminals. Terminals not defined through a system definition are *dynamic* terminals.

The IMS ETO Support feature applies to all VTAM terminals except for MSC VTAM, MTO terminals, and XRF surveillance links. These terminals, and terminals that are supported by non-VTAM access methods, still require an IMS system definition process to introduce changes. The number of terminals that you define dynamically is limited only by your resource constraints.

With IMS ETO Support, you can:

- Improve system availability by reducing the scheduled down time that is associated with adding or deleting VTAM terminals

- Improve IMS security by relating output message queues (LTERMs) to users rather than to terminals
- Reduce the number of macro statements that are required for static, system-defined VTAM terminals
- Reduce the amount of virtual storage used for IMS ETO Support terminals and users by allocating storage only when it is required
- Add new terminals and users without terminating and cold starting IMS TM

IMS ETO Support descriptors are not generated during large system definition (LGEN). A standard system definition, however, can build IMS ETO Support descriptors that support terminals that are currently generated statically, allowing an easy transition to the IMS ETO Support environment.

Dynamic allocation of VTAM terminals does not require the use of VTAM TYPE, TERMINAL, NAME, VTAMPOOL, or SUBPOOL macros, or the use of MSC remote NAME macros, for the system definition stage 1 input. Instead, you include IMS ETO Support with the ETOFEAT keyword on the IMSCTRL macro, which causes IMS TM to generate descriptors. IMS TM uses descriptors to build the required control blocks and queues that are associated with terminal definition. With IMS ETO Support terminals, configuration information from VTAM control blocks is dynamically merged with information from IMS ETO Support descriptors when a user signs on to IMS TM. Terminal control blocks are not built for IMS ETO Support terminals until an ACF/VTAM session is established between the terminal and IMS TM, or until a user structure is built. User control blocks are not built until a user signs on to a terminal.

Requirement: Static terminal definition is still required for:

MTO and secondary master terminals

MSC physical/logical links

XRF ISC Surveillance link

Non-VTAM devices

IMS ETO Support descriptors are described in "Enabling IMS ETO Support for ACF/VTAM terminals" on page 186.

Defining an FDBR region

Fast Database Recovery (FDBR) provides quick access to shared database resources in a sysplex environment that might otherwise be locked by a failed IMS until the failed IMS is restarted.

The FDRMBR= keyword in the DBC and IMS startup procedures identifies the FDBR region to the IMS PROCLIB data set with a two-digit suffix (FDRMBR=xx). The IMS PROCLIB data set member is DFSFDRxx.

The DFSFDRxx member specifies the options used by FDBR. The IMS PROCLIB data set can contain multiple instances of DFSFDRxx, but each DFSFDRxx member in the IMS PROCLIB data set must have a unique, two-digit suffix. FDBR, IMS, or DBC procedures identify which DFSFDRxx member to use.

The FDR procedure executes an FDBR region. The FDR procedure is like those used to define IMS systems (for example, the IMS procedure or the DBC procedure). FDBR is supported in DBCTL and DB/DC environments.

Two parameters, CSAPSB and DLIPSB, in the FDR procedure have a different result than when they are specified for the IMS and DBC procedures. When the FDR procedure specifies CSAPSB and DLIPSB, the sum of their values defines the PSB pool size. When PSB is also specified, the larger value (PSB or the sum of CSAPSB and DLIPSB) is used.

Both the 64-bit pools and non-resident 31-bit pools in IMS active systems and their associated FDBR and XRF alternate systems must have identical contents. Any parameters and values that you specify on the active IMS system must be the same on the alternate IMS system. For example, the parameters specified in the DFSVSMxx, DFSDFxxx, and DFSPBxxx IMS PROCLIB data set members must be the same, and the sizes of the PSB and DMB pools must be the same.

Related reference:

"DFSFDRxx member of the IMS PROCLIB data set" on page 702

"DBC procedure" on page 523

"IMS procedure" on page 560

"Parameter descriptions for IMS procedures" on page 448

"FDR procedure" on page 555

Configuration options for FDBR regions

There can be one FDBR region for each IMS subsystem in the data sharing group. The FDBR region should be on a separate z/OS system from the IMS that it tracks. This is to protect the FDBR region during a z/OS system failure on the system that contains the tracked IMS.

You can configure FDBR in several ways. The following figure shows FDBR regions placed with IMS subsystems. Each FDBR region tracks an IMS subsystem on a different z/OS system. This configuration has the advantage of low resource requirements.

In this configuration, if IMS1 fails, FDBR1 detects the failure and recovers any databases that IMS1 was using when it failed. If z/OS1 fails (with both IMS1 and FDBR4), FDBR1 recovers the database resources for IMS1, and FDBR4 can be established on another system until z/OS1 is restarted.

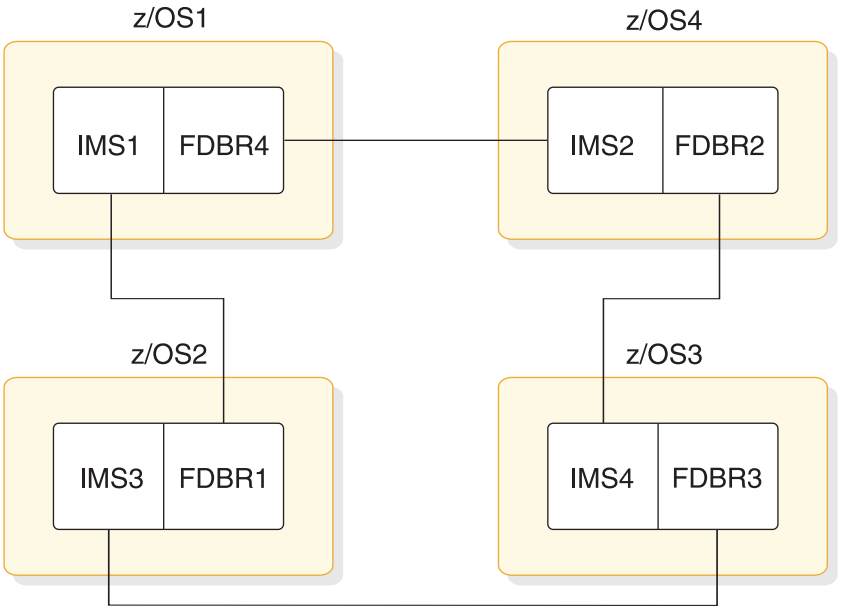


Figure 9. System configuration 1 for FDBR tracking of IMS subsystems

The following figure shows all FDBR regions on a separate z/OS system from the tracked IMS subsystems. This configuration requires an additional hardware resource (the extra z/OS system). If a z/OS system failure occurs on z/OS1, z/OS2, or z/OS3, none of the FDBR regions are affected.

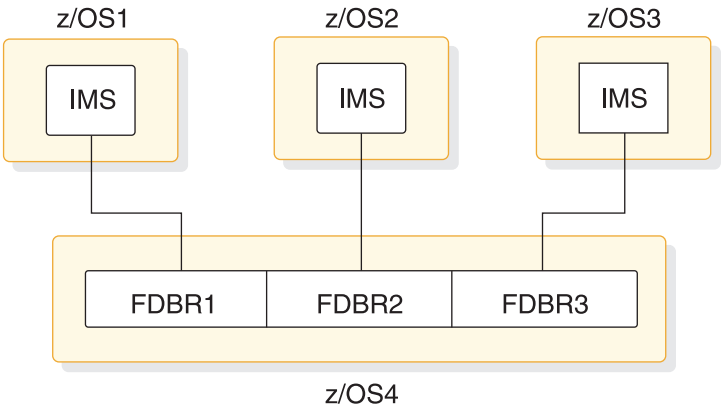


Figure 10. System configuration 2 for FDBR tracking of three IMS subsystems

Enabling an IMS subsystem for FDBR

To reduce the amount of time the sharing subsystems must wait, you can use Fast Database Recovery (FDBR) regions to enable DB/DC and DBCTL subsystems.

Enabling a DB/DC subsystem for FDBR

To enable a DB/DC subsystem for FDBR, specify the FDRMBR= parameter in the IMS procedure. The FDRMBR= parameter defines the DB/DC system as FDBR-capable.

Note the IMSID of the IMS system that FDBR is to track. Specify this IMSID in the control statement for the DFSFDRxx member of the IMS PROCLIB data set. Match the IMSID to the ID that is specified for the IMSID EXEC parameter.

Restrictions:

- If both FDRMBR= and HSBID= (XRF configuration) parameters are specified in the IMS procedure, the FDRMBR= parameter is ignored.
- For Fast Path systems using FDBR, the Fast Path buffer managers for the tracking system and the active system must be the same: both must be 64-bit or both must be 31-bit. Make sure that the tracking system and the active system have the same value for the FPBP64= parameter of the DFSDFxxx member of the IMS PROCLIB data set.

Enabling a DBCTL subsystem for FDBR

To enable a DBCTL subsystem for FDBR, specify the FDRMBR= parameter in the DBC procedure. The FDRMBR= parameter defines the DBCTL system as FDBR-capable.

Note the IMSID of the IMS system that FDBR is to track. Specify this IMSID in the control statement for the DFSFDRxx member of the IMS PROCLIB data set. Match the IMSID to the ID that is specified for the IMSID EXEC parameter.

Restrictions:

- If both FDRMBR= and DBRSE= (DBCTL standby configuration) parameters are specified in the DBC procedure, the FDRMBR= parameter is ignored.
- For systems using Fast Database Recovery (FDBR), the buffer manager of the tracking system must be the same as the buffer manager of the active system.

Related reference:

"DFSFDRxx member of the IMS PROCLIB data set" on page 702

"DBC procedure" on page 523

"IMS procedure" on page 560

"Parameter descriptions for IMS procedures" on page 448

"FASTPATH section of the DFSDFxxx member" on page 685

IMSplex requirements for FDBR

A Fast Database Recovery (FDBR) system must have the appropriate definitions to use the Common Service Layer (CSL), a Structured Call Interface (SCI), or the global online change function.

In an IMSplex with the CSL, you must define the FDBR system with access to CSL. An SCI must be defined on the operating system of the FDBR system.

If you use the global online change function in an IMSplex, any FDBR system that is tracking the active system must also be defined with global online change and it must use the same OLCSTAT data set as the active system. If the IMS system does not use RM services (defined with RMENV=N), the active IMS system must exclusively own the OLCSTAT data set. The OLCSTAT data set can include the IMSID of only the active IMS system.

Specifying enqueue and dequeue requirements

Protecting database integrity when the same database record is concurrently updated requires you to estimate the number of events to be stored (enqueued) and released (dequeued) in internal storage.

The online IMS system protects the integrity of the database when concurrently running programs are updating the same database record. A record of the inter-level update events is maintained for program isolation in tables defined in the control program storage. These are termed enqueue/dequeue tables and consist of 24-byte entries in a z/OS system. As programs reach synchronization points, the entries are freed and the storage can be reused. You need to estimate the number of events that might be recorded for concurrent execution of programs that might process against the same segment type.

The storage is allocated above the 16 MB line.

Specifying security options

Use the SECURITY macro or initialization EXEC parameters to implement general security decisions during system definition. Use the TERMINAL and TRANSACT macros to make resource-specific security decisions.

The security options that you specify must be a part of the overall security design. For a discussion about how to implement security for IMS, including what you need to do to use RACF and security exit routines, see IMS security (System Administration).

You make security choices at various stages of the IMS life cycle. During system definition, use the SECURITY macro or initialization EXEC parameters to implement general security decisions. You can also use the IMSGEN and COMM macros to define security options; however, the SECURITY macro definitions or the initialization EXEC parameters take precedence.

The TERMINAL and TRANSACT macros allow you to make resource-specific security decisions. You can also specify certain general security decisions using keyword parameters such as ISIS, RCF, SGN, TRN, and others in the startup procedures. You can make other security decisions using the DFSDCxxx members of the IMS PROCLIB data set.

The TYPE keyword of the SECURITY macro or the RCF or ISIS initialization EXEC parameters allow you to specify the security facilities (RACF or exit routines) that you want to use for security checking for dependent regions, transactions, commands, and signon.

To enable or disable support for mixed-case passwords, specify the optional PSWDC=M, R, or U parameter on the DBC, DCC, and IMS procedures. M specifies that IMS supports mixed-case passwords. R, which is the default, specifies that IMS uses whatever is defined for mixed-case passwords in RACF. U specifies that IMS forces all passwords to upper case. See “Parameter descriptions for IMS procedures” on page 448 for a description of the PSWDC parameter.

Including RSR in the IMS system

Remote Site Recovery (RSR) enables you to recover quickly from an interruption of computer services at an active (primary) site. RSR supports recovery of IMS DB full-function databases, IMS DB Fast Path DEDBs, IMS TM message queues, and the IMS TM telecommunications network.

The system definition process for RSR involves setting up a global service group (GSG) and a Transport Manager instance (TMI), both of which can be defined in several different places. This topic describes the system definition steps for RSR.

To enable RSR, specify:

1. RSRFEAT=RLT or RSRFEAT=DLT in the IMSCTRL macro.
2. A global service group (GSG) name. As the examples in Table 14 show, you can specify a GSG name for RSR in several places:
   - The IMSCTRL macro
   - The DLIBATCH procedure
   - The DBBBATCH procedure
   - The DFSRSRxx member of the IMS PROCLIB data set
   - On certain DBRC commands

   Table 14 describes how to specify the GSG name for RSR using a GSG name of *accounts*.

Table 14. Specifying the global service group (GSG) name for RSR

Procedure	Result
Specify GSGNAME=accounts in the IMSCTRL macro.	The default name *accounts* is specified.
In the DBBBATCH or DLIBATCH procedure, specify the GSGNAME statement as GSG=payroll, . This name must match the GSGNAME parameters.	The default name *accounts* is overridden.
In the DFSRSRxx member of the IMS PROCLIB data set, specify the GSGNAME parameter as GSGNAME (payroll). The parameters must match the statement in the DBBBATCH or DLIBATCH procedure.	The default parameters are overridden.
Enter DBRC commands using the GSG name payroll or the default name account: INIT.DB GSGNAME (payroll) INIT.DBDS GSGNAME (payroll) CHANGE.DB GSGNAME (payroll) CHANGE.DBDS GSGNAME (payroll)	Specifying the override names generates the same results as using the default name.
Enter ILS commands: START ILS(payroll)	Specifying the override names generates the same results as using the default name.

3. A TMI name. Table 15 shows where you can specify a TMI name:
   - The IMSCTRL macro
   - The DLIBATCH procedure
   - The DBBBATCH procedure
   - The DFSRSRxx member of the IMS PROCLIB data set
   - On certain Transport Manager Subsystem (TMS) commands
   - On certain isolated log sender commands

   In Table 15, ELX1 is an example of a TMI name. When you issue the TMS command, you can use the default or overridden TMI name.

Table 15. Specifying the Transport Manager Instance (TMI) name for RSR

Procedure	Result
Specify UNITYPE=(3270,LOCAL) in the TYPE macro.	The default name is specified.
Specify the TMINAME statement, TMI=ELX1, in the DBBBATCH or DLIBATCH procedure. This name must match TMINAME parameter.	The default parameter is overridden.

Table 15. Specifying the Transport Manager Instance (TMI) name for RSR (continued)

Procedure	Result
Specify the TMI parameter TMI=ELX1 in the DFSRSRxx member of the IMS PROCLIB data set. The parameter must match the statement in the DBBBATCH or DLIBATCH procedure.	The default parameter is overridden.
Enter TMS commands using the TMI name ELX1: SET INSTANCE(ELX1)	Specifying the override names generates the same results as using the default name.

After setting or overriding the GSG name and TMI name, you can use these names in various DBRC and TMS commands and isolated log sender commands.

A single active online subsystem environment does not require a separate system definition process for the tracking subsystem. A separate tracking subsystem system definition process is required if any of the following situations is true:

- More than one active subsystem must be tracked, and no one active subsystem system definition process defines all the resources needed by the tracking subsystem.

  This separate system definition requirement can be eliminated if the scope of the system definition of one of the active subsystems is broadened to define the additional resources.
- Batch DL/I jobs use databases that are not defined in the active subsystem.
- You want to eliminate database definitions for databases that are not tracked from the tracking subsystem.
- You want to reduce the data communications definitions for the tracking subsystem.

If your GSG name is *payroll* for your payroll system and your TMI name is *ELX1*, the DBRC, TMS, and ILS commands are:

- DBRC commands:

  ```
  INIT.GSG GSGNAME(accounts)
  INIT.DB DBD(db1) GSGNAME(payroll)
  CHANGE.DB DBD(db1) GSGNAME(payroll)
  INIT.DBDS DBD(dedb1) AREA(area1) GSGNAME(payroll)
  CHANGE.DBDS DBD(dedb1) AREA(area1) GSGNAME(payroll)
  ```
- TMS and ILS commands:

  ```
  SET INSTANCE(ELX1)
  START ILS(payroll)
  ```

If the RSR feature is installed and you do not want to use it, you must override RSR enablement by specifying RSR(NO) in the DFSRSRxx member of the IMS PROCLIB data set. DFSRSRxx includes other RSR-related parameters and overrides. For more information about the DFSRSRxx member of the IMS PROCLIB data set, see "DFSRSRxx member of the IMS PROCLIB data set" on page 720.

Chapter 4. Allocating and cataloging IMS system resources

When installing an IMS DB/DC environment, you must allocate and catalog IMS system libraries and online data sets.

Although this activity is done independently of the actual system definition processing, you need to coordinate this activity with several parts of system definition stage 1 input.

Planning for the definition of IMS data sets requires you to predict the direct access storage to be allocated based on your anticipated application program work load. Some of the data sets you estimate are affected by the design decisions made for system definition. Most of these data sets are automatically generated as DD statements within the members of IMS.PROCLIB.

Restriction: IMS supports partitioned data sets extended (PDSEs) for only these libraries: PGMLIB, SMPLTS, and SDFSJLIB.

Note: The External Subsystem Attach Facility (ESAF) supports PDSE load library data sets.

IMS system data sets for online change

Planning for the definition of IMS data sets requires you to predict the direct access storage to be allocated based on your anticipated application program work load. Some of the data sets you estimate are affected by the design decisions made for system definition. Most of these data sets are automatically generated as DD statements within the members of IMS.PROCLIB.

Restriction: IMS supports partitioned data sets extended (PDSEs) for only these libraries: PGMLIB, SMPLTS, and SDFSJLIB.

Note: The External Subsystem Attach Facility (ESAF) supports PDSE load library data sets.

In many installations, it is important that the online system be available during a large portion of the day. IMS provides two methods for adding, deleting, and replacing certain resources online without shutting down your IMS system. The methods are:

- Dynamic resource definition (DRD)
- Online change (OLC)

  An online change performs changes on a local IMS (called local online change) or to IMS systems in an IMSplex (called global online change). For an overview of performing online changes in these environments, see *IMS Version 11 System Administration*.

Adding, deleting, or changing IMS resources involves changes to the control blocks set up for these resources. If you use online change, making additions, deletions, or changes requires a MODBLKS system definition. Within a MODBLKS system definition, you specify changes to keyword parameters on the DATABASE, APPLCTN, TRANSACT, and RTCODE macro statements. When designing a DBCTL or DCCTL environment, use the information in this topic as it applies to

© Copyright IBM Corp. 1974, 2015

your system. For a DBCTL environment, no MFS facility exists, and the TRANSACT and RTCODE macros do not apply. A DCCTL environment has no database facilities; therefore, the DATABASE keyword does not apply. A MODBLKS system definition generates the control block members for resources that are to be added or changed online. These control blocks, stored in the IMS.MODBLKS data set, are used by the IMS control region and the MSC Verification utility when an online change to your IMS system is requested.

Before you can use online change, you must create three copies of each of the following libraries:

- IMS.MODBLKS—the library that contains the control blocks to support online change of databases, programs, transactions, and MFS formats
- IMS.ACBLIB—the library that contains database and program descriptors
- IMS.FORMAT—the library that contains your MFS maps produced by the MFS Language and Service utilities

These libraries are for the exclusive use of IMS offline functions and are called the *staging libraries*. Two copies are made of each library, producing data sets with a data set name suffixed with an A and a B, for example, IMS.FORMATA and IMS.FORMATB. These two copies of each library are used by the IMS online system.

Note: If you have multiple copies of the staging IMS.ACBLIB in an IMSplex, each copy of the IMS.ACBLIB must be identical.

When IMS IVP processing completes, the staging libraries and the IMS A libraries are identical, and the A libraries are referred to as the active libraries. IMS draws its execution information from the A libraries. The B libraries, which are not used at this time, are referred to as the inactive libraries.

Figure 11 on page 113 illustrates how libraries are used when you change your system online:

1. You apply changes to the staging libraries.
2. The staging libraries are then copied to the inactive (B) libraries using the Online Change utility.
3. Operator commands are issued to cause the B libraries to become the active libraries; the old active (A) libraries become the inactive libraries.

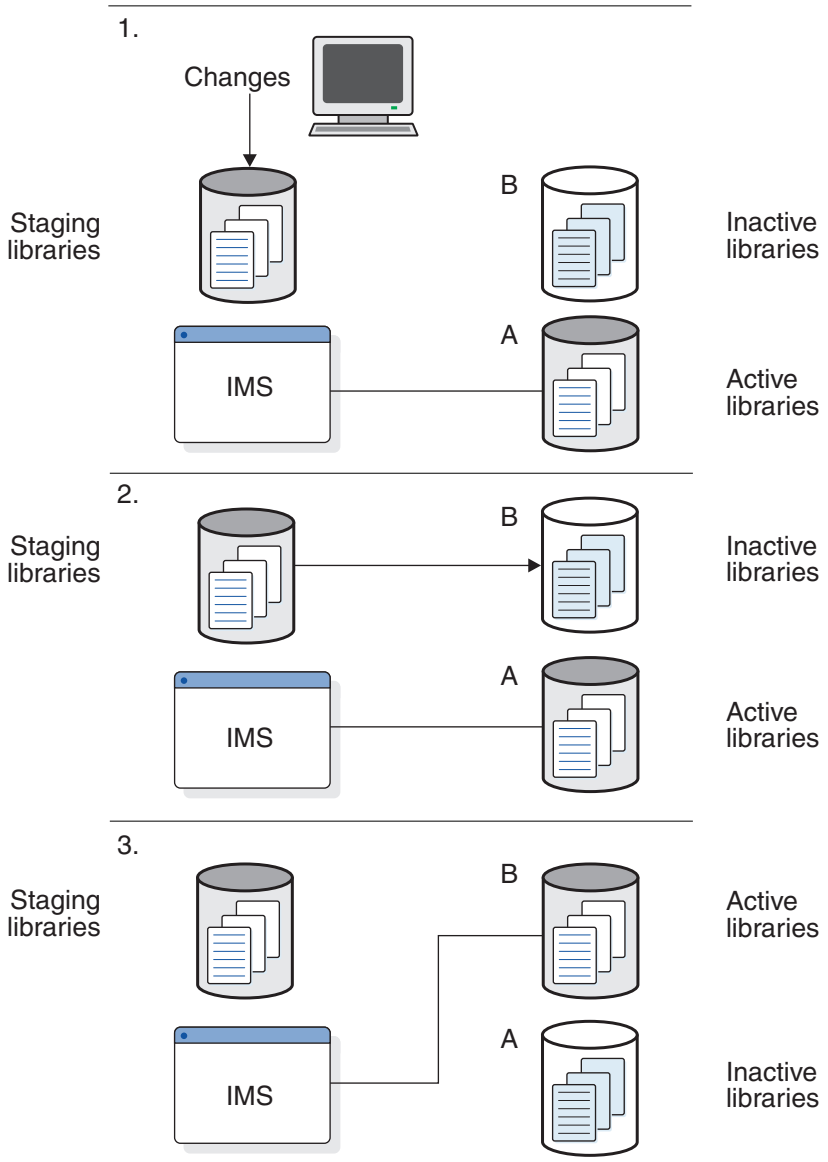


Figure 11. How libraries are used when you change your system online

The process above is repeated as necessary. When you choose to add, replace, or delete any IMS resources, you apply your changes to the offline staging libraries by running either:

- A MODBLKS system definition, if you have added, changed, or deleted application programs, full-function databases, DEDBs, or routing codes
- An ACBGEN, if you have added, changed, or deleted any databases or programs
- The MFS Language and Service utilities, if you have added, changed, or deleted any MFS format definitions

You can apply changes to IMS.FORMAT, IMS.ACBLIB, or IMS.MODBLKS independently or in combination. IMS.MODBLKS is changed by the MODBLKS system definition.

After the sequence of commands (/MODIFY for local online change or INITIATE OLC for global online change) has been issued to cause the previously inactive

libraries to become the active libraries, your previously active libraries now become the inactive libraries. They are not destroyed until they are overwritten by the next online change sequence. You can return to the inactive libraries if back up and recovery are necessary, or if an incorrect definition occurs during your online change.

IMS monitors which set of libraries is currently active. If local online change is enabled, this information is kept in a status data set, IMS.MODSTAT. If global online change is enabled, this information is kept in the IMSPLEX.OLCSTAT data set. All IMS systems in an IMSplex that share the IMSPLEX.OLCSTAT data set must specify the same value on the ACBSHR= parameter, which is specified on both the DFSCGxxx and DFSDFxxx PROCLIB data set members.

After an online change successfully completes, it persists across all types of IMS restarts. Additionally, the new resources can be easily maintained by running an SMP/E JCLIN against the Stage 1 output stream produced by your MODBLKS system definition to record the contents of the new system definition in your SMP/E control data set. This ensures that any maintenance applied to your IMS system is applied to the currently active IMS system. Do not manage the online change data sets with a migration/recall system that might recall the data set to a volume other than the one to which it was originally allocated. If you do so, IMS might be unable to warm start or emergency start the system.

Global resource serialization considerations

Include all IMS data set names in the global resource serialization SYSTEMS exclusion resource name lists (RNLs). Do not include the DBRC RECON data set or the OLDS and WADS names in the RESERVE conversion RNL.

JES considerations

If you use JES3, include all IMS data sets and databases in the RESDSN statement.

Initializing system data sets when not using online change

If you do not plan to use the online change function, you do not need to maintain the full set of staging, active, and inactive libraries. You need to manage only the staging libraries. You do not need to make copies for the active data sets, which would have the same contents.

You need to modify the JCL, generated in the IMS member of the IMS PROCLIB data set, for the online execution for the following ddnames:

- MODBLKSA
- MODBLKSB
- IMSACBA
- IMSACBB
- FORMATA
- FORMATB

1. For each of these DD statements, use a DSN parameter pointing to a staging library. For example, ddnames MODBLKSA and MODBLKSB use DSN=IMS.MODBLKS, and ddnames FORMATA and FORMATB use DSN=IMS.FORMAT. If you plan to use terminals in MFSTEST mode, the DD statements for the MFS library that contain the formats under test (ddnames IMSTFMTA and IMSTFMTB) have the staging library (IMS.FORMAT) concatenated to IMS.TFORMAT.

2. Initialize either the IMS.MODSTAT or IMSPLEX.OLCSTAT data set:
   a. If global online change is not enabled, initialize the IMS.MODSTAT data set appropriately, which is most conveniently done using the INITMOD procedure. This procedure initializes IMS.MODSTAT so that the ddnames with suffix A are set to be the active libraries.
   b. If global online change is enabled, initialize the IMSPLEX.OLCSTAT data set instead of the IMS.MODSTAT data set. When creating and allocating the OLCSTAT data set, make sure that an End of File (EOF) mark is placed at the beginning of the data set. Failure to do so can produce unpredictable results. To place an EOF mark at the beginning of the data set, use program IEBGENER. Do not use program IEFBR14 to allocate the OLCSTAT data set. IEFBR14 does not place an EOF mark at the start of the data set. An alternate method of allocating the OLCSTAT data set is to use the ALLOCATE function of ISPF utilities. The ALLOCATE function places an EOF mark correctly at the beginning of the data set.

IMS online data sets

An IMS system requires that you define several different types of online data sets. Use the IMS macro statements, including DATABASE, IMSCTF, and LINEGRP to define online data sets that meet your business requirements.

The online IMS data sets you need to define are shown in Table 16.

Table 16. List of online IMS data sets.

ddname	Data Set/Content	Prerequisite
Databases[1, 3]	Database data sets	DATABASE macro, ACBLIB content, use of DL/I address space
DFSESL	Optional use for external subsystems	APF library authorization
DFSOLPnn[1]	Online log data set (primary)	Archiving, dynamic allocation
DFSOLSnn[1]	Online log data set (dual)	Archiving, dual logging
DFSTRAnn	External Trace	
DFSWADSn[1]	WADS data set and spares	Dual WADS logging, dynamic allocation
DUMP	Diagnostic storage dump	Installation standards
FORMATA/B[2]	From staging Format library	MFSUTL procedure execution, Online Change utility
IEFRDER	IMS.JOBS (PDS)	Use of IMSRDR to start regions
IMSACBA/B	From staging IMS.ACBLIB	ACBGEN procedure execution, Online Change utility, use of DL/I address space
IMSMON[1]	IMS Monitor output	IMSCTF macro
IMSRDS	Restart data set	IMSCTF macro
IMSTFMTA/B[2]	Test message formats (PDS)	MFSTEST procedure execution, Online Change utility
LGMSG[2]	Long message	Message sizes
MATRIXA/B	From staging security tables	Online Change utility
MODBLKSA/B	From staging IMS.MODBLKS	Modified for online change
MODSTAT	Active library list	INITMOD procedure, Online Change utility

Table 16. List of online IMS data sets (continued).

ddname	Data Set/Content	Prerequisite
MSDBCP1[2, 3]	Fast Path MSDB checkpoint	Size of MSDBs
MSDBCP2[2, 3]	Fast Path MSDB checkpoint	Size of MSDBs
MSDBDUMP[2, 3]	Fast Path MSDB output	Size of MSDBs and operations
MSDBINIT[2, 3]	Fast Path MSDB input	MSDB Maintenance utility
PRINTDD	System output	Installation standards
PROCLIB	IMS.PROCLIB (PDS)	Procedures and initialization, use of DL/I address space
QBLKS[2]	Message queue blocks	Transaction traffic
SHMSG[2]	Short message	Message sizes
Spool names[2]	Spool output (IMS.SYSOnnn)	LINEGRP macro
STEPLIB	IMS.SDFSRESL (PDS)	APF library authorization

1. This ddname is not required if dynamic allocation macros are coded.
2. This ddname does not apply to DBCTL.
3. This ddname does not apply to DCCTL.

Allocating ACBLIB data sets

ACBLIB data sets contain the application control blocks (ACBs), which describe IMS applications, and data management blocks (DMBs), which describe databases and the applications that can access them.

You can allocate active and inactive ACBLIB data sets by using either JCL or DFSMDA members.

If you use the DFSMDA member for the allocation of IMSACB, you can dynamically allocate active and inactive ACBLIBs and remove the JCL DD statements for them from their IMS and DL/I procedures.

Do not mix allocation methods. For example, you cannot allocate one ACBLIB data set with JCL and the other with DFSMDA. When the IMS control region is initialized, IMS checks whether any of the ACBLIBs are allocated using JCL. If either IMSACBA or IMSACBB is allocated in the control region or DL/I region using JCL, both must be allocated using JCL. After IMS determines that one of the ACBLIB DD statements is present, no DFSMDA members are processed. If JCL is used for IMSACBA but not for IMSACBB, IMS terminates with abend U0023.

The IMSACBA and IMSACBB DD statements in the JCL for the FDR, DBC, DCC, DLISAS, and IMS procedures take precedence over the DFSMDA member. For dynamic allocation to occur, you would need to remove the IMSACBA and IMSACBB DD statements in the JCL for the FDR, DBC, DCC, DLISAS, and IMS procedures (and keep the DFSMDA member).

Allocating ACBLIB data sets using JCL

If you allocate ACBLIBs using JCL, IMS opens the active ACBLIB when the control region is initialized. The inactive ACBLIB is opened only after an online change procedure is initiated. However, because the JCL DD statement is in the IMS JCL

procedure, the inactive ACBLIB data set is always allocated. If you have defined a DL/I SAS, the DD statements for the IMS ACBLIBs are also in the DL/I JCL procedure.

Ensure that the same ACBLIB data sets are specified on the control region JCL and the DL/I region JCL. If an inconsistency occurs in the naming of the ACBLIB data sets specified on the control region JCL and the DL/I region JCL, the IMS control region initialization terminates with abend U0023.

Allocating ACBLIB data sets using DFSMDA

When you use DFSMDA members, the active ACBLIB is allocated when the control region is initialized. The inactive ACBLIB is not allocated until an online change procedure is initiated. However, because the inactive ACBLIB is not allocated during normal IMS processing, you can resize the inactive ACBLIB data sets, add data sets to the concatenation, or change data sets in the concatenation without bringing IMS down.

When you use DFSMDA, the IMS ACBLIB data sets are allocated with disposition of "SHR."

Two DFSMDA members must be defined:

- A DFSMDA member for the data sets in the IMSACBA concatenation.
- A DFSMDA member for the data sets in the IMSACBB concatenation.

The DFSMDA member names correspond to the DD statements for the ACBLIBs. For example, DFSMDA member IMSACBA contains all the data sets in the IMSACBA concatenation. DFSMDA member IMSACBB contains all the data sets in the IMSACBB concatenation.

If you use DFSMDA members and the dynamic allocation of the active ACBLIB fails, IMS terminates with abend U0023.

If DFSMDA members are used to allocate the ACBLIB data sets, and no DFSMDA member is found for the active ACBLIB, the IMS control region initialization terminates with abend U'0071' and message DFS0887A.

Related concepts:

Building the application control blocks (ACBGEN) (Database Administration)

Related reference:

"DFSMDA macro" on page 325
"IMS procedure" on page 560
"DLISAS procedure" on page 551
"DBC procedure" on page 523
"DCC procedure" on page 532
"FDR procedure" on page 555

Application Control Blocks Maintenance utility (System Utilities)

Database Description (DBD) Generation utility (System Utilities)

Program Specification Block (PSB) Generation utility (System Utilities)

Dynamically allocating the ACB staging library for ACBLIB member online change

Use the DFSMDA member with TYPE=IMSACB to dynamically allocate the ACBLIB staging library for use with ACBLIB member online change.

You can dynamically allocate the ACBLIB staging library using the DFSMDA member TYPE=IMSACB. By using TYPE=IMSACB, you only have to specify the data set name for the staging ACBLIB. A dynamic allocation member with the name of IMSACB is created for the staging ACBLIB.

Restriction: The staging ACBLIB cannot be concatenated.

Restriction: Do not create database MDA members with a DDNAME=IMSACB or with DISP=OLD.

IMS dynamically allocates the staging ACBLIB with DISP=SHR.

To ensure that the staging ACBLIB can be allocated by your IMS control region, you can use either of the following methods:

- Create a DFSMDA member for ACBLIB staging library. For example:

  ```
  DFSMDA TYPE=INITIAL
  DFSMDA TYPE=IMSACB,DSNAME=STAGING.LIBRARY
  DFSMDA TYPE=FINAL
  ```

- Add an IMSACB DD statement to your IMS procedure. For example:

  ```
  //IMSACB DD DSN=STAGING.LIBRARY,DISP=SHR
  ```

  If you use the IMSACB DD statement, it takes precedence over the DFSMDA member settings.

Related reference:

"DFSMDA macro" on page 325

"IMS procedure" on page 560

Dynamically allocating the IMSACBA and IMSACBB library data sets

Use the DFSMDA member with TYPE=IMSACBA and TYPE=IMSACBB to dynamically allocate the IMS IMSACBA and IMSACBB library data sets. Using DFSMDA to dynamically allocate IMSACBA and IMSACBB allows you to resize the inactive ACBLIB data sets, add data sets to the concatenation, or change data sets in the concatenation without bringing IMS down.

Restrictions:

- Do not create database MDA members with a DDNAME=IMSACB.
- Dynamic allocation statements for the ACBLIBA and ACBLIBB data sets can be combined in the same job. They cannot be combined with other statements to dynamically allocate any other IMS data set.

IMS dynamically allocates the IMS IMSACBA and IMSACBB library data sets with DISP=SHR.

To ensure that the IMS IMSACBA and IMSACBB library data sets can be dynamically allocated by your IMS control region, perform the following procedure:

- Create DFSMDA members for the IMSACBA and IMSACBB library data sets. The following is a sample of the DFSMDA statements:

```
DFSMDA TYPE=INITIAL
DFSMDA TYPE=IMSACBA
DFSMDA TYPE=DATASET,DSNAME=IMSTESTL.ACB1
DFSMDA TYPE=DATASET,DSNAME=IMSTESTL.ACB2
DFSMDA TYPE=FINAL
END

DFSMDA TYPE=INITIAL
DFSMDA TYPE=IMSACBB
DFSMDA TYPE=DATASET,DSNAME=IMSTESTL.ACB3
DFSMDA TYPE=DATASET,DSNAME=IMSTESTL.ACB4
DFSMDA TYPE=FINAL
END
```

- Remove the IMSACBA and IMSACBB DD statements from your FDR, DBC, DCC, DLISAS, and IMS procedures. Sample JCL of the statements you should remove is shown below:

```
//IMSACBA DD DSN=IMSTESTL.ACB1,DISP=SHR
    DD DSN=IMSTESTL.ACB2,DISP=SHR
//IMSACBB DD DSN=IMSTESTL.ACB3,DISP=SHR
    DD DSN=IMSTESTL.ACB4,DISP=SHR
```

  If you left in the IMSACBA and IMSACBB DD statements, they would take precedence over the DFSMDA member, and no dynamic allocation would occur.

Related reference:

"DFSMDA macro" on page 325

"IMS procedure" on page 560

"DLISAS procedure" on page 551

"DBC procedure" on page 523

"DCC procedure" on page 532

"FDR procedure" on page 555

Allocating log data sets

IMS creates logs of activity that are based on online executions. You must allocate sufficient data sets on DASD for the log data associated with your online activity, including online log data sets, system log data sets, and log write-ahead data sets.

For online IMS executions, allocate the IMS log to multiple data sets on DASD. Log records are initially written to an online log data set (OLDS), and then copied (archived) to the system log data set (SLDS). An SLDS can be on DASD or tape. Batch users can allocate a log (also known as the system log data set) to DASD or tape.

A data set containing log records that reflect completed operations not yet written to an online log data set is a write-ahead data set (WADS). For log write-ahead, provide the WADS. You can specify log write-ahead options in the DCLWA keyword of the TRANSACT macro. Log records created by IMS can be written to a WADS before the results of processing are externalized. Thus, a WADS contains a copy of committed log records in the online log data set buffers that have not yet been written to an OLDS.

You do not need DD statements for this log and the system output log (IEFRDER and IEFRDER2) for online IMS executions; you must remove the DD statements from your JCL. With batch, however, do not change the DD statements for logging. If you specify a secondary log in the IMSCTF macro, the ddnames for the primary

and secondary log data sets must be IEFRDER and IEFRDER2. The system rounds the BLKSIZE for IEFRDER and IEFRDER2 data sets to a doubleword boundary (a multiple of eight).

If you specify MONITOR in the IMSCTF macro, the IMSMON DD statement is used for both the DB and IMS Monitor data sets. You can allocate the IMSMON data set on DASD or tape (SL or SUL). You need a minimum of two buffers. If the block size you specify is smaller than the system-calculated minimum, the latter is used. The block size is rounded up to a doubleword boundary (a multiple of eight). You can specify the IMSMON data set through a JCL DD statement or a DFSMDA dynamic allocation member. If the block size is dynamically allocated, the default is 4096. If it is JCL allocated and DCB=BLKSIZE=NNNN is not specified in the IMSMON DD statement, the default block size is 1048 even if a larger block size is preallocated.

If you do not specify BLKSIZE, or if BLKSIZE=0 is coded in the JCL, the default for batch log data sets is LRECL=4092 and BLKSIZE=4096.

If you are using a 64-bit capable environment with OLDS blocksize that is a multiple of 4 K, and you allocate a tape device for this IMSMON, it must be a 64-bit capable device. If the tape device is not 64-bit capable, the monitor does not start and you receive the following error message: DFS2201I OPEN ERROR FOR IMSMON.

Allocating online log data sets (OLDSs)

OLDSs are required for online IMS execution. When allocating space for OLDS, consider how many data sets your system needs, the track size of the storage device, and whether you intend to use dual logging.

- The maximum number of OLDSs is 100.
- Because the OLDS can be required for restart, it cannot be a temporary data set.

  Define the initial set of OLDSs to be acquired by restart initialization in the OLDSDEF control statement in the DFSVSMxx member of IMS.PROCLIB. You can dynamically allocate this set of OLDSs, or specify them through DD statements.
- Single or dual online logs can be specified by using the OLDSDEF control statement in the DFSVSMxx member of IMS.PROCLIB.
- The only specific naming requirement for an OLDS is that it be unique.

  However, ddnames for the OLDS must be of the form DFSOLP*nn* for primary OLDS, and DFSOLS*nn* for secondary OLDS, where *nn* can be any numeric value.
- An OLDS must be a single volume and extent, and at least three data sets must be allocated. However, if an OLDS is to be stopped and started with /STA and /STO commands, DFSMDA members must exist with IMS.SDFSRESL for each such data set. You must provide DFSMDA members for all OLDSs.
- The minimum number of buffers that you can specify is 2, with a maximum of 9,999. Specify the number of OLDS buffers on the OLDSDEF control statement in the DFSVSMxx member of PROCLIB data set. The default number of buffers is 5.
- DASD space for each OLDS must be contiguous, and secondary extents are not permitted. Pairs of OLDSs (primary and secondary) must have the same space allocation.

If you use dual logging, allocate at least six data sets with corresponding numeric values, with a maximum of 200 possible. You can dynamically allocate an

additional OLDS using the /START OLDS master terminal operator command. If you use dynamic allocation you should preallocate and catalog candidate data sets, and specify data set names using the dynamic allocation macro, DFSMDA. You must provide a DFSMDA member for each OLDS.

Recommendation: Consider assigning enough OLDS space to each OLDS so that it almost fills an SLDS volume at the end of each archive process. If the size of an OLDS exceeds the capacity of a tape volume, additional tape mounts are required. If the IMS online system is active, the Log Archive utility attempts to access the OLDS while the OLDS is still allocated to the IMS online system. You can use DISP=OLD only if you can allocate sufficient OLDS space to hold all the log records generated by the online system between startup and shutdown. Archiving must then be performed while the online system is not active.

OLDS block sizes must be equal. Predefine the OLDS with block size, logical record length (LRECL), and record format specified at definition time. The OLDS LRECL must equal the OLDS block size minus 4 bytes (LRECL=BLKSIZE–4). The OLDS record format must be variable blocked (VB), and block size must meet the following requirements:

- It must be a minimum of 6 KB and a multiple of 2048. If IMS is running in z/Architecture® mode, log buffer storage is only fixed above 2 GB if the block size is a multiple of 4096.
- It must not exceed a maximum of 30720 bytes, because this is the largest multiple of 2048 supported by BSAM.
- At a minimum, its length must be the same as the length of the largest log record, plus 20 bytes. The largest log record length is a function of the block size for the message queue data sets, the EMH terminal buffer size, and the DEDB control interval size.

The main factor that determines OLDS block size is the track size of the OLDS devices. The OLDS block size cannot exceed the OLDS track size.

The WADS temporarily holds partially filled OLDS buffers, which means that only full OLDS buffers are written to the OLDS. Therefore, choose a large OLDS block size to use DASD space more efficiently.

The following table provides some recommended OLDS block sizes (in multiples of 2048) that maximize DASD space utilization for several DASD devices. Table 17 also provides information about blocks per track and bytes of log data per track.

Table 17. Recommended OLDS block sizes

Device type	OLDS block size	Blocks per track	Bytes of log data per track
3380	22,528	2	45,056
3390	26,624	2	53,248
3390	18,432	3	55,296
9340	22,528	2	45,056

The following table provides recommended OLDS block sizes for device types 3380 and 3390 if IMS is running in z/Architecture mode, in which the OLDS block sizes must be multiples of 4096. This table also provides information about blocks per track and bytes of log data per track.

Table 18. Recommended OLDS block sizes for 3380 and 3390 device types in z/Architecture mode

Device type	OLDS block size	Blocks per track	Bytes of log data per track
3380	20,480	2	40,960
3380	12,288	3	36,864
3390	24,576	2	49,152
3390	16,384	3	49,152

Log initialization ensures that the block size specified in the OLDS data set control block (DSCB) data set is large enough to handle the maximum length log record. If the block size is too small, an abend can occur.

To change the OLDS block size, archive all OLDS data, and scratch and reallocate each OLDSs to ensure that all OLDS block sizes remain identical. Also use the DELETE.LOG DBRC command to remove the OLDS from the DBRC RECON data set.

Formatting newly initialized (reinitialized) volumes for an OLDS

Before using a newly initialized or reinitialized volume for an online log data set (OLDS), you must format the volume or space occupied by the OLDS. There are several techniques for formatting the volume or space.

Attention: If a newly initialized (or reinitialized) volume is to contain an OLDS, before it can be used in the online production system, you must format the volume or space occupied by the OLDS. If it is not formatted, or if the block size of the new OLDS is not the same as the existing OLDSs, severe performance degradation and excessive device and channel utilization can be expected until the OLDS is filled once. This problem is noticeable during emergency restart and XRF tracking/takeover.

When formatting the volume or space, the blocks must have the same size as that specified in your OLDS definition. Each block must start with a halfword that is filled with the block size (in hexadecimal), and the remainder of each block must be filled with zeros.

Although IMS does not provide a formatting utility, many techniques for formatting are available, such as:

- Copy an existing OLDS (of the same size) into the new OLDS.
- Copy an existing volume into the new volume, rename the OLDS to a new name, and delete unrelated VTOC entries.
- Use another IMS subsystem to fill the OLDS (turn on all traces to the log, and issue checkpoint commands until the OLDS is filled).
- Write your own program to write at least 1 byte of data in each track on the volume, or to fill the OLDS with the maximum number of LRECL blocks.
- Use the z/OS IEBDG utility to format the OLDS.

Related concepts:

z/OS: DFSMSdfp IEBDG test data generator utility

Allocating the write-ahead data set (WADS)

The write-ahead data set (WADS) is a DASD data set containing a copy of log records reflecting committed operations in the OLDS buffers that have not yet been written to the OLDS. WADS space is continually reused after the records it contains are written to the OLDS. The recommended size for your WADS can vary by storage device.

You can specify this required data set by JCL, or you can dynamically allocate it. The WADS ddname is DFSWADS*n*, where *n* is a number 0 - 9. If you define multiple instances of a WADS, they are used in the WADS DD statement suffix sequence as indicated by the *n* in the ddname. Preallocate the WADS on DASD supporting Count Key Data (CKD) architecture and then format the WADS with a /NRE or /ERE FORMAT WA command at least once before it is used. Each WADS must be on the same device type and have the same space allocation.

Recommendation: For increased performance, allocate each WADS on a minimally used device and data path.

Dual WADSs provide an alternative source of input to the OLDS in case an error occurs on one WADS while IMS uses it to close the OLDS. You can use single or dual WADS logging with either single or dual OLDS logging. Specify single or dual WADS logging by using the WADS=S or WADS=D parameter. Define at least two WADSs before enabling dual WADS logging.

You can define up to 10 WADSs to use as spares. IMS automatically switches to a spare WADS if the current WADS becomes unusable after an error occurs. If a write error occurs, logging to the WADS continues if there is at least one WADS is available (for single logging) or two WADSs are available (for dual logging). If you use dual logging, define at least three WADSs so that IMS can activate the spare if a write error occurs. For additional resiliency, define each WADS on a different hardware device.

Define the initial set of WADSs to be acquired by restart initialization in the WADSDEF control statement in the DFSVSMxx member of the IMS PROCLIB data set.

Tracks in the WADS data set are used in groups. A track group can contain data from only one OLDS block at any given time. The size of a WADS track group depends on the OLDS block size and the WADS segment size. IMS sets the WADS segment size to 4 KB if the operating system can run in 64-bit mode and the OLDS block size is divisible by 4 KB, otherwise IMS sets the segment size to 2 KB. Use the following formula to calculate the size of a WADS track group:

```
Number of tracks in a WADS track group = (OLDS block size/WADS segment size) + 1
```

The WADS must be large enough to hold at least one WADS track group for each OLDS block that fits on an OLDS track. Use the WADS track group size (or the number of tracks in a WADS track group) to calculate the recommended minimum WADS sizes using the following formula:

```
Minimum WADS size (in tracks) = (number of tracks in a WADS track group) x
 (number of OLDS blocks per track)
```

Table 19 on page 124 provides the calculated recommended minimum WADS sizes for 32-bit mode operating systems based on the OLDS block size and on the DASD device type being used.

Table 19. Recommended minimum WADS sizes for 32-bit mode

OLDS Block Size	WADS with OLDS on 3380	WADS with OLDS on 3390
6 KB	28 tracks or 2 cylinders	36 tracks or 3 cylinders
8 KB	25 tracks or 2 cylinders	30 tracks or 2 cylinders
10 KB	24 tracks or 2 cylinders	30 tracks or 2 cylinders
12 KB	21 tracks or 2 cylinders	28 tracks or 2 cylinders
14 KB	24 tracks or 2 cylinders	24 tracks or 2 cylinders
16 KB	18 tracks or 2 cylinders	27 tracks or 2 cylinders
18 KB	20 tracks or 2 cylinders	30 tracks or 2 cylinders
20 KB	22 tracks or 2 cylinders	22 tracks or 2 cylinders
22 KB	24 tracks or 2 cylinders	24 tracks or 2 cylinders
24 KB	13 tracks or 1 cylinder	26 tracks or 2 cylinders
26 KB	14 tracks or 1 cylinder	28 tracks or 2 cylinders
28 KB	15 tracks or 1 cylinder	15 tracks or 1 cylinder
30 KB	16 tracks or 2 cylinders	16 tracks or 2 cylinders

Table 20 provides the calculated recommended minimum WADS sizes for 64-bit mode operating systems based on the OLDS block size and on the DASD device type being used.

Table 20. Recommended minimum WADS sizes for 64-bit mode

OLDS Block Size	WADS with OLDS on 3380	WADS with OLDS on 3390
8 KB	15 tracks or 1 cylinders	18 tracks or 2 cylinders
12 KB	12 tracks or 1 cylinders	16 tracks or 2 cylinders
16 KB	10 tracks or 1 cylinders	15 tracks or 1 cylinders
20 KB	12 tracks or 1 cylinders	12 tracks or 1 cylinders
24 KB	7 tracks or 1 cylinders	14 tracks or 1 cylinders
28 KB	8 tracks or 1 cylinders	8 tracks or 1 cylinders

The maximum number of WADS tracks that are ever used is calculated by the following formula:

```
Maximum number of tracks =
     ([OLDS block size/WADS segment size] + 1) x (number of OLDS buffers)
```

The maximum amount of space that is used for each WADS is large enough to contain 9999 OLDS buffers.

Use the range of the recommended minimum size from the table (or by using the minimum WADS size formula) for allocating a WADS. Obtain the maximum size by using the maximum number of tracks formula. Most installations find that four to five cylinders are appropriate.

Allocating the system log data set (SLDS)

A system log data set (SLDS) is the single or dual log data set that is created by IMS batch execution on tape or DASD. The SLDS is dynamically allocated to the

address space if needed for restart. Define the SLDS (with ddname IMSLOGR) through the dynamic allocation macro DFSMDA.

An SLDS is also one of the output data sets created when the Log Archive utility is used to archive an OLDS. The Log Archive utility can also be used to copy a batch log (SLDS) from DASD to tape (or another DASD data set).

When the Log Archive utility is used to archive an OLDS to tape, you can force the primary and secondary SLDS volumes to contain the same data by specifying the number of log blocks per volume. SLDS block size can be different from the block size of the OLDSs being archived, but the block size of the primary SLDS must be the same as the secondary SLDS block size.

If 3480 tape drives are used for logging, they are forced to run in tape-write-immediate mode.

If SMS-managed generation data sets (GDS) are used for the SLDS, certain error conditions might cause the SLDS to be overwritten. For batch allocations of SMS GDS, the data set is cataloged in deferred roll-in status at step allocation time, and rolled-in at step deallocation time. If a power failure occurs after the SLDS has been written and closed, but before step deallocation, IMS assumes the SLDS is valid; however, SMS does RECLAIM processing at the next allocation. RECLAIM processing means that a data set in deferred roll-in status is reused. For DISP=NEW, the new data would overwrite the existing data.

Allocating the restart data set

The restart data set contains information that enables IMS to determine from which checkpoint to restart the system.

IMS uses the restart data set during IMS restart to determine from which checkpoint to restart the system. The restart data set is required. A minimum of five tracks must be allocated to the restart data set because it contains the checkpoint-ID table and other control information.

Setting the TOD clock during IPL

The time-of-day (TOD) clock setting is critical to IMS log integrity and the proper functioning of database recovery. Use caution when setting the TOD clock during system IPL to avoid database integrity and recovery problems.

Attention: Setting the Greenwich mean time (GMT) clock value back at IPL time can cause severe database integrity and recovery problems. Issuing a SET CLOCK command to change the local time, for example at the end of daylight saving time, has no effects on IMS recoverability.

The time-of-day (TOD) clock setting is critical to IMS log integrity and the proper functioning of database recovery, IMS restart, and XRF tracking/takeover. Never set the TOD clock to a time earlier than the immediate prior shutdown or failure without taking actions to reset the recovery base. You can reset the recovery base by invalidating the existing log, image copy, and change accumulation data sets. If the TOD clock must be set to a time earlier than the previous shutdown or failure, you must complete the following procedure to reset the recovery base:

1. Reallocate a different block size for the OLDS data sets.
2. Reinitialize the DBRC RECON data set.
3. Make image copies of all database data sets.

4. Cold start IMS.

Issuing a SET CLOCK command does not reset the TOD clock. You can set the TOD clock only at system IPL either by changing the setting of the sysplex timer (external time reference or ETR); or by replying to the IPL prompts for setting the clock with the GMT option. Therefore, you do not need to reset the recovery base if you issue a SET CLOCK command when the TOD setting must be changed for daylight saving time (for example).

Message queue data set allocation in DB/DC and DCCTL environments

The amount of DASD space allocated to the message queue data sets depends on how many transaction codes and logical terminal names you specify during system definition, and how many short and long messages are to be held by the system during any period.

You can change the amount of direct access storage space allocated to the message queue data sets before a cold start of IMS. Reallocation of the message queue data sets with a warm start requires the use of the FORMAT and BUILDQ parameters with either the /NRESTART or /ERESTART command. Allocating less space (than in the previous execution) before an /NRESTART or /ERESTART BUILDQ might cause the restart to abend.

You can allocate up to 10 data sets for the long message queue and 10 data sets for the short message queue. Each data set requires an additional DD statement.

Ensure that all data sets of a given message queue type are the same size. If the data sets have different sizes, the smallest size is used for all. This can reduce the available space of a message queue.

If you change the number of data sets, or if you rename any of the message queue data sets, you must restart the system.

The Queue Manager Concurrent I/O component provides multiple normal short and long message queue data sets. This facility is optional, and you can invoke it by providing 1 - 10 DD cards for the normal short and long message queue data sets.

In order to provide Queue Manager Concurrent I/O:

- IMS initialization allows multiple physical data sets to be viewed as one logical data set.
- You can view both the physical data set and the logical structure.

For single-mode transactions, a message space is available as soon as it is processed by an application program (for example, the program terminates normally or requests the next message).

For multiple-mode transactions, the message spaces are available only after the application program that processes them terminates normally or takes a checkpoint.

For logical terminal messages, a given message space is made available after the successful receipt of this message by the terminal device.

The number of records to be reserved in each data set to allow the system to shut down depends on message throughput and the number of regions scheduled.

Recommendations:

- If you use emergency restart procedures using BLDQ, reallocate logical record size and data set spaces carefully. Allocate enough space to the data set to hold log records relating to message queue activity occurring between checkpoints. The BLDQ procedure always restores the message queue entries to the relative position in the respective queue data sets at the time saved. If the logical record or data set size is decreased, you might be unable to restart in some situations.
- Do not manage the QBLKS, SHMSG, and LGMSG queue data sets with a migration/recall system that might recall the data sets to a volume other than the one to which they were originally allocated. If you do so, IMS might be unable to warm start or emergency start the system.
- Secondary allocation is not allowed for message queue data sets.

The normal short and long message queues allow only one DD card for each.

To prevent message queue overflow due to looping application programs, the Queue Manager and the Queue Space Notification exit routine (DFSQSPC0) monitor the number of buffers assigned to each unit of work (UOW). When a UOW exceeds its buffer limit, the Queue Space Notification exit routine takes action to prevent further inserts by that UOW, and an 'A7' status code is returned to the application program.

Migrating IMS messages to a different release or configuration of IMS

You can use the IBM IMS Queue Control Facility for z/OS (QCF) to migrate messages from one release of IMS to another. QCF is also used after a cold start to migrate messages either to a new configuration or after maintenance.

Monitoring and controlling high message queue users with the IBM IMS Queue Control Facility for z/OS

You can monitor and take action to prevent high message queue users. Use the User Queue Space Notification exit routine (IQMRH0 linked as DFSQMRH0) and the QCF ISPF interface. Using this exit routine helps prevent queue usage from reaching critical thresholds.

Related Reading: For more information about the IBM IMS Queue Control Facility for z/OS, see *IMS Queue Control Facility for z/OS User's Guide*.

Additional restrictions in an XRF environment

Message queue data sets in an XRF environment have two additional restrictions:

- The number of data sets allocated for the short and long message queues must be the same on the primary and the alternate subsystems.
- The names for the message queue data sets must be different on the primary and alternate subsystems. These data sets cannot be shared between subsystems.

Message queue data set secondary allocation

Several factors affect the usage of IMS.QBLKS records. For example, the requirement for multiple temporary destinations when using program isolation can

cause an increase in the space requirements. The space requirements for the IMS.QBLKS data set depend on your installation.

The amount of direct access space required for the IMS.SHMSG and IMS.LGMSG data sets is dependent on message throughput. The disk space is reusable as soon as the message to which it was allocated has been processed and it is no longer required for recovery.

Message queue data set space should be allocated in terms of contiguous cylinders for most efficient operation. Secondary allocation is ignored unless the secondary space has been preallocated (that is, multiple volume data set with preallocated space on both volumes). Allocate each message queue data set on a separate direct access device or next to each other, with IMS.QBLKS in the center, on the same direct access device.

Message queue data set allocation restrictions

If emergency restart procedures using BUILDQ are to be used, you must carefully reallocate logical record and data set spaces. The BUILDQ procedure always restores the message queue entries to the relative position in the respective queue data sets they had at the time they were saved. If the logical record or data set size has been decreased, it might be impossible to perform the restart.

Allocating OSAM data sets

To allocate OSAM (overflow sequential access method) single or multiple volumes, use JCL when the data set is loaded using the SPACE parameter.

If your installation control of DASD storage and volumes is such that the OSAM data sets must be reserved ahead of time, or if you decide that a message queue data set requires more than one volume, the OSAM data sets can be preallocated.

Restrictions:

- Do not specify DCB parameters.
- If the data set is to be expanded beyond the preallocate space, a secondary quantity must be specified during preallocation. Queue data sets are constrained to only the space that is preallocated.

If you are preallocating a multiple-volume data set, allocate extents on all volumes to be used. The end of the data set must be correctly indicated in the data set control block (DSCB) on the last volume.

The suggested method is to use the IEFBR14 utility once for each volume on which space is required; do not just use the IEFBR14 utility and specify a DD statement for a multivolume data set. This action only puts an extent on the first volume and does not indicate which volume is the last volume of the data set.

You can allocate OSAM data sets to take advantage of z/OS DFSMS support for large format sequential data sets, which can exceed more than 65 535 tracks per volume. This allows more data to be stored on fewer volumes, helping to minimize sequential data sets that grow large and span many volumes of storage hardware.

To enable support for DFSMS large sequential data sets, specify DSNTYPE=LARGE in the JCL that allocates the OSAM data sets, and bring the data sets online as follows:

- If the new data sets are to be used for an OLDS or a message queue, cold start IMS.
- If the new data sets are to be used for a database, use a database reorganization process (unload and reload) to bring the new data sets online.

"Sample OSAM data set allocation JCL" displays the recommended OSAM data set allocation JCL. If you are allocating a large sequential OSAM data set, see "Sample JCL to allocate a large sequential OSAM data set."

If the OSAM data sets must be cataloged, use IEHPROGM or Access Method Services (AMS) to ensure that all volumes are included in the catalog entry.

Attention: Do not reuse multivolume OSAM data set extents without scratching and reallocating the space first. Otherwise, an invalid end-of-file mark can be left in the DSCB of the last volume of the data set, which causes an embedded EOF mark somewhere in the middle of the data set.

Sample OSAM data set allocation JCL

```
//OSAMALL  JOB
//S1      EXEC PGM=IEFBR14
//EXTENT1   DD DSNAME=OSAM.SPACE,DISP=(,KEEP),
//             UNIT=SYSDA,VOLSER=AAAAAA,
//             SPACE=(CYL,(10,5))
//S2      EXEC PGM=IEFBR14
//EXTENT2   DD DSNAME=OSAM.SPACE,DISP=(,KEEP),
//             UNIT=SYSDA,VOLSER=BBBBBB,
//             SPACE=(CYL,(15,5))
⋮
//LAST    EXEC PGM=IEFBR14
//EXTENTL   DD DSNAME=OSAM.SPACE,DISP=(,KEEP),
//             UNIT=SYSDA,VOLSER=LLLLLL,
//             SPACE=(CYL,(15,5))
```

Sample JCL to allocate a large sequential OSAM data set

```
//OSAMALBG   JOB
//S1         EXEC  PGM=IEFBR14
//EXTENT1  DD  VOL=SER=AAAAAA,SPACE=(CYL,(20,5)),UNIT=3390,
//           DSN=OSAM.LARGE.SPACE,DISP=(,KEEP),DSNTYPE=LARGE
```

Related tasks:

"Defining large sequential data sets" on page 141

Related reference:

z/OS: EAV enhancements

Allocating VSAM data sets

To define Virtual Storage Access Method (VSAM) data sets, use the z/OS AMS DEFINE CLUSTER command.

VSAM database data sets are defined by an AMS DEFINE CLUSTER command.

Related reading: This command and all its parameters are described in *z/OS DFSMS Access Method Services for Catalogs*.

Sharing of VSAM data sets is specified by the DEFINE CLUSTER SHAREOPTIONS keyword. IMS VSAM databases that use data sharing must be

defined with at least SHAREOPTIONS (3,3). This allows IMS to access the VSAM VSI so that any extensions to the VSAM data set are known by all IMS sharing systems.

VSAM data sets opened for update by XRF-capable IMS online systems must also use at least SHAREOPTIONS (3,3), in order for extensions to the VSAM data set to be tracked by the alternate system. Because VSAM data sets opened for input are not extended by VSAM, the VSAM VSI is not required. SHAREOPTIONS (3,3) can be used even if the online system is XRF capable. SHAREOPTIONS (3,3) is not necessary for Fast Path DEDBs; SHAREOPTIONS (2,3) can be used for this environment.

You can enable VSAM data sets to use extended address volumes (EAVs) that are available in z/OS V1.12 or later. To enable a VSAM data set to use EAVs, specify an EAV volume on the VOLSER parameter when you allocate the data set. In addition, you can specify the attribute EATTR to indicate whether the data set supports extended attributes.

Allocating data sets in extended address volumes

IMS supports the allocation of only VSAM data sets in the extended address space (EAS) on the extended address volume (EAV).

Resource definition data sets

Resource definition data sets (RDDSs) contain resource definitions and resource descriptor definitions, which can be imported from or exported to IMS systems to define a standard set of attributes for resources.

Dynamic resource definition (DRD) uses the resource definition data set (RDDS) to contain the resource definitions and *resource descriptor* definitions (templates that define standard attributes for resources) for a single IMS system. IMS systems can export resources that are either defined by the batch system definition process, or created or updated dynamically, to an RDDS. These resources can then be imported from the RDDS into an IMS system during cold start processing.

The IVP program that builds a sample IMS system includes support for DRD; using the IVP, you can define and allocate two BSAM data sets which can be used as resource definition data sets (RDDSs) to save IMS resource definitions. For information about the IVP sample jobs, see *IMS Version 11 Installation*.

The definitions in an RDDS are in a binary format. RDDSs are not supported in IMS FDBR regions or on IMS RSR tracking systems.

Recommendation: The RDDS is a BSAM data set that is defined with the RDDSDSN= parameter in the DYNAMIC_RESOURCES section of the DFSDFxxx IMS.PROCLIB member. A minimum of two data sets must be used, but three is recommended. In an IMSplex environment, each IMS in the IMSplex should have its own set of RDDSs.

The data sets are allocated dynamically using DFSRDDDD as the DD name. Each data set contains a header record followed by multiple resource records.

In an XRF environment, the XRF alternate must have its own set of resource definition data sets defined. The XRF alternate processes X'22' log records and applies the changes to its resources. When X'4098' (end checkpoint) log records are

processed on the alternate IMS system, the log record is checked to determine whether changes have occurred since the last checkpoint. If so, autoexport is initiated.

DBCTL warm standby systems require their own set of RDDSs if you intend to use AUTOEXPORT or AUTOIMPORT. Because DBCTL warm standby systems are started with an /ERE command, the resource definitions are initially loaded from the log records of a failed IMS active system. The initial checkpoint that is created at the end of restart can initiate an AUTOEXPORT request to export all resource and descriptor definitions in the IMS.MODBLKS data set to an RDDS, which can then be used to recover resources if the DBCTL warm standby system (now the active system) needs to be cold started.

When creating and allocating the RDDS data sets, ensure that an End of File (EOF) mark is placed at the beginning of the data set. Failure to do so can produce unpredictable results. To place an EOF mark at the beginning of the data set, use program IEBGENER. The sample job skeleton DFSRDDAL, which is in the IMS.SDFSSLIB data set, can be used for this step.

The BLKSIZE parameters of the SYSUT1 and SYSUT2 DD statements can be variable. The JCL shown below sets BLKSIZE=32760, which is the maximum size allowable for BSAM data sets. Using the maximum size minimizes the number of I/O requests when importing or exporting to an RDDS and is the recommended value.

The SYSUT1 DD statement should contain the following values:

- RECFM=VB
- BLKSIZE=*value*, where *value* is a number between 4096 and 32 760, inclusive.

The SYSUT2 DD statement should contain the following values:

- The data set name and VOLSER of the RDDS to be created.
- RECFM=VB
- BLKSIZE=*value*, where *value* is the same number that is specified on the SYSUT1 DD statement.
- LRECL=*value*, where *value* is always the BLKSIZE value minus 4.

An alternate method of allocating the RDDS data sets is to use the ALLOCATE function of ISPF utilities. The ALLOCATE function places an EOF mark correctly at the beginning of the data set.

Do not use program IEFBR14 to allocate the RDDS data sets. IEFBR14 does not place an EOF mark at the start of the data set.

Sample JCL for allocating an RDDS is shown below. This sample JCL, which allocates 3 RDDSs, is shipped with IMS as DFSRDDAL in the IMS.SDFSSLIB data set.

```
//ALLOC1 EXEC PGM=IEBGENER
//SYSUT1 DD   DUMMY,BLKSIZE=32760,RECFM=VB
//SYSUT2 DD   DSN=IMSTESTL.RDDS1,
//            DCB=(RECFM=VB,LRECL=32756,BLKSIZE=32760),
//            UNIT=SYSDA,VOL=SER=TSTVOL,
//            DISP=(,CATLG),SPACE=(TRK,(10,10))
//SYSPRINT DD SYSOUT=*
//SYSIN  DD   DUMMY
```

As a rule, an RDDS uses 33 percent more space than an IMS.MODBLKS data set that contains an equivalent set of resource definitions.

If you specify a BLKSIZE value other than 32 760, you must also ensure that:

- BLKSIZE is less than or equal to 32 760.
- BLKSIZE is greater than or equal to 4096.
- RECFM=VB.

Recommendation: All RDDS block sizes should have the same value. If the RDDS data sets are defined with different block sizes, a slight performance impact can occur during autoexport processing due to the extra overhead required for reblocking the records.

Changing RDDS block sizes

You can change the block size of the RDDS without bringing down IMS.

This procedure assumes that autoexport is enabled to RDDS.

To change the RDDS block size, complete the following steps:

1. Save a copy of the existing system RDDS by using the IEHPROGM utility with the RENAME function. Sample JCL is shown below.

   In the JCL, the data sets IMSPLEX1.IMS1.RDDS1, IMSPLEX1.IMS1.RDDS2, and IMSPLEX1.IMS1.RDDS3 on 3390 volumes 111111, 222222, and 333333, respectively, are renamed to SAVE.RDDS1, SAVE.RDDS2, and SAVE.RDDS3, respectively.

   ```
   //RDDSNAME JOB .........
   //STEP1 EXEC PGM=IEHPROGM
   //SYSPRINT DD SYSOUT=A
   //DD1 DD VOLSER=111111,UNIT=SYSDA,DISP=OLD
   //DD2 DD VOLSER=222222,UNIT=SYSDA,DISP=OLD
   //DD3 DD VOLSER=333333,UNIT=SYSDA,DISP=OLD
   //SYSIN DD *
    RENAME DSNAME=IMSPLEX1.IMS1.RDDS1,VOL=3390=111111,NEWNAME=SAVE.RDDS1
    RENAME DSNAME=IMSPLEX1.IMS1.RDDS2,VOL=3390=222222,NEWNAME=SAVE.RDDS2
    RENAME DSNAME=IMSPLEX1.IMS1.RDDS3,VOL=3390=333333,NEWNAME=SAVE.RDDS3
   ```
2. Scratch and reallocate all RDDSs with the new block size using either TSO or the IEBGENER job provided in the DFSRDDAL skeleton JCL. The JCL is shipped in the IMS.SDFSSLIB data set.
3. Create a dummy resource descriptor to force an autoexport at the next checkpoint by issuing the command CREATE PGMDESC NAME(DUMMY) LIKE(DESC(DFSDSPG1)).
4. Issue the /CHE command.
5. Delete the dummy resource by issuing the command DEL PGMDESC NAME(DUMMY).

After autoexport completes, message DFS3371I is issued to the system console, indicating that the autoexport was successful.

If IMS is terminated before autoexport successfully completes, it can be restarted with a warm or emergency restart. Both warm and emergency restarts rebuild the resource and descriptor definitions in the IMS.MODBLKS data set from the IMS log records and automatically export them to a system RDDS at the first system checkpoint.

If a cold start is necessary, perform the following steps:

1. Rename the RDDSs to their original names by using the IEHPROGM, swapping the DSNAME and the NEWNAME data set name values.
2. Bring up IMS and issue the IMS cold start command /NRE CHKPT 0.
3. Change the block sizes using the procedure above.

Allocating IMS spool data sets

Message processing programs can require their own printed output to be separate from any system-wide SYSOUT (spooled) output. JES2 and JES3 provide SYSOUT data set support. IMS provides for spooled output data sets, as well as a print utility (DFSUPRT0), which can be scheduled during execution of the online system.

IMS spool data sets are on direct access storage devices (DASD). When allocating IMS spool data sets, use the IEBGENER utility to ensure that they are properly initialized (empty).

When allocating IMS spool data sets, code the combination of DISP=(NEW,CATLG) and DSORG=PS on the DD statement.

```
//FORMAT   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=IMS spool data set name,DISP=(,CATLG),
//            UNIT=SYSDA,SPACE=space info
//SYSUT1   DD DUMMY
//SYSIN    DD DUMMY
```

Allocate space for IMS spool data sets as required, but do not specify secondary allocation. You need DCB parameters DSORG=PS and RECFM=UM. If not supplied, these parameters are set automatically. You can specify block size in the DD statement, but it can be adjusted downward by the system, if larger than the system definition specification.

Records written to this data set are standard z/OS variable-length blocked (VBM) records. The designation of the undefined record format (UM) specification reduces the buffer space requirement in the IMS control region. The minimum block size is 20 bytes, which is sufficient for one print line. The maximum block size is the track size of the device on which the data set is allocated.

Recommendation: Allocate at least two data sets.

IMS uses EXCP to maintain the end-of-file (EOF) mark on the subsequent track of the spool data sets to support online access (TSO browsing).

Restriction: EXCP does not support partitioned data sets extended (PDSEs), extended format data sets, or z/OS UNIX System Services file system data sets.

Defining spool line groups

You specify, in system definition, a LINEGRP macro to be dedicated to IMS spool output. Associated with the LINEGRP macro are LINE, TERMINAL, and NAME macro specifications. The specification requirements for one such group are illustrated in the following table.

Table 21. Example of IMS spooled data sets in system definition

Macro	Coding	Comments
LINEGRP	DDNAME = (SPOOL1, SPOOL2) UNITYPE = SPOOL	Two IMS spool data sets
LINE	BUFSIZE = 1200	Buffer size in bytes
TERMINAL	AUTOSCH	Optional, specified if automatic scheduling
NAME	RPT10	Use LTERM names that show nature of output

System definition execution automatically generates appropriate DD statements in the IMS procedure in IMS.PROCLIB. The ddnames are those given in the LINEGRP macro, and the data set names are of the form IMS.SYSnn. The order of the ddnames in the Stage 1 input stream determines the incremented value of *nn*. If, in the example shown in the table above, the LINEGRP macro is the first spool line group, the data set name for the ddname SPOOL2 is IMS.SYS02.

System definition also automatically generates procedures named DFSWTnnn, members in IMS.PROCLIB that are tailored to the print operation for the data sets implied in each line group. Referring to the same example, a member of the IMS.JOBS data set named IMSWT000 invokes DFSWT000, because it is the first individual job to print output for a spool line group.

The default values for job class and message class used for execution of the IMSWTnnn procedures are derived from the parameters of the MAXREGN keyword on the IMSCTRL macro. You must review these generated procedures for your installation's output class requirements. The DFSWTnnn procedures are the executable portions that are invoked for each IMSWTnnn member.

For spool lines, the logical record length (LRECL) specification must be the maximum segment length desired +8, and the block size must be at least equal to LRECL+10. Assign a nonzero value to LRECL. Message segments are truncated at a value of LRECL+4. For example, if the buffer size you specify in the LINE macro is 132, block size can be 116, and LRECL 106. The total size of the data sets must be at least as large as the largest possible message. If the physical block size of the data set is larger than the buffer size specified in the LINE macro during IMS system definition, IMS adjusts the block size (BLKSIZE) downward to the specified BUFSIZE -10.

When all IMS spool data sets defined for a line group are full, IMS shuts down the line and sends a message (DFS998I) to the master terminal that the physical terminal is inoperative. If you specify the AUTOSCH option in the TERMINAL macro during system definition, a spool print program is scheduled as each data set is filled.

Estimating the spooled requirements

To estimate the IMS spooled requirements for your system:

- Determine the LTERM names that the application programs use. There might be output that is unique to a particular program or an agreement among several programs to use the LTERM for their online printed output.
- Obtain estimates of the volume of output so that you can make appropriate space allocations for the DASD data sets.

- Assess the maximum output buffer size to handle the output without unnecessary buildup.
- Obtain some indication of the turnaround requirements for the output and whether the output is to be accumulated or produced in small batches. This information helps you decide how many data sets are needed to support the operation of each group of output (at least two data sets are recommended) and how to schedule the printing operation.

Allocating required IMS.SYSnn data sets

You allocate 1 to 99 DASD data sets with appropriate space specifications. One of the factors that determines the availability of the data set for printing is the size of the primary space. You can either allocate sufficient primary space with any additional output planned to spill to alternative data sets, or you can allocate secondary space for unusual amounts of output.

Allocate and catalog these data sets along with your other online system data sets. The data set names are in the form IMS.SYSnn, beginning with IMS.SYS01.

Implementing spool line groups in an XRF environment

To properly implement spool data sets in an XRF complex, note the following considerations:

- Separate spool data sets must be used for the active and alternate IMS subsystems.
- The appropriate DD statements must be added to the execution procedures for the active and alternate IMS subsystems.
- Separate JOBS data sets must be used for the active and alternate IMS subsystems.
- Separate IMSRDR procedures must be used for the active and alternate IMS subsystems (use the PRDR= execution parameter).
- The IMSRDR procedures used for the active and alternate IMS subsystems must reference the appropriate JOBS data set.
- The IMSWTnnn members of the JOBS data sets must reference the appropriate spool data sets. Depending upon the names chosen for the spool data sets, the SYS2= parameter in the DFSWTnnn procedure can be used to access the correct data sets.

Allocating direct output lines

You can specify record format, logical record length, and block size when you allocate direct output lines.

For direct SYSOUT lines defined to IMS, you can use any valid output device supported by the operating system's BSAM. You can specify the following record formats: F, FM, FB, FBM, FBS, FBSM, V, VM, VB, and VBM. You can specify block sizes, but these are adjusted downward at execution time if they are larger than system-definition maximums.

For fixed-format records, the system-defined buffer size must be at least 20 bytes longer than the DCB block size for the data set. For variable-length records, the buffer size must be 16 bytes longer than the desired block size, including Block Descriptor Word and Record Descriptor Word. To accommodate the data to be written, you can select logical record specifications that are restricted as follows:

- For fixed-format records, the block size must be an even multiple of logical record length.
- For unblocked variable-format records, maximum logical record length equals block size minus 4, and must include the RDW (4 bytes).

Table 22 lists device types and the corresponding default data set values for direct output data sets. If you do not supply DCB parameters, these default record format, logical record length, and block size values apply.

Table 22. Default data set attributes for direct output data sets

Device Type	RECFM	LRECL	BLKSIZE
3211	VM	137	141
2540P	V	84	88 (note 1)
2400 series tape	VBM	125	(note 2)
DASD	VBM	125	1/4 Track

Note:

1. Control characters are not supported.
2. Block size only depends on system-definition buffer size. Each segment is treated as a logical record. When you specify blocking, all segments of a message are contained within a block, unless the block size is not large enough.

   Fixed-length segments are padded with trailing blanks. If blocking is used, the balance of the block is also padded when a message does not have the same number of segments as logical records in the block.

   Tape blocks are not shorter than 18 bytes, regardless of the record format.

   Because volume switching is provided by operator command when tape is used, specify a large value (for example, 99) for the volume count subparameter of the VOLUME keyword on the associated DD statement. In an IMS system in which binary synchronous devices are also operating, and only one tape drive is allocated, timeout problems can occur.

Initializing the RECON data set for DBRC

Before your IMS system can operate, you must initialize the RECON data set, where system log and database status are recorded.

To prepare for online operation, ensure that the RECON data set, in which system log and database status are recorded, is ready. To initialize the RECON data sets:

1. Define two RECON data sets (named RECON1 and RECON2) and a spare data set (named RECON3).
2. Identify the database data sets to be tracked.
3. Record a starting set of data set levels.

Access Method Services parameters define the VSAM KSDS data sets referenced as RECON1, RECON2, and RECON3. Then the INIT.RECON command is used to write the required header record.

Related reading: For more information about the allocation parameters, see *IMS Version 11 System Administration*. All three RECON data sets can be dynamically allocated.

A separate record is required for each data set of the databases that are to be controlled. The image copy, reorganization, and recovery operations call upon the current information in the RECON data set records. Using the INIT.DBDS command, you specify:

- The database name, the data set DD name, and the relevant data set names.
- The number and reuse characteristics of image-copy data sets that are to be maintained.
- Indicators of how the data sets are to be allocated by the IMS system.
- The member names of procedures to execute database utilities.

Using other commands in the INIT group creates other records for initial change accumulation, image copy, and system log data set names. For example, use the INIT.DB command to specify the share level for a database and to register a database with DBRC.

For system definition purposes you need to coordinate the entries on DATABASE macro statements, the online JCL requirements, and the database maintenance procedures.

To use IMS Version 11, you must have an IMS Version 11 RECON. Either initialize a new set of RECON data sets, or if you are migrating from an earlier release of IMS, upgrade the RECON data sets to the IMS Version 11 format by using the CHANGE.RECON UPGRADE command.

Important: If you modified IMS Version 10 DBRC skeletal JCL members, you can use them in IMS Version 11 and later. However, DBRC skeletal JCL members supplied with IMS Version 10 and later are not compatible with IMS Version 9 and earlier.

For information about the DBRC-related commands, see *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands*.

Allocating XRF data sets

If your installation is using the IMS Extended Recovery Facility (XRF), in which an alternate IMS system acts as a backup to an active IMS system, there are a number of considerations for allocating data sets. Some data sets must be shared between the two systems, others must be replicated, and others can optionally be replicated.

Three main XRF requirements for placing your IMS data sets are:

- Availability of data sets during tracking and takeover

  An XRF complex consists of two systems that must sometimes access the same data sets or identical copies of the same data sets. Therefore, use of XRF requires that you load some data sets on DASD shared by the two systems.

  Recommendation: Load other data sets on shared DASD. However, you can switch some data sets through a switching device or maintain separate copies of them.
- Prevention of single points of failure

  Use of XRF requires that you maintain and constantly synchronize separate copies of some data sets for the two systems.
- Accessibility of data sets to one IMS system

  Recommendation: Keep the data sets unique to one system on local DASD.

Mandatory sharing of system data sets

Use of XRF requires that some IMS system data sets, such as the system logs, be available to both the active and the alternate IMS subsystems during the tracking phases. Use of XRF requires that others, such as the DEDB data sets, be present immediately at takeover.

The following data sets must reside on DASD that active and alternate IMS subsystems share:

CRITICAL DL/I DATABASE (DFSMDA definitions)
DEDB AREA
DFSOLPxx (DFSMDA definitions are recommended)
DFSOLSxx (DFSMDA definitions are recommended)
DFSWADSx (DFSMDA definitions are recommended)
IMSRDS
IMSRDS2
MODSTAT
MODSTAT2
MSDBINIT
RECON1 (DFSMDA definitions are recommended)
RECON2 (DFSMDA definitions are recommended)
RECON3 (DFSMDA definitions are recommended)

These data sets must be accessible to both subsystems through the catalog structure. Also, do not store an OLDS, WADS, or restart data set (RDS) on volumes containing data sets (IMS or otherwise) that can be subject to a RESERVE operation. Keep such data sets separated.

Mandatory replication of data sets

Certain IMS execution data sets contain information unique to only one subsystem. Replicate these data sets, so each active and alternate IMS subsystem has its own unique data sets. Store these data sets on local, non-shared DASD, and define them in a separate catalog structure. The data sets in this category are:

IMSMON
LGMSGx
LGMSGL
MSDBCP1
MSDBCP2
MSDBCP3
MSDBCP4
MSDBDUMP
QBLKS
QBLKSL
SHMSGx
SHMSGL
SPOOLx
SYSABEND
SYSUDUMP

If your XRF configuration requires that both IMS subsystems be executable on either CPC, these data sets must be on shared or switchable DASD, and in a catalog structure accessible to both subsystems.

Optionally replicating data sets

To avoid single points of failure, you can duplicate certain other IMS execution data sets and store them in non-shared local DASD. Data sets in this category are:

DBDLIB (used by DL/I batch)
FORMATA
FORMATB
IMSACBA
IMSACBB
IMSTFMTA
IMSTFMTB
JOBS (used in the IMSRDR procedure)
MODBLKSA
MODBLKSB
PGMLIB
PROCLIB
PSBLIB (used by DL/I batch)
SDFSRESL
SDXRRESL
TCFSLIB
OTHER STEPLIB DATA SETS

If your XRF configuration requires that both IMS subsystems be executable on either CPC, these data sets must be on shared or switchable DASD and in a catalog structure accessible to both subsystems.

Other data sets impacted by XRF

When planning your XRF configuration, it is important to consider the possible impact on the other IMS data sets. Also examine the impact on activities other than online execution, such as IMS system definition and the application of SMP/E service. Table 23 provides information about data sets in this category, including descriptions and whether they are managed by SMP/E.

Table 23. Other data sets impacted by XRF

Data set	Description	Managed by SMP/E
IMS.FORMAT	message format library	No
IMS.MODBLKS	created by SYSDEF	Yes
IMS.OBJDSET	created by SYSDEF	No
IMS.OPTIONS	created by SYSDEF; used by SMP/E and SYSDEF	No
IMS.PROCLIB	created by SYSDEF	No
IMS.REFERAL	used with FORMAT	No
IMS.SDFSMAC	created by SMP/E	Yes
IMS.SDFSRESL	created by SYSDEF and SMP/E	Yes

Table 23. Other data sets impacted by XRF (continued)

Data set	Description	Managed by SMP/E
IMS.TFORMAT	test message format library	No

Some of these data sets appear in earlier lists in this section. You must avoid possible synchronization conflicts.

It is essential to synchronize the DFSMDA members in the IMS.SDFSRESL, or associated libraries, across the XRF complex.

RACF profile data sets

The Resource Access Control Facility (RACF) profile data sets must be on DASD shared by the active and alternate IMS subsystems. To avoid single points of failure, use the RACF backup facility to keep a second copy of these data sets also on shared DASD.

For additional information about RACF, see *z/OS Security Server RACF General User's Guide*.

Defining a HALDB indirect list data set

An IMS High Availability Large Database (HALDB) uses indirect pointers rather than direct pointers during reorganization processing. A VSAM KSDS (key sequenced data set) called the indirect list data set (ILDS) stores the indirect pointers. Use the z/OS DEFINE CLUSTER command to define the ILDS.

The use of logical relationships or secondary indexing presents challenges in High Availability Large Database (HALDB) reorganization processing. After a HALDB partition is reorganized, the change in segment locations in the reorganized partition potentially invalidates all the pointers to those segments, whether the pointers are from other database records within the same HALDB partition, other HALDB partitions, or HALDB secondary indexes. In order to eliminate the need to update pointers throughout other database records when a single HALDB partition is reorganized, HALDB introduces the use of indirect pointers.

After a reorganization, direct pointers that have become invalid are updated by using the indirect pointers upon the first reference to the segments that have moved. A new system index data set, which serves as a repository for indirect pointers, is introduced for HALDB. This system index data set is called the Indirect List Data Set (ILDS).

The ILDS is a VSAM KSDS with a 9-byte key. There is one ILDS per partition in a PHDAM, or PHIDAM database. During a reorganization reload or migration reload of segments involved in inter-record pointing, an entry called Indirect List Entry (ILE) is created in the ILDS for each of these segments that is reloaded. Each ILE is 50 bytes in length and contains pointers and control information. The following is a sample of IDCAMS input that is used for defining an ILDS.

```
DEFINE CLUSTER ( -
         NAME (FFDBPRT1.XABCD010.L00001) -
         TRK (2,1) -
         VOL (IMSQAV) -
         FREESPACE (80,10) -
         KEYS (9,0) -
         RECSZ (50,50) -
```

```
  REUSE -
  SHAREOPTIONS (3,3) -
  SPEED ) -
DATA ( -
  CISZ (8192) ) -
INDEX ( -
  CISZ (2048) )
```

NAME
Defines the HALDB Partition Base Name (FFDBPRT1.XABCD010), ILDS reference (.L), and HALDB Partition ID (00001).

KEYS A required parameter value. Specifies a key size of 9 bytes at offset 0 into the LRECL.

RECSZ
A required parameter value. Specifies a record size of 50 bytes, the length of an ILE. REUSE must be specified for all HALDB VSAM data sets.

FREESPACE
Provides for free space on initial loads and after CI and CA splits.

To compute the size of an ILDS, multiply the size of an ILE with the total number of physically paired logical children, logical parents of unidirectional relationships, and secondary index target segments.

Defining large sequential data sets

Sequential data sets can become large and span many volumes of storage hardware. However, z/OS Data Facilities Storage Management Subsystem (DFSMS) supports more than 65,535 tracks per volume of sequential data sets, which enables more data to be stored on fewer volumes. IMS supports large format data sets.

Sequential data sets are used in IMS primarily for:

- GSAM databases
- Sequential OSAM databases (DSORG=PS - physical sequential)
- Long and short message queues
- Online log data sets (OLDS)
- External trace data sets

Migration to large format data sets can be done only at data set creation time.

The following process shows one possible approach to defining large sequential data sets:

1. Determine which existing GSAM data sets that use the BSAM access method and which sequential OSAM data sets span more than one volume because they are larger than 65,535 tracks.
2. Determine if you have hardware that supports more than 65,535 tracks.
3. Add the DSNTYPE=LARGE parameter to the DD statement in the JCL that defines these data sets.
4. Allocate the new data sets.
5. Bring the new data sets online.
   - If you are using the new data sets for an online data set (OLDS) or a message queue, perform a cold start of IMS to bring the new data sets online.

- If you are using the new data sets for a database, use a database reorganization process (unload and reload) to bring the new data sets online.

DD statement examples

Here are a few DD statement examples that include the DSNTYPE=LARGE statement that is required to define large format sequential data sets:

DD statement for GSAM data sets

```
//ODASD    DD UNIT=SYSDA,VOL=SER=LRGVS1,DISP=(,KEEP),
//         SPACE=(CYL,(4200,5)),DSN=IMSTESTL.GSAM.VARIAB01,
//         DSNTYPE=LARGE,
//         DCB=(LRECL=800,BLKSIZE=80,BUFNO=5,RECFM=FB)
```

DD statement for OSAM data sets

```
//AJOSAMDB DD  DSN=IMSTESTL.AJOSAMDB,UNIT=SYSDA,
//             DISP=(NEW,CATLG),DSNTYPE=LARGE,
//             SPACE=(CYL,(4500,100)),VOL=SER=LRGVS1
```

Related concepts:

"Allocating OSAM data sets" on page 128

Falling back from large sequential data sets

If necessary, you can fall back from using large sequential data sets.

To fall back from using large sequential data sets:

1. Define data sets with less than 65,535 tracks on one volume.
2. Perform a database reorganization process (unload from the large sequential data sets and reload onto multiple volumes) if the large sequential data sets are used for databases.
3. Remove the JCL that defines the large sequential data set or sets.
4. Cold start IMS to bring the multivolume data set or sets online.

Defining Message Formatting Service

Message Formatting Service (MFS) is an editing facility that allows application programs to handle simple logical messages instead of device-dependent data, which simplifies the application development process. These topics describe MFS-related system definition and programming considerations.

The IMS macros that are used to define specifications for MFS are described in the following table.

Table 24. IMS macros that are used to define MFS specifications

IMS macro	MFS specification
BUFPOOLS	• The size of the message format buffer pool. • The number of fetch request elements (FRE) to be used for loading MFS control blocks into the message format buffer pool (with the FORMAT=, COMM=, and FRE= keywords). The following I/O pool requirements for MFS are provided by the MFS language utility using message DFS1060I: • Largest line buffer size for 3270 displays • Largest work buffer size for an input message • Largest work buffer size for an output message For efficient operation in the virtual control region, the areas acquired for buffer pools during initialization can be fixed in address space during initialization.
MSGQUEUE	MSGQUEUE is used to specify the maximum size output segment that is properly processed by MFS.
IMSGEN	IMSGEN is used to specify MFSTEST parameters if the COMM macro is not used. The SUFFIX= operand specifies the suffix of the MFS device characteristics table that is generated when TYPE or TERMINAL macros define device symbolic names (that is, TYPE=3270-An) during system definition.
COMM	COMM is used to specify the following facilities of MFS: • Exits for user-written field and segment editing routines to be included in the generated system • IMS-provided formatting for MFS on the 3270 or SLU 2 master terminal The IMS-provided formatting for the master terminal is provided if OPTIONS=(...,FMTMAST,...) is specified and the 3270 or SLU 2 display used as the master terminal is a 24x80 screen defined as TYPE=(3270,2) or 3270-An with SIZE=(24,80). The name used for the EDITNAME parameter cannot be the same as any MID name.
TYPE	The TYPE, SIZE, FEAT, and OPTIONS keywords can be used in the TYPE and TERMINAL macro. The OPTIONS= operand of the TYPE macro is used to request MFS support for a set of terminal devices described in the subsequent TERMINAL macros. If 3270 or SLU 2 terminals are defined, MFS support is automatic. The TYPE, SIZE, and FEAT keywords define the 3270 and SLU 2 devices using the device symbolic name and generate the MFS device characteristics table containing the device type symbolic name, associated screen size, and physical terminal features. TYPE keywords provide definition for subsequent TERMINAL macros. For NTO, specify UNITYPE=NTO on the TYPE macro.

Table 24. IMS macros that are used to define MFS specifications (continued)

IMS macro	MFS specification
TERMINAL	The COMPTn= operand can be used to request MFS support for a specific component of a terminal defined as a secondary logical unit. For NTO, specify PU=xx where xx is LUNS, 2741, 2740-1, or TTY. **For TYPE UNITYPE=SLUTYPE1** COMPTn=(...,MFS-SCS1,...) must be specified for the MFS SCS1 formatting option for the console, printer, or print data set component. COMPTn=(...,MFS-SCS2,...) must be specified for the MFS SCS2 formatting option for the reader, punch, or transmit data set component. **For TYPE UNITYPE=SLUTYPEP** COMPTn=(...,DPM-An,...) must be specified for the MFS DPM formatting option for the remote controller component. COMPTn=(...,MFS-SCS1,...) must be specified for the MFS SCS1 formatting option for the remote controller component. **For TYPE UNITYPE=LUTYPE6** COMPTn=(...,DPM-Bn,...) must be specified for MFS ISC formatting.

The following is a sample definition that would allow DPM-An to be used with a remote program. The definition is for a secondary logical unit with two components.

```
TYPE      UNITYPE=SLUTYPEP

TERMINAL  NAME=LUX,OPTIONS=(FORCRESP,ACK,NOBID),
          OUTBUF=256,COMPT1=(PROGRAM2,DPM-A3,7),
          COMPT2=(PROGRAM1,DPM-A4,1)

NAME      LTERMQ,COMPT=1

NAME      LTERMY,COMPT=2
```

PROGRAM1 is printed output operation; PROGRAM2 is paged output operation.

The OUTBUF= specification would allow RCDCTL to be up to 256 bytes in all format definitions for this secondary logical unit.

The following is a sample definition that would allow DPM-An or SCS1 to be used with a remote program. For this secondary logical unit, the DPM-An definition is for the first two components, and the SCS1 definition is for the third component.

```
TYPE      UNITYPE=SLUTYPEP

TERMINAL  NAME=LUX,OPTIONS=(FORCRESP,ACK,NOBID),
          OUTBUF=256,COMPT1=(PROGRAM2,DPM-A3,7),
          COMPT2=(PROGRAM1,DPM-A4,1),
          COMPT3=(PROGRAM1,MFS-SCS1,IGNORE)

NAME      LTERMQ,COMPT=1

NAME      LTERMY,COMPT=2

NAME      LTERMS,COMPT=3
```

PROGRAM1 is an unprotected operation at the end of a message for paged output or a printer (SCS1) operation; PROGRAM2 is a protected operation at the end of a message for paged output.

The OUTBUF= specification would allow RCDCTL to be up to 256 bytes in all DPM format definitions for this secondary logical unit.

If OUTBUF is not large enough for an entire data page, then more than one VTAM SEND is required to send the page.

Implementing Message Formatting Service

After defining the Message Formatting Service, you can create an index directory to control blocks in the staging library, you can define several format libraries on different physical or logical devices, and you can display the contents of the message format buffer pool counters.

INDEX directory

The INDEX function of the MFS service utility creates an index directory to specified control blocks in the staging library, IMS.FORMAT. The IMS initialization process uses the index directory to construct the MFS dynamic directory in a private storage area. This space is separate from the message format buffer pool.

If a requested control block is not found in the message format buffer pool, MFS looks for an entry in the MFS dynamic directory. If found, the entry gives the relative track address for the indexed control block. This allows MFS to issue a direct read without going through the active format library's PDS directory. If the entry is not found, MFS reads the PDS directory, then reads the requested control block. MFS also creates an entry in the MFS dynamic directory for the control block.

The number of entries in the MFS dynamic directory continually increases until the MTO uses the command /CHANGE DIRECTORY MFS. This purges all the entries that were added to the MFS dynamic directory after online initialization.

Error conditions include:

- IMS abending during initialization if storage is unavailable for the MFS dynamic directory
- The /MODIFY COMMIT command failing during online change if storage is unavailable for the MFS dynamic directory or the PDS directory index
- No entries being added to the MFS dynamic directory during online operation if storage is unavailable for the MFS dynamic directory. To add entries to the MFS dynamic directory again, issue a /CHANGE DIRECTORY MFS command, or restart IMS.

Use of concatenated format libraries

You can define several format libraries on different physical or logical devices, thus permitting faster (concurrent) input/output processing. IMS limits the number of data sets that can be concatenated for MFS format libraries, IMS.FORMATA, and IMS.FORMATB to 16. All the concatenations must have like attributes.

Phase 2 of the MFS language utility implicitly deletes an old control block from a format library when a new control block with the same name is defined for that

library. However, if the new control block is defined for a different library than the one the old copy is in, there is no implicit deletion. Thus, old and new copies for control blocks can exist simultaneously in different format libraries.

One way to prevent this duplication is to explicitly delete duplicate control blocks, using the MFS service utility SCRATCH function. Another way is to add a DD statement in the MFSUTIL phase 2 JCL for each format library (up to 15) that is to have old control blocks deleted when new versions of those control blocks are defined for the staging library, IMS.FORMAT. Phase 2 makes the deletions and updates the index directory $$IMSDIR (if there is one) in each library to reflect the new contents.

The new DD statements must have labels of the form, FORMATn, where n is a number from 1 to 15, inclusive. Phase 2 only compresses and adds control blocks to the format library defined with a DD statement labeled FORMAT. (FORMAT is the original staging library.)

At IMS online initialization, the index directories ($$IMSDIR) of each of the format libraries are merged into a single MFS dynamic directory for all the format libraries. If a requested control block is not listed in the MFS dynamic directory, MFS reads one PDS directory block from each concatenated library, in sequence, until it finds the control block. This can mean many additional reads if the control block resides on one of the last concatenated libraries.

Pool statistics

Use the /DISPLAY POOL command to display the contents of the message format buffer pool counters. The statistics displayed for directory and fetch input/output operations and pool space use should help in determining the appropriate pool size and number of FREs.

Related reading: See *IMS Version 11 System Administration* for formulas to calculate the format buffer pool size and the number of FREs.

Chapter 5. Specifying IMS execution parameters

You can specify control region execution parameters in the appropriate control region JCL, or in the IMS, DBC, and DCC members of the IMS PROCLIB data set. Parameters specified in these members override any system definition. The parameters are not positional.

Use the PARM1= and PARM2= parameters on the EXEC statements for the control region to specify the JCL for IMS online execution.

Recommendation: For optimal performance, use the PARM1= and PARM2= parameters rather than creating your own JCL. These parameters are specified symbolically for the control region.

System control and performance EXEC parameters for the IMS control region

Using IMS EXEC parameters, you can control many IMS system resources such as number of active regions, performance options, z/OS options, DLISAS options, dump formatting options, IRLM options, Fast Path settings, subsystem options, and storage pool definitions.

This topic describes the EXEC parameters that can be used to control IMS system resources.

Identifying the nucleus

The SUF parameter is generated as a null value for the IMS procedure, indicating that the default nucleus name has a one-character suffix of 0. If you are using an alternative control program, you must specify this parameter.

Overriding the number of active regions

Use the PST parameter to override the expected number of regions that are to be in operation during the online execution. Additional regions can be dynamically allocated, up to the maximum allowable number permitted by your operating system. If you do not specify the PST parameter, the default number of regions is the number specified for the MAXREGN keyword in the IMSCTRL macro.

Specifying performance options

Three parameters contribute to general performance strategy. The FIX= parameter specifies a two-character suffix for the member DFSFIXxx in the IMS PROCLIB data set. This suffix indicates which member should describe all modules and control blocks that are to be page fixed. The EXVR= parameter allows you to page-fix buffers used for the management of message queues. The parameter is generated with a null value; you must specify EXVR=1 to indicate the page-fixing action. Similarly, the PRLD= parameter specifies a two-character suffix for the member DFSMPLxx, where all preloaded modules are listed.

© Copyright IBM Corp. 1974, 2015

Specifying z/OS options

Several parameters apply only to the z/OS operating system. The SRCH parameter allows you to take advantage of any special library structure to optimize the search for loaded modules. You can override the default value of 0 with a value of 1 if you want the JPA and LPA to be searched before IMS program libraries.

To reduce the amount of z/OS CSA (common storage area) that IMS uses you can use one of the following techniques:

One technique is to specify LSO=Y on the IMS procedure to cause some control blocks and some IMS modules that are used for DL/I processing to be loaded into the control program's private storage. To allow for this transfer, use the following sum to increase the size of the control region:

```
220K + OSAM buffer pool + VSAM buffers + enqueue/dequeue tables
```

Specifying DL/I separate address space options

An additional variation of the local storage option is to use the DL/I separate address space. You do this by specifying LSO=S. This address space contains most of the DL/I code, control blocks, and database buffers for full-function databases. Again, z/OS cross memory services are used.

Specifying dump formatting options

In a DC environment, you can request the following types of dump outputs for errors that terminate IMS: SDUMP, SYSMDUMP, SYSABEND, or SYSUDUMP. To do this, specify the FMTO startup parameter in combination with z/OS dump DD statements.

For SYSMDUMP, you should provide operational procedures for saving and formatting dumps; otherwise, you can overlay a SYSMDUMP if you must restart IMS before the previous SYSMDUMP is transferred.

You can also request dump outputs for some errors that do not terminate IMS. Your choice of dump depends on several factors: the type of failure, the FMTO parameter option, and the IMS spin-off and z/OS dump DD statements that have been selected. For more information about the FMTO parameter and about using dumps in these situations, see *IMS Version 11 System Utilities*.

Specifying IRLM options

If an execution of the IMS online system is to use IRLM as lock manager or participate in block-level sharing, specify IRLM=YES. Specify the z/OS subsystem name for the IRLM associated with this control region on the IRLMNM parameter.

You cannot use the UHASH parameter with the IRLM to specify the name of an alternative Fast Path hashing module. The UHASH parameter is ignored, and DBFLHSH0 is always used as the hashing module.

Enabling Fast Path

Use FP= to enable or disable Fast Path.

The FPCTRL macro, although allowed for compatibility, is ignored by system definition. Fast Path is enabled by setting FP=Y in the DFSPBxxx member of the

IMS PROCLIB data set or by specifying FP=Y as a startup parameter. The default is FP=N, so you must explicitly specify FP=Y to enable Fast Path.

If you specify FP=N and you attempt to use a Fast Path resource or command, results are unpredictable.

To override the initial value coded on DFSPBxxx, use the FP= parameter at execution time.

Specifying subsystem identification parameters

Use the SSM parameter to reference a member in the IMS PROCLIB data set which identifies the DB2® for z/OS subsystems that can be accessed from application programs executing in dependent regions. The one- to four-character suffix that you specify, together with the currently assigned name for IMSID, forms the member name. The member contains entries, each identifying a DB2 for z/OS subsystem (by its z/OS subsystem name). All subsystems to be accessed from programs executing in dependent regions must be identified by an entry.

Depending on its entry, a region can access all, some, or none of these subsystems. To allow a dependent region to access all subsystems identified to the IMS control region, do not specify the SSM parameter for the dependent region or have the SSM entry specify the same member as the IMS procedure. To allow a dependent region to access only selected DB2 for z/OS subsystems, specify an SSM parameter on the dependent region procedure that points to a member containing only those specific subsystems. To prevent a dependent region from having access to any subsystem identified to the IMS control region, specify a member containing no entries.

See *IMS Version 11 System Administration* for details about how to specify the individual entries and construct the member of the IMS PROCLIB data set.

Modifying storage pool definitions for the storage manager

You can set an upper expansion limit for the AOIP, HIOP, CIOP, CMDP, SPAP, DYNP, LUMP, LUMC, FPWP, and EMHB storage manager pools by using the appropriate parameters at execution time. IMS establishes these storage pool definitions without an upper expansion limit, because they are dynamic storage pools that expand and contract as needed during execution.

Use caution in specifying upper expansion limits. If an upper limit is too low, IMS might abend. Under normal circumstances, a pool should never reach its upper limit. The intent of the upper limit is to keep pools from consuming so much storage that an out-of-storage condition occurs.

Use the SPM=nn parameter to specify the suffix for DFSPMnn. DFSPMnn identifies the member of the IMS PROCLIB data set that overrides the storage pool definitions established by IMS.

Dynamic resource definition EXEC parameters

You can use the DFSDF= EXEC parameter to override dynamic resource definition (DRD) values that were specified on the DFSDFxxx member of the IMS PROCLIB data set.

Use DFSDF= to identify the DRD PROCLIB member.

By using DFSDF= as an EXEC parameter in the control region, you can override the suffix for DFSDFxxx that was provided during system definition.

Database and PSB exec parameters for the control region

The IMS procedure includes parameters that enable you to control database buffer requirements, override predefined buffer sizes, and manage performance.

Three kinds of parameters in the IMS procedure that are related to database and PSB processing are:

- Pointers to database buffer requirements
- Overrides of predefined buffer sizes
- Performance options

Database buffer requirements

You can use the VSPEC parameter of the IMS procedure to predefine buffer pool requirements for databases that use the OSAM or VSAM access method.

Use the VSPEC parameter to specify a two-character suffix for member DFSVSMxx in the IMS PROCLIB data set. This member predefines the buffer pool requirements for databases that use OSAM or VSAM as the access method. The IMS procedure generates a default value of 00 that points to a general- purpose buffer definition in member, but you can change this suffix.

See "IMS buffer pools" on page 179 to specify subpool sizes.

Buffer sizes for databases that use OSAM or VSAM

Performance studies often indicate a need to increase DMB or PSB buffer space. You can override the system definition values to implement this change.

Several parameters enable you to redefine the size of buffer pools that hold DMBs and PSBs. Specify a value that represents the size as a number of 1024-byte blocks. For example, a value of 18 represents the size as 18432 bytes (18 x 1024 = 18432). IMS then rounds up this number based on the nearest whole number of pages on which these bytes fit. For example, if one page in z/OS equals 4096 bytes, the value of 18 blocks would result in the allocation of 20480 bytes. This value overrides the value defined in system definition. The calculation is as follows:

Formula

```
18 x 24 = 18432/4096 = 4.5 pages (round up to 5)
5 x 4096 = 20480 bytes
```

If you use the DL/I address space option, two PSB pools exist: one in the z/OS common area and one in the DL/I address space. Use the SASPSB parameter on the BUFPOOLS macro to specify sizes of the PSB pools during IMS system definition; use the CSAPSB and DLIPSB parameters on the IMS procedure to override the size of the PSSB pools. If the DL/I address space is not used, specify the size of the single PSB pool using the PSB= parameter on the BUFPOOLS macro; override this value with the PSB= parameter on the IMS procedure.

If Fast Path resources are used, an EPCB pool must contain extensions to Fast Path PCBs. For each PCB referencing a DEDB or MSDB database, an EPCB is required. This pool is used only by MPP regions, not by IFP or BMP regions.

Use the parameters of the IMS procedure in Table 25 to override the buffer pool sizes that were specified at system definition time with the BUFPOOLS macro.

Table 25. Overriding buffer pool sizes with database and PSB parameters

IMS Procedure Parameter	System Definition Specification	
	Macro	**Parameter**
DMB	BUFPOOLS	DMB
PSB	BUFPOOLS	PSB
PSBW	BUFPOOLS	PSBW
CSAPSB	BUFPOOLS	SASPSB(size1)
DLIPSB	BUFPOOLS	SASPSB(size2)
EPCB	BUFPOOLS	EPCB

The IMS procedure is generated with null value parameters for these storage areas. The sizes default to their system definition values.

Performance options for data management and program specification blocks

You can override the system definition specifications for individual database DMBs and PSBs using the RES parameter on the IMS procedure.

System definition lets you specify individual database DMBs and PSBs to be resident with the DATABASE and APPLCTN macros, respectively. You can override this specification with the RES parameter. The IMS procedure generates this as a null parameter; the default is RES=Y.

Request that all the control blocks are not to be made resident by specifying RES=N. You might do this when executing a test system or if storage is temporarily constrained.

Data communication EXEC parameters for the control region

Using EXEC parameters, you can override definitions for data communication that were initially set during IMS system definition.

Three kinds of parameters in the IMS procedure are related to data communication:

- Overrides of predefined buffer sizes
- Overrides and options particular to Message Format Service (MFS)
- Performance options

Buffer sizes for data communications

By monitoring communication traffic you might decide to increase buffer space. Use the parameters of the IMS procedure shown below to allocate virtual storage and override the buffer pool sizes predefined in system definition.

Table 26. Overriding buffer pool sizes with data communication parameters.

IMS procedure parameter	System definition specification	
	Macro	**Parameter**
FBP[1]	BUFPOOLS	FORMAT(size1)
QBUF[2]	MSGQUEUE	BUFFERS
RECA	COMM	RECANY

[1] Specify a value representing the number of 1024-byte blocks.

[2] By increasing the number of buffers (used for the message queue data sets), you can reduce the frequency of I/Os.

Overrides of predefined parameters

Monitoring communication traffic often results in a decision to increase buffer space or the number of buffers. The following execution parameters are provided to override the size and number of some of the buffer pools predefined at system definition:

FBP
Overrides the size specified on the FORMAT(size1) keyword of the BUFPOOLS macro.

QBUF
Message queue buffer. Adjusts the size of the message queue buffer pool. QBUF enables you to override the number of blocks to hold messages in the control program's storage. By increasing the number of buffers used for the message queue data sets, you can reduce the frequency of I/Os. The default number of buffers is specified in the MSGQUEUE macro.

EMHL
Specifies the EMH buffer length for Fast Path transactions.

RECASZ
Specifies the size of the Receive Any buffer.

RECA
Overrides the number of Receive Any buffers predefined in system definition by the RECANY keyword of the COMM macro.

SAV
Specifies the allowed maximum number of concurrently active device I/Os.

ETO parameters

Use the following parameters to define ETO options:

ALOT
Specifies the amount of time allotted before a terminal in session is automatically logged off if no user successfully signs on.

ASOT
Specifies the amount of time allotted before a signed-on user is automatically signed off if no input or output activity occurs.

LHTS
Specifies the number of slots for the CNT/LNB/RCNT hash table.

NHTS
Specifies the number of slots for the VTCB hash table.

UHTS
Specifies the number of slots for the SPQB hash table.

DLQT
Specifies the number of days allotted before a queue containing data that has not been allocated is classified as a potential dead-letter queue.

ETO
Specifies whether terminals and queues that are not defined to IMS are to be supported.

DSCT
Specifies the suffix of the descriptor member DFSDSCTy.

Performance options
Use the following parameters to define performance options:

VAUT
Specifies the use of VTAM-Authorized Path.

NLXB
Specifies that parallel sessions are added during system startup.

FESTIM
Overrides the timeout value for Front End Switching; this value is predefined during system definition.

MFS options

When you use MFS, you can adjust the capacity of the system by adjusting the availability of format blocks. Use the FRE parameter to specify the maximum number of active blocks in the message format buffer pool. You need to coordinate this number with the value specified for the FBP pool size parameter.

Performance options for VTAM systems

The VAUT parameter offers a performance option to systems using VTAM. You can override the null value generated in the IMS procedure and specify the use of VTAM Authorized Path.

The NLXB parameter allows you to add parallel sessions during system startup. The number you specify is added to the number defined in the SESSION keyword of the MSPLINK macro.

The FESTIM parameter overrides the timeout value for front-end switching that is predefined in system definition. You specify the parameter on the COMM macro with a value from 1 second to 300 seconds.

Fast Path EXEC parameters in DCCTL or DB/DC

Use Fast Path EXEC parameters to specify database buffer sizes and specify DEDB options in a DCCTL or IMS DB/DC environment.

Note: Although the DCCTL environment does not support Fast Path databases, it does support Fast Path processing and transactions.

The PARM1= and PARM2= positional parameters for the control region's EXEC statement that can be used to specify the Fast Path parameters are shown below.

Table 27. Categories and purpose of Fast Path control region parameters for DCCTL and DB/DC.

Category	Parameter	Purpose
MSDB load	MSDB	Specify the DBFMSDBx suffix
Database buffer sizes	BSIZ	Specify the common size of a buffer
	DBBF	Specify the maximum number of buffers
	DBFX	Specify system buffer allocation
DEDB options	OTHR	Specify the number of DEDB updates that can be concurrently waiting for I/O after sync point
	LGNR	Specify the maximum DEDB buffer alterations before CI logging
EPCB	EPCB	Specify the size of the EPCB pool
EMHL	EMHL	Specify the size of the EMHL buffer

When an emergency restart is performed, the values specified for the EXEC parameters MSDB, BSIZ, DBBF, OTHR, and LGNR, and the contents of the member DBFMSDBx, must remain unchanged from the last normal start. These values are used when reestablishing buffer contents from checkpoint records.

MSDB loading

The MSDB parameter enables you to control which MSDBs must be loaded for the current online session and how many segments each MSDB requires. The parameter is generated with a null value, implying that the MSDBs are loaded directly from a Fast Path system data set. You specify a one-character suffix that points to an IMS.PROCLIB member, DBFMSDBx, containing the detailed requirements.

If the MSDB requirements for an online session are a subset of the MSDBs, or if the number of segments for any MSDB is to be increased to reserve space for dynamic terminal-related MSDBs, you must coordinate the content of the DBFMSDBx member and the MSDB parameter value. The control statements in the DBFMSDBx member must explicitly name all MSDBs required to be loaded. The control statements also enable you to individually select MSDBs for page fixing.

LU 6.2 devices allow read-only access to dynamic MSDBs. LU 6.2 devices cannot access terminal-related MSDBs.

Terminals defined dynamically with ETO are not available to terminal-related MSDBs.

Database buffer sizes

Using the DBBF and BSIZ parameters, you manually specify the total buffers available for the IMS online system's processing of DEDB activity. Dependent regions then claim portions of this allocation. Use DBFX to specify additional buffers (not separate pools) that are set aside and page-fixed at when the control region is initialized. These buffers allow for asynchronous processing.

If you prefer to have IMS automatically manage Fast Path buffer pools, enable the Fast Path 64-bit buffer manager using the DFSDFxxx PROCLIB member.

DEDB options

The OTHR parameter specifies the number of asynchronous output threads. You can revise the estimate of DEDB updates that occupies buffer space while they are waiting to be committed to the area data sets after sync point logging.

The OTHR= parameter can have any value from 1 to 32,767. If a value is not specified, the default is 255. If `0` or a value greater than `32767` is specified, the default is 2. If a value from `1` to `32767` is specified, the specified value is used.

The LGNR parameter determines a maximum number of individually logged DEDB alterations made in a buffer. Above that number, the whole control interval (CI) is logged. The default value is 7, and you can specify a number from 7 to 99. If the application program is using large CIs, the LGNR value can be increased to economize on the system log activity.

Related reference:

"FASTPATH section of the DFSDFxxx member" on page 685

"Parameter descriptions for IMS procedures" on page 448

Fast Path EXEC parameters in DBCTL

Use Fast Path EXEC parameters to specify database buffer sizes and specify DEDB options in a DBCTL environment.

The PARM1= and PARM2= positional parameters for the control region's EXEC statement that can be used to specify the Fast Path parameters in a DBCTL region are shown below.

Table 28. Categories and purpose of Fast Path control region parameters for DBCTL.

Category	Parameter	Purpose
Database buffer sizes	BSIZ	Specify the common size of a buffer
	DBBF	Specify the maximum number of buffers
	DBFX	Specify system buffer allocation
DEDB options	OTHR	Specify the number of DEDB updates that can be concurrently waiting for I/O after sync point
	LGNR	Specify the maximum DEDB buffer alterations before CI logging
EPCB	EPCB	Specify the size of the EPCB pool

When an emergency restart is performed, the values specified for the EXEC parameters BSIZ, DBBF, OTHR, and LGNR must remain unchanged from the last normal start. These values are used when reestablishing buffer contents from checkpoint records.

Database buffer sizes

Using the DBBF and BSIZ parameters, you manually specify the total buffers available for the IMS online system's processing of DEDB activity. Dependent regions then claim portions of this allocation. Use DBFX to specify additional buffers (not separate pools) that are set aside and page-fixed at when the control region is initialized. These buffers allow for asynchronous processing.

If you prefer to have IMS automatically manage Fast Path buffer pools, enable the Fast Path 64-bit buffer manager using the DFSDFxxx PROCLIB member.

DEDB options

The OTHR parameter specifies the number of asynchronous output threads. It allows you to revise the estimate of DEDB updates occupying buffer space while waiting to be committed to the area data sets after sync point logging.

The OTHR= parameter can have any value from 1 to 32,767. If a value is not specified, the default is 255. If `0` or a value greater than `32767` is specified, the default is 2. If a value from `1` to `32767` is specified, the specified value is used.

The LGNR parameter determines a maximum for the number of DEDB alterations made in a buffer that are individually logged. Above that number, the whole control interval (CI) is logged. The default value is 7, and you can specify a number from 7 to 99. If the application program is using large CIs, the LGNR value can be increased to economize on the system log activity.

Related reference:

"Parameter descriptions for IMS procedures" on page 448

Fast Path dependent region parameters in DCCTL or DB/DC

The positional parameters (PARM=) for a Fast Path dependent region in DCCTL or DB/DC environments are described in this topic.

Note: The DCCTL environment does not support Fast Path databases. It does support Fast Path processing and transactions.

The PARM= positional parameters for Fast Path dependent region's EXEC statements are shown in Table 29.

Table 29. Fast Path dependent region parameters for DCCTL and DB/DC

Category	Parameter
Database and PSB	
	NBA
	OBA
	MBR
	PSB
Data communications	none
Region control and performance	
	IFP
	OPT
	PRLD
	PREINIT
	STIMER
	DIRCA
	CPUTIME
	DBLDL
	SSM
	ALTID
	PARDLI
Recovery and restart	
	TLIM
	SOD

Table 29. Fast Path dependent region parameters for DCCTL and DB/DC (continued)

Category	Parameter
Security options	
	IMSID

The EXEC statement's first positional parameter, IFP, causes this region to be active for Fast Path processing only. The MBR parameter specifies the name of the message-driven application program.

Two database-related parameters, NBA and OBA, control how many of the total Fast Path buffers this region can appropriate for its use. If you are using the Fast Path 64-bit buffer manager, the combined values of NBA and OBA make up the maximum number of buffers reserved for storage. If you are not using the Fast Path 64-bit buffer manager, DBBF controls the maximum number of buffers reserved for storage.

The NBA parameter specifies how many buffers are reserved for DEDB and MSDB processing by this region. This normal allotment is obtained at region startup and must be available from the total number specified for the control region. Specify a number from 1 to 9 999. If you are not using the Fast Path 64-bit buffer manager, coordinate the value across all regions that are to be concurrently active and specify the total on the DBBF parameter. If too few buffers are available from the DBBF total, the dependent region abends. If you are using the Fast Path 64-bit buffer manager, IMS controls the expansion of the buffer pools, and the DBBF parameter is not valid.

The OBA parameter specifies how many buffers are requested from the control region as overflow when the normal allotment is used up. The control region page fixes the largest number of overflow buffers specified for all active BMPs and CCTL threads. If you are not using the Fast Path 64-bit buffer manager, the buffers are made available to only one program at a time. If you are using the Fast Path 64-bit buffer manager, multiple dependent regions can extend into the overflow buffers concurrently.

Both the NBA and OBA parameters are generated with a 0 (zero) value so that you must respecify the individual values.

Fast Path parameters in BMP and CCTL regions in DBCTL

The Fast Path positional parameters (PARM1= and PARM2=) for batch message processing and CCTL regions in a DBCTL environment are described in this topic.

The PARM1= and PARM2= positional parameters for a BMP region's EXEC statements are shown below. The Fast Path parameters in the DRA startup table that the CCTL must use when it connects to a DBCTL environment are also shown.

Table 30. Fast Path dependent region parameters for DBCTL.

Category	Parameter	Purpose
Database and PSB	NBA	Specifies the number of database buffers
	OBA	Specifies the number of overflow buffers
	MBR	Specifies name of message-driven program
	PSB	Specifies PSB name

Table 30. Fast Path dependent region parameters for DBCTL (continued).

Category	Parameter	Purpose
CCTL	CNBA	Total number of buffers for this CCTL
	FPB	Number of database buffers each thread uses (from the CNBA total)
	FPOB	Number of overflow buffers each thread might need

Two database-related parameters, NBA and OBA, control how many of the total Fast Path buffers this region can appropriate for its use.

The NBA or CNBA parameter specifies how many buffers are reserved for DEDB processing. This normal allotment is obtained at region startup and must be available from the total number specified for the control region. Specify a number from 1 to 9999. If you are not using the Fast Path 64-bit buffer manager, coordinate the value across all regions that are to be concurrently active and specify the total on the DBBF parameter. If too few buffers are available from the DBBF total, the dependent region abends. If you are using the Fast Path 64-bit buffer manager, IMS controls the expansion of the buffer pools, and the DBBF parameter is not valid.

After a CCTL is connected, each CCTL thread request for Fast Path PSBs receives an allotment of buffers.

The OBA or FPOB parameter specifies how many buffers are requested from the control region as overflow when the normal allotment (NBA, FPB) is depleted. The control region page fixes the largest number of overflow buffers specified for all active BMPs and CCTL threads. If you are not using the Fast Path 64-bit buffer manager, the buffers are made available to only one program at a time. If you are using the Fast Path 64-bit buffer manager, multiple dependent regions can extend into the overflow buffers concurrently. Choose a value from 1 to 9999 that is a suitable common value for all regions.

Both the NBA and OBA parameters are generated with a 0 (zero) value so that you must respecify the individual values.

Related reading: For more information about DEDB buffer considerations in a DBCTL environment, see *IMS Version 11 Database Administration*.

Online DEDB utility region parameters in DCCTL, DBCTL, or DB/DC

There are several positional parameters used by the Fast Path Online DEDB utility in DCCTL, DBCTL, and DB/DC environments.

The procedure FPUTIL uses the first few positional parameters defined for the IMSFP procedure, but interprets several of the parameters as control for the online DEDB utilities. Table 31 shows these parameters.

Table 31. Generated values and purposes for FPUTIL procedure parameters

FPUTIL Parameter	Generated Value	Purpose
IFP	IFP	Specifies Fast Path region
DBD	N/A	Specifies the DEDB name
N/A	DBF#FPU0	Utility program name

Table 31. Generated values and purposes for FPUTIL procedure parameters (continued)

FPUTIL Parameter	Generated Value	Purpose
REST	00	Restart indicator
N/A	00	No overflow buffers
N/A	Null	Default startup, ask operator
N/A	1	Allows one abend
DIRCA	02	2 KB block for PCB
PRLD	Null	No preload
N/A	0	No time limit

The function performed by the utility depends on the input control statements. A series of DEDB areas can be scanned, have dependent segments deleted, or have the units of work reorganized. If the utility is to be restarted, the REST parameter is coded nonzero.

Recovery-related EXEC parameters for the control region

The EXEC parameters that you can use to control the kind of recovery that is performed for the current system execution include ARC=, DBRC=, DBRCNM=, WADS=, QTU=, AND QTL=.

ARC=

Use the ARC parameter to specify whether automatic archiving of the online log data sets (OLDS) is to be performed. Automatic archiving is recommended; however, you can arrange for the MTO to monitor the availability of the OLDS and perform archiving when necessary. Specify ARC=0 for no automatic archiving; otherwise, specify a one- or two-digit number representing the number of full (closed) OLDSs that invoke automatic archiving for those data sets.

DBRC=

Batch and utility regions use the DFSIDEF0 module during initialization. The DFSIDEF0 module that is shipped in the ADFSSMPL library contains a batch and utility region default of DBRC=YES. This value is coded on the DFSIDEF macro.

You are not required to use the DFSIDEF0 module. If you do not use the module, or if the module cannot be loaded at initialization time, IMS defaults to DBRC=YES. Therefore, if DBRC is to be used for your batch and utility regions, then creating the DFSIDEF0 module is not required.

For online environments (DBCTL, DCCTL, or DB/DC), the only valid value is DBRC=YES.

For batch and utilities, you might want to change the default or current value for DBRC.

For batch and utilities. the default is DBRC=Y. To change the default value for DBRC, you can do either of the following:

- Assemble and bind DFSIDEF0 into the IMS execution library; you can specify DBRC=Y, DBRC=N, or DBRC=FORCE. If you specify RMODE, you must specify RMODE=24 and AMODE=24

- Override the value in a DFSPBxxx PROCLIB member. You can specify only DBRC=N or DBRC=Y.

  To do this, you must define the RGSUF= parameter in your DLIBATCH or DBBBATCH procedure.

To change the current value for DBRC, override the value in JCL. You can specify only DBRC=N or DBRC=Y.

Sample JCL to assemble and bind the DFSIDEF0 module is shown below:

```
//ASSEMBLE EXEC PGM=ASMA90,PARM='NOOBJ,DECK'
//SYSLIB DD DSN=IMS.SDFSMAC,DISP=SHR
//SYSPUNCH DD DISP=OLD,DSN=IMS.OBJDSET(DFSIDEF0)
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD UNIT=SYSDA,DISP=(,DELETE),SPACE=(CYL,(15,15))
//SYSIN    DD *
IDEF0    TITLE 'DFSIDEF0 - IMS INSTALLATION DEFAULTS BLOCK'
DFSIDEF0 CSECT
         SPACE 1
         DFSIDEF TYPE=BEGIN
         DFSIDEF TYPE=PARM,DBRC=YES
*****    DFSIDEF TYPE=PARM,DBRC=NO
*****    DFSIDEF TYPE=PARM,DBRC=FORCE
         DFSIDEF TYPE=END
         END   DFSIDEF0
//STEP1 EXEC PGM=IEWL,
//            PARM='SIZE=(880K,64K),NCAL,LET,REUS,XREF,LIST'
//SYSPRINT DD  SYSOUT=*
//SYSPUNCH DD DSN=IMS.OBJDSET,DISP=SHR
//SYSLMOD  DD DSN=IMS.SDFSRESL,DISP=SHR
//SYSUT1   DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSPUNCH)),SPACE=(CYL,(10,1))
//SYSLIN   DD *
 INCLUDE SYSPUNCH(DFSIDEF0)
 NAME DFSIDEF0(R)
```

DBRCNM=

Specify a name (do not accept the default) for the start procedure for the DBRC region on the DBRCNM parameter. The default is a null parameter indicating a procedure name of DBRC. Also as part of your installation's recovery strategy, the AUTO parameter specifies whether to use automatic restart of the IMS online system. You need to override the generated default value (N) by specifying AUTO=Y if you plan to use automatic restart.

QTU= and QTL=

For a non-shared queues environment, the QTU and QTL parameters are associated with protection of the message queues. Use them to adjust the values in the message queue space notification exit routine (DFSQSPC0). Given the number of records reserved on each message queue data set for automatic shutdown (specified in system definition with the SHUTDWN parameter on the MSGQUEUE macro), a finite number of residual records are available for message use. The notification exit routine keeps track of the current level of utilization. If the usage exceeds the upper limit (expressed as a percentage of records used), the MTO is notified so that steps can be taken to reduce the backlog. The MTO is also notified if the usage falls below the lower limit.

For a shared queues environment, the QTU and QTL parameters are used to monitor the device relative record number (DRRN) in-use count. If the high threshold value is reached, a DFS2281I message is issued, and all messages

retrieved from shared queues are put in an IWAIT state. When the DRRN in-use count becomes less than the low threshold value, a DFS2282I message is issued, and the messages in an IWAIT state are then posted.

The QTU and QTL values are one or two-digit percentages. QTU overrides the upper threshold value and QTL overrides the lower one (100 is allowed for QTU). Both upper and lower threshold values are generated with null values; the defaults are those of the IBM-supplied exit routine—75% and 60%, respectively. Unless your installation establishes its own exit routine to control the MTO notification, you should probably use these defaults. In the long term, you can tune them to fixed values based on the feedback from the statistics given in the `/DISPLAY POOL` command; enter this command during peak periods of activity.

WADS=

The WADS parameter indicates whether the IMS online system is to use a single write-ahead data set (WADS) or dual data sets. The WADS helps protect the integrity of the online logging. In case of write errors, additional data sets can be defined up to a total of 10. Specify WADS=S for single recording or WADS=D for dual recording.

Security-related EXEC parameters for the control region

Several parameters control the kind of security checking that is done during the current execution by acting as switches for different types of security.

Moreover, these parameters determine how much flexibility the MTO has to override the choice of security checking. You must coordinate the setting of these switches with both overall security design and operational procedures. The parameters are RCLASS, SECCNT, TRN, SGN, RCF, ISIS, ASOT, ALOT, AOI1, AOIS, and TCORACF.

All the values generated for the IMS procedure parameters specify no security. You need to reset them to match the security design. You must also coordinate the level of the security tables with the suffix identifier for the nucleus.

Related reading: Operational restrictions for the MTO, and the parameter values and their meaning are discussed in *IMS Version 11 System Administration*.

The IMSID parameter is partially related to security and to operations. The one- to four-character value that you specify uniquely identifies the control region. Dependent regions executing under control of this nucleus must specify the same identifier. The definition default is IMSA.

Any message sent to the console from subsystem execution is identified by this name; therefore, the name chosen should be unique to other subsystems executing in the z/OS system, including any batch executions. The IMSID name should not be the same as the procedure name in order to avoid confusion as to the source of the message.

EXEC parameters for IMS message processing regions

Using IMS EXEC parameters, you can control many IMS message processing region characteristics including data communications, PSBs, performance, recovery and restart, and security.

These EXEC parameters do not apply to DBCTL.

The PARM= positional parameters for a message processing region's EXEC statement are shown in Table 32. See the DFSMPR procedure for more information.

Table 32. Categories and purpose of message processing region parameters

Category	Parameter	Purpose
Database and PSB	PCB	Specify size of pool for PSB copy
	VALCK	Check validity of addresses in DL/I calls
	NBA	Specify Fast Path database buffers
	OBA	Specify Fast Path overflow buffers
Data Communication	CL1	Specify priority of message class
	CL2-4	Specify up to three lower-priority message classes
Region Control and Performance	LOCKMAX	Turn off locking limitations
	MSG	Specify MSG for message region
	OPT	Coordinate region start with control region
	OVLA	Retain overlay supervisor
	PRLD	Specify suffix of DFSMPLxx for program preload
	PREINIT	Specify suffix of DFSINTxx for modules to be preinitialized in readiness for takeover
	STIMER	Specify timer options
	DBLDL	Specify maximum number of IMS.PGMLIB directory entries to be saved
	SOD	Specify output class for spin-off dump
	SSM	Specify access to DB2 for z/OS subsystems
	ALTID	Specify alternate IMS system
	PWFI	Activate pseudo wait-for-input
	APARM	Specify application parameter
Recovery and restart	SPIE	Allow user program SPIE during DL/I call
	TLIM	Specify number of abnormal terminations allowed
Security options	IMSID	Override IMS subsystem identifier

PSB-related EXEC parameters for a message processing region

The PCB parameter is an optional parameter that specifies the size of the interregion communication area. This area holds the PCBs included in the PSBs that are used by programs executing in this region. If you do not specify this parameter, the default value is 000. This allocates an area large enough to hold all the PCBs for the largest PSB defined to this IMS system and present in the active IMS.ACBLIBA/B. To specify a value, enter the number of 1024-byte blocks as a three-digit number (for example, 001).

The default value for the VALCK parameter signifies that address validity checking is not to be performed for DL/I calls issued by the application programs in this region. (An address is invalid if it is either lower than the lowest address not in the z/OS nucleus or higher than the highest address in virtual storage.) With adequate testing of and controls over the DL/I call parameter coding, validity checking should not be necessary.

Data communication EXEC parameters for a message region

A message region requires the set of four parameters, CL1=, CL2=, CL3=, CL4=, to specify transaction classes. Express each class as a three-digit number. The first message class for the region causes all messages assigned to that class to be selected first as eligible for scheduling. Only when all possibility of scheduling a transaction in that class has been exhausted does scheduling begin for the second message class. Priorities determine the order that programs are to be selected for scheduling into the region. The message classes you specify need to be coordinated with your transaction scheduling rules and the numbers entered with the PRTY keyword on the TRANSACT macro.

Region control EXEC parameters for a message region

MSG is the first positional parameter for a message processing region. When transactions are encountered on the queue, programs are automatically scheduled in these regions, if the message class priority is suitable.

The parameter OPT= helps you control the region startup. If, for some reason, the control region is not active or is terminating when the MPP region is invoked, you can have the MTO decide whether to start the MPP region again (the default), let it wait until the control region is ready, or cancel it.

Use STIMER= to invoke timer facilities. If you use the non-default LSO option, LSO=S, both application program and DL/I processing times are recorded. Most DL/I processing time is not included when you use the default, LSO=Y. A value of 1 causes a timer sequence to be issued once for each DL/I call that can adversely affect the performance of the region. This parameter is used with the TIME= parameter on the //JOB statement.

Recommendation: Because a timeout of a message region causes an abnormal termination of the message region, have processing limits coded within the application program.

Use the SSM= parameter, which is required when using DB2 for z/OS group names to point to a member in IMS.PROCLIB that identifies the DB2 for z/OS subsystems that can be accessed from this MPP region. To prevent any access to subsystems from an MPP scheduled into this region, use the name of a null member (no entries). The SSM= default, a null value, allows the MPP region to

attach to any of the subsystems declared to the IMS control region. If necessary, coordinate this parameter with the corresponding SSM parameter in the IMS procedure. Refer to the description in "System control and performance EXEC parameters for the IMS control region" on page 147.

Use the APARM= parameter to specify execution time parameters unique to this dependent region. This parameter specifies a character string for the application program or Data Capture exit routine. The APARM= can be used to specify the frequency for checkpoint calls. The APARM= can also be used to pass parameters to the Data Capture exit routine to indicate whether the data should be captured.

The maximum length of the parameter is 32 characters; the parameter must be enclosed in single quotation marks (') if special characters are used; embedded commas are not allowed. The INQY call with the ENVIRON subfunction is used to receive the APARM character string.

Related reading: For more information, see *IMS Version 11 Application Programming*.

Use the SOD parameter to specify the output class for a spin-off dump. In this way, you can obtain a printed copy of the storage immediately at abnormal termination of the message region immediately rather than being placed on general SYSOUT queues.

Performance-related EXEC parameters for a message region

Four performance-related options are available. The parameter OVLA= enables you to retain a copy of the overlay supervisor in the message region. The default is to have the overlay control loaded each time you schedule an overlay application program in the region. If you have arranged the message priorities so that an overlay program is likely to be scheduled in the region, you should use this option.

Use the DBLDL= parameter to minimize the cost of program loading by increasing the number of program directory entries maintained in the message region. The effect is to reduce the I/O to the program library (and the search) to obtain the direct address of the program. The default is 20.

Use the PRLD= parameter to specify the two-character suffix of the member DFSMPLxx in IMS.PROCLIB. This member contains the list of programs that are to be preloaded. The effect is to reduce the repetition of I/O for program load. The region address space retains the modules.

PREINIT allows you to specify the two-character suffix of the member DFSINTxx in IMS.PROCLIB. This member contains the preinitialization modules to are to receive control.

The type of program load selected depends on the order of the following:

1. Preload list
2. Standard z/OS fetch

The fourth performance-related parameter is the PWFI= (Pseudo WFI) parameter. Specifying PWFI=Y can potentially reduce CPU time by eliminating the termination and rescheduling of resources. On a message GU call, if MODE=SNGL is specified on the TRANSACT macro and no more messages are available, IMS checks to see if other work needs to be done for the region. If no other work is

available and the resources owned by the region are not needed by another region with work to do, the region becomes idle and waits until another message is queued (wait-for-input mode). It does not return a "QC" status code to the application program. If the new message is for the transaction that is scheduled in the region, IMS returns the new message to the application program without having to terminate and reschedule the resources.

If the next message that comes in is not for the transaction that is scheduled in the region, but it can be processed by the region, there is a delay before the new transaction can be processed. Because termination of the previous message is delayed until after the new message comes in, the region must go through termination before it can schedule the new transaction.

Recovery-related EXEC parameters for a message region

The TLIM parameter addresses a problem with an application program that causes an abnormal termination. Because the program can be scheduled many times into a region as a result of transactions in the queue, you need to be able to stop the operation of this region. The value for TLIM specifies the limiting number of abnormal terminations permitted. In the case where the application program has a SPIE in effect, the SPIE option allows it to be left on during the DL/I call or for it to be turned off during the DL/I call and reinstated when returning to the application program. For performance reasons, it is not recommended to turn it on and off for each DL/I call. With PL/I Release 5 or later, you can use the PL/I SPIE facility without having IMS reset the SPIE on each DL/I call.

Security-related EXEC parameters for a message region

The parameter IMSID is related to security and to operations. You specify the 1- to four-character identifier for the name of the control region (the name given for the IMSID parameter). If the value does not match the current IMSID of any operating control region, the message region is not scheduled.

EXEC parameters for IMS batch message processing regions

Using EXEC parameters, you can override definitions for IMS batch message processing regions that were initially set during IMS system definition, including definitions for databases, PSBs, data communications, region control and performance, recovery and restart, and security.

The PARM= positional parameters for the BMP region's EXEC statement are shown in Table 33. See the IMSBATCH procedure for more information.

Table 33. Categories and purpose of BMP region parameters

Category	Parameter	Purpose
Database and PSB	MBR	Specify name of batch message program
	PSB	Specify PSB name
	TEST	Check validity of addresses in DL/I calls
	NBA	Specify Fast Path database buffers
	OBA	Specify Fast Path overflow buffers
Data Communication	IN	Specify an input transaction queue
	OUT	Specify an output transaction queue

Table 33. Categories and purpose of BMP region parameters (continued)

Category	Parameter	Purpose
Region Control and Performance	BMP	Specify BMP for batch message region
	OPT	Coordinate region start with control region
	LOCKMAX	Specify locking limitations
	PRLD	Specify suffix of DFSMPLxx for program preload
	PREINIT	Specify suffix of DFSINTxx for modules to be preinitialized in readiness for takeover
	STIMER	Specify timer options
	CPUTIME	Specify processing time limit
	PARDLI	Specify region for DL/I processing
	DIRCA	Specify interregion communication size
	SSM	Specify access to DB2 for z/OS subsystems
	ALTID	Specify alternate IMS system
	APARM	Specify application parameter
Recovery and Restart	SPIE	Allow user program SPIE during DL/I call
	CKPTID	Specify checkpoint for program restart
Security options	IMSID	Override IMS subsystem identifier

PSB-related EXEC parameters for a batch message region

The MBR= parameter specifies the name of the program and is often the same as the PSB name. This parameter is required. The BMP has flexibility in using a program and PSB combination. This allows you to test modifications of the BMP using a temporary program name. You can also use a different PSB with the same program.

The PSB= parameter is optional if it matches the MBR name. If an APPLCTN macro has been included for the PSB specifying BATCH as the program type, the PSB parameter specifies one of several PSBs for a given BMP. You should identify application programs that can process several types of transactions. These programs often require larger amounts of virtual storage, and you can need to adjust the size of the region.

The TEST= parameter is required, but the IMSBATCH procedure generates a 0 to specify that validity checking of the addresses in a DL/I call is not performed. (An address is invalid if it is either lower than the lowest address not in the z/OS nucleus or higher than the highest address in virtual storage.) Adequate testing of the program and controls over the DL/I call parameter list coding should make the generated option (no validity checking) appropriate.

The NBA= parameter specifies the number of Fast Path database buffers to be made available, up to the maximum number available. If you are using the Fast Path 64-bit buffer manager, it manages the segmentation of the buffer pool into subpools. Otherwise, the buffers are taken from the global pool that is defined by the DBBF= parameter in the IMS control region procedure.

The OBA= parameter specifies the number of additional database buffers to be made available in the CSA/ECSA, up to the maximum number available. Fast Path allocates only the largest OBA specification from all currently active regions. If you are using the Fast Path 64-bit buffer manager, be aware that multiple regions can access the overflow buffers concurrently.

Data communication EXEC parameters for a batch message region

For the batch message program, you have the flexibility of declaring that the input transaction queue is to be made available to the program at execution time. You do this by specifying one transaction code as the value for the IN parameter.

Some BMP programs do not access a message queue; however, they have a requirement to send output to a terminal or to generate transactions to be processed by other application programs. Specify the LTERM name or transaction code (as appropriate) on the OUT parameter.

A null value for the IN parameter prevents the program from accessing message queues. When you specify the transaction code for the IN parameter, the program has no restrictions on generated transactions or output messages.

Region control EXEC parameters for a batch message region

BMP= is the first positional parameter for a batch message processing region. These regions are not scheduled automatically. They must be invoked by the operator.

The OPT= parameter helps you control the start of a batch message region. If the control region is not active when the BMP region is invoked, you must decide whether to wait for a control program, cancel the batch message region, or ask the operator to decide. This might occur when JCL in the z/OS job stream for the BMP region is scheduled before the control region has completed its initialization or when the control region is terminating. Specifying 'wait' is a risk because the z/OS resource is reserved until the control region resumes. If you are starting the region from the master terminal, the default generated for the IMSBATCH procedure is satisfactory.

The optional PRLD= parameter specifies the two-character suffix for the member DFSMPLxx in IMS.PROCLIB, which lists the preloaded modules. It is optional; it is generated as a null parameter for the IMSBATCH procedure. For BMP regions that do not have other programs scheduled in them, PRLD provides no performance gain.

The PREINIT= parameter specifies the two-character suffix of the member DFSINTxx in IMS.PROCLIB. This member contains the preinitialization modules that are to receive control.

The optional STIMER= and CPUTIME= parameters are generated as null parameters, their defaults are no use of the timer.

The optional PCB= parameter specifies the size of the interregion communication area. This area holds the PCBs included in the PSBs that are used by programs executing in this region. If you do not specify this parameter, the default value is 000. This allocates an area large enough to hold all the PCBs for the largest PSB

defined to this IMS system and present in the active IMS.ACBLIBA/B. To specify a value, enter the number of 1024-byte blocks as a three-digit number (for example, 001).

Use the SSM parameter to reference a member in IMS.PROCLIB that identifies the DB2 for z/OS subsystems that can be accessed from this BMP region. To prevent any access to a DB2 for z/OS subsystem from the BMP, use the name of a null member (no entries). The default (a null value) allows the BMP region to attach to any of the subsystems declared to the IMS control region. If necessary, coordinate this parameter with the corresponding SSM parameter in the IMS procedure. Refer to the description in "System control and performance EXEC parameters for the IMS control region" on page 147.

The FMTO= parameter and the type of z/OS dump DD statements selected determine whether IMS dumps are formatted online or offline.

Use the APARM= parameter to specify execution-time parameters unique to this dependent region. This parameter specifies a character string for the application program or Data Capture exit routine. For batch or BMP regions, the APARM= can be used to specify the frequency for checkpoint calls. The APARM= can also be used to pass parameters to the Data Capture exit routine to indicate whether the data should be captured.

The maximum length of the parameter is 32 characters; the parameter must be enclosed in single quotation marks (') if special characters are used; embedded commas are not allowed. The INQY call with the ENVIRON subfunction is used to receive the APARM character string.

Recovery-related EXEC parameters for a batch message region

For a BMP program, you can use the CKPTID= parameter to set a restart position for the program processing. The IMSBATCH procedure generates CKPTID as a null parameter. To invoke restart, create a special version of the procedure containing the exact checkpoint identification. This can be a 1- to 8-character extended checkpoint identifier generated by the program itself, or a 14-character time stamp identifier issued within a message from checkpoint processing. Alternatively, you can specify the value *LAST* as the identifier to cause the last recorded checkpoint to be used. One technique is to restart BMPs with JCL entered in the system reader and have the CKPTID symbolic parameter coded for the EXEC statement.

Related reading: For more details regarding BMP restart for BMPs that use extended checkpoint, refer to *IMS Version 11 Operations and Automation*.

In the case where the application program has a SPIE in effect, you specify whether the SPIE is to remain on or off during the DL/I call. Negated SPIEs are reinstated before returning to the application program.

Recommendation: For performance reasons, do not turn the SPIE on and off for each DL/I call. With PL/I Release 5 and later releases, you can use the PL/I SPIE facility without having IMS reset the SPIE on each DL/I call.

Security-related EXEC parameters for a batch message region

The parameter IMSID is related to security and to operations. You specify the 1-to 4-character identifier for the name of the operational IMS DB/DC environment. If

the value does not match the current IMSID of any operating control region, the batch message region is not scheduled.

Chapter 6. Tailoring the IMS system to your environment

Tailoring your IMS system includes tailoring the procedure library, buffer pools, PL/I modules, sequential buffering, high-speed sequential processing, DBCTL, ETO, and HALDB.

Tailoring the IMS PROCLIB data set

The IMS PROCLIB data set contains procedures that are built during system definition. The procedures are also called members of the IMS PROCLIB data set, and they execute various system operations. You can modify these members to meet your system needs.

Output from Stage 1 system definition includes a list of updates to be made to the IMS PROCLIB data set, which is comprised of several different members. Stage 2 system definition then applies those updates to the appropriate members of the IMS PROCLIB data set. To tailor the IMS PROCLIB data set to meet your system needs, you can modify the IMS PROCLIB data set in one of three ways:

- Tailor the contents of the IMS PROCLIB data set members before implementing Stage 2 system definition.
- Directly alter the contents of the IMS PROCLIB data set members.
- Use the IMS Syntax Checker.

During system definition, you can initially use default values for the parameters (*startup* parameters), but you can also specify them or change the value before starting the region using symbolic PARM1= and PARM2= parameters (*executive*, or EXEC, parameters). For example, individual region control or tuning recommendations might necessitate overriding the initial or default values.

IMS PROCLIB data set members generated by system definition

When you perform an IMS system definition, IMS generates members of the IMS PROCLIB data set if PROCLIB=YES is specified (or defaulted to) on the IMSGEN macro, and you are generating a CTLBLKS, NUCLEUS, ON-LINE, or ALL type of system definition. If you previously made changes to a member of the IMS PROCLIB data set and want to save those changes, specify PROCLIB=NO on the IMSGEN macro.

If you perform a CTLBLKS, NUCLEUS or ON-LINE system definition and made changes that affect line groups, unit addresses, or spool output data sets, specify PROCLIB=IMS on the IMSGEN macro to replace only the IMS member, which executes the control region.

The generated members of the IMS PROCLIB data set can be grouped into several categories depending on their use, as shown in the table that follows.

© Copyright IBM Corp. 1974, 2015

Table 34. Categories of generated IMS PROCLIB data set members

Category	PROCLIB member	What the member is used for
Online operation	DBC	Executes the DBCTL region
	DBRC	Executes the DBRC address space
	DCC	Executes the DCCTL region
	DFSJBP	Executes the Java™ non-message-driven batch address space
	DFSJMP	Executes the Java message program address space
	DFSMPR	Executes the message processing region
	DFSWTnnn	Prints IMS spool data sets
	DLISAS	Executes the DL/I address space
	DXRJPROC	Executes the IRLM region
	FPUTIL	Executes DEDB utilities online
	IMS	Executes the control region
	IMSBATCH	Executes the batch message region
	IMSFP	Executes the Fast Path dependent region
	IMSMSG	Executes the message region
	IMSRDR	Reads the IMSMSG job
	IMSWTnnn	Executes DFSWTnnn, a procedure that executes the Spool SYSOUT Print utility program (DFSUPRT0)

Table 34. Categories of generated IMS PROCLIB data set members (continued)

Category	PROCLIB member	What the member is used for
Online initialization	DFSCGxxx	Specifies CSL parameters
	DFSDCxxx	Specifies data communication options
	DFSDFxxx	Specifies dynamic resource definitions, diagnostics and statistics, CSL parameters, shared message queue, and CQS address space parameters
	DFSDRFnn	List for defining DREF requirements
	DFSFDRxx	Specifies FDBR region execution parameters
	DFSFIXnn	List for page fixing blocks and modules
	DFSHSBxx	Specifies XRF options used by the active and alternate subsystems in XRF complex
	DFSINTxx	Identifies preinitialization modules
	DFSMPLxx	Defines a preload program list
	DFSORSxx	Specifies system-related startup parameters for the recovery manager
	DFSPBDBC	Specifies DBCTL execution parameters
	DFSPBDCC	Specifies DCCTL execution parameters
	DFSPBIMS	Specifies DB/DC execution parameters
	DFSRSRxx	Specifies Remote Site Recovery execution parameters
	DFSSPMxx	Specifies storage pool management parameters
	DFSSQxxx	Specifies shared message queue and CQS address space parameters
	DFSVSMxx	Defines buffer pools, logging attributes, and other options
	DFSYDT	Specifies OTMA descriptors
	DFS62DT	Specifies LU 6.2 device descriptors
Online preparation and maintenance	ACBGEN	Maintains the ACBLIB
	IMSDALOC	Maintains dynamic allocation
	IMSMSV	Offline utility for MSC
	MFSDCT	Maintains MFSDCT and new MFS default formats
	MFSRVC	Maintains MFS service and index
	MFSUTL	Maintains the MFS library. Related procedures include MFSBTCH1/2, MFSTEST, MFSBACK, and MFSREST
	OLCUTL	Prepares IMS for online change
Batch operation	DBBBATCH	Executes stand-alone DL/I batch processing region
	DLIBATCH	Executes stand-alone DL/I batch processing region with block building
Alternative batch execution	IMSCOBGO	Compiles, binds, and executes (under batch) a COBOL program
	IMSPLIGO	Compiles, binds, and executes (under batch) a PL/I program

Table 34. Categories of generated IMS PROCLIB data set members (continued)

Category	PROCLIB member	What the member is used for
Database definition and access	DBDGEN	Maintains the DBDLIB
	DBFMSDBx	Specifies MSDBs
	FPUTIL	Executes online DEDB utilities
	MFDBDUMP	Program to list sample databases
	MFDBLOAD	Batch execution of a program to load the sample databases
Application program preparation	CBLTDLI	Binds control statements for COBOL
	DFSJVMAP	Maps IMS Java application name to OMVS path name for Java class file
	IMSCOBOL	Compiles and binds COBOL program
	IMSPLI	Compiles and binds PL/I program
	PLITDLI	Binds control statements for PL/I
	PSBGEN	Maintains the PSBLIB

Controlling the IMS PROCLIB data set

Carefully examine the procedures generated as a result of your system definition.

Although many of the procedures generated in the IMS PROCLIB data set require alteration before they can be used in direct execution of the online system, they do provide a convenient start to the task of defining execution JCL. Many of the members have content that directly indicates the options you specified in the system definition stage 1 input. For example, the online execution member IMS includes a DD statement for IMSMON if the IMSCTF macro specifies LOG=MONITOR.

You should carefully examine the procedures generated as a result of your system definition.

Renaming the IMS PROCLIB data set members

Rename the IMS PROCLIB data set members according to your installation's requirements. The names can follow a convention that suggests ownership by a particular application system or a convention that has an implied sequence. For example:

IMSIMAG
Initial procedure for image copies of database

IMSCTL
Control region startup (IMS renamed)

IMSTXLD
BMP to preload a transaction queue (IMSBATCH)

IMSMSG1
Message region startup (IMSMSG)

IMSBCH1
Low-priority BMP (IMSBATCH)

IMSMSG2
Second message region when required (IMSMSG)

IMSWT000
Spool output print procedure (named by IMS)

The system operator or master terminal operator (MTO) uses the procedure names to start the job on z/OS.

To develop these members, you need to either rename the members in the IMS PROCLIB data set or create new members. In some installations, the procedures are added to SYS1.PROCLIB. One option of the NODE keyword on the IMSGEN macro allows you to substitute an alternative library naming convention, so that your base PROCLIB data set can be named LEGAL.PROCLIB.

Altering members of your PROCLIB data set

You must alter the JCL content of the members of the PROCLIB data set. The updates you apply follow naming conventions for your IMS systems and the required DD statements.

Table 35 summarizes the ways you can tailor the PROCLIB data set member, excluding arrangements for batch executions. In all procedures, you might want to add JCL comment statements for additional documentation. Your PROCLIB data set might not contain all the members listed below.

Table 35. Tailoring actions for PROCLIB data set members for batch executions

PROCLIB data set member	Tailoring action
IMS	• Checks and overrides the EXEC PARM parameter. • Specifies region size again. • Moves to SYS1.PROCLIB. • Adds database DD statements, if not dynamically allocated. • Specifies logging data sets again; allows for dynamic allocation.
IMSRDR	• Adjusts all symbolic parameters for message region start by system operator. • Moves to SYS1.PROCLIB.
DBC	• Checks and overrides EXEC PARM parameter. • Specifies region size again. • Moves to SYS1.PROCLIB. • Adds database DD statements, if not dynamically allocated. • Specifies logging data sets again, allow for dynamic allocation.
DBRC	• Adds RECON DD statements, if not dynamically allocated. • Renames the procedure, if desired, and specifies this name with the DBRCNM parameter of the IMS procedure. • Adjusts procedures JCLOUT and JCLPDS. • Moves to SYS1.PROCLIB.
DFSDCxxx	Specifies data communication options.

Table 35. Tailoring actions for PROCLIB data set members for batch executions (continued)

PROCLIB data set member	Tailoring action
DFSFDRxx	• Defines the IMSID tracked by the active FDBR. • Defines whether FDBR applies to DEDB areas defined as SHARELVL=0|1. • Defines z/OS cross-system coupling facility group name for the active IMS system and the FDBR tracking region. • Specifies the number of seconds before FDBR goes into timeout status. • Specifies whether control blocks for DEDB processing are allocated in private storage or ECSA.
DLISAS	• Adds full-function database DD statements and removes the corresponding statements from the IMS procedure. (No changes are necessary for dynamically allocated databases). • Specifies sizes of two PSB pools using the DLIPSB and CSAPB parameters of the IMS procedure. • Moves the pools to SYS1.PROCLIB. • If the procedure is renamed, specifies this name with the DLINM parameter of the IMS procedure.
IMSMSG	• Adjusts JOB statement parameters for DFSMPR execution. • Moves to IMS.JOBS
DFSMPR	Checks and specifies EXEC PARM parameters.
IMSBATCH	• Checks and specifies EXEC PARM parameters. • Add z/OS DD statements.
DFSVSMxx	• Defines buffer pools and other options. • Allows use of sequential buffering. • Specifies initial WADS and OLDS and dual logging. • Coordinates with IMS procedure. • Enables long busy handling.
DFSHSBxx	Establishes IMS active and alternate subsystems in XRF complex.
DFSDRFxx	Defines DREF requirements list.
DFSFIXxx	Defines page fix list.
DFSINTxx	Defines, loads, and executes user-written exits during dependent region initialization.
DFSMPLxx	Defines preload program list.
DFSSQxxx	• Defines PROCLIB data set member for CQS address space. • Defines subsystem name for CQS address space. • Defines name for shared expedited message handler queues. • Defines primary structure containing shared message queues. • Defines XCF IMS shared queues group name.
IMSWTnnn	• Adjusts JOB statement parameters for DFSWTnnn execution. • Moves to IMS.JOBS.
DFSWTnnn	Checks and specifies SYSOUT class for spooled output.
DXRJPROC	Establishes procedures for each required IRLM address space.

Table 35. Tailoring actions for PROCLIB data set members for batch executions (continued)

PROCLIB data set member	Tailoring action
DFSPBIMS	• Renames to DFSPBxxx, where xxx is determined by IMS procedure RGSUF=xxx. • Defines IMS control region execution parameters.
DFSPBDBC	• Renames to DFSPBxxx, where xxx is determined by DBC procedure RGSUF=xxx. • Defines DBC control region execution parameters.
DFSPBDCC	• Renames to DFSPBxxx, where xxx is determined by DCC procedure RGSUF=xxx • Defines DCC control region execution parameters.

Preparing for IMS job execution

In preparation for executing the IMS procedure as a system task, the members IMS, IMSRDR, DBC, DCC, DBRC, and DLISAS are moved to SYS1.PROCLIB. Additional generated members of the IMS PROCLIB data set, especially IMSMSG and the set of IMSWTnnn members for spool output, need to be tailored to satisfy your installation's requirements. IMSMSG invokes the procedure DFSMPR, and the IMSWTnnn members invoke DFSWTnnn. To enable message region and spool output jobs to be started with IMS commands or from the system console, these members, after tailoring, are moved to the IMS.JOBS data set, which is concatenated with SYS1.PROCLIB.

Preparing IMS PROCLIB data set member DFSPBxxx

IMS PROCLIB data set member DFSPBxxx contains IMS control region execution parameters. However, the values specified on the EXEC statement override (but do not nullify) any parameters specified in DFSPBxxx. To build or update a DFSPBxxx member, use the IMS Syntax Checker.

IMS also creates sample DFSPBxxx members during system definition. The sample members are as follows:

- DFSPBIMS for IMS DB/DC
- DFSPBDBC for DBCTL
- DFSPBDCC for DCCTL

These samples contain all the valid parameters for the specified IMS control region. To use the DFSPBxxx member, code RGSUF=xxx on the invocation of the IMS procedure.

Tailoring Fast Path execution procedures in DB/DC or DCCTL

As a result of defining Fast Path application programs in system definition macros, the contents of the IMS PROCLIB data set include two procedures for executing Fast Path dependent regions in a DB/DC or DCCTL environment:

IMSFP
To execute a region containing a Fast Path application program

FPUTIL
To execute online DEDB utilities

You must tailor both of these procedures to the requirements of individual programs. Also, several of the other members of the IMS PROCLIB data set have EXEC statement parameters that specifically apply to Fast Path or need to reflect Fast Path requirements. The additional tailoring actions are summarized in the table below.

Table 36. Tailoring actions for Fast Path procedures

IMS PROCLIB data set member	Tailoring action
IMS	Specify buffering and output thread limits Respecify region size. Add appropriate DEDB DD statements. Allocate MSDB initialization data set.
DFSMPR	Specify buffering reserved for this region.
IMSBATCH	Specify buffering reserved for this region.
IMSFP	Specify the application program and region size. Specify buffering reserved for this region.
FPUTIL	Specify the DEDB name and restart indicator. Provide the utility control statements.
DFSFIXxx	Page fix list for Fast Path control blocks.
DBFMSDBx	List of MSDBs and segments to be loaded.

Tailoring Fast Path execution procedures in DBCTL

After defining Fast Path application programs in system definition macros, the contents of the IMS PROCLIB data set in a DBCTL environment include FPUTIL, a procedure that executes online DEDB utilities.

This procedure must be tailored to the requirements of individual programs. Also, several of the other members of the IMS PROCLIB data set have EXEC statement parameters that specifically apply to Fast Path or need to reflect Fast Path requirements. The additional tailoring actions are summarized in the table below.

Table 37. Tailoring actions for Fast Path procedures in a DBCTL environment

IMS PROCLIB data set member	Tailoring action
DBC	Specify buffering and output thread limits Respecify region size Add appropriate DEDB DD statements
IMSBATCH	Specify buffering reserved for this region
FPUTIL	Specify the DEDB name and restart indicator Provide the utility control statements
DFSFIXxx	Page fix list for Fast Path control blocks

Controlling modifications to the IMS PROCLIB data set

Using operator commands keeps the initial operating instructions simple and avoids complex symbolic parameter data entry. However, you must control the exact content of the JCL residing in the IMS PROCLIB data set. For example, the master terminal operator enters:

```
/START REGION IMSBCH1
```

The procedure IMSBCH1 must be correctly coordinated to a known BMP, appropriate PSB and transaction queue, other system options, and identifying parameters.

Your control responsibilities can include auxiliary procedures for database reorganization, recovery, or system output control. Further, the modification level of each procedure must be coordinated to the actual production environment. For example, if an application program is modified and requires more dependent region storage, you must coordinate the program library and IMS PROCLIB data set changes. One technique is to include JCL comment statements that document the date and the reason for the change.

Take special care to check that the physical changes you make in IMS PROCLIB data set members. Many of the DD statements extend over several input records and involve positional parameters, so you need to make more than a cursory examination of a change. You can use a data dictionary to record and maintain the IMS PROCLIB data set members. Changes can then be checked at the terminal or by reviewing listings of the changed members.

Related concepts:

Chapter 14, “IMS Syntax Checker,” on page 283

IMS buffer pools

Three types of buffer pools control the IMS DL/I buffering services: the VSAM shared resource pool, the OSAM buffer pool, and the OSAM Sequential Buffer (SB) pool.

This topic describes the parameters that define the size and content of the IMS buffer pools. Each data set can have no more than one open ACB (VSAM access method control block).

VSAM constructs the VSAM shared resource pool based on parameters provided by the VSAM BLDVRP macro, which is issued during IMS initialization. This pool contains buffers to be used for VSAM data sets (both index and data components) and the input/output-related control blocks necessary to perform VSAM requests. The buffers are combined in subpools. All buffers within a subpool are of equal length.

The OSAM buffer pool is required for IMS online and batch operations. This pool contains buffers to be used for OSAM data set data components and the input/output-related control blocks necessary to perform OSAM requests. The buffers are combined in subpools. All buffers within a subpool are of equal length.

IMS dynamically constructs the OSAM Sequential Buffer (SB) pool when application programs or utilities using SB are active. To allow online applications to use SB, you must provide a SBONLINE control statement in the DFSVSMxx member of the IMS PROCLIB data set for an IMS DB/DC or DBCTL environment.

Each application program or utility that is using SB has its own SB buffer pools. You can control the use of SB by an application program or utility using either PSBGEN, control statements in the //DFSCTL data set, or an SB Initialization exit routine.

VSAM subpool definition

VSAM subpools are defined using the VSRBF control statement, one of the control statements in the DFSVSMxx member of the IMS PROCLIB data set.

Define VSAM subpools using the VSRBF control statement, which is described in "DFSVSMxx member of the IMS PROCLIB data set" on page 738. Specify the required subpools in the DFSVSAMP data set for batch environments or in the DFSVSMxx member of the IMS PROCLIB data set for DB/DC environments. The VSAM buffers and control blocks are above the 16-MB line in most environments.

The minimum number of subpools is one and the maximum is 11. The minimum IMS-calculated number of buffers in a subpool is three, and the maximum is 32,767. This calculation is based on region type, number of PCBs, number of data set groups, and database organization.

A database is in sequential mode if all the following conditions are true:

- The database operates in a batch type region (DLI or DBB)
- The database is referred to by only 1 PCB in the PSB
- The database organization is HISAM, single-segment HISAM, INDEX, or HIDAM accessed through unqualified GN calls

Buffer sizes can be 0.5 KB, 1 KB, 2 KB, 4 KB, 8 KB, and multiples of 4 KB up to 32 KB. IMS accepts any valid CI size up to 32 KB, but always uses a buffer size equal to or larger than the CI size. For example, a 30 KB CI uses a 32 KB subpool. If no VSAM files are used, the VSAM subpools need not be defined. A buffer handler pool is always built.

During DL/I database open, a data set is assigned a specific buffer subpool based on the CI size. The CI size must be equal to or less than the buffer size for the subpool assigned. The data and index components of a key-sequenced data set (KSDS) can be assigned to different subpools if their CI sizes are different and corresponding subpools exist. VSAM can assign a larger CI size to the INDEX component than to the DATA component when defining KSDS data sets that have small record sizes. For example, VSAM can assign a 512-byte DATA CI size and a 4096-byte INDEX CI size. You must be sure to define subpools that accommodate both INDEX and DATA components of a KSDS. (A listing of the VSAM catalog shows the CI sizes assigned to the components.) A single subpool can be defined with buffers large enough to contain the longest CI, or you can define several subpools that more nearly fit the different-sized CIs used by the programs.

Related reading: Refer to *IMS Version 11 System Administration* for additional information about buffer pool structure and buffering techniques.

OSAM subpool definition

OSAM subpools are defined using the IOBF and DBD statements. Specify the required subpools in the DFSVSAMP data set for batch environments, or in the DFSVSMxx member of the IMS PROCLIB data set for DB/DC environments.

OSAM subpools are defined using the IOBF and DBD statements. Specify the required subpools in the DFSVSAMP data set for batch environments, or in the DFSVSMxx member of the IMS PROCLIB data set for DB/DC environments. You can define multiple OSAM subpools as having the same buffer size (that is, the length parameters specified on the IOBF subpool definition statements are identical); then, you can direct given data sets to specific subpools. To take advantage of this flexibility, the subpool must have a user-defined ID (using the ID subparameter) and a DBD statement coded to identify the data set.

If you do not want to assign data sets to specific subpools, do not code the DBD subpool definition statement. If you define multiple OSAM subpools having the same buffer size and ID, IMS generates only one subpool for that buffer size. Within this subpool, the number of buffers is equal to the sum of the number of buffers specified for these subpools. The OSAM buffers and control blocks are above the 16 MB line. In a batch environment, the OSAM buffers and control blocks are located in the private address space. You must allocate a region size that is large enough to accommodate your OSAM buffer requirements.

OSAM buffer pool compatibility definition

The OSAM buffer pool consists of one or more subpools that are defined using IOBF parameter statements. These statements are required in a DB/DC environment. Default member DFSVSM00 is used if the VSPEC parameter is not coded, but these statements are optional in a batch environment.

Omitting IOBF statements in a batch environment causes the OSAM subpools to be generated based on the following factors:

- The DMBs loaded for execution
- The values specified by the BUF= parameter on the EXEC statement

The BUF= parameter specifies the size of the OSAM buffer pool that is to be used in the subpool compatibility definition calculations. The size specified by the BUF= parameter does not necessarily define the actual size of the buffer pool. The specified size influences the eventual buffer pool size.

The buffer pool is generated as follows:

- IMS inspects each DMB loaded for execution for an OSAM DCB. The buffer length from the DCB is rounded up to the next appropriate subpool size (.5 KB, 1 KB, 2 KB, and so on) and is used to create a potential subpool. DCBs with similar length requirements increment the occurrence count for that subpool.
- The potential number of buffers for each potential subpool is derived from the ratio of actual to least occurrences, multiplied by 4. The minimum number of buffers allowed per subpool is four.
- IMS allocates the actual number of subpool and buffers according to the previous ratio allocation using the buffer pool size specified by the BUF= parameter. The objective is to allocate at least four buffers per subpool without exceeding the specified size. The process begins with the subpool with the largest buffer size and continues to the subpool with the smallest. Depending on the specified buffer pool size, it might not be possible to generate all required subpools. IMS allocates only those subpools (largest to smallest), with a minimum of four buffers each, that fit in the specified size. A minimum of one subpool with a minimum of four buffers is allocated regardless of the buffer pool size defined. This subpool has a buffer size capable of handling the largest buffer size requirement.

Example: The BUF= parameter is specified as 10 KB. A scan of the loaded DMBs and the block size required by the OSAM DCBs indicates two subpools are required: one with 1 KB buffers, and one with 4 KB buffers. If each subpool has the minimum number of buffers, the total storage for the buffer pool is 20 KB. The size specified (by BUF=) is 10 000 bytes. To support all requests, the subpool having 4 KB buffers must be allocated. The subpool with 1 KB buffers is not allocated. The buffer pool is allocated having one subpool with four 4 KB buffers. The size is 16 KB (plus control areas), even though 10 000 bytes are specified.

Specifying VSAM and OSAM subpools

Use control statements in the DFSVSAMP data set or the DFSVSMxx member of the IMS PROCLIB data set to specify VSAM and OSAM subpools.

The following example shows the control statements to construct a single OSAM subpool with four 2 KB buffers each and two VSAM subpools. The first subpool contains four 2048-byte buffers, and the second subpool contains fifteen 1024-byte buffers. The new KSDS records use mass insert.

```
IOBF=(2K,4,N,N)
VSRBF=2048,4
VSRBF=1024,15
OPTIONS,INSERT=SEQ
```

For information about specifying these control statements, see "DFSVSMxx member of the IMS PROCLIB data set" on page 738.

For IMS batch environments, the //DFSVSAMP data set provides the control statements listed in the example. Additionally, the //DFSVSAMP data set provides the GTF trace option. GTF trace records reflect the stream of I/O requests to the OSAM buffer handler.

In an IMS environment, the DFSVSMxx member of the IMS PROCLIB data set provides the statements. Do not place DFSVSAMP in the same PDS as your user application files. This causes an ABEND0C4 in DFSRTM00. For more information about GTF trace records, refer to *IMS Version 11 System Administration*.

Specifications for OSAM sequential buffering

The SBONLINE control statement allows use of OSAM Sequential Buffering (SB) in an IMS DB/DC or DBCTL environment.

For information about specifying the SBONLINE control statement, see "DFSVSMxx member of the IMS PROCLIB data set" on page 738.

One or more of the following factors controls the use of SB by a particular application program or utility:

- PSBGEN
- Control statements in the //DFSCTL data set
- An SB Initialization exit routine

Related reading: For more information, see *IMS Version 11 Database Administration*.

See also "Specifying sequential buffering control statements" on page 184 for more information about how to request use of SB through control statements in the //DFSCTL data set of the IMS batch region or IMS dependent online region.

Creating and sizing the 64-bit storage pool

Creating the 64-bit storage pool is a system definition task. Before you can specify an appropriate size on the ACBIN64 parameter in the DFSDFxxx member of the IMS PROCLIB data set, you must perform some calculations.

You can make your calculations based either on DASD allocation or on the sizes of the 31-bit non-resident pool sizes.

To calculate how much 64-bit storage must be allocated for PSB and DMB ACB members, calculate how much storage is being used for all the non-resident PSBs and non-resident DMBs in the ACB library. DEDB DMBs always reside in 31-bit storage. Since DMBs and PSBs both reside in the same 64-bit pool, sufficient storage must be allocated to accommodate all these members.

The following are some examples:

- The current DASD space allocation for all the non-resident PSBs and non-resident DMBs in the ACB library is 2150 cylinders on a 3390-9, which is approximately 1.9 GB of DASD space. To have sufficient space to include as many ACB members as possible in the 64-bit pool, specify ACBIN64=2.
- There are 57 000 ACB members in the ACBLIB data set. The cumulative size of these ACB members is approximately 850 MB. The size of the non-resident DMB pool is 250 MB. The size of the non-resident PSB pool is 300 MB. Therefore, specify ACBIN64=1.

Making IMS and IMSRDR procedures accessible to z/OS

The IMS and IMSRDR procedures initiate the IMS control program region and message regions. These procedures are placed in IMS.PROCLIB by system definition and must be added to SYS1.PROCLIB.

z/OS with JES2 or JES3: Concatenate IMS.PROCLIB with SYS1.PROCLIB, or move the following procedures to SYS1.PROCLIB:

- IMS
- IMSRDR
- DBC
- DBRC address space procedure (as specified in DBRCNM= in the IMSCTRL macro)
- DL/I address space procedure (as specified in DLINM= in the IMSCTRL macro)
- DXRJPROC - IRLM procedure
- DFSMPR
- DFSWTnnn

The IMS PROCLIB data set should be in the master catalog, otherwise, ensure that the volume serial number and unit are on the DD statement. If this is not done, JES2 is unable to initialize.

Any dependent region jobs (IMSMSG, IMSWTnnn) to be specified in the IMS `/START REGION` command must be added to IMS.JOBS.

Related reference:

"DBC procedure" on page 523

"DBRC procedure" on page 529

"DFSMPR procedure" on page 547
"DLISAS procedure" on page 551
"DXRJPROC procedure" on page 554
"IMS procedure" on page 560
"IMSRDR procedure" on page 584

Organizing PL/I modules for use with the PL/I Optimizer

You can decrease response time for those IMS applications that use the PL/I Optimizer by organizing the PL/I modules.

You can organize PL/I modules by:

- Using several different program libraries, one for each region. Put only those modules required by the application in the library. Include in that library all supporting modules (such as the PL/I transient library modules).
- Concatenating the PL/I library into the message region STEPLIB.
- Putting the required supporting modules in the link pack area. This is the recommended long-term solution for a virtual environment.
- Using IMS PRELOAD on the modules.

Attention: Do not use the PL/I Optimizing Compiler for multitasking during binding. Do not use SYS1.PLITASK as a SYSLIB data set.

Specifying sequential buffering control statements

You can use six different sequential buffering control statements, SBPARM, SBIC, SBCO, SBSNAP, SBESNAP, and SNAPDEST, to specify and override definitions related to sequential buffering.

SB control statements allow you to:

- Specify which I/O operations should be buffered by the SB buffer handler
- Override the default number of buffer sets to increase buffering performance
- Experiment with SB specifications without changing exit routines or regenerating the PSB
- Use SB problem determination aids

SB control statements are in a data set with the ddname of //DFSCTL. You can provide a //DFSCTL DD statement in the JCL of the IMS batch or IMS dependent online region. The //DFSCTL DD statement must point to a sequential data set or a member of a partitioned data set. The record format of the //DFSCTL data set must be F, FB, or FBS.

Sequential buffering can be invoked by default for IMS utilities, such as online image copy, without requiring SB control statements, through the use of the SB Initialization exit routine.

Related reading:

- For the syntax and keywords associated with the SB control statements, see "Sequential buffering control statements" on page 805.
- For the description of the SB Initialization exit routine, see *IMS Version 11 Exit Routines*.

Using high-speed sequential processing control statements

The following is an introduction to high-speed sequential processing (HSSP) control statements.

HSSP control statements allow you to:

- Set up the environment in which you process a selected PCB with HSSP
- Create an image copy of a designated DEDB area, reducing the amount of logged data
- Restrict access of a specific HSSP or non-HSSP application program to only designated DEDB areas

You can specify the following HSSP control statements:

SETO The SETO control statement allows you to specify the options in processing a PCB with HSSP.

SETR The SETR control statement specifies the processing range of PCBs to a database during scheduling of an application program.

HSSO, HSSR, and HSSD control blocks are built from SETO and SETR statements. These control blocks—specifically those that represent image copy data sets and those that are used for UOW locking—are formatted for offline dumps. These control statements are in the DFSCTL data set.

Related reading: For the syntax and keywords associated with the HSSP control statements, see "High-speed sequential processing control statements" on page 812.

Supporting CCTL users with DBCTL databases

DBCTL is an IMS facility that allows access to DL/I full-function databases, HALDB, and data entry databases (DEDBs) from a subsystem called a coordinator controller subsystem (CCTL).

A CCTL is the transaction management subsystem that communicates with DBCTL using the database resource adapter (DRA).

"Preparing a CCTL" on page 186 describes the installation tasks required of the CCTL to enable the DRA.

Related reading: See *IMS Version 11 Communications and Connections* for information about CCTL programming requirements for using the DRA interface and installation concerns related to IBM CICS® Transaction Server for z/OS as a CCTL.

Figure 12 illustrates the relationship of the DRA interface, CCTL, and DBCTL.

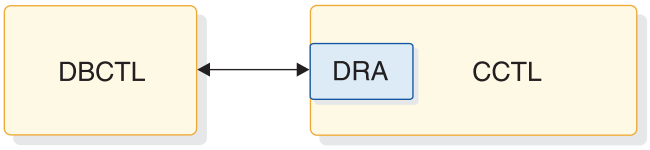


Figure 12. DRA interface to DBCTL

As part of the DBCTL definition process, you should consider CCTL requirements such as database, PSB, or buffer pools. For sample input for DBCTL stage 1 definition, see “IMS DBCTL environment” on page 280.

Preparing a CCTL

You must take steps to prepare a CCTL for enabling a DRA environment.

Perform the following procedure:

1. Confirm that the DRA startup/router routine (DFSPRRC0) is in a CCTL load library. The routine can be copied from the IMS.SDFSRESL, which is built by the IMS definition process. Alternately, the IMS.SDFSRESL can be concatenated in the CCTL step library.
2. Confirm that the DFSPZPxx load module is in a CCTL load library; this load module is the DRA startup table. The *xx* is the startup table name suffix that the CCTL specifies on the DRA initialization request. The source code for DFSPZPxx is DFSPZP00 in the IMS distribution library, DLIB. After all modifications have been made to DFSPZPxx, it is assembled using IMS DLIBS. A default load module, DFSPZP00, is in the IMS.SDFSRESL. It has default values for all but two of the required DRA initialization parameters, all of which can be overridden on the initialization (INIT) request itself.

   The rest of the DRA code must reside in a load library that is dynamically allocated by DFSPRRC0. The DDNAME and DSNAME of this load library are specified in the startup table or initialization request. The default DSNAME is IMS.SDFSRESL because all DRA code resides here throughout the IMS definition process.

Related reading: For the sample source code for DFSPZP00 and explanations of the parameters specified through the DFSPRP macro, see “Database resource adapter startup table for CCTL regions” on page 817.

Enabling IMS ETO Support for ACF/VTAM terminals

IMS Extended Terminal Option Support (IMS ETO Support) requires at least one user descriptor and one logon descriptor.

Related reference:

“DFSDSCMx member of the IMS PROCLIB data set” on page 689

“DFSDSCTy member of the IMS PROCLIB data set” on page 701

ETO descriptors

When you are enabling ETO support in your installation, you can optionally request that IMS produce ETO descriptors during system definition. You do this with the ETOFEAT keyword in the IMSCTRL macro.

Descriptors contain information about terminal devices and users that is required to establish dynamic sessions. Four types of descriptors are:

Logon

MFS Device

MSC

User

Descriptors that are created during system definition are placed in member DFSDSCMx, which resides in the IMS.PROCLIB data set. If you perform

subsequent system definitions of the same stage 1 input deck, the DFSDSCMx members are overwritten. If you use TSO or a z/OS utility to create descriptors, you can place the descriptors in member DFSDSCTx to avoid loss when member DFSDSCMx is replaced. You can also use TSO or an z/OS utility to update descriptors in DFSDSCMx that were created during system definition.

The suffix for DFSDSCMx is always the nucleus suffix. The default suffix for DFSDSCTy defaults to 0; however, you can specify it with the DSCT= JCL parameter. Each descriptor statement must be 80 characters in length. All statements are translated to uppercase.

Related tasks:

Administering the Extended Terminal Option (Communications and Connections)

Related reference:

"IMSCTRL macro" on page 340

Transaction Manager exit routines (Exit Routines)

"DFSDSCMx member of the IMS PROCLIB data set" on page 689

ETO descriptor definition rules

Follow these rules when defining ETO descriptors.

- Separate one keyword or parameter set from another with one or more blanks.
- Do not include embedded blanks within a keyword and its parameters.
- Separate a keyword from its parameters with an equal sign (=).
- Do not abbreviate keywords.
- Separate multiple parameters for single keywords with commas.
- Specify at least one parameter for each keyword.
- No more than 512,000 records (excluding comments) are permitted for each individual descriptor.

You can continue a keyword and its parameter set to the next statement if no intervening blanks appear at the end of the first statement or at the beginning of the parameters of the next statement. A continued statement still has the same descriptor type and name in columns 1-10; the continued specification begins in column 12. If you specify keywords, they must be accompanied by parameters. Keywords followed by blanks or commas are invalid.

Parentheses are optional if they follow a keyword that supports multiple parameters and only if the first parameter is specified. Otherwise, they are required.

If the keyword has positional parameters, you can use commas when parameters are specified after a missing positional parameter. Following is an example of valid use:

```
LTERMA=(ABC,,1)
```

This example is valid because LTERMA has positional parameters.

The following example is invalid:

```
LTERMB=(ABC,,1,)
OPTIONS=(TRANSRESP,)
OPTIONS=(,SYSINFO)
```

The LTERMB example is invalid because the comma must be followed by a parameter. The OPTIONS examples are invalid because the parameters for the OPTIONS keyword are not positional.

ETO descriptor overrides

If a descriptor is defined in both DFSDSCTy and DFSDSCMx, the descriptor in member DFSDSCTy overrides the descriptor in member DFSDSCMx.

No message is issued.

If a descriptor is defined more than once in either member, error message DFS3661 is issued. IMS uses the last descriptor defined. If the same descriptor is defined once in member DFSDSCTy and one or more times in member DFSDSCMx, the descriptor in member DFSDSCTy is used, and no error message is issued.

Logon descriptors

Logon descriptors provide IMS with information about the physical characteristics of the terminals that establish logon sessions.

When you request that ETO descriptors be created during system definition, a logon descriptor is created for each VTAM TYPE and TERMINAL macro set.

IMS also creates a default logon descriptor during system definition, which reflects the TYPE and TERMINAL macro statements that define the unique characteristics of the largest number of terminals of that type.

For terminals that cannot be represented by the default logon descriptor, IMS creates unique logon descriptors.

IMS does not create ETO logon descriptors for the following terminal types:

- Any terminal defined as a primary or secondary MTO
- Any LUTYPE6 terminal defined as the XRF ISC link

You can keep and use the unique logon descriptors that IMS creates or you can discard them. To discard the unique logon descriptors you can:

- Operate the terminals based on the default logon descriptor
- Create one or more of your own installation default logon descriptors and code an installation Logon exit routine (DFSLGNX0) to select the required descriptor for logging on to a terminal

The system definition process generates up to 37 default logon descriptors for each device type. ETO descriptors are not generated during large system definition. The suffix for each descriptor defaults to 0; the last character of the descriptor name ensures that the name is unique. This character is a blank for the most common descriptors, and then 0 through 9, and A through Z, for a total of 37 possible descriptors.

Terminal definitions that do not match the 37 most common terminal definitions generate comment statements with an asterisk in the first position and the rest of the statement is shifted over one position.

All user descriptors created during system definition are generated as comment statements to avoid getting error messages caused by a statically defined LTERM name.

Typically, one generic logon descriptor exists per unique LU type and option configuration. After the descriptor is created, the same descriptor can support any number of logons for the same LU type and option configuration.

The format of a logon descriptor is

```
L Descriptor name Parm(1) Parm(2) Parm(3)
L Descriptor name Parm(4) Parm(5) Parm(n)
```

The parameters that are required for a descriptor cannot fit into one record. You might need to use multiple records with the same descriptor name to define your descriptor.

Related tasks:

Creating logon descriptors (Communications and Connections)

MFS device descriptors

Use MFS device descriptors to define characteristics of terminals added dynamically that have different characteristics from statically defined terminals.

MFS device descriptors are used by the MFS Device Characteristics Table (DCT) utility. This utility updates screen information, for example 3270 screen sizes and feature information, in the DCT and generates new MFS default formats without system definition.

Related reading: See *IMS Version 11 System Utilities* for more information about how to use the MFS DCT utility.

MFS device descriptor format

The format of an MFS device descriptor is

```
D Descriptor name Parm(1) Parm(2) Parm(3)
```

MFS device descriptor parameters

The three parameters supported for MFS device descriptors are TYPE=, SIZE=, and FEAT=.

Example of MFS device descriptors

```
Col        Col
1          12
D          TYPE=3270-A04 SIZE=(43,80) FEAT=IGNORE
```

See the parameter descriptions in “Parameter descriptions for IMS procedures” on page 448 for more information.

MSC descriptors

MSC descriptors relate remote NAME macros to MSC links defined during system definition.

When you use ETOFEAT to specify that descriptors are to be created during system definition, an MSC descriptor is created for each MSNAME macro statement.

MSC descriptor format

The format of an MSC descriptor is

```
M Link name name1 name2 namen
```

An MSC descriptor contains a link name rather than a descriptor name. The link name is the name of the retrieved descriptor. The name parameters represent any valid remote LTERM name that can be accessed through a link name. The parameters that are required for a link name cannot fit into one record. You might need to use multiple records with the same link name to define your descriptor.

MSC descriptor parameters

The parameters that are supported for MSC descriptors are as follows:

descriptor type
Specifies that the descriptor is MSC (M).

descriptor name
Specifies the link name from the MSNAME macro.

remote LTERM name
Specifies the name of a logical terminal associated with a physical terminal defined in a remote IMS system.

User descriptors

User descriptors are another form of descriptor created during system definition for ETO support. Node user descriptors are essentially migration aids, allowing message queues and options associated with a particular terminal to remain unchanged during migration.

User descriptors are another form of descriptor created during system definition for ETO support. Three types of user descriptors are:

- DFSUSER
- Node
- Installation-created

The node user descriptors are essentially migration aids, allowing message queues and options associated with a particular terminal to remain unchanged during migration. They do not, however, provide output security.

Related reading: For information about the format of user descriptors, refer to *IMS Version 11 Communications and Connections*.

User descriptors are generated from each VTAM TERMINAL or VTAMPOOL SUBPOOL macro statement. Those descriptors created from VTAM TERMINAL macro statements, such as node user descriptors, have the same name as the terminal. Descriptors created from VTAMPOOL SUBPOOL macro statements have the same name as the subpool.

For user descriptors created during IMS system definition from the SUBPOOL macro statement, the response option (TRANSRESP, NORESP, or FORCRESP) cannot be set. This is because the response option is defined on the TERMINAL statement for static definitions. You need to add the appropriate response option for your installation to any user descriptor created from a SUBPOOL statement.

DFSUSER descriptor

When you request that IMS build ETO descriptors, IMS creates a single default user descriptor, DFSUSER, which defines default user characteristics. The DFSUSER descriptor represents the most common set of user options. With this descriptor, IMS can dynamically create a user and message queue structure for a signon request when no other user descriptor is available. The message queue name is the same as the user ID. You can also use DFSUSER to add queue names and other options through the DFSSGNX0 Signon exit routine.

Related reading: Refer to *IMS Version 11 Exit Routines* for more information about how to use the DFSSGNX0 exit routine.

Recommendation: Use the DFSUSER descriptor for most of your users after your migration to ETO is completed.

Node user descriptors

Node user descriptors are created during system definition. One is generated for each terminal in the IMS system definition. Node user descriptors are created even if they match DFSUSER options, except for ISC terminals defined for parallel-session support.

Installation-created descriptors

Installation-created user descriptors are designed by your installation to meet your criteria. The name of the installation-created user descriptor is the same as the user ID.

User descriptor format

The format for a user descriptor, regardless of whether it is created by installation, node, or DFSUSER, is:

```
U username parm1 parm2 parmn
```

`U` is the descriptor type (USER), and `Username` is DFSUSER, the user ID, or the node user name from the name field in the TERMINAL or SUBPOOL macro statement supplied by the installation.

`parm (1...n)` can be any of the following parameters: ASOT=, LTERM=, OPTIONS=, AUTLGN=, AUTLMOD=, AUTLDESC=, AUTLID=

Specifying the number of hash table slots

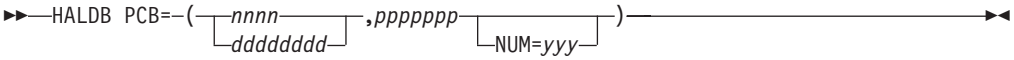
The number of hash table slots that IMS builds for a statically defined resource depends on the total number of resources; you can specify additional slots.

The number of hash table slots built by IMS for a statically defined resource (such as LTERMs, physical terminals, and users) is one-eighth the total number of the resource. To specify additional slots required for a resource created dynamically (conversations, LTERMs, physical terminals, and users), as a rule-of-thumb use one-eighth of the estimated total number of a dynamic resource. You might need to adjust this value. For information about setting the number of slots for specific hash tables, see the LHTS, NHTS, and UHTS parameters in “IMS procedure” on page 560 and “DCC procedure” on page 532.

HALDB selective partition processing

You can restrict the processing of DL/I calls to a single HALDB partition or a range of HALDB partitions by using a DD statement with the ddname DFSHALDB to pass control statements. DFSHALDB must be provided in the JCL of the batch job, the BMP (Batch Message Processing dependent online region), or the JBP (Java Batch Processing dependent online region).

Control Statements for HALDB selective partition processing



Each HALDB control statement must have a PCB keyword that contains the required parameters. The required parameters for an individual control statement must be on one line; no continuation is allowed. The input can consist of multiple HALDB control statements. There should be no duplication of DB PCB numbers. In the event of a duplication, the control statement that has been read the most recently overrides the previous statement.

Any HALDB control statement that is syntactically correct results in an entry within a table. The maximum number of entries in the table is 20. All subsequent statements that are read, even though syntactically correct, are ignored and result in a U0201 abend, unless a statement is a duplicate of an entry that is already in the table.

Parameter descriptions for HALDB selective partition processing

nnnn
The DB PCB number as the relative number of the DB PCB defined in the PSB.

dddddddd
The DB PCB label or name.

ppppppp
The partition name. This parameter is required.

NUM=*yyy*
The range of consecutive partitions that this PCB is restricted to using, starting with the named partition. The range of consecutive partitions is defined as the partition selection order, which is the next partition selected starting from the target partition named in the DFSHALDB statement. The next partition is determined using either the high keys defined for the HALDB or the processing order defined by the partition selection exit. This parameter is optional.

The following examples show how to use HALDB selective partition processing statements.

DFSHALDB for single partition restriction

```
HALDB PCB=(4,POHIDKA)
HALDB PCB=(PCBNUM2,POHIDJA)
```

DFSHALDB for range partition restriction

```
HALDB PCB=(3,PVHDJ5A,NUM=4)
HALDB PCB=(PCBNUM7,PVHDJ5B,NUM=3)
```

DFSHALDB for independent processing of partitions

```
PRINT NOGEN
PCB TYPE=DB,DBDNAME=G1CSTP,PROCOPT=A,KEYLEN=100,PCBNAME=XXCSTP
SENSEG NAME=CUSTOMER
SENSEG NAME=DISTRICT,PARENT=CUSTOMER
SENSEG NAME=CUSTLOCN,PARENT=CUSTOMER
SENSEG NAME=ADDRLINE,PARENT=CUSTLOCN
SENSEG NAME=CUSTORDN,PARENT=CUSTLOCN
SENSEG NAME=CUSTINVN,PARENT=CUSTOMER
SENSEG NAME=PAYMENTS,PARENT=CUSTOMER
SENSEG NAME=ADJUSTMT,PARENT=CUSTOMER

PCB TYPE=DB,DBDNAME=G1CSTP,PROCOPT=A,KEYLEN=100,PCBNAME=IICSTP,
             PROCSEQ=CSTCY02
SENSEG NAME=CUSTOMER


PSBGEN LANG=ASSEM,PSBNAME=G1ACSTP,CMPAT=YES
END
```

Report generated for HALDB selective partition processing

When you use HALDB selective partition processing, a report called "HALDB Selective Partition Processing" is generated in the SYSHALDB data set. This report shows the control statements that have been issued and the reason for accepting or rejecting each statement. Control statements that have been validated and accepted are shown as "Syntactically correct." Other messages that might appear for syntactically correct statements, and their accompanying messages, are shown in the following table:

Table 38. Messages provided in the report generated for HALDB selective partition processing

Message	Explanation
Duplicate, overrides previous statement	A HALDB statement for the same PCB was already found. The current statement overrides the previous HALDB statement.
Ignored, number of valid statements exceeds 20	More than 20 HALDB statements were provided, but only 20 statements are allowed. Reduce the number of HALDB statements to 20 or fewer, and run the job again. This message results in an abend U0201.
NUM parameter must be non-zero numeric	The partition range specified in the NUM keyword must be a non-zero value from 1 to 999.
NUM value exceeds three digits	The partition range specified in the NUM keyword must be a non-zero value from 1 to 999.
An equal sign must follow NUM keyword	An equal sign must follow the NUM keyword in the HALDB statement. Add an equal sign to the HALDB statement.
The NUM keyword is missing	A comma was found after the partition name, but the NUM keyword was not present. Either verify the syntax of the positional parameters in the HALDB statement, or add the NUM keyword and the range of partitions for the restriction.

Table 38. Messages provided in the report generated for HALDB selective partition processing (continued)

Message	Explanation
NUM parameter is missing	The NUM keyword was found, but the NUM parameter value was not present. Either verify the syntax of the positional parameters in the HALDB statement, or add the NUM keyword and the range of partitions for the restriction.

For HALDB control statements that are not syntactically correct (statements that are processed and rejected), the messages and explanations that are issued are shown in the following table:

Table 39. Messages provided in the report generated for syntactically incorrect HALDB statements

Message	Explanation
No HALDB statement type	The DFSHALDB data set did not contain a HALDB statement. Add a HALDB statement to prevent this error.
A space must follow HALDB statement type	The HALDB statement requires a space after HALDB and before the PCB keyword.
PCB keyword missing	The required keyword PCB was not found. The PCB keyword must be present to process the HALDB statement successfully.
Equal sign must follow PCB keyword	An equal sign did not follow the PCB keyword. The equal sign must follow the PCB keyword to process the HALDB statement successfully.
Open parenthesis must follow equal sign	An open parenthesis did not follow PCB=. The open parenthesis must follow the PCB= to process the HALDB statement successfully.
Second parameter may be missing	The HALDB partition must be provided. Either add the partition name, or verify that the syntax of the positional parameters is correct.
First parameter exceeds four digits	The DB PCB number cannot exceed a four-digit value. Change the DB PCB number to the correct DB PCB number.
Delimiter is not a comma	A comma is missing between parameter values. The comma is used as a delimiter for the positional parameters. Either add the comma, or verify that the syntax of the positional parameters is correct.
Partition name must start with an alpha	The HALDB partition name must begin with a alphabetic character. Add the partition name or verify the syntax of the positional parameters is correct.
Delimiter is not a close parenthesis	A closing parenthesis is missing from the HALDB statement. Add a closing parenthesis around the PCB parameters.

Table 39. Messages provided in the report generated for syntactically incorrect HALDB statements (continued)

Message	Explanation
Partition name exceeds seven characters	The HALDB partition name must be seven or fewer characters. Either add the partition name, or verify that the syntax of the positional parameters is correct.
Invalid character in partition name	The HALDB partition name contains an invalid character. Either add the partition name, or verify that the syntax of the positional parameters is correct.
Statement contains all spaces	The HALDB statement is missing. Add a valid HALDB statement.
Invalid statement input	A HALDB statement was found, but it does not appear to be complete. Verify the syntax of the HALDB statement and the positional parameters specified.
Space must follow close parenthesis	A space must follow the closing parenthesis. Add a space after the closing parenthesis.
First parameter missing	The PCB number or label is missing. Either add the PCB name or label, or verify that the syntax of the positional parameters is correct.
Comma and part name missing	Only the PCB number or label was provided in the HALDB statement. Either add the partition name, or verify that the syntax of the positional parameters is correct.
Partition name is missing	The HALDB partition name must be provided in the HALDB statement. Either add the partition name, or verify that the syntax of the positional parameters is correct.
Partition name starts with numeric	The HALDB partition name must begin with an alphabetic character. Either add the partition name, or verify that the syntax of the positional parameters is correct.
First parameter must not be zero	The PCB number must be a non-zero number. Add a non-zero number for the DB PCB number.
Comment statement	An asterisk was found in column one of the HALDB statement. This statement was skipped and considered a comment.

After all of the statements are validated, the job abnormally terminates with an abend code of U0201.

Chapter 7. CQS definition and tailoring

The tasks for defining and tailoring the Common Queue Server (CQS) involve working with macros, procedures, and other system-oriented information.

This topic contains Product-sensitive Programming Interface information.

CQS's support of multiple clients

A single CQS address space can support as many as 32 different clients that are on the same z/OS image.

One CQS address space can support multiple clients. Examples of clients are IMS and RM.

As many as 32 different clients on the same z/OS image can connect to coupling facility structures through a single CQS by using the CQSCONN request. For example, as many as 32 different IMS control regions can specify the same CQS=xxx parameter in the DFSSQxx member of the IMS PROCLIB data set. IMS starts the CQS address space if it is not currently active. If CQS is already active, IMS registers with the active CQS address space and does not start an additional CQS address space. Be sure that not more than 32 IMS systems or RMs specify the same CQS.

Defining structure size for CQS connections

Use the INITSIZE and SIZE parameters of the z/OS CFRM policy to define the size of the CQS structures.

The size of the structures to which CQS connects is defined in the CFRM policy by defining the `INITSIZE` (initial size of the structure) and `SIZE` (maximum size of the structure) parameters. If enough free space does not exist for this INITSIZE value, the size of the structure becomes that of the available space in the coupling facility.

To determine what structure size to define in the CFRM policy, you can use the System z® Coupling Facility Structure Sizer Tool (CFSizer). CFSizer is a web-based application that calculates the structure size based on the input data you provide. CFSizer is available at System z Coupling Facility Structure Sizer Tool (CFSizer) web page.

You can use the z/OS structure alter process to change the structure size or to redistribute the objects within the structure after it has been defined. For information, see Using structure alter for CQS (System Administration).

Calculating resource structure entry and element values to maximize storage

Several different resources can be stored in the resource structure. It is important to create a resource structure of sufficient size to accommodate these resources. Use the recommendations in this topic to calculate a resource structure of appropriate size.

© Copyright IBM Corp. 1974, 2015

An IMS system with a defined resource structure stores transaction names in the resource structure. Additional resources can be stored in the resource structure depending on the IMS functions that you enable, as shown in the following table:

Table 40. Resources stored in the resource structure

IMS function that is enabled	Resources that are stored in the resource structure	Comments
IMS sysplex terminal management	• APPC descriptor • node name • remote and local system identifications / names in a Multiple Systems Coupling (MSC) environment • user ID • CPIC transaction • user information	Resources are not stored if sysplex terminal management is disabled by defining STM=NO on the DFSDCxxx PROCLIB data set member.
IMS global online change	Global online change stores a resource on the resource structure to manage the global online change process.	Resources are not stored unless global online change is enabled by specifying OLC=GLOBAL on the DFSCGxxx PROCLIB data set member.
IMS global status for areas, databases, and transactions	• global area information • global database information • global transaction information	Resources are not stored unless global status for areas, databases, and transactions is enabled by using either: • The COMMON_SERVICE_LAYER section of the DFSDFxxx PROCLIB member data set and specifying GSTSDB=(Y) to enable global database status, GSTSAREA=(Y) to enable global area status, and GSTSTRAN=(Y) to enable global transaction status. • The DFSCGxxx PROCLIB data set member.

Each resource is stored on the resource structure using a 128-byte entry, and either zero, one, or more 512-byte data elements. The 128-byte entry contains 64 bytes for z/OS control information, and 64 bytes for user data in an adjunct area. Both IMS and CQS use a portion of the 512-byte data element as a prefix. The remaining bytes are available for client data.

Use the entry-to-element ratio when allocating the resource structure to reserve portions for entries and data elements. The more accurate the ratio is for actual resources stored on the resource structure, the less storage is wasted. The number of entries is equal to the number of resources. The number of data elements depends on the number of resources for each resource type.

The information and formulas in the following table can help you determine the ENTRY and ELEMENT values to define in the CQSSGxxx PROCLIB data set member. For each resource type used by your installation, add the entry value to the entry sum and add the element value to the element sum. To calculate the ENTRY value, locate the resource types used by your installation in the table and add the entry value for the resource types to the system resource value. To

calculate the ELEMENT value, locate the resource types used by your installation and add the element value for the resource types to the system resource value.

For example, if you do not use sysplex terminal management (STM=NO), the ENTRY value is the number of transactions plus 8 (system resources), and the ELEMENT value is 14 (system resources).

Table 41. Formulas for calculating resource structure entry and element values.

Resource type	System definition status, if applicable	Entry value	Element value	Meaning
APPC descriptor	STM=NO	0	0	Unique APPC descriptors in the IMSplex
	STM=YES	Number of descriptors	0, if the number of IMS systems is less than 3. If the number of IMS systems is greater than 2, the count is (the number of IMS systems) multiplied by (the number of IMS systems - 2) divided by 29 (rounded up).	
Area	GSTSAREA=(N)	0	0	Area with global status. This can be the number of Fast Path areas you plan to use to maintain global status, or the total number of areas.
	GSTSAREA=(Y)	Number of Fast Path areas		
Database	GSTSDB=(N)	0	0	Databases with global status. This can be the number of databases that you plan to use to maintain global status, or the total number of databases.
	GSTSDB=(Y)	Number of databases		
Global Online Change	OLC=LOCAL	0	0	Global Online Change in progress.
	OLC=GLOBAL	1	1	
Lterm	STM=NO	0		System generated and dynamic logical terminals.
	STM=YES	Number of LTERMs [1]	0	
MSName	STM=NO	0	0	Unique system-generated MSNames in the IMSplex.
Node	STM=NO	0	0	System generated and dynamic nodes
	STM=YES	Number of VTAM nodes [1]	Number of ISC nodes with multiple parallel sessions multiplied by (the node's maximum parallel sessions - 1) divided by 29, rounded up.	

Table 41. Formulas for calculating resource structure entry and element values (continued).

Resource type	System definition status, if applicable	Entry value	Element value	Meaning
Serial program		Number of serial programs	0	System generated and dynamic programs that are defined with a scheduling type of SERIAL
System resources		8 + (*x* multiplied by 3) + (*y*) + (*z*) + *q* [3]	14 + (*x* multiplied by 2) [4]	IMS, CQS, and RM resources *x* The number of IMS systems *y* The number of CQSs *z* The number of RMs **q** The number of structures
Transactions		Number of transactions	0	System generated, non-CPIC transactions
CPIC transactions	STM=NO	0	0	CPIC transactions invoked by APPC
	STM=YES	Number of CPIC transactions	0, if the number of IMS systems is less than 3. If the number of IMS systems is greater than 2, the value is the number of CPIC transactions multiplied by (the number of IMS systems - 1) divided by 29, rounded up.	
User ID	STM=NO	0	0	Maximum number of user IDs signed on.
	STM=YES	Number of user IDs if SGN is not either G, M, or Z.		
User	STM=NO	0	0	Dynamic users and unique static ISC subpools
	STM=YES	Number of users [1]	Number of users [2]	
User - static node	STM=NO	0	0	Unique static single-session users
	STM=YES	Number of static users [1]	Number of static users [2]	

Table 41. Formulas for calculating resource structure entry and element values (continued).

Resource type	System definition status, if applicable	Entry value	Element value	Meaning

Note:

1. The entry value is the maximum number of LTERMs or nodes where:
   - the assigned node is logged on.
   - the assigned node has significant status
   - the assigned user is signed on.
   - the LTERMs have significant status.
2. The element value is normally one per user or static node user, but it is greater than one if either the number of LTERMs is excessive, the number of held conversations is excessive, or both. Each user or static node user resource stores a minimum of the sum of the following values:

   ```
   176 bytes in a data element +
   16 bytes for each associated LTERM +
   52 bytes for each held conversation +
   196 bytes if a Fast Path transaction is in progress
   ```

   If the number of bytes exceeds the amount available for client data in a data element, more data elements are used.
3. The IMS, RM, and CQS system resource entries include:
   - CQSGLOBAL
   - CQSGLOBALCONN
   - CQSLOCALcqsid (one per CQS)
   - CQSRECOVER (one per structure type MSGQ, EMHQ, and resource)
   - CSLRGBL
   - CSLRLrmid (one per RM)
   - CSLRMPRCprocessid (one per IMSplex-wide process)
   - CSLRRTYP
   - DFSGBLPLXPN
   - DFSGBLSYS
   - DFSSTMGBL
   - DFSSTMLimsid (one per IMS)
   - Primary master terminal (one per IMS)
   - Secondary master terminal (one per IMS)
4. The IMS, RM, and CQS system resource elements include:
   - CQSGLOBAL
   - CQSGLOBALCONN
   - CSLRGBL
   - CSLRMPRCprocessid (one per IMSplex-wide process)
   - CSLRRTYP (9)
   - DFSGBLSYS
   - Primary master terminal (one per IMS)
   - Secondary master terminal (one per IMS)

If the entry or element value exceeds the maximum value of 65 535, reduce the entry or element value to a value that is less than 65 535. If the value of the elements divided by the entries is greater than 120, reduce the values so that the value of the elements divided by the entries is less than or equal to 120. For example, if your calculation results in a ratio of 131 070 entries to 65 535 elements, this can be reduced to a ratio of two entries to one element, because 131 070

divided by 65 535 is equivalent to 2 divided by 1. If you want to allocate a structure large enough to support your maximum number of IMS systems, RMs, CQSs, and resources, use the maximum values in the table above to calculate the values from the formula.

Make sure that your structure's number of entries and elements is greater than the calculated entry and element values prior to activating STM. If it is not, you can use the z/OS SETXCF command to make your structure bigger.

Preparing to start the CQS address space

Before you can start the Common Queue Server (CQS) address space, you must define CQS parameters, and you must create and activate the necessary z/OS policies.

Because they are a part of the defined IMS system, CQS and Base Primitive Environment (BPE) are automatically linked into IMS.SDFSRESL when you run the JCLIN job stream.

You must define CQS parameters before starting the CQS address space. Define the parameters using either of the following:

- CQS initialization parameters member of the IMS PROCLIB data set (CQSIPxxx)
- CQS startup procedure (CQSINIT0 or BPEINI00)

1. Create a coupling facility resource management (CFRM) policy that defines the structures to which you want CQS to connect. The CFRM policy specifies the name, size, attributes, and location that the structure is to be assigned when it is allocated.
2. Define the following z/OS policies:
   - Sysplex failure management (SFM) policy
   - System logger (LOGR) policy

     Note: In the LOGR policy, specify the MAXBUFSIZE parameter as 65272 bytes to ensure that the system logger can write the largest possible log record that is written by CQS.
   - Automatic restart management (ARM) policy
3. Activate the CFRM policy using the following command:
   ```
   SETXCF START,POLICY,TYPE=CFRM,POLNAME=CONFIG01
   ```
   The structure is allocated when the first CQS connects to it.
4. If you are using the SFM policy, activate it using the following command:
   ```
   SETXCF START,POLICY,TYPE=SFM,POLNAME=SFMPOL
   ```
5. Create the CQS and BPE members of the IMS PROCLIB data set.
6. Define all CQS execution data sets.
7. Customize your CQS environment; determine which exits you want to use and then write the exits, which can include the following:

   CQS initialization-termination user-supplied exit routine

   CQS client connection user-supplied exit routine

   Queue overflow user-supplied exit routine for CQS

   CQS structure statistics user-supplied exit routine

   CQS structure event user-supplied exit routine

   CQS statistics available through the BPE statistics user exit

8. Authorize connections to CQS structures.
9. Update the z/OS program properties table.
10. Plan security.

Related reference:

"CQSIPxxx member of the IMS PROCLIB data set" on page 614

Chapter 8. CSL definition and tailoring

You can define the Common Service Layer using several IMS PROCLIB data sets; after they are defined, they should be started in a specific sequence.

The IMS Common Service Layer (CSL) comprises several address spaces that provide system services to IMS. These address spaces include:

- Open Database Manager
- Operations Manager
- Resource Manager
- Structured Call Interface

Defining the Common Service Layer using IMS PROCLIB data set members

The Common Service Layer (CSL) can be defined and tailored using several different IMS PROCLIB data set members.

The following IMS PROCLIB data set members can be used to define settings and values for the Common Service Layer (CSL) and its managers: Open Database Manager, Operations Manager, Resource Manager, and Structured Call Interface. The uses described here are specifically for the CSL.

BPECFG*xx*
: The CSL ODBM, OM, RM, and SCI all use the Base Primitive Environment (BPE). Use the BPE configuration member of the IMS PROCLIB data set to configure the BPE execution environment parameters for each CSL manager. Each CSL manager can have its own BPE configuration and user exit definition members, or they can share common members.

CQSIP*xxx*
: Use the IMSPLEX() and NAME= parameters on this IMS PROCLIB data set member to define the IMSplex name.

CQSSG*xxx*
: Use the RSRCSTRUCTURE() parameter on this IMS PROCLIB data set member to define a resource structure for the Resource Manager.

CSLDC*xxx*
: Use the CSLDCxxx IMS PROCLIB data set member to configure the ODBM data store connections to IMS systems.

CSLDI*xxx*
: Use the CSLDIxxx IMS PROCLIB data set member to specify parameters that initialize the ODBM address space. Certain parameters within CSLDIxxx can be overridden with the ODBM execution parameters.

CSLOI*xxx*
: Use the CSLOIxxx IMS PROCLIB data set member to specify parameters that initialize the OM address space. Certain parameters within CSLOIxxx can be overridden with the OM execution parameters.

© Copyright IBM Corp. 1974, 2015

CSLRIxxx
Use the CSLRIxxx IMS PROCLIB data set member to specify parameters that initialize RM. Some parameters within CSLRIxxx can be overridden with RM execution parameters.

CSLSIxxx
Use the CSLSIxxx IMS PROCLIB data set member to specify parameters related to initialization of the SCI address space. Certain parameters within CSLSIxxx can be overridden using SCI execution parameters.

DFSCGxxx
All the parameters on this IMS PROCLIB data set member specify parameters related to the Common Service Layer (CSL), including the Operations Manager (OM), the Resource Manager (RM), and the Structured Call Interface (SCI).

DFSDCxxx
Use the AOS= parameter on this IMS PROCLIB data set member to enable IMSplex users to execute transactions that are originated by APPC or OTMA on a back-end system. Use the SRMDEF= parameter to globally save the status of a resource every time the significant status changes. Use the STM= parameter to have IMS the Resource Manager resource structure to manage TM resources.

DFSDFxxx
Specify CSL parameters in the CSL section of the DFSDFxxx IMS PROCLIB data set member or on the DFSCGxxx IMS PROCLIB data set member. All parameters that are valid in the DFSCGxxx member of the IMS PROCLIB data set are valid in the DFSDFxxx member of the IMS PROCLIB data set. Either the DFSCGxxx IMS PROCLIB data set member or the CSL section in the DFSDFxxx IMS PROCLIB data set member initiate the CSL. If you specify the CSL parameters on both IMS PROCLIB data set members, the values in DFSCGxxx take precedence. You should specify OM command security instead of IMS security. All IMSplex members that are in the same IMSplex group sharing databases, message queues, or both, must specify the same values except OLC=, which specifies either LOCAL or GLOBAL.

DFSPBxxx
Use the CSLG= parameter to specify a three-character suffix for the DFSCGxxx member of the IMS PROCLIB data set.

DFSVSMxxx
Use the OCMD= and CSLT= parameters on this IMS PROCLIB data set member to activate traces related to IMSplex activity.

EXITDEF
Each of the CSL managers use BPE services to define and manage calls to user exit routines. You can externally specify the user exit routine modules to be called for a particular exit routine type using EXITDEF statements in BPE user exit members of the IMS PROCLIB data set.

Sample ODBM user exit list member of the IMS PROCLIB data set

A sample ODBM BPE user exit list member of the IMS PROCLIB data set is shown in the following example.

```
*****************************************************************
* ODBM USER EXIT LIST PROCLIB MEMBER                            *
*****************************************************************
```

```
#-----------------------------------------------------------------#
# DEFINE 1 ODBM INIT/TERM USER EXIT: ZDINTM00                     #
#-----------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZDINTM00),COMP=ODBM)

#-----------------------------------------------------------------#
# DEFINE 1 ODBM INPUT USER EXIT: ZINPUT00                         #
# WITH AN ABEND LIMIT OF 8.                                       #
#-----------------------------------------------------------------#
EXITDEF(TYPE=INPUT,EXITS=(ZINPUT00),ABLIM=8,COMP=ODBM)

#-----------------------------------------------------------------#
# DEFINE 1 ODBM OUTPUT USER EXIT: ZOUTPUT0                        #
#-----------------------------------------------------------------#
EXITDEF(TYPE=OUTPUT,EXITS=(ZOUTPUT0),COMP=ODBM)
```

Sample OM BPE user exit list member of the IMS PROCLIB data set

In the sample OM BPE user exit list member of the IMS PROCLIB data set shown below, an OM input user exit, ZINPUT00, is defined with an abend limit of 8.

```
*******************************************************************
* OM USER EXIT LIST PROCLIB MEMBER                                *
*******************************************************************

#-----------------------------------------------------------------#
# DEFINE 1 OM INPUT USER EXIT: ZINPUT00                           #
# WITH AN ABEND LIMIT OF 8.                                       #
#-----------------------------------------------------------------#
EXITDEF(TYPE=INPUT,EXITS=(ZINPUT00),ABLIM=8,COMP=OM)
```

Sample RM BPE user exit list member of the IMS PROCLIB data set

In the sample RM BPE user exit list member of the IMS PROCLIB data set shown below, an RM client connection user exit and initialization-termination exit are defined. The client connection user exit has an abend limit of 8.

```
*******************************************************************
* RM USER EXIT LIST PROCLIB MEMBER                                *
*******************************************************************

#-----------------------------------------------------------------#
# DEFINE 1 RM CLIENT CONNECTION USER EXIT: ZRCLNCN0               #
# WITH AN ABEND LIMIT OF 8.                                       #
#-----------------------------------------------------------------#
EXITDEF(TYPE=CLNTCONN,EXITS=(ZRCLNCN0),ABLIM=8,COMP=RM)

#-----------------------------------------------------------------#
# DEFINE 1 RM INIT/TERM USER EXIT: ZRINTM00                       #
#-----------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZRINTM00),COMP=RM)
```

Sample SCI BPE user exit list member of the IMS PROCLIB data set

In the sample SCI BPE user exit list member of the IMS PROCLIB data set shown below, an SCI initialization-termination user exit is defined.

```
*******************************************************************
* SCI USER EXIT LIST PROCLIB MEMBER                               *
*******************************************************************
#-----------------------------------------------------------------#
```

```
# DEFINE 1 SCI INIT/TERM USER EXIT: ZSINTM00                    #
#--------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZSINTM00),COMP=SCI)
```

Sequence for starting CSL address spaces

The CSL comprises several address spaces, some of which need to be started either before or after other CSL and IMS address spaces.

The recommended starting sequence for the CSL is:

1. Start the SCI address space and IRLM.
2. Start the OM address space and, if your IMSplex includes queue structures and resource structures, CQS.
3. If needed, start the RM address space, which manages resources that are shared by multiple IMS systems in an IMSplex.
4. Start the IMS control region. The IMS control region then automatically initiates the DBRC address space.
5. If needed, start any other optional IMSplex components, such as ODBM (which routes database access requests to IMS DB in DB/TM and DBCTL configurations) and IMS Connect.

If CQS is started before the SCI address space, message CQS0001E is issued. See *IMS Version 11 Messages and Codes, Volume 2: Non-DFS Messages*.

Chapter 9. IMS Connect definition and tailoring

This topic describes the tasks of defining and tailoring IMS Connect. It provides detailed information about configuring and customizing the IMS Connect environment, as well as guidelines and procedures for using and invoking IMS Connect.

IMS Connect provides the following services:

- TCP/IP gateway to IMS, including support for:
  - TCP/IP access to IMS DB using Structured Call Interface (SCI) connections to Open Database Manager
  - TCP/IP access to IMS TM using z/OS cross-system coupling facility connections to Open Transaction Manager Access (OTMA)
- Local access to IMS for client applications running on WebSphere Application Server for z/OS clients using the z/OS Program Call Interface
- TCP/IP access to IMS type-2 commands through SCI and the IMS Operations Manager OM interface

Related concepts:

Overview of IMS Connect (Communications and Connections)

Sample JCL to start IMS Connect

You can use this sample JCL to start IMS Connect as a z/OS procedure or job.

You invoke IMS Connect using either a z/OS procedure or a z/OS job. If you start multiple instances of IMS Connect with the same configuration, a connection outage can occur. To avoid starting the same IMS Connect address space more than once, start IMS Connect by running a z/OS job with a unique z/OS initiator class assigned to it, rather than starting the connection as a procedure. The following sample starts IMS Connect as a z/OS job:

```
//HWS01    JOB   MSGLEVEL=1,TIME=1440,CLASS=Y,USERID=&USERID
//*******************************************************************
//*  BRING UP IMS CONNECT USING A JOB                               *
//*******************************************************************
//HWS01    EXEC HWS,SOUT=A
```

Two examples of starting IMS Connect as a z/OS procedure are shown below. In the first example, the JCL statements illustrate that both IMS Connect and BPE have configuration members (HWSCFG00 and BPECFGHT). In addition, PGM=HWSHWS00 is required to run IMS Connect in authorized supervisor state and key 7.

```
//HWS       PROC  RGN=4096K,SOUT=A,
//             BPECFG=BPECFGHT,
//             HWSCFG=HWSCFG00
//*
//*******************************************************************
//*  BRING UP AN IMS CONNECT USING HWSHWS00                         *
//*******************************************************************
//STEP1    EXEC PGM=HWSHWS00,REGION=&RGN,TIME=1440,
//             PARM='BPECFG=&BPECFG,HWSCFG=&HWSCFG'
//STEPLIB  DD   DSN=SDFSRESL,DISP=SHR
//         DD   DSN=CEE.SCEERUN,UNIT=SYSDA,DISP=SHR
```

© Copyright IBM Corp. 1974, 2015

```
//         DD   DSN=SYS1.CSSLIB,UNIT=SYSDA,DISP=SHR
//         DD   DSN=GSK.SGSKLOAD,UNIT=SYSDA,DISP=SHR
//PROCLIB  DD   DSN=USER.PROCLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=&SOUT
//SYSUDUMP DD   SYSOUT=&SOUT
//HWSRCORD DD   DSN=HWSRCDR,DISP=SHR
```

In the second example, the BPE program properties table entry, BPEINI00 is used instead of HWSHWS00. When you use BPEINI00, the address space runs in authorized supervisor state and key 7. IMS Connect uses BPEINI00 unless you explicitly identify PGM=HWSHWS00, as in the example above. If you use BPEINI00, you must also add the BPEINIT=HWSINI00 parameter, which identifies the HWS BPE initialization parameters module. A sample startup procedure using BPEINI00 is shown below.

```
//HWS       PROC  RGN=4096K,SOUT=A,
//            BPECFG=BPECFGHT,
//            HWSCFG=HWSCFG00
// *
// ************************************************************
// *  BRING UP AN IMS CONNECT USING BPEINI00
// ************************************************************
//STEP1    EXEC PGM=BPEINI00,REGION=0M,TIME=1440,
//             PARM='BPECFG=&BPECFG,BPEINIT=HWSINI00,
//             HWSCFG=&HWSCFG'
//STEPLIB  DD   DSN=SDFSRESL,DISP=SHR
//         DD   DSN=CEE.SCEERUN,UNIT=SYSDA,DISP=SHR
//         DD   DSN=SYS1.CSSLIB,UNIT=SYSDA,DISP=SHR
//         DD   DSN=GSK.SGSKLOAD,UNIT=SYSDA,DISP=SHR
//PROCLIB  DD   DSN=USER.PROCLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=&SOUT
//SYSUDUMP DD   SYSOUT=&SOUT
//HWSRCORD DD   DSN=HWSRCDR,DISP=SHR
```

Note: The IMS.SDFSRESL library is required in the STEPLIB, because all IMS Connect modules reside in the IMS.SDFSRESL library. IMS Connect requires the CEE.SCEERUN, SYS1.CSSLIB, and GSK.SGSKLOAD libraries (which are the C execution and z/OS system SSL libraries) only when SSL support is used.

Configuring IMS Connect

IMS Connect supports communication between one or more TCP/IP clients and IMS systems. You can configure multiple IMS systems on multiple z/OS images within a single sysplex and distribute the client request to the IMS systems (data stores).

1. Authorize the Authorized Program Facility (APF).
   a. Create and run a JCL job that authorizes IMS.SDFSRESL library to the APF.
   b. Issue the z/OS command `SETPROG APF,ADD,DSNAME=IMS.SDFSRESL,VOLUME=IMS001`.
2. Update the z/OS Program Properties Table (PPT). Updating the PPT allows IMS Connect to run in authorized supervisor state and in key 7.
   a. Edit the SCHEDxx member of the SYS1.PARMLIB data set.
   b. Add the required entries to the z/OS PPT. For TCP/IP communications only, add the following entry in the z/OS PPT:

```
PPT PGMNAME(HWSHWS00)     /* PROGRAM NAME = HWSHWS00          */
         CANCEL           /* PROGRAM CAN BE CANCELED          */
         KEY(7)           /* PROTECT KEY ASSIGNED IS 7        */
         SWAP             /* PROGRAM IS SWAPPABLE             */
         NOPRIV           /* PROGRAM IS NOT PRIVILEGED        */
         DSI              /* REQUIRES DATA SET INTEGRITY      */
```

```
        PASS            /* CANNOT BYPASS PASSWORD PROTECTION */
        SYST            /* PROGRAM IS A SYSTEM TASK          */
        AFF(NONE)       /* NO CPU AFFINITY                   */
        NOPREF          /* NO PREFERRED STORAGE FRAMES       */
```

If you are using local option for client communications, either by itself or with TCP/IP communications, add the following entry in the z/OS PPT:

```
PPT PGMNAME(HWSHWS00)     /* PROGRAM NAME = HWSHWS00           */
        CANCEL            /* PROGRAM CAN BE CANCELED           */
        KEY(7)            /* PROTECT KEY ASSIGNED IS 7         */
        NOSWAP            /* PROGRAM IS NOT SWAPPABLE          */
        NOPRIV            /* PROGRAM IS NOT PRIVILEGED         */
        DSI               /* REQUIRES DATA SET INTEGRITY       */
        PASS              /* CANNOT BYPASS PASSWORD PROTECTION */
        SYST              /* PROGRAM IS A SYSTEM TASK          */
        AFF(NONE)         /* NO CPU AFFINITY                   */
        NOPREF            /* NO PREFERRED STORAGE FRAMES       */
```

The only difference between the two PPT entries is the use of SWAP or NOSWAP.

c. IPL the z/OS system again or issue the z/OS SET SCH= command to make the changes effective.

3. Enable Internet Protocol Version 6 (IPV6) for IMS Connect.
   a. Ensure that IMS Connect is running.
   b. Customize the BPXPRMxx member, as follows:

```
FILESYSTYPE Type(INET) Entrypoint(EZBPFINI)
NETWORK DOMAINNAME(AF_INET)
DOMAINNUMBER(2)
  MAXSOCKETS(2000)
  TYPE(INET)

NETWORK DOMAINNAME(AF_INET6)
DOMAINNUMBER(19)
  MAXSOCKETS(3000)
  TYPE(INET)
```

   c. Recycle the TCP/IP stack. For more information about customizing this member, see *z/OS UNIX System Services Planning*.
   d. Customize the IMS Connect configuration member using the IPV6 parameter.
   e. For each of the READ subroutines shown below that you use, determine whether the EXPREA_IPV6 bit is turned on in the EXPREA_FLAG2 field of the READ subroutine. If it is turned on, IPV6 is enabled.
      - HWSJAVA0
      - HWSSMPL0
      - HWSSMPL1
   f. Map EXPREA_SOCKET6 to the AF_INET6 socket address structure. See "READ subroutine" in *IMS Version 11 Exit Routines* for more information.

   The IP address format when IPV6 is enabled is described in "VIEWHWS command" in *IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands*.

4. Create an IMS Connect configuration member in IMS.PROCLIB data set to hold the configuration statements that IMS Connect uses during initialization. You can specify the following configuration statements for IMS Connect to use:
   - ADAPTER
   - DATASTORE

- HWS
- IMSPLEX
- ODACCESS
- TCPIP

After completing these steps, you can define IMS security and optionally enable the IMS Connect XML Message Conversion support.

Related reference:

"HWSCFGxx member of the IMS PROCLIB data set" on page 782

Defining IMS Connect security

Properly defining security for IMS Connect using RACF, and assigning IMS Connect z/OS UNIX System Services superuser privileges ensures that IMS Connect can open ports.

You can start IMS Connect as a job or as a procedure. If the data store (which is IMS) is RACF-protected, you have to start IMS Connect as a job with the JOB statement specifying a valid *USERID* to make the connection from IMS Connect to IMS, or you can use the RACF-started procedure table. The *USERID=&userid* parameter specified in the JOB card of the IMS Connect job JCL is used as the security vehicle to ensure IMS Connect access to IMS. *&USERID* must have READ access to *IMSXCF.group.member*. IMS OTMA provides security for the IMS z/OS cross-system coupling facility connection by defining and permitting *IMSXCF.group.member* in the RACF *FACILITY* class.

To configure security for the local option using RACF, you must add HWS.ICON_NAME as the SAF facility class name.

ICON_NAME is how IMS Connect is defined in the ID parameter of the HWS statement in the IMS Connect configuration member. The resource that must access IMS Connect is the WebSphere Application Server, and UPDATE authority is required to update the RACF profile.

When you add HWS.ICON_NAME as the SAF facility class name, IMS Connect security is in effect and requires IMS Connect access through the local option to be authorized. IMS Connect security is in effect even if RACF=Y has not been specified in the IMS Connect configuration member of the IMS PROCLIB data set (HWSCFGxx) or the SETRACF command has not been issued.

IMS Connect must have z/OS Unix System Services () superuser privileges so IMS Connect can ensure compatibility between the IMS Connect MAXSOC parameter and the UNIX System Services MAXFILEPROC parameter.

The IMS Connect MAXSOC parameter is related to the UNIX System Services parameter MAXFILEPROC. The values of MAXSOC and MAXFILEPROC must be compatible. If the values of each parameter are not compatible, IMS Connect cannot open any ports. You can ensure compatibility between MAXSOC and MAXFILEPROC by granting IMS Connect UNIX System Services superuser privileges, which allows IMS Connect to change the value of the MAXFILEPROC parameter automatically. You can grant UNIX System Services superuser privileges to IMS Connect by using the RACF command `ALTERUSER` to assign an OMVS segment with a UID of 0 to the user ID of the IMS Connect started task.

Alternatively, your UNIX System Services administrator can adjust the value of MAXFILEPROC directly in the BPXPRMxx member of the z/OS SYS1.PARMLIB data set.

Related reading: For more details about OTMA and IMS Connect security, see *IMS Version 11 Communications and Connections*.

Setting up AT-TLS SSL for IMS Connect

You can set up IBM z/OS Communications Server Application Transparent Transport Layer Security (AT-TLS) to set up Secure Socket Layer (SSL) on TCP/IP connections to IMS Connect. Setting up AT-TLS is the recommended method for enabling SSL for IMS Connect.

Perform the following procedure to set up SSL for IMS Connect by setting up AT-TLS:

1. On each z/OS system where you run IMS Connect, create a server key ring with a server certificate and the necessary certificate authority certificates.
2. Create Policy Agent files:
   a. Create a Policy Agent main configuration file containing a TcpImage statement for the server stack. Set the TcpImage statement to point to the image configuration file.
   b. Create a Policy Agent image configuration file containing a TTLSConfig statement for the server stack. Set the TTLSConfig statement to point to the TTLSConfig policy file.
   c. Create and configure a Policy Agent TTLSConfig policy file, and add the AT-TLS policy statements to this file. Most of the SSL settings are in this file. Set the security suite you want to support, and specify which ports to use SSL.
   d. Store all the configuration files in a z/OS UNIX System Services file system directory that can be accessed when you start the Policy Agent.
3. Activate the RACF SERVAUTH class. Issue the following command from TSO:

   ```
   SETROPTS CLASSACT(SERVAUTH)
   ```

4. Set up InitStack access control:
   a. Define the EZB.INITSTACK.*SYSNAME.TCPNAME* profile for each AT-TLS stack.
   b. Permit administrative applications to use the stack before AT-TLS is initialized.

   The following is sample JCL to set up InitStack access control (based on the member EZARACF in sample data set SEZAINST):

   ```
   //TLSRACF JOB MSGLEVEL=(1,1),USER=USERNAME,PASSWORD=PASSWORD,
   //        CLASS=A,MSGCLASS=A
   //*
   /*ROUTE PRINT THISCPU/IMSNAME
   //   EXEC PGM=IKJEFT01
   //SYSTSPRT DD SYSOUT=*
   //SYSABEND DD SYSOUT=*
   //SYSTSIN DD *
   SETROPTS RACLIST (SERVAUTH)
   SETROPTS CLASSACT(SERVAUTH)
   SETROPTS GENERIC (SERVAUTH)
   RDEFINE SERVAUTH EZB.INITSTACK.SYSNAME.TCPNAME UACC(NONE)
   PERMIT EZB.INITSTACK.SYSNAME.TCPIP CLASS(SERVAUTH) ID(*) ACCESS(READ) -
          WHEN(PROGRAM(PAGENT,EZAPAGEN))
   ```

```
SETROPTS GENERIC(SERVAUTH) REFRESH
SETROPTS RACLIST(SERVAUTH) REFRESH
SETROPTS WHEN(PROGRAM) REFRESH
//
```

From the z/OS console, you can find the *SYSNAME* in the SYS= field. To find the *TCPNAME*, issue the `D A,L` command.

5. Enable AT-TLS. Perform one of the following steps:
   - Add the following lines to the SYS1.TCPPARMS(PROFILE) member:
     ```
     ;Enable AT-TLS support
     TCPCONFIG TTLS
     ```
   - Put the following lines in a separate file and run the `V TCPIP,,O,SYS1.TCPPARMS(`*`filename`*`)` (VARY) command:
     ```
     ;Enable AT-TLS support
     TCPCONFIG TTLS
     ```

     Note: If you perform this step, skip steps 6 and 8.
6. If TCP/IP and IMS Connect are running, stop TCP/IP and IMS Connect. To stop IMS Connect, issue the IMS Connect WTOR command CLOSEHWS.
7. Configure and start the syslog daemon (syslogd):
   a. Review your syslogd configuration (/etc/syslog.conf) to verify that messages written by the Policy Agent and TCP/IP stacks are saved in the intended files. AT-TLS syslogd messages are written to the daemon facility by default. A sample configuration file is provided by z/OS in /usr/lpp/tcpip/samples/syslog.conf.
   b. Start syslogd with the following command:
      ```
      /usr/sbin/syslogd -f /etc/syslog.conf &
      ```

   Note: To stop this job, run the following command: `kill` *`process_ID`*. To find the *process_ID*, run the following command: `ps -A`.
8. Start the TCP/IP stack. Run the following command from the z/OS console:
   ```
   S TCPIP
   ```
9. Start the Policy Agent and verify that there were no policy errors in processing the policy files:
   a. Run the following command:
      ```
      /usr/sbin/pagent -c /etc/sysname_pagent.conf -l SYSLOGD &
      ```

      The pagent executable is in /usr/sbin.
   b. Verify that the TCP/IP stack has received the AT-TLS policy and has released console message EZZ4248E.

   Note: To stop this job, run the following command: `kill` *`process_ID`*. To find the *process_ID*, run the following command: `ps -A`.
10. Update the IMS Connect configuration member (HWSCFGxx) and start IMS Connect:
    a. In HWSCFGxx, specify the ports that you want to use for SSL as regular non-SSL ports. Perform one or both of the following steps:
       - If you are connecting to IMS TM, specify the ports in the PORT and PORTID parameters on the TCPIP statement.
       - If you are connecting to IMS DB, specify the ports in the DRDAPORT parameter on the ODACCESS statement.

If a specified port matches one of the SSL ports from the *sysname*_pagent_TTLS.conf file, an SSL client can connect to that port. If no specified ports match an SSL port from the *sysname*_pagent_TTLS.conf file, only non-SSL clients can connect to it.

b. If you are migrating from using IMS Connect's SSL function to using AT-TLS, then remove the SSLPORT parameter and specify the port number in the PORT or PORTID parameter. Here is an example that specifies port 8888 to use IMS Connect's SSL function:

```
TCPIP=(HOSTNAME=TCPIP,PORTID=(9999,9998,LOCAL), SSLPORT=(8888),
```

To change port 8888 to use AT-TLS, remove SSLPORT and specify port 8888 in the PORTID parameter:

```
TCPIP=(HOSTNAME=TCPIP,PORTID=(9999,9998,8888,LOCAL),
```

IMS Connect treats port 8888 as a normal port. Traffic coming through this port does not go through IMS Connect's SSL function. All the SSL processing would be done by the z/OS AT-TLS stack instead.

c. Start IMS Connect by running JCL.

What to do next

- Configure the IMS Universal database resource adapter for SSL support in a container-managed environment.
- Configure IMS Universal drivers for SSL support in a stand-alone environment.

Related concepts:

z/OS: TLS/SSL security

z/OS: AT-TLS policy configuration

Related tasks:

Configuring the IMS Universal Database resource adapter for SSL support in a container-managed environment (Application Programming)

Configuring IMS Universal drivers for SSL support in a stand-alone environment (Application Programming)

Related reference:

"HWSCFGxx member of the IMS PROCLIB data set" on page 782

z/OS: TcpImage and PEPInstance

IBM Redbooks: Defining the TcpImage statements

z/OS: TTLSConfig statement

z/OS: TCP/IP stack initialization access control

z/OS: VARY TCPIP,,START or VARY TCPIP,,STOP

z/OS: Configuring the syslog daemon

z/OS: Starting and stopping the Policy Agent

Configuring XML conversion support for IMS Connect clients

You must configure IMS Connect to convert the input and output messages between an IMS Connect client and an IMS application from XML to a language-specific structure, when the IMS Connect client is IMS Enterprise Suite SOAP Gateway.

The following information does not apply to IMS Connect clients that are IMS Universal drivers or custom-built applications that use the Distributed Relational Database Architecture™ (DRDA®) protocol.

Prerequisites:

- You must have the COBOL copybook or the PL/I source files for the IMS application program that processes the input messages or issues a callout request.
- You must increase the IMS Connect region size to accommodate the storage used by the XML converters.
- You must have IMS Enterprise Suite SOAP Gateway.

IMS Connect can convert the XML data contained in the message from an IMS Connect client into the COBOL or PL/I data used by an IMS application program. The COBOL or PL/I data in the corresponding output message is also converted back to the XML data that the IMS Connect client expects. This XML conversion support enables IMS to accept IMS Connect client messages in an XML format without modifying IMS application programs to support XML natively.

The XML conversion function supports conversion of single-segment and multi-segment messages. See the SOAP Gateway and IBM Rational® Developer for System z documentation for the restrictions that apply to multi-segment message support.

Restrictions:

- IMS Enterprise Suite SOAP Gateway is the only supported client.
- Inbound and outbound messages must be encoded in UTF-8 because IMS Enterprise Suite SOAP Gateway supports only UTF-8. This encoding must match the encoding of the XML Converter that is being requested.
- IMS Connect provides XML conversion support for commit mode 1 and sync level 0 messages.

The XML conversion function supports conversion of single-segment and multi-segment messages. See the SOAP Gateway and Rational Developer for System z documentation for the restrictions that apply to multi-segment message support.

To configure IMS Connect to convert XML data from the client into COBOL or PL/I IMS application program data, you must perform the following basic steps:

1. Include the ADAPTER configuration statement, ADAPTER=(XML=Y), in the IMS Connect configuration member HWSCFGxx.
2. Specify the HWSSOAP1 user message exit in the EXIT= parameter of the TCPIP configuration statement.
3. Define the XML adapter as a BPE exit routine for IMS Connect by coding a BPE exit list IMS PROCLIB data set member and specifying it in the BPE configuration parameter IMS PROCLIB data set member:
   a. Create a BPE exit list IMS PROCLIB data set member with any name, for example HWSEXIT0. In the BPE exit list IMS PROCLIB data set member, define the XML Adapter (HWSXMLA0) as an exit by setting the following EXITDEF statement:

      ```
      EXITDEF(TYPE=XMLADAP,EXITS=(HWSXMLA0),ABLIM=8,COMP=HWS)
      ```

All parameters must be coded as shown except for ABLIM, which sets the number of times the XML adapter can abend before it is disabled.

b. Set the BPE exit list IMS PROCLIB data set member in the BPE configuration parameter IMS PROCLIB data set member by adding an EXITMBR statement. For example, if the BPE exit list IMS PROCLIB data set member is HWSEXIT0, then add the following statement to the BPE configuration member:

```
EXITMBR=(HWSEXIT0,HWS)
```

4. Configure the z/OS Unicode Conversion Services to support character conversions from UTF-8 to EBCDIC and from EBCDIC to UTF-8. Most likely, your z/OS system administrator performs this task. For more information about z/OS Unicode support, see *z/OS Unicode Services User's Guide and Reference*.
5. Generate the XML converter to convert the XML data to the COBOL or PL/I data that the IMS application expects. XML converters are COBOL or PL/I application programs that are based on the COBOL copybook or PL/I source code of the IMS application program that processes the transactions between the IMS Connect client and the IMS application.

   The recommended method is by using the separately licensed tool Rational Developer for System z to generate the converters. Each IMS application requires a unique XML converter. For IMS Enterprise Suite SOAP Gateway, you can find an example of an XML converter in the IMS Enterprise Suite SOAP Gateway Phone Book Sample. You can download the sample from the IMS Enterprise Suite SOAP Gateway home page.
6. Compile and link the XML converter into an APF-authorized data set that is concatenated to the STEPLIB in the IMS Connect startup JCL. When linking the XML converter, specify an additional program entry name for an internal service as an ALIAS in the link job. The additional program entry name has the same name as the converter, except that the last character is an X. For example, if the converter name is CNVNAMED, the additional program entry name is CNVNAMEX.
7. Restart IMS Connect if both of the following conditions are true:
   - You are updating an existing XML converter (the converter exists in the data set).
   - The converter has already been loaded by IMS Connect (the converter was called before).
8. Configure your IMS Enterprise Suite SOAP Gateway and provide the generated XML converter. For detailed configuration instructions for IMS Enterprise Suite SOAP Gateway, see the XML-formatted IMS messages documentation.

Example IMS Connect configuration statements:

The following example shows the IMS Connect configuration statement specifications required to enable the XML conversion function of IMS Connect:

```
************************************************
* HWS EXAMPLE OF INCLUDING XML ADAPTER SUPPORT
************************************************
HWS=(ID=HWS8,RACF=Y,XIBAREA=20)
TCPIP=(HOSTNAME=MVSTCPIP,RACFID=RACFID,
PORTID=(9999,LOCAL),MAXSOC=2000,TIMEOUT=8800,
EXIT=(HWSSMPL1,HWSSOAP1))
ADAPTER=(XML=Y)
************************************************
```

Example JCL to compile and link a COBOL program:

The following example JCL compiles and links an XML converter (for COBOL):

```
//COBLNK JOB (PLS,81038),'user',CLASS=A,REGION=4096K,
// MSGLEVEL=(1,1),MSGCLASS=A,NOTIFY=user
//*************************************************************
//* JOB TO COMPILE AND LINK A COBOL PROGRAM
//*************************************************************
//COMP EXEC PGM=IGYCRCTL
//STEPLIB DD DSN=COBOL.V3R4.X,DISP=SHR
//SYSIN DD DSN=COBOL.SOURCE(NNNNNNND),DISP=SHR
//SYSLIN DD DSN=&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,
// DISP=(MOD,PASS),
// DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)
//SYSUT1 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSUT2 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSUT3 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSUT4 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSUT5 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSUT6 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSUT7 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
/*
//*************************************************************
//LKED EXEC PGM=IEWL,COND=(4,LT,COMP),
// PARM='LIST,LET,XREF,MAP'
//SYSLIB DD DSN=CEE.SCEELKED,DISP=SHR
// DD DSN=CEE.SCEEBIND,DISP=SHR
// DD DSN=CEE.SCEEBND2,DISP=SHR
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
// DD DDNAME=SYSIN
//SYSLMOD DD DSN=IMSCONN.AUTHLIB,DISP=SHR
//SYSUT1 DD UNIT=VIO,SPACE=(1024,(120,120),,,ROUND)
//SYSPRINT DD SYSOUT=*
//LKED.SYSIN DD *
ENTRY NNNNNNND
ALIAS NNNNNNNX
NAME NNNNNNND(R)
/*
```

For information about the XML, COBOL, and PL/I data structures of converted input and output messages, see *IMS Version 11 Application Programming*.

Related concepts:

IMS Connect XML message conversion (Communications and Connections)

Related reference:

"HWSCFGxx member of the IMS PROCLIB data set" on page 782

Configuring the IMS Base Primitive Environment for IMS Connect

The IMS Connect address space is built on top of the Base Primitive Environment (BPE).

Changing the BPE configuration parameter member of the IMS PROCLIB data set

The BPE configuration parameter member of the IMS PROCLIB data set defines BPE execution environment settings for the IMS Connect address space. You specify the IMS PROCLIB data set member name by coding BPECFG=*member_name* on the EXEC PARM= statement in the IMS Connect address space startup JCL, as shown in the following example:

```
EXEC HWSHWS00,PARM='BPECFG=BPECFGHW'
```

You can use the BPE configuration parameter member of the IMS PROCLIB data set to specify the following items:

- The language used for BPE and IMS Connect messages
- The trace level settings for BPE and IMS Connect internal trace tables

The keywords that are available for the BPE configuration parameter member of the IMS PROCLIB data set are LANG= and TRCLEV=.

Avoid coding statements in the BPE configuration member that specify definitions for the same resources more than one time. For example, do not specify multiple TRCLEV statements for the same trace table type, or multiple EXITMBR statements for the same IMS Connect. BPE uses the last statement it encounters in the member. Any values that are specified on earlier duplicate statements are ignored. A message BPE0017I is issued for each duplicate found.

The IMS Connect Recorder trace table, RCTR, requires explicit commands, and therefore, is not processed if a default TRCLEV statement, TRCLEV=(*, ,HWS), is defined for IMS Connect. Recorder trace table configuration must be explicitly specified with a separate, additional TRCLEV statement. For example: `TRCLEV=(RCTR,MEDIUM,HWS)`.

Recommendation: Enable the RCTR trace by issuing the BPE UPDATE TRACETABLE command. Use the BPE configuration member only if you must capture the trace before the BPE UPDATE TRACETABLE command can be issued.

Formatting incore trace tables

IMS Connect trace tables are *incore* tables, which can be formatted from a dump of an IMS Connect address space using the IMS Connect dump formatter. The traces are formatted by the standard BPE formatting services.

Example of a configuration file for BPE

A sample BPE configuration data set is shown in the following figure.

```
********************************************************************
* CONFIGURATION FILE FOR BPE                                       *
********************************************************************

LANG=ENU                              /* LANGUAGE FOR MESSAGES     */
                                      /* (ENU = U.S. ENGLISH)      */

#
# DEFINITIONS FOR BPE SYSTEM TRACES
#

TRCLEV=(*,LOW,BPE)                    /* DEFAULT TRACES TO LOW     */
TRCLEV=(AWE,HIGH,BPE)                 /* AWE SERVER TRACE ON HIGH  */
TRCLEV=(CBS,MEDIUM,BPE)               /* CTRL BLK SRVCS TRC ON MED */
TRCLEV=(DISP,HIGH,BPE,PAGES=12)       /* DISPATCHER TRACE ON HIGH  */
                                      /* WITH 12 PAGES (48K BYTES) */
#
# DEFINITIONS FOR IMS CONNECT TRACES
#

TRCLEV=(*,HIGH,HWS)      /* DEFAULT ALL IMS CONNECT TRACES TO HIGH     */
TRCLEV=(HWSI,MEDIUM,HWS) /* BUT RUN IMS CONNECT TO IMS OTMA TRACE...   */
TRCLEV=(HWSW,MEDIUM,HWS) /* AND SERVER TO IMS CONNECT TRACE AT MEDIUM  */
TRCLEV=(RCTR,MEDIUM,HWS) /* AND SET RECORDER TRACE TO MEDIUM BECAUSE   */
                         /* IT IS NOT PROCESSED WITH THE ALL DEFAULT   */
```

Related reference:

BPE UPDATE TRACETABLE command (Commands)

Allocating IMS Connect log record data sets

IMS Connect writes log records to the HWSRCDR data set.

Recommendation: Instead of setting up IMS Connect to write log records to the HWSRCDR data set, use the Base Primitive Environment (BPE) to manage the output of the IMS Connect Recorder Trace facility. Configure BPE for an external trace of IMS Connect.

To allocate the HWSRCDR data set from TSO, use the following settings:

```
                                Data Set Information
Command ===>

Data Set Name . . . .: HWSRCDR

General Data                          Current Allocation
 Volume serial. . . .: USER01          Allocated cylinders: 1
 Device type  . . . .: 3390            Allocated extents. : 1
 Organization . . . .: PS
 Record format. . . .: FB
 Record length. . . .: 1440
 Block size . . . . .: 14400          Current Utilization
 1st extent cylinders: 1               Used cylinders . . : 1
 Secondary cylinders : 5               Used extents . . . : 1

 Creation date  . . .: 2009/10/01
 Expiration date  . .: ***None***
```

Refer to the *Program Directory for Information Management System Transaction and Database Servers* for the recommended allocations for the required IMS Connect libraries.

Related tasks:

Configuring BPE for an external trace of IMS Connect (Diagnosis)

JCL to print IMS Connect RECORDER output

IMS Connect provides a line trace function to capture data that is received from and sent to a client.

The line trace contains a copy of the first 670 bytes of the data as it is passed to the user message exit and upon return from the user message exit. Line traces are intended for use in problem resolution.

Use the `RECORDER` command to activate and terminate the line trace function.

The following sample JCL illustrates how to print the line trace data set:

```
//IDCAMS   JOB  JOB 1,IDCAMS,MSGLEVEL=1,CLASS=K,TIME=1440
//SELECT   EXEC PGM=IDCAMS
//DD1      DD   DSNAME=HWSRCDR,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
  PRINT    INFILE(DD1)
```

For information about the IMS Connect Event Recorder exit routine (HWSTECL0), see *IMS Version 11 Exit Routines*.

Chapter 10. Accessing external subsystem data

These topics describe the installation and definition tasks that you need to perform if you are an IMS user that wants to access external subsystem data, such as data that resides in DB2 for z/OS.

Defining your external subsystems to IMS

In order for IMS to initiate contact with an external subsystem, you must add a subsystem member (SSM) to the IMS PROCLIB data set for each external subsystem with which IMS communicates.

This SSM contains an entry for each external subsystem with which IMS communicates. In the case of DB2 for z/OS with several possible definitions for the same DB2 for z/OS system, the member contains one access for Java regions in IMS and the other for other region types. When defining DB2 for z/OS databases, this member contains one entry for each DB2 for z/OS subsystem, or an entry that defines a group of DB2 for z/OS subsystems.

You can also establish simultaneous access to the same DB2 for z/OS subsystem if the IMS PROCLIB data set member contains a definition for both connections.

Define the SSM PROCLIB member (or members) using the IMS PROCLIB data set that is allocated before Stage 2 of IMS system definition and the z/OS utility IEBUPDTE. The first 1 - 4 characters of this PROCLIB member's name must match the IMSID, as specified by the IMSID parameter on the IMSCTRL macro statement, or it must be overridden using the execution parameter IMSID=. The last four positions represent any unique installation-defined identifier. This identifier is appended to the IMSID to construct the member name.

After defining the SSM PROCLIB member, before you can use it you must:

1. Insert a DD statement for IMS PROCLIB with the DD name PROCLIB in the step execution JCL.
2. Specify the correct PROCLIB member on the EXEC statement parameter SSM= or on the `/START SUBSYS SSM` command.

Each entry within the PROCLIB member consists of a blocked or unblocked 80-character record. Information for each record begins in position 1 of the record.

Specifying the SSM= EXEC parameter

Use the SSM= EXEC parameter to indicate which member of the IMS PROCLIB data set contains an entry for the external subsystem you want to attach.

Use the SSM= EXEC parameter to indicate which member of the IMS PROCLIB data set contains an entry for the external subsystem you want to attach. You can specify the SSM= EXEC parameter on the EXEC procedures of that control region, or for an MPP, BMP, or IFP-dependent region.

The SSM= EXEC parameter is required for an MPP, BMP, IFP, JMP, or JBP region when the subsystem member defines a DB2 for z/OS group.

© Copyright IBM Corp. 1974, 2015

If the SSM= EXEC parameter is not specified when the IMS subsystem is initialized, the ESAF or RRSAF component is not activated. You can later activate the ESAF component and enable external subsystem support in one of two ways:

- Shut down IMS and restart it, specifying the SSM= EXEC parameter.
- Use the /START SUBSYS SSM command to inform IMS of the subsystem member. This action activates the ESAF component and attempts a connection to the external subsystems defined in the SSM member of the IMS PROCLIB data set.

Restriction: You can activate the RRSAF component and enable DB2 for z/OS support only by shutting down IMS and restarting it with the SSM= EXEC parameter specified.

For an MPP or an IFP dependent region that is currently active but not connected, you must stop and start the region for a connection for that dependent region.

When SSM= is specified for the control region, any dependent region running under the control region can attach to the external subsystems named in the member of the IMS PROCLIB data set pointed to by the SSM= parameter.

When SSM= is specified for an MPP, BMP, or IFP region, it provides a filter that determines whether a given application program running in that dependent region can access external subsystem resources. Only the subsystems defined in the subsystem member can be connected to the dependent region. If a subsystem is defined in the named dependent region member, but not in the control region member, a message is sent to the MTO during dependent region initialization and a subsystem connection is not made for this dependent region entry.

For example, if a subsystem entry for a given DB2 for z/OS subsystem exists in the named dependent region member, but not in the control region member, then a message is sent to the MTO during dependent region initialization and a subsystem connection is not made for this dependent region entry.

If SSM is specified on the control region but the installation wants a particular dependent region to be prohibited from attaching to any external subsystem, a member of the IMS PROCLIB data set must be created having no entries, and its ID specified in the dependent region. Application programs in this dependent region cannot attach to any external subsystem.

When the member of the IMS PROCLIB data set contains a null record, you receive the following informational message:

```
DFS3600I UNABLE TO INITIATE THE EXTERNAL SUBSYSTEM TABLE, RC=36
```

If an invalid member name is specified or if the record format is incorrect, no external subsystem is attached, but initialization continues. You can then enter the /START SUBSYS SSM command, specifying the correct subsystem name. If the /START SUBSYS SSM command is issued with an invalid subsystem member name, re-enter the command with a valid name.

Restriction: You cannot use the /START SUBSYS SSM command for external subsystems attached by RRSAF. If an invalid member name is specified or if the record format is incorrect, you must restart IMS with the corrected member name or record format.

If any record in the SSM member of the IMS PROCLIB data set is invalid, a DFS3600 message is issued and processing continues. Valid records allow an

external subsystem to be attached. Invalid records in the SSM member of the IMS PROCLIB data set are ignored but can be corrected. Those external subsystems can be attached to the IMS control region using the /START SUBSYS command.

Defining the language interface module

In order for an IMS application program to use the language interface module, the application program modules must be bound using the IMS reenterable DL/I language interface. IMS provides the DFSLI macro, which you can use to generate a language interface module. This topic describes the DFSLI macro and its parameters.

In order for an IMS application program to use the language interface module, the application program modules must be bound using the IMS reenterable DL/I language interface. The true attributes must be specified; for example, if a module is reusable only, do not specify "RENT."

Restriction: IMS does not support dynamic calls to the language interface.

To make dynamic calls, provide the RMODE value at bind time so that the language interface is shown below the 16 MB line.

The IMS DL/I language interface is not reenterable. Any IMS application programs that were designed to be reenterable or serially reusable can use external subsystem resources only after being bound with the IMS language interface.

Important: IMS and DB2 for z/OS share a common alias name, DSNHLI, for the language interface module. Keep this in mind when concatenating your libraries to ensure the correct order.

- If you are running an IMS dependent region with an application program using the COBOL dynamic option, the IMS library must be concatenated first.
- If you are running an application program strictly under DB2 for z/OS, be sure the DB2 for z/OS library is concatenated first.

For subsystems other than DB2 for z/OS, IMS provides a language interface module, DFSLI000, which supports the external subsystem interface. DFSLI000 passes a language interface token (LIT) value of SYS1. The installation can use this module or it can define its own language interface if it wants to use a LIT value other than SYS1. When two or more external subsystems are accessed by the IMS system, the installation must define its own language interface modules because each subsystem has a unique LIT.

IMS provides the DFSLI macro to assist the installation in generating a language interface module. The code necessary to perform the language interface function is generated in the DFSLI macro expansion. The IMS macro library must be supplied when the macro statements are compiled to generate the module. The format of this macro is as follows, where user-supplied values must be substituted for the x's.

```
DFSLIxxx   DFSLI   TYPE=V2DB,LIT=xxxx
           END
```

where:

DFSLI*xxx*
Specifies the CSECT name for the module.

Recommendation: Make the CSECT name match the bind name of the language interface module.

For the IMS-supplied language interface module, the value for this parameter is DFSLI000.

TYPE=
Specifies the language interface type to IMS. The only value that you can enter for this parameter is `V2DB`.

LIT=
Specifies the language interface token that relates this language interface to an external subsystem with their respective specifications in a PROCLIB member entry as the LIT and SSN parameters. The IMS-supplied language interface module DFSLI000 uses a LIT value of `SYS1`.

Example: When IMS issues an external subsystem request, IMS knows the target subsystem by the LIT used in the request. For example, consider the case of an IMS dependent region accessing two external subsystems ESS1 and ESS2.

- You already have a default language interface module DFSLI000, as specified by the default LIT, `SYS1`. You now generate a second language interface DFSLI001 with a LIT of `SYS2`.
- Define two entries in the PROCLIB SSM member. The first entry points to ESS1 with `LIT=SYS1`. The second entry points to ESS2 with `LIT=SYS2`.
- Bind IMS application programs accessing ESS1 with the IMS-provided language interface module DFSLI000.
- Bind IMS application programs accessing ESS2 with the language interface module that you have generated, DFSLI001.

Although a region can communicate with two or more external subsystems, an IMS application can access only the external subsystem referred to in the bound language interface. However, you can alter the SSM member to route application requests to a different external subsystem from the subsystem defined in the language interface. Alternatively, you can use the DFSESS entry point in the language interface module.

Specifying trace options

Use the SUBS= parameter on the OPTIONS statement to specify trace options for external subsystems.

If you want to trace the connection and disconnection activities of IMS to the external subsystem, the parameter SUBS= on the OPTIONS statement must be set to ON. Specifying OUT on this parameter causes trace information to be written to the log. If this parameter is omitted, traces can be turned on and off with the `/TRACE` command.

Specifying DB2 for z/OS groups for IMS online regions

IMS supports the use of DB2 for z/OS data sharing groups.

You can use DB2 for z/OS group names in all dependent regions. The DB2 for z/OS group name is used to call z/OS Name Services. If z/OS returns a positive response to the group inquiry, IMS selects a member of the DB2 for z/OS group that is connected to the IMS control region and connects that DB2 for z/OS to the

dependent region under the following condition: The DB2 for z/OS name that was selected was not explicitly defined by another statement with the SSM member.

If that condition is not met, IMS ignores the statement that caused the z/OS Name Services call, and no message is issued.

IMS dependent regions examine the subsystem name that is specified in their SSM member. If the control region is connected to a DB2 for z/OS with that subsystem name, the dependent region connects to it. If the control region is not connected to it, the dependent region assumes that it might be a DB2 for z/OS group attach name. In this case, the dependent region uses this name to access the z/OS Name Services to find any DB2 for z/OS subsystem on the z/OS image that is using this group attach name. It also checks to see if its control region is connected to this DB2 for z/OS and, if so, connects to the specific DB2 for z/OS.

If you are using DB2 for z/OS groups:

- SSM= must be specified when the dependent region starts.
- The name specified on SSN= must be a one- to four-character DB2 for z/OS group name.
- When CRC= is specified and a system is selected from a DB2 for z/OS group, the CRC is changed to that of the control region's corresponding SSM member statement.

IMS BMP, JMP, JBP, MPP, and IFP (dependent regions) support DB2 group attach. The control region does not. For the control region, you must point to the correct subsystem name running on the same LPAR with the IMS.

In the SSM for the control region, you include one of the DB2 for z/OS subsystems in the DB2 for z/OS group that you may use. The SSM for the control region can only specify one member of the group. You cannot specify the DB2 for z/OS group name in the SSM for the control region. In the SSM for the dependent region you can specify the DB2 for z/OS group name and IMS (with help from z/OS) determines if the DB2 for z/OS group can connect or not.

A DB2 for z/OS group name cannot be the same as any DB2 for z/OS subsystem name.

An IMS control region cannot connect to more than one DB2 for z/OS in the same group; however, different IMS control regions on the same z/OS can connect to different instances of DB2 for z/OS in the same group.

A BMP that accesses DB2 for z/OS can be submitted to run on any IMS system in a Parallel Sysplex® environment without requiring modification of the SSM parameter or the SSM member of the IMS PROCLIB data set. BMPs that run with different instances of DB2 for z/OS in a data sharing group can use the same SSM member referring to the same DB2 for z/OS. The IMS control region must specify the specific DB2 for z/OS subsystem name or names in its SSM member. IMS control regions cannot use the DB2 for z/OS group attach name.

Examples

For the following two examples, the DB2 for z/OS group name is GRP1 and contains DB2 for z/OS subsystems DB21 and DB22:

Example 1

1. Specify the following in Control Region 1's SSM PROCLIB member:

   ```
   SST=DB2,SSN=DB21,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
   ```

2. Specify the following in Dependent Region 1 of Control Region 1's PROCLIB member:

   ```
   SST=DB2,SSN=GRP1,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
   ```

   An application can run here and get to DB21.

3. Specify the following in Control Region 2's SSM PROCLIB member:

   ```
   SST=DB2,SSN=DB22,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
   ```

4. Specify the following in Dependent Region 2 of Control Region 2's PROCLIB member:

   ```
   SST=DB2,SSN=GRP1,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
   ```

Example 2

1. Specify the following in Control Region 1's SSM PROCLIB member:

   ```
   SST=DB2,SSN=DB21,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
   SST=DB2,SSN=DB22,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
   ```

2. Specify the following in Dependent Region 1 of Control Region 1's PROCLIB member:

   ```
   SST=DB2,SSN=GRP1,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
   ```

In this example, IMS connects to DB2 for z/OS subsystem DB21 or DB22, depending on which DB2 for z/OS is available. The dependent region uses z/OS Name Services to determine if GRP1 is a DB2 for z/OS group, check if any of the systems defined in the Control Region's SSM are within that group, and connect to the first one that is active. If both DB2 for z/OS systems are active, connected to IMS, and part of the same DB2 for z/OS group, then the connection is made to the first one in the list. If GRP1 is not found to be a DB2 for z/OS group using z/OS Name Services, then an attempt is made to connect to GRP1 as a specific DB2 for z/OS.

Specifying DB2 for z/OS for IMS batch regions

The SSM= parameter, instead of the DDITV02 DD statement, can be used in IMS DL/I and DBB batch regions to specify the DB2 for z/OS connection parameters.

The subsystem member in PROCLIB is the same format as for online; however, only a single DB2 for z/OS can be specified. The language interface token (LIT) used is SYS1, and the following additional parameters are passed to DB2 for z/OS by IMS in batch regions:

Connection Name
IMS uses the job name for the batch region as the connection name to provide uniqueness within the z/OS system for DB2 for z/OS.

Plan
IMS uses the application program name as the plan name. The resource translation table (RTT) can be used to translate the program name to the desired plan name if the plan has another name.

Program Name
The application program name can be specified in one of the following ways:

- DSNMTV01 can be specified as the application program name for the batch region. When this method is used, control is given to DSNMTV01. When the DB2 for z/OS environment is established, control is passed to the application program.
- The application program name can be specified for the batch region. When this method is used, the SSM= parameter must be specified on the batch region parameters. Control is given to DSNMTV01 directly to establish the ESS environment. When the DB2 for z/OS environment is established, control is passed to the application program specified in the batch region.

The subsystem member allows a batch region to use DB2 for z/OS without specifying the DB2 for z/OS parameters and application program in the batch region JCL. However, the DDITV02 DD statement can still be used. In fact, the SSM= parameter is overridden if both the DDITV02 DD statement is used and DSNMTV01 is the batch application program name.

The DDOTV02 DD statement must be used for messages and diagnostic information regardless of how the DB2 for z/OS connection parameters are specified.

Preparing DB2 for z/OS for use with IMS

You must perform several tasks to prepare DB2 for z/OS for use with IMS.

Providing appropriate tables in your ESS

It is the responsibility of the DB2 for z/OS user to provide the external subsystem module table, and optionally, the resource translation table in the DB2 for z/OS subsystem.

It is also the user's responsibility to make the names of these tables known to the person who is responsible for defining the IMS PROCLIB data set member entries. The person defining the IMS PROCLIB data set member entries includes the names of these tables as values for the parameters ESMT and RTT, respectively.

Placing DB2 for z/OS modules and tables in the appropriate library

IMS loads the modules and tables specified in the external subsystem module table from the DB2 for z/OS libraries when the SSM EXEC parameter is specified in an IMS region. To ensure that this process is successful, you must make the DB2 for z/OS libraries, which must be APF-authorized, available to IMS.

For this purpose, you can either add the DB2 for z/OS libraries to the JOBLIB/STEPLIB/LINKLIST concatenation or create a DFSESL DD statement for this purpose. IMS does not automatically generate a DFSESL DD statement.

If the JOBLIB/STEPLIB/LINKLIST concatenation is not authorized, you must use the DFSESL DD statement. For online IMS regions, the subsystem library or libraries must be concatenated after the library containing the IMS modules (typically IMS.SDFSRESL). When multiple subsystems are connected, additional subsystem data sets can be concatenated.

Note: The DFSESL DD statement is used by IMS to load specific modules or tables and is not part of a general library search.

Example:

```
//DFSESL DD  DISP=SHR,DSN=IMS.SDFSRESL
//       DD  DISP=SHR,DSN=DSNxxx.DSNLOAD
//       DD  DISP=SHR,DSN=DSNyyy.DSNLOAD
```

IMS.SDFSRESL is first, followed by the subsystem libraries.

For IMS batch regions, the DFSESL DD statement is not used. The DB2 for z/OS libraries must be in both the JOBLIB/STEPLIB/LINKLIST concatenation and the DFSRESLB DD statement for authorized modules.

If you use the COBOL II dynamic option, the DB2 for z/OS libraries must be added to the JOBLIB or STEPLIB regardless of authorization. The DD statement that specifies IMS.SDFSRESL must precede the reference to DSNxxx.DSNLOAD in the JOBLIB or STEPLIB. If the JOBLIB or STEPLIB is not authorized, you must also use a DFSESL DD statement.

Configuring JMP and JBP regions for DB2 for z/OS data access

JMP and JBP applications can access DB2 for z/OS databases. Before your JMP and JBP applications can access DB2 for z/OS databases, you must configure the JMP and JBP region and use the DB2 Resource Recovery Services attachment facility (RRSAF) to attach DB2 for z/OS to IMS.

This topic describes how to set up a JMP or JBP region to access DB2 for z/OS databases. It does not describe how to set up DB2 for z/OS for access from IMS. For information about setting up DB2 for z/OS for Java application access, see *DB2 for z/OS Application Programming Guide and Reference for Java*. You must create a DB2 for z/OS plan for each PSB (typically each Java application) that is used to access DB2 for z/OS.

For JMP or JBP applications to have DB2 for z/OS access, you must attach DB2 for z/OS to IMS using the DB2 Recoverable Resource Manager Services attachment facility (RRSAF). Unlike other dependent regions, JMP and JBP regions do not use the External Subsystem Attach Facility (ESAF).

DB2 for z/OS provides different JDBC drivers:

- JDBC/SQLJ driver for DB2 for z/OS with JDBC 2.0 support (called the DB2 for z/OS JDBC/SQLJ 2.0 driver), which allows access to DB2 for z/OS databases only when IMS is on the same z/OS image as DB2 for z/OS. This is a type 2 JDBC driver.
- JDBC/SQLJ driver for DB2 for z/OS with JDBC 1.2 support (called the DB2 for z/OS JDBC/SQLJ 1.2 driver), which allows access to DB2 for z/OS databases only when IMS is on the same z/OS image as DB2 for z/OS. This is a type 2 JDBC driver.
- DB2 for z/OS Universal JDBC driver, which allows access to DB2 for z/OS databases from IMS systems that are on different z/OS images from DB2 for z/OS when you use the IMS Universal drivers type 4 connectivity. You can also use the type 2 implementation of this driver for access to DB2 for z/OS databases when IMS is on the same z/OS image as DB2 for z/OS.

All these drivers are referred to in this topic as DB2 for z/OS JDBC drivers.

Identifying DB2 for z/OS JDBC drivers

For type 2 JDBC drivers, you must use the default connection URL in the application program. For example, `jdbc:db2os390:` or `db2:default:connection`.

For type 4 JDBC drivers, you can use a specific connection URL in the application program.

With RRSAF, the dependent region builds an attachment thread to DB2 for z/OS using z/OS Resource Recovery Services (RRS). RRS coordinates the commits of the updates that the application program makes to both IMS and DB2 for z/OS resources. IMS is a participant, not the coordinator, of these updates and commits.

Attaching a DB2 for z/OS subsystem to IMS using RRSAF

To attach a DB2 for z/OS subsystem to IMS using RRSAF so that IMS JMP and JBP regions can access DB2 for z/OS databases, follow this procedure.

1. Create an IMS PROCLIB data set member for information about the DB2 for z/OS subsystem. The member name must follow the same naming conventions you follow when you attach DB2 for z/OS with ESAF.
2. In the IMS PROCLIB data set member, define the following three parameters for the DB2 for z/OS subsystem that JMP and JBP applications need access to:

   `SST=DB2,SSN=`*`db2name`*`,COORD=RRS`

   You can provide two different definitions for the same DB2 for z/OS system
   - An ESAF definition for non-Java regions
   - An RRSAF definition for Java regions
3. In the class path of the DFSJVMMS member, which is provided by IMS in a sample library, add the following paths:
   - Path to the .zip file of the DB2 for z/OS JDBC driver
   - Path to the .zip file and .zip file name of the DB2 for z/OS JDBC driver

   For example:

   ```
   -Djava.class.path=>
   /usr/lpp/db2/db2710/classes: >
   /usr/lpp/db2/db2710/classes/db2j2classes.zip
   ```
4. In the DFSJVMEV member, which is provided by IMS in a sample library, add the path to the SO file of the DB2 for z/OS JDBC driver to the LIBPATH= environment variable. For example:

   ```
   LIBPATH=/usr/lpp/db2/db2710/lib
   ```
5. Add the following parameters to the IMS control region EXEC statement:

   `SSM=`*`name`*`RRS=Y`
6. In the DFSJMP or DFSJBP procedure of the region that has access to DB2 for z/OS, add the DFSDB2AF DD statement to point to the DB2 for z/OS libraries, which must be APF-authorized.

Related Reading: For details about the IMS PROCLIB data set member and procedure parameters, see Chapter 17, “Members of the IMS PROCLIB data set,” on page 587. For information about the DB2 for z/OS JDBC drivers, see *DB2 for z/OS Application Programming Guide and Reference for Java*.

Chapter 11. Installing the Transport Manager Subsystem

An example of JCL statements used to install the Transport Manager Subsystem (TMS) are provided in this topic.

To install the TMS, place the initial command stream into a data set referenced by the TMS's SYSIN DD statement. This allows you to have the component start be an automated process. The JCL procedure must be tailored to your local conventions and placed in a library accessible by the z/OS START command. "Sample Transport Manager job control statements" provides an example of job control statements for the TMS.

Related reading: For a description of the IMSPLEX parameter, see "Parameter descriptions for IMS procedures" on page 448.

Sample Transport Manager job control statements

```
  //TMS     PROC SYS2=,CMD=TMSTART
//        EXEC PGM=DFSMVCR0,PARM=(ELX[,'IMSPLEX=ccccc'])
//STEPLIB DD   DISP=SHR,DSN=IMS.&SYS2.SDFSRESL
//SYSIN   DD   DISP=SHR,DSN=IMS.&SYS2.PROCLIB(&CMD)
//SYSOUT  DD   SYSOUT=A
//SYSTSPRT DD  SYSOUT=A
//SYSABEND DD  SYSOUT=A
//*
```

Defining the Transport Manager Subsystem to IMS

You can define Transport Manager Subsystem (TMS) settings when you start the TMS or after it is running.

The TMS requires no offline definition. All TMS definition can be performed when you start the TMS, or while the TMS is running, using TMS commands. However, some VTAM definition is required before you can use the Transport Manager subsystem.

You cannot name the TMS the same name as the TM start-up member in SYS1.PROCLIB.

Defining the Transport Manager Subsystem to VTAM

In order for the Transport Manager Subsystem (TMS) to use VTAM, certain VTAM definition work is required. A sample VTAM APPL definition is provided.

In order for the TMS to use VTAM, certain VTAM definition work is required. Because the VTAM publications describe VTAM definition specifics, the example provided in "Sample VTAM APPL definition" on page 233 shows the relationships between products and indicates which parts of the VTAM definition are important.

Related reading: See *IMS Version 11 Communications and Connections* for related information.

APPL
: In SYS1.VTAMLST, a group of application program minor nodes must be defined for use by TMS and IMS systems. An example is shown in "Sample

© Copyright IBM Corp. 1974, 2015

VTAM APPL definition" on page 233. This definition is performed with the APPL definition statement. The "name" on the APPL statement corresponds to the TMS SET command APPLID (VTAM application id) name. Each APPL statement must have three numeric characters (beginning with 001) appended to the one-to-five character name specified in the SET APPLID command. The number of APPL names defined must be at least one greater than the maximum number of IMS subsystems and ILS instances expected to execute concurrently on this CPC.

Other key specifications for APPL are:

- AUTH=VPACE to allow session pacing.
- MODETAB must specify the member name of a logon mode table in SYS1.VTAMLIB that contains entries appropriate for the TMS.
- EAS specifies the number of VTAM sessions that each ACB can have. It is defined as follows:

```
2 x (maximum number of concurrent active IMS subsystems +
          maximum specification of the MAXCONV value) + 2
```

  A value of 64 should be more than enough.
- ENCR=NONE should be specified unless you want VTAM encryption.
- MAXPVT=1024 allows VTAM to use 1 MB of TMS or IMS private area storage.
- PARSESS=YES is required and allows parallel sessions.
- PRTCT correlates with "vtam-acb-password" on the TMS SET PASSWORD command.
- SRBEXIT=YES must be specified.
- VPACING=n (start with 10) to limit VTAM buffer usage while allowing reasonable bandwidth.
- VTAMFRR=NO must be specified.

MODEENT

Defines an entry in a mode table when combined with MODETAB and MODEEND macros, assembled and link edited into SYS1.VTAMLIB. An example is shown in "Sample VTAM APPL definition" on page 233. For the TMS, the primary value is to obtain a class of service for a conversation, thus choosing a network route.

- LOGMODE specifies the logon mode table entry name, correlating to DLOGMOD on the VTAM APPL definition or to MODENAME specified by a TMS component.
- COS specifies the name of an entry in the class of service table.
- SRCVPAC should be zero so that APPL VPACING= controls pacing on the PLU-SLU sessions.
- SSNDPAC should be non-zero so that APPL VPACING= controls pacing on the SLU-PLU sessions.
- TMDEFLT must be defined as the default logon mode table entry name.

COS

Defines an entry in a class of service table when combined with COSTAB and COSEND macros, assembled and link edited into SYS1.VTAMLIB. The recommended transmission priority to specify with virtual routes is 2.

If you have enough hardware connectivity between sites, use multiple MODEENT entries and more than one class of service to select different virtual routes. This spreads different TMS conversations among the hardware.

COS could be used to avoid satellite links (because of latency), particularly for LOGGER conversations.

IOBUF
Defines the size and number of VTAM I/O buffers. Both the size and number have a significant impact on processor consumption and effective bandwidth. 1 KB buffers provide much better performance than small buffers, but requires more storage than small buffers where network traffic consists of many small messages. The average RU size for logger to log router conversations is close to the OLDS blocksize (typically 22 KB).

"Sample VTAM MODETBL definition" provides a sample VTAM mode table definition for the TMS.

Sample VTAM APPL definition

```
*********************************************************************
* SYS1.VTAMLST APPL definition for Transport Manager                 *
* Corresponds with a Transport Manager command of:                   *
*   SET APPLID(TMP2) APPLCOUNT(9)                                    *
*                                                                    *
*********************************************************************
         VBUILD TYPE=APPL
TMP2001  APPL  EAS=64,PARSESS=YES,SRBEXIT=YES,                       C
               APPC=YES,AUTOSES=1,DMINWNL=32,DMINWNR=32,DSESLIM=64,  C
               MAXPVT=1024K,VPACING=10,VTAMFRR=NO,                   C
               AUTH=(VPACE),MODETAB=MODELUTM,DLOGMOD=TMDEFLT
TMP2002  APPL  EAS=64,PARSESS=YES,SRBEXIT=YES,                       C
               APPC=YES,AUTOSES=1,DMINWNL=32,DMINWNR=32,DSESLIM=64,  C
               MAXPVT=1024K,VPACING=10,VTAMFRR=NO,                   C
               AUTH=(VPACE),MODETAB=MODELUTM,DLOGMOD=TMDEFLT
         .
         .
         .
TMP2009  APPL  EAS=64,PARSESS=YES,SRBEXIT=YES,                       C
               APPC=YES,AUTOSES=1,DMINWNL=32,DMINWNR=32,DSESLIM=64,  C
               MAXPVT=1024K,VPACING=10,VTAMFRR=NO,                   C
               AUTH=(VPACE),MODETAB=MODELUTM,DLOGMOD=TMDEFLT
```

Sample VTAM MODETBL definition

```
*********************************************************************
*  MODULE NAME:  MODELUTM                                           *
*                                                                   *
*  DESCRIPTIVE NAME:                                                *
*    LOGON MODE TABLE FOR Remote Site Recovery(RSR) TRANSPORT MANAGER *
*                                                                   *
*  FUNCTION:  THIS TABLE IS USED BY VTAM TO DETERMINE THE BIND      *
*     PARAMETERS FOR A SESSION.                                     *
*                                                                   *
*  NOTES:                                                           *
*     SET PACING TO 10 IN DEFAULT ENTRY.                            *
*     SET SSNDPAC NON-0, SRCVPAC 0,                                 *
*       THIS CAUSES APPL VPACING TO BE USED.                        *
*                                                                   *
*********************************************************************
MODELUTM  MODETAB
*********************************************************************
*  DEFAULT - SEND PACING OF ¬0 FORCES USE OF APPL VPACING           *
*********************************************************************
         MODEENT LOGMODE=TMDEFLT,FMPROF=X'13',TSPROF=X'07',           X
               SSNDPAC=X'3F',SRCVPAC=X'00', TUNE ADAPT SESSION PACING  X
               RUSIZES=X'8D8D',  8 * 2**D = 8*8192=65536               X
               PSNDPAC=X'00',               TUNE ADAPT SESSION PACING  X
               TYPE=0,                      NEGOTIABLE BIND            X
               PSERVIC=X'060200000000000000000300',                    X
```

```
|                                PRIPROT=X'B0',SECPROT=X'B0',COMPROT=X'D0B1'
|                         TITLE 'SNASVCMG'                                    *@R495812*
|                       ********************************************************************
|                       *        LOGMODE TABLE ENTRY FOR SPECIAL VTAM APPC SESSIONS THAT     *
|                       *        SUPPORT CNOS.  TAKEN FROM SYS1.SAMPLIB(ISTINCLM)            *
|                       *                                                     @R495812*
|                       ********************************************************************
|                       SNASVCMG MODEENT LOGMODE=SNASVCMG,FMPROF=X'13',TSPROF=X'07',           X
|                                PRIPROT=X'B0',SECPROT=X'B0',COMPROT=X'D0B1',              X
|                                RUSIZES=X'8585',ENCR=B'0000',                             X
|                                PSERVIC=X'060200000000000000000300'          *@R495812*
|                         MODEEND
|                         END
```

Chapter 12. Setting up IMS for diagnostics

IMS can process large amounts of work efficiently; however, IMS can experience problems that need to be diagnosed and corrected. For these types of problems, IMS displays symptoms that can help you with your diagnosis, but, in order to obtain that information, you need to set up your system correctly.

Setting the size of the z/OS trace tables

When you set up your IMS system for diagnostics, you need to consider your settings for z/OS trace tables as well.

z/OS system trace table

The z/OS system trace table is valuable for a large variety of problem types. The system trace table is page-fixed storage, and the default size is only 64 KB. You must ensure that there are enough real page frames for this specification.

Set z/OS system trace table size to 999 KB by issuing the z/OS command `TRACE ST,999K` in the z/OS COMMNDxx member of the SYS1.PARMLIB data set.

See *z/OS MVS System Commands* for more information.

z/OS master trace table

The master trace table maintains the most recently issued operator messages. It provides a view of external events at the time of failure. Ensure that the master trace table is large enough to span most error time frames. The default size is only 24 KB, which allows approximately 336 messages. A 500 KB specification allows approximately 7000 messages.

The master trace uses Subpool 229 Key 0 High Private Pageable Storage of the master scheduler address space.

Set the z/OS master trace table size to 500 KB by specifying the command `TRACE MT,500K` in the SCHEDxx member of the SYS1.PARMLIB data set.

See *z/OS MVS Diagnosis: Tools and Service Aids*, *z/OS MVS Initialization and Tuning Reference*, and *z/OS MVS System Commands* for complete details.

Common Storage Tracker

When you set up your IMS system for diagnostics, you should also start the z/OS common storage tracking function.

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:

© Copyright IBM Corp. 1974, 2015

```
VSM TRACK CSA(ON)
```

See *z/OS MVS Diagnosis: Tools and Service Aids*, *z/OS MVS Initialization and Tuning Reference*, and *z/OS MVS System Commands* 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

When you set up your IMS system for diagnostics, you should ensure that the space defined for the internal supervisor call (SVC) dump is adequate, and if not, use CHNGDUMP MAXSPACE to modify it.

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 or larger is standard for large multi-address space SVC Dumps.
    - One way to estimate the amount of storage needed for the SVC dump is to calculate the total amount of global storage in use by adding the amount of storage used by all the regions (at peak usage) as viewed by online monitor programs.

  See *z/OS MVS Diagnosis: Tools and Service Aids*, *z/OS MVS Initialization and Tuning Reference*, and *z/OS MVS System Commands* 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.

Automatic dump data set allocation

When you set up your IMS system for diagnostics, ensure that the automatic dump data set has been allocated by using the COMMNDxx member of the SYS1.PARMLIB data set.

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 *z/OS MVS Diagnosis: Tools and Service Aids* and *z/OS MVS System Commands* 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.

Because system dumps for IMS Version 11 and later generally contain more address spaces than do dumps for previous IMS versions, you might need to increase the amount of virtual storage that an SVC dump can use to capture volatile virtual storage data, summary dump data, and component-specific data before writing the dump to DASD.

Diagnostic setup recommendations for IMS

When you set up your IMS system for diagnostics, consider not only the IMS diagnostic settings, but also the z/OS systems settings that impact IMS diagnostics.

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 for z/OS 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.

Note: Make sure that you use FMTO=D so that you get a system dump if IMS abends. Do not rely on SYSMDUMP, SYSABEND, or SYSUDUMP as your primary source of diagnostic information. These dumps only dump a single address space, and sometimes they do not provide adequate storage for complete diagnosis of abends.

SYSMDUMP DD

- Specify the SYSMDUMP 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 SYSMDUMP 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, DL/I, and Lock traces on. Perform one of the following:

  - The DL/I 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, SCHD=ON
    ```

  - Use the IMS /TRA SET ON TABLE *nnnn* command, where *nnnn* is alternately = DISP, SCHD, 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 LATC=ON for the LATCH trace in the DFSVSM*xx* member of the IMS.PROCLIB data set.

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 IMS.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 241.

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 increases.

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.

CQS trace setup recommendations

There are specific recommendations for optimally setting up your CQS (Common Queue Server) system.

Trace environment - conservative

You can specify the CQS execution parameter BPECFG=*nnnnnnnn* in a way that is optimal for conservative trace environments.

Specify the following trace entries within the BPE configuration parameter member of the IMS PROCLIB data set:

```
--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 *
```

Related reference:

"BPE configuration parameter member of the IMS PROCLIB data set" on page 587

Trace environment - more aggressive

You can specify the CQS execution parameter BPECFG=*nnnnnnnn* in a way that is optimal for more aggressive trace environments.

Specify the following trace entries within the BPE configuration parameter member of the IMS PROCLIB data set:

```
--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*/
```

Related reference:

"BPE configuration parameter member of the IMS PROCLIB data set" on page 587

Installing the IMS Dump Formatter

You can install the IMS Dump Formatter by updating several DD concatenations.

Prerequisite: Before you begin installing the IMS Dump Formatter, make sure that you have IPCS already functioning with ISPF/PDF.

Perform the following procedure:

1. Install the IMS Dump Formatter by updating the DD concatenations listed in Table 42:

Table 42. DD concatenations to update

DDNAME	Data Set to Be Added	Contents of Data Set
SYSPROC	IMS.SDFSCLST	CLISTs
	IMS.SDFSEXEC	REXX EXEC
ISPMLIB	IMS.SDFSMLIB	Messages
ISPPLIB	IMS.SDFSPLIB	Panels
ISPTLIB	IMS.SDFSTLIB	Tables
IPCSPARM	IMS.SDFSMAC	All IMS macros **Note:** DFSIPCSP is the only member that is used from this data set.
TASKLIB	IMS.SDFSRESL	Formatting modules **Note:** The TASKLIB concatenation is specified as part of the IPCS command invocation.
SYS1.PARMLIB (for IPCSPR00 member)	MACLIB DD and IMS.SDFSMAC	

2. Add the Offline Dump Formatting module name to the Print Dump Exit Control Table in the SYS1.PARMLIB member BLSCECTX.

   The IMS Offline Dump Formatting module should already be defined to z/OS as part of z/OS preconditioning for IMS. If the definition is missing, update the BLSCECTX member of SYS1.PARMLIB with the following:

   ```
   EXIT EP(DFSOFMD0) VERB(IMSDUMP) ABSTRACT('IMS analysis')
   ```

For more information about BLSCECT (the Formatting exit routines for dump and trace analysis for IPCS), see *z/OS MVS Initialization and Tuning Reference*.

The IMS IVP job IV_D202T also provides an example.

Making IMS.SDFSRESL available

After updating the DD concatenation for IMS.SDFSRESL, you need to make it available.

To make IMS.SDFSRESL available, use one of the following methods:

- Add the IMS.SDFSRESL data set to the ISPF ISPLLIB
- Add the IMS.SDFSRESL data set to STEPLIB in the TSO logon procedure
- Use TSOLIB from TSO READY to establish IMS.SDFSRESL in the search list:

  ```
  TSOLIB ACTIVATE DATASET('IMS.SDFSRESL')
  ```

If you do not want to add the entire IMS.SDFSRESL data set to the concatenation, you can concatenate a data set that contains both DFSABND0 and DFSOFMD0

load modules from the highest IMS version level with the alias modules from the IMS.SDFSRESL for lower-level IMS versions that you want to format.

The following example shows how to concatenate DFSABND0 and DFSOFMD0 from IMS Version 11 and provide format support for IMS Version 10:

```
DFSABND0 DFSAB111 DFSAB101
DFSOFMD0
DFSOF111
DFSOF101
```

For more information, see *z/OS TSO/E Command Reference*.

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 the DFSVSMxx member of the IMS PROCLIB data set.
- 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 "DFSVSMxx member of the IMS PROCLIB data set" on page 738 for more information.

Controlling the volume of traces

Control the volume of the traces using the VOLUME parameter on the /TRACE TABLE command.

The parameter can be set to high, medium, low, or error, where high generates the largest volume of trace entries, and error generates the smallest volume of trace entries.

For details about the /TRACE command parameters, see *IMS Version 11 Commands, Volume 2: IMS Commands N-V*. For details about the OPTIONS statement in the DFSVSAMP or DFSVSMxx data set, see "DFSVSMxx member of the IMS PROCLIB data set" on page 738.

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 helpful to you in diagnosing various problems that might occur in your environment.

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

Activating 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 "IMSFP procedure" on page 578.
- 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.

Writing 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.

Related Reading: For more information about diagnosing BPE component-related problems, see *IMS Version 11 Diagnosis*.

Creating output data sets with correct attributes

When you specify particular attributes, you can use the DFSTRA01, DFSTRA02, and DFSTRA0T data sets to hold your trace data.

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

DSORG
PS (Physical Sequential)

RECFM
VB

LRECL
4016

BLKSIZE
A formula of: (LRECL*N)+4. The block size must be a multiple of the LRECL (4016), with the additional 4 bytes for the block descriptor word.

Recommendation: Use a BLKSIZE of 20,084, which is five logical records in length (4016 bytes, multiplied by 5), plus the block descriptor word (4 bytes). The BLKSIZE of 20,084 is recommended for current DASD because it is equal to one-half track.

Recommendation: Allocate these data sets as a single extent, meaning contiguous tracks. Do not specify secondary allocation.

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

DSORG
PS (Physical Sequential)

RECFM
VB

LRECL
4016

BLKSIZE
A formula of: (LRECL*N)+4. The block size must be a multiple of the LRECL (4016), 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 tracing for BPE-managed address spaces

The member of the IMS PROCLIB data set that you specify using the BPECFG= parameter in the execution parameters for the BPE-managed address spaces defines configuration parameters to the Base Primitive Environment (BPE), including trace levels.

Perform the following procedure to set up tracing for BPE-managed address spaces:

1. To set the trace level for BPE-managed trace tables in the CQS, IMS Connect, ODBM, OM, RM, SCI, and BPE-based DBRC address spaces, use the TRCLEV= statement in the BPE configuration parameter member of the IMS PROCLIB data set. The TRCLEV= parameter is used in the BPE configuration member of the IMS PROCLIB data set 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 (common queue server), ODBM (Open Database Manager), OM (Operations Manager), RM (Resource Manager), SCI (Structured Call Interface), IMS Connect, and BPE-based Database Recovery Control (DBRC) support.

   Trace tables are written to internal memory in the BPE address space unless you enable BPE external tracing for the trace table.
2. Enable BPE external tracing (writing trace data to an external data set as well as to memory). You must complete both of the following tasks:
   a. Define, or modify, the BPE configuration parameter member (BPECFG) of the IMS PROCLIB data set to include the external trace parameters:
      1) Add the EXTTRACE statement to the BPE configuration parameter member (BPECFG) of the IMS PROCLIB data set. This statement defines the external trace data set to IMS.
      2) **Optional:** update the TRCLEV statement for each trace table that is to be externalized by adding the EXTERNAL=YES keyword in the BPE configuration parameter member (BPECFG) of the IMS PROCLIB data set. This keyword causes the specified tables to be written to the external trace data set from the beginning of the address space. The default is EXTERNAL=NO. BPE trace tables can also be set to be written to the external trace data sets dynamically, by issuing the BPE UPDATE TRACETABLE command with the EXTERNAL(YES) parameter after the address space has started.
   b. Define a generation data group (GDG) for the external trace data set. Use the IDCAMS DEFINE GENERATIONDATAGROUP command to define the GDG. The value that you specify for the NAME keyword in this command must match the data set name in the GDGDEF parameter of the EXTTRACE statement of the BPE configuration parameter member (BPECFG) of the IMS PROCLIB data set. If the data control block (DCB) characteristics of a GDG data set are not defined anywhere on the system, you must also define a prototype data set with the appropriate DCB characteristics on the volume containing the system catalog. Here is an example of how that JCL would look:

```
//PROTOTYP JOB ...
//STEP1 EXEC PGM=IEFBR14
//BLDDSCB DD DSN=BPEEXTRC.GDG01,
// DISP=(NEW,KEEP),
// UNIT=SYSDA,
// VOL=SER=PAGE01, <=System catalog volume
// SPACE=(TRK,(0)),
// DCB=(DSORG=PS,RECFM=VB,BLKSIZE=24580,LRECL=24576)
```

You can issue the IDCAMS DEFINE GENERATIONDATAGROUP command from an IDCAMS job, a TSO session, or a TSO command batch job, as shown in the following three examples, respectively.

```
//DEFGDG   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
   DEFINE GENERATIONDATAGROUP -
      (NAME(BPEEXTRC.GDG01) -
       NOEMPTY -
       SCRATCH -
       LIMIT(255))
```

```
DEFINE GENERATIONDATAGROUP (NAME(BPEEXTRC.GDG01) NOEMPTY SCRATCH LIMIT(255))
```

```
//DEFGDG     JOB  ...
//STEP1      EXEC PGM=IKJEFT01
//SYSTSPRINT DD   SYSOUT=A
//SYSTSIN    DD   *
DEFINE GENERATIONDATAGROUP (NAME(ICFUCAT1.GDG01) NOEMPTY SCRATCH LIMIT(255))
```

For more information about the DEFINE GENERATIONDATAGROUP command, see *z/OS DFSMS Access Method Services for Catalogs*.

Setting up the IMS abend search and notification function

Use the IMS abend search and notification function to notify personnel of abnormal termination (abend) and to research and retrieve links to information about abends.

The abend search and notification function enables IMS to send an email or text message to a predesignated recipient in order to:

- Notify the correct people about an abnormal termination (abend)
- Provide additional information about the abend

The abend search and notification function can also be used to research and retrieve links to information about abends which include IMS product documentation, technical notes in an IBM technical support database and information in the preventive service planning (PSP) database.

To set up and enable the IMS abend search and notification function:

1. Allocate and authorize the following two data sets:
   - A runtime data set that contains the information from the choices made during the setup process when working with the IMS abend search and notification system setup panel. The name of this runtime data set is included in all the JCL jobs that are generated by the IMS abend search and notification function. The DCB for runtime data set is RECFM=FB,LRECL=80.

     Users of the IMS abend search and notification system setup panel need update access to the runtime data set. All other users of the IMS abend search and notification function need read-only access. Examples of such users are:

- Started tasks that initiate abend-event driven invocation of the function
- Users of the IMS abend search and notification on-demand panels
- The user ID associated with the IMS control region

- A skeleton data set that contains the name of the IMS abend search and notification procedure. Access authorizations to the skeleton data set are the same as the authorizations needed for the runtime data set. The DCB for skeleton data set is RECFM=FB,LRECL=80.
- A skeleton data set that contains the following members:
  - - DFSSPRCI, which specifies the name of the IMS abend search and notification procedure.
  - - DFSSPCLI, which specifies a JCLLIB statement that points to where the procedure resides.

  The DCB for the skeleton data set is RECFM=FB,LRECL=80.

  The skeleton data set must be concatenated to the ISPSLIB. Some system environment configurations might prevent the concatenation. In that case, name the skeleton data set *hlq*.IASNSLIB, where *hlq* is the high-level qualifier that is used to install IMS. After it is named, the data set is automatically concatenated to ISPSLIB for all users of the IMS abend search and notification panel.

  The installer of IMS abend search and notification must therefore have RACF ALTER authorization for the high-level qualifier that is used to install IMS. ALTER allows the installer to read, write, create, or delete the data set.

2. Access and fill out the IMS abend search and notification system setup ISPF panel to populate the runtime data set and skeleton data set. The IMS abend search and notification system setup panel can be accessed by either of the following methods:
   - Select **IMS abend search and notification** from the IMS Application Menu.
   - From the ISPF option **6**, enter the following command, where *hlq* is the high-level qualifier used to install IMS:

     ```
     exec 'hlq.SDFSEXEC(DFSRASN0)' 'HLQ(hlq)'
     ```

   The IMS Abend Search and Notification panel is displayed, as shown in the following example figure.

```
                  IMS Abend Search and Notification      IMS Version 11.1
Command ===>


                                                       TIME....10:58:45
                                                       DATE....2010/03/10
                                                       USERID..userid
Select one of the following tasks and press ENTER .




     Tasks . . _  1. IMS ASN System Setup
                  2. IMS ASN On-Demand Interface



To Exit this menu, press the END key.
For Help information, place cursor on any field and press PF1 .
```

Figure 13. IMS Abend Search and Notification panel

Type 1 in the field **IMS ASN System Setup** and press Enter. Here is an example of the IMS abend search and notification system setup panel:

```
IMS abend search and notification - system setup
  COMMAND ===>

   Identify IMS ASN skeleton lib data set (must be concatenated to ISPSLIB):
  *Skeleton lib DS ____________________________________________________________

   Identify the following data sets and members (must be accessible at runtime):
  *Runtime  DS ____________________________________________________________
  *PROCLIB  Mbr ________  Restore? _     *URLS     Mbr ________  Restore? _
  *SYSUT1   Mbr ________  Restore? _     *CONTROL  Mbr ________  Restore? _
  *SMS      Mbr ________  Restore? _     *SMSCNTL  Mbr ________  Restore? _

   Identify the fully qualified name of your installation's IMS.SDFSRESL:
  *IMS.SDFSRESL DSN ___________________________________________________________

   E-mail and Recipient Information
  *Recipient e-mail address. . . ______________________________________________
   Specify additional addresses? _               (Y-Yes/N-No)
   SMS recipient address . . . . ______________________________________________
   Specify additional SMS rcpts? _               (Y-Yes/N-No)
   External Writer ID for Local SMTP ________
   FROM e-mail ________________________________________________________________
   SMTP server ____________________________________________________ Port _____
```

Use the IMS abend search and notification system setup panel to perform the following tasks:

- Name and customize the IMS abend search and notification procedure JCL (the sample shipped in IMS.SDFSSLIB as DFSIASN0)
- Name and customize one or more SYSUT1 PDS members with the email message text. You can either use the IBM-supplied SYSUT1 PDS member as is, or customize it.
- Name and customize one or more CONTROL PDS members with email routing information. You can either use the IBM-supplied CONTROL PDS member as is, or customize it.
- Name and customize one or more SMS PDS members with the text for the text messages. You can either use the IBM-supplied SMS PDS member as is, or customize it.
- Name and customize one or more SMSCNTL PDS members with the text-message routing information. You can either use the IBM-supplied SMSCNTL PDS member as is, or customize it.
- Name and customize one or more URLS PDS members with the URL information to be mapped into the email and text messages. You can either use the IBM-supplied URLS PDS member as is, or customize it.

The IMS Abend Search and Notification System Setup panel has general online help as well as field-sensitive online help. To access the online help, place the cursor on the command line (for general help) or in an input field (for field-sensitive help) and press F1.

For detailed information about the DFSIASN0 procedure and above PDS members, see "DFSIASN0 procedure" on page 541.

3. Specify IASNPROC= *member_name* in the diagnostics section of the DFSDFxxx member of the IMS PROCLIB data set, where *member_name* is the eight-character name of the IMS abend search and notification procedure. This task can be performed using the Syntax Checker or by manually inserting the output of the IMS abend search and notification system setup panel (which resides in the DFSMSxxx member of the runtime data set) into the DFSDFxxx

member of the IMS PROCLIB data set. For information about the DFSDFxxx member of the IMS PROCLIB data set, see "DFSDFxxx member of the IMS PROCLIB data set" on page 672.

4. Copy the IMS abend search and notification procedure, which was created and customized through the use of the system setup panel, to a concatenated z/OS procedure library.
5. Provide the name of the skeleton data set to the users of the IMS abend search and notification on-demand interface by one of the following methods:
   - Concatenate the data set name in the ISPSLIB DD statement in the logon procedures used by the TSO users
   - Concatenate the data set name directly by statements in the default CLIST of those TSO users
   - Name the skeleton data set using the format *hlq*.IASNSLIB, where *hlq* is the high-level qualifier that is used to install IMS. After it is named, the data set is automatically concatenated to ISPSLIB for all users of the IMS abend search and notification panel. If you perform setup a second time and use a data set name other than *hlq*.IASNSLIB for the skeleton data set, and *hlq*.IASNSLIB still exists, ISPF looks for the skeleton library members in *hlq*.IASNSLIB before looking in the other data set.
   - Provide the TSO users a program that performs the concatenation

To disable the IMS abend search and notification function, set *member_name* to blanks or a null value in the IASNPROC= parameter in the diagnostics section of the DFSDFxxx member of the IMS PROCLIB data set.

Customizing the IMS abend search and notification email

You can customize the email notification that is sent, adding a specific URL link, by modifying both the URLS and the SYSUT1 members.

You can access and modify the members by doing one of the following:

- Running the setup panel, which displays the customizable members in sequence for editing.
- Directly accessing the members in ISPF in the Runtime partitioned data set using the names specified in the setup panel.

To create a link to be inserted into the SYSUT1 member, modify the URLS member:

1. Access the URLS member. The URLS member is the second member in sequence from setup panel.
2. Insert the link using the following format:

```
*This is a new link
#NEWURL,#PARM=(value1,value2)
http://www.new.com/
```

   A sample is shown in the figure below.

   The first line, which is preceded with an asterisk (*), is a comment line. Following the comment line is the name of the link, preceded with #. The name can be up to eight alphanumeric characters, including the #. Following the link name is the URL link to be associated with the name.

   Optionally, you can specify a conditional statement, by specifying #PARM=(value1,value2,...). Use the following syntax:

   - Specify a comma after the name to separate the name and the conditional statement.

- Following the comma, specify a parameter that exists in your particular customization, preceded with #.
- Following the parameter, specify "=" or "!=" depending on if you want to check for equality or inequality.
- Following the "=" sign or "!=", specify one or a set of values to be checked for the specified parameter, enclosed within parentheses.

  A value cannot exceed 20 characters.

  If you have multiple URLs with the same name, the foremost URL that satisfies the conditional statement is mapped into the formatted email.

  For equality, if any of the value matches a part of the parameter's value, the conditional statement is satisfied. For inequality, if none of the value matches a part of the parameter's value, the conditional statement is satisfied.

This creates a link that can be inserted into the SYSUT1 member to become part of the IMS ASN email

This link replaces any occurrence of the link name that exists in the SYSUT1 member, if no PARM conditions are specified or if the PARM conditions are satisfied. An example of how to insert the name into the SYSUT1 member is in the figure below.

```
------------------------------------------------------------------------------
   File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
 EDIT       IBMER.IASN.SDFSIASN(DFSIAURL) - 01.01          Columns 00001 00072
 Command ===>                                                Scroll ===> HALF
 000052 http://www-1.ibm.com/support/docview.wss?rs=0&dc=DB540&q1=HMK1010
 000053 &uid=isg1_IMS1010_HMK1010&loc=en_US&cs=utf-8&lang=
 000054
 000055 *Order PTF electronically
 000056 #PTFURL
 000057 https://techsupport.services.ibm.com/server/390.elecptforder
 000058
 000059 *New Link
 000060 #MSGURL,#MSG=(BPE,CQS,CSL,DHB,DFS,DSP,DXR,ELX,HWS,MDA,PCB,PGEN)
 000061 http://www.ibm.com/support/knowledgecenter/SSEPH2/welcome
```

Figure 14. Adding an email link in the IMS Abend Search and Notification panel using the URSL member

```
------------------------------------------------------------------------------
   File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       KINLAU.IASN.SDFSIASN(DFSIAEML) - 01.02          Columns 00001 00072
Command ===>                                                 Scroll ===> HALF
000204       <li>
000205 <a href="
000206 #GENURL
000207 ">General search results for #GEN
000208 </a></li>
000209 %GEN
000210     </ul></font>
000211     </td></tr></table>
000212     </td>
000213   </tr>
000214   <tr>
000215     <td width="375">
000216     <table border="0" cellspacing="0" cellpadding="5"><tr><td>
000217     <font size="-2" face="Verdana, Arial, Helvetica, sans-serif">
000218     <a href="
000219     #PSPURL
000220     ">Search the
000221     Preventative Maintenance Planning (PSP) Database</a>
000222     for specific installation tips, high impact or pervasive problems
000223     and service recommendations.</font><br>
------------------------------------------------------------------------------
```

Figure 15. HTML formatting using the URSL member

To add a link in the email using the SYSUT1 member:

1. Access the SYSUT1 member. The SYSUT1 member is the third member in sequence from the setup panel and follows the URLS member.
2. Scroll down to the text and HTML sections. These two sections are marked by “Content-Type: text/plain” and “Content-Type: text/html”, respectively.
3. Add the following text in the text section.

   ```
   9. This is a new link
   #NEWURL
   ```

4. Add the HTML syntax in the HTML section. You must follow HTML syntax rules in the HTML section.

   ```
   <li>
   <a href="
   #NEWURL
   ">This is a new link.
   </a></li>
   ```

   In the HTML section, the name of the link #NEWURL must be on a line by itself.

This creates a link at the bottom of the list in each IMS ASN email.

```
------------------------------------------------------------------------------
   File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
 EDIT       KINLAU.IASN.SDFSIASN(DFSIAEML) - 01.01          Columns 00001 00072
 Command ===>                                                 Scroll ===> HALF
 000078
 000079 8. General search results for #GEN
 000080 #GENURL
 000081 %GEN
 000082
 000083 9. This is a new link
 000084 #NEWURL
 000085
 000086 ------------------------------------------------------------------------
 000087
 000088 Search the Preventative Maintenance Planning (PSP) Database
 000089 #PSPURL
 000090 for specific installation tips, high impact or pervasive
 000091 problems and service recommendations.
 000092
 000093 Can't find an answer?
 000094 Contact our support team at:
 000095 https://www.ibm.com/support/servicerequest/Home.action
 000096
 000097 Looking for fixes?
------------------------------------------------------------------------------
```

Figure 16. Adding an email link in the IMS Abend Search and Notification panel using the SYSUT1 member

```
------------------------------------------------------------------------------
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       KINLAU.IASN.SDFSIASN(DFSIAEML) - 01.01          Columns 00001 00072
Command ===>                                                 Scroll ===> HALF
000204       <li>
000205 <a href="
000206 #GENURL
000207 ">General search results for #GEN
000208 </a></li>
000209 %GEN
000210       <li>
000211 <a href="
000212 #NEWURL
000213 ">This is a new link.
000214 </a></li>
000215     </ul></font>
000216     </td></tr></table>
000217     </td>
000218   </tr>
000219   <tr>
000220     <td width="375">
000221     <table border="0" cellspacing="0" cellpadding="5"><tr><td>
000222     <font size="-2" face="Verdana, Arial, Helvetica, sans-serif">
000223     <a href="
------------------------------------------------------------------------------
```

Figure 17. HTML formatting using the SYSUT1 member

Chapter 13. IMS system definition examples

The examples of IMS system definitions provided here reflect stage 1 input to the assembler during an IMS system definition.

The IMS system definition macro statements shown in the following example are the only statements necessary for the system definition for a z/OS database system.

```
Characteristics                                          Column
to define        Macro statements prepared                 72


IMS              IMSCTRL  SYSTEM=(VS/2,(BATCH,DB/DC),390)
Configuration    IMSGEN  ASM=HLASM,ASMPRT=ON,                X
                       LKPRT=(XREF,LIST),                    X
                       JOBCTL=(4,C,T),                       X
                       PRTY=6,PROCLIB=YES
```

IMS DB/DC environment

This example of an IMS DB/DC system definition includes definitions for data communication, system configuration, databases, and application programs.

A sample IMS DB/DC environment is the basis for this example of an IMS DB/DC system definition. An overview of the IMS system is followed by system details and macros.

In this example, the system has the following characteristics:

- 46 application programs
- 67 transaction codes against those application programs
- 16 databases

Data communication characteristics

The line groups and terminals that comprise the example IMS DB/DC system definition are provided.

The data communication characteristics of the sample environment include:

- Four local SYSOUT line groups
- Six 3270 local VTAM terminals
- Six 3270 remote VTAM terminals
- Five 3600 (or FINANCE) VTAM terminals
- Three SLU 1 VTAM terminals
- Two NTO terminals
- Two SLU 2 VTAM terminals
- Two SLU P VTAM terminals
- Two LU 6.1 VTAM terminals

System configuration macro statements

The macro statements associated with the example of the IMS DB/DC system definition are provided.

© Copyright IBM Corp. 1974, 2015

The system configuration macros in "Sample system configuration macro statements" select IMS system functions for z/OS, allowing five regions to operate simultaneously. They specify 16 transaction code classes, and the number of concurrently operating subtasks optimized to equal the number of specified communication lines divided by two.

Sample system configuration macro statements

```
Characteristics                                    Column
to define        Macro statements prepared            72

IMS              IMSCTRL SYSTEM=(VS/2,(ALL,DB/DC),390),  X
Configuration          MAXREGN=(5,52K,A,3),              X
                       MCS=13,DESC=2,MAXCLAS=16,         X
                       IMSID=IMSA
                 IMSCTF SVCNO=(,254,255),                X
                       APNDG=(,ZZ),                      X
                       RDS=(LGDK,2048),                  X
                       CPLOG=500000

System Buffers   BUFPOOLS PSB=8000,PSBW=6000,            X
and Data Sets          DMB=11000,FRE=28,                 X
                 MSGQUEUE DSETS=LGDK,                    X
                       BUFFERS=(20,1152),                X
                       SHUTDWN=150
```

Database and application macro statements

The macro statements associated with the databases and application programs of the example IMS DB/DC system definition are provided.

The macro statements shown in "Sample database and application macro statements" define DL/I databases.

Sample database and application macro statements

```
Characteristics                                    Column
to define        Macro statements prepared            72

DL/I Databases

DH41SK01,        DATABASE DBD=(DH41SK01,DH41SK02,DH41TS01)
  DH41SK02,
  DH41TS01

DI31LM01         DATABASE DBD=DI31LM01

DI21PART,        DATABASE RESIDENT,                      X
  DH41SK03,            DBD=(DI21PART,DH41SK03,DX41SK01)
  DX41SK01

DI41SK01,        DATABASE DBD=(DI41SK01,DI41SK02)
  DI41SK02

DD41SK01,        DATABASE DBD=(DD41SK01,DD41TS01,DX41TS01)
  DD41TS01,
  DX41TS01

LTERMINL,        DATABASE DBD=(LTERMINL,LTINDXDB)
  LTINDXDB

TESTABLE,        DATABASE DBD=(TESTABLE,TTINDXDB)
  TTINDXDB
```

DL/I application programs

An example of the macro statements that can be used to define the characteristics associated with DL/I application programs is provided.

In the following example, each DL/I application program is shown with associated transaction codes and macros.

```
Characteristics                                  Column
to define        Macro statements prepared         72


Programs and
Transaction
Codes

HSBASK41         APPLCTN PSB=HSBASK41,PGMTYPE=BATCH
  (Batch)

HIMAJC01         APPLCTN PSB=HIMAJC01
  TPPL1          TRANSACT CODE=TPPL1,PRTY=(8,8)

HIMAJC03         APPLCTN PSB=HIMAJC03
  TUBE           TRANSACT CODE=TUBE,PRTY=(8,8),      X
                       SPA=100,MODE=SNGL
  TUBFD          TRANSACT CODE=TUBFD,PRTY=(8,8),     X
                       SPA=100,                      X
                       MODE=SNGL
  TUBFC          TRANSACT CODE=TUBFC,PRTY=(8,8),     X
                       SPA=100,                      X
                       MODE=SNGL
  TUBMA          TRANSACT CODE=TUBMA,PRTY=(8,8),     X
                       SPA=20000,MODE=SN GL

HIMALM01         APPLCTN PSB=HIMALM01,               X
                       SCHDTYP=PARALLEL
  DLI            TRANSACT CODE=DLI,PRTY=(5,10,5),    X
                       PROCLIM=(10,10),              X
                       MSGTYPE=(SNGLSEG,RESPONSE)
  ICS            TRANSACT CODE=ICS,PRTY=(5,12,5),    X
                       PROCLIM=(10,100),             X
                       MODE=SNGL
  DLN            TRANSACT CODE=DLN,PRTY=(0,8,3),     X
                       PROCLIM=(10,100)
  IMS            TRANSACT CODE=IMS,PRTY=(5,12,5),    X
                       PROCLIM=(1,100),              X
                       MSGTYPE=SNGLSEG,PARLIM=1

CONALTR          APPLCTN PSB=CONALTR
  CONAL          TRANSACT CODE=CONAL,MODE=SNGL,      X
                       PRTY=(8,8),SPA=100
  CONALF         TRANSACT CODE=CONALF,MODE=SNGL,     X
                       PRTY=(8,8),                   X
                       SPA=100
LKMDFS00         APPLCTN PSB=LKMDFS00
  STL            TRANSACT CODE=STL,PRTY=(5,12,5),    X
                       PROCLIM=(8,100),INQ=YES

LKMDFS10         APPLCTN PSB=LKMDFS10,PGMTYPE=BATCH
  LKM            TRANSACT CODE=LKM,PRTY=(0,0),       X
                       MSGTYPE=SNGLSEG

MR1              APPLCTN PSB=MR1
  MR1            TRANSACT CODE=MR1,PRTY=(8,8)

MR2              APPLCTN PSB=MR2
  MR2            TRANSACT CODE=MR2,PRTY=(8,8)
```

```
MR3             APPLCTN PSB=MR3
  MR3           TRANSACT CODE=MR3,PRTY=(8,8)

MR4             APPLCTN PSB=MR4
  MR4           TRANSACT CODE=MR4,PRTY=(8,8)

MR5             APPLCTN PSB=MR5
  MR5           TRANSACT CODE=MR5,PRTY=(8,8)
```

Characteristics to define	**Macro statements prepared**	**Column 72**

```
Programs and
Transaction
Codes

MR6             APPLCTN PSB=MR6
  MR6           TRANSACT CODE=MR6,PRTY=(8,8)

TESTIMSD        APPLCTN PSB=TESTIMSD
  TSTCONV       TRANSACT CODE=TSTCONV,PRTY=(8,8),   X
                      SPA=100,MODE=SN GL
  TSTNORM       TRANSACT CODE=TSTNORM,PRTY=(8,8)

REQIMSD         APPLCTN PSB=REQIMSD
  TREQ          TRANSACT CODE=TREQ,PRTY=(8,8),      X
                      SPA=100,MODE=SNGL

LOADLT          APPLCTN PSB=LOADLT,PGMTYPE=BATCH
  (Batch)

LOADTT          APPLCTN PSB=LOADTT,PGMTYPE=BATCH
  (Batch)

STLECHO         APPLCTN PSB=STLECHO
  ECHO          TRANSACT CODE=ECHO,PRTY=(5,10,5),   X
                      PROCLIM=(10,10),              X
                      MSGTYPE=SNGLSEG
  ECHO          TRANSACT CODE=ECHO,PRTY=(5,10,5),   X
                      PROCLIM=(10,10)

MR8             APPLCTN PSB=MR8
  MR8           TRANSACT CODE=MR8,PRTY=(8,8)
```

Characteristics to define	**Macro statements prepared**	**Column 72**

```
Programs and
Transaction
Codes

                RESIDENT PSBS

RESPSB1         APPLCTN PSB=RESPSB1,RESIDENT
  RESPSB        TRANSACT CODE=RESPSB,PRTY=(8,8)

RESBAL01        APPLCTN PSB=RESBAL01,               X
                      SCHDTYP=PARALLEL,RESIDENT
  BAL01         TRANSACT CODE=(BAL01,BAL02,BAL03),  X
                      PRTY=(8,8)
  BAL02
  BAL03

RESBAL02        APPLCTN PSB=RESBAL02,               X
                      SCHDTYP=PARALLEL
```

```
  BALTRAN        TRANSACT CODE=BALTRAN,PRTY=(8,8),   X
                       PARLIM=1

PSBEXINT         APPLCTN PSB=PSBEXINT
  TRANEXNT       TRANSACT CODE=TRANEXNT,PRTY=(8,8)

INTCON           APPLCTN PSB=INTCON
  ANY            TRANSACT CODE=ANY,PRTY=(8,8),SCHD=3
  HI             TRANSACT CODE=HI,PRTY=(8,8),SCHD=2
  EQHI           TRANSACT CODE=EQHI,PRTY=(8,8),      X
                       SCHD=1
  NXCLASS        TRANSACT CODE=NXCLASS,PRTY=(8,8),   X
                       SCHD=4

CUTOFF           APPLCTN PSB=CUTOFF
  SIX            TRANSACT CODE=SIX,PRTY=(6,6)
  SEVEN          TRANSACT CODE=SEVEN,PRTY=(7,7)
  EIGHT          TRANSACT CODE=EIGHT,PRTY=(8,8)
  NINE           TRANSACT CODE=NINE,PRTY=(9,9)

NXTCLASS         APPLCTN PSB=NXTCLASS,PGMTYPE=(,,2)
  ONE            TRANSACT CODE=ONE,PRTY=(1,1)
```

The system includes one 3600 application program.

```
Characteristics                                Column
to define        Macro statements prepared       72


Programs and
Transaction
Codes

TEST3600         APPLCTN DOPT,PSB=TEST3600,          X
                       PGMTYPE=TP
  CIFINQ         TRANSACT CODE=CIFINQ,PRTY=(1,2,2),  X
                       MSGTYPE=SNGLSEG,              X
                       PROCLIM=(,30),INQUIRY=YES,    X
                       SEGNO=256,SEGSIZE=4096
  SAVDBT         TRANSACT CODE=(SAVDBT,SAVDPT),      X
                       PRTY=(1,2,2),                 X
                       MSGTYPE=SNGLSEG,              X
                       PROCLIM=(,30),INQUIRY=NO,     X
                       SEGNO=10,SEGSIZE=80
BRNCHTOT         TRANSACT CODE=BRNCHTOT,             X
                       PRTY=(1,2,2),MSGTYPE=SNGLSEG, X
                       PROCLIM=(,30),INQUIRY=YES,    X
                       SEGNO=256,SEGSIZE=132

PGM3741          APPLCTN PSB=PGM3741,PGMTYPE=TP
  TX3741         TRANSACT CODE=TX3741

DFSSAM02         APPLCTN PSB=DFSSAM02,               X
                       PGMTYPE=(TP,,4)
  PART           TRANSACT CODE=PART,PRTY=(7,10,2),   X
                       INQ=(YES,NORECOV)

DFSSAM03         APPLCTN PSB=DFSSAM03
  DSPINV         TRANSACT CODE=DSPINV,PRTY=(7,10,2), X
                       INQ=YES,MSGTYPE=(,,4 )

DFSSAM04         APPLCTN PSB=DFSSAM04,PGMTYPE=(,,4)
  ADDPART        TRANSACT CODE=(ADDPART,DLETPART),   X
                       PRTY=(7,10,2)
  DLETPART
  ADDINV         TRANSACT CODE=(ADDINV,DLETINV),     X
                       PRTY=(7,10,2),MSGTYPE=(,,5)
```

```
  DLETINV

DFSSAM05         APPLCTN PSB=DFSSAM05,PGMTYPE=(,,3)
  CLOSE          TRANSACT CODE=CLOSE,PRTY=(7,10,2)

DFSSAM06         APPLCTN PSB=DFSSAM06
  DISBURSE       TRANSACT CODE=DISBURSE,             X
                       PRTY=(7,10,2),MSGTYPE=(,,2)

DFSSAM07         APPLCTN PSB=DFSSAM07,PGMTYPE=(,,2)
  DSPALLI        TRANSACT CODE=DSPALLI,              X
                       PRTY=(7,10,2),INQ=YES
```

Data Communication macro statements

These topics describe several line group configurations and include the required macro statements for each configuration.

Local SYSOUT line groups

Macro statements to define local SYSOUT line groups are provided in this topic.

Figure 18 shows local SYSOUT line groups.

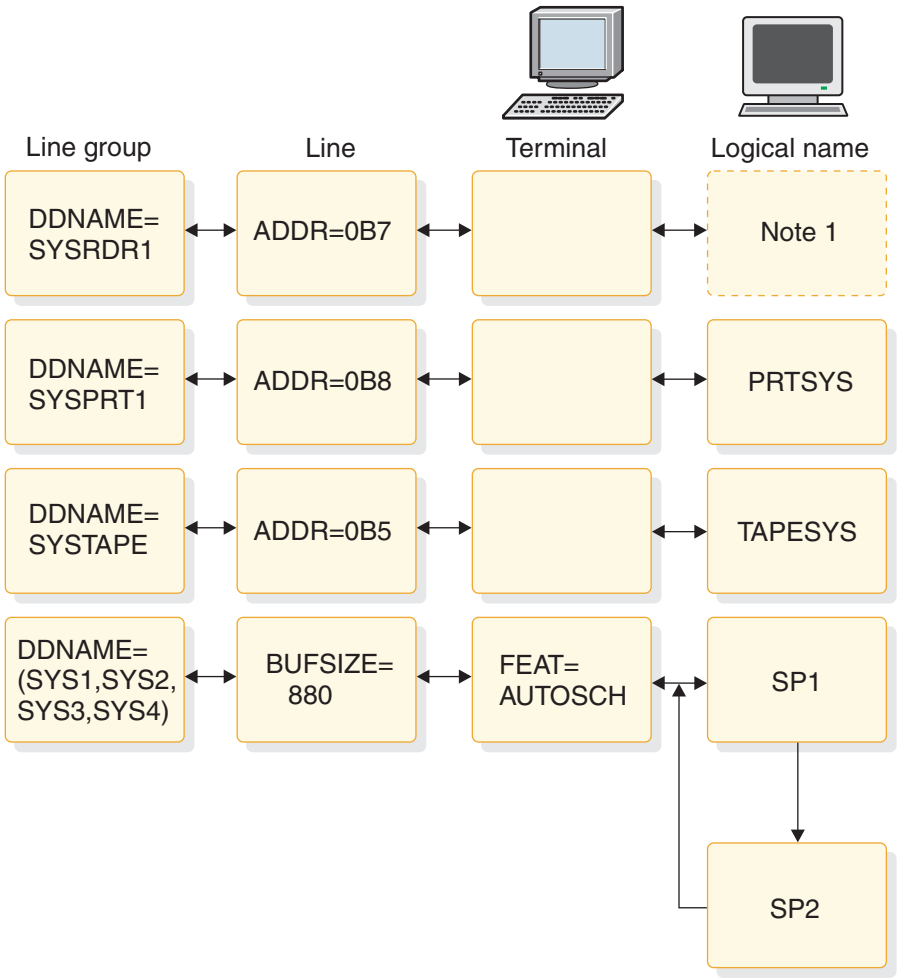


Figure 18. Local SYSOUT line groups

Sample macro statements for local SYSOUT line group

```
LINEGRP DDNAME=SYSRDR1,UNITYPE=READER
LINE   ADDR=0B7
TERMINAL LTERM=LCLRDR1
LINEGRP DDNAME=SYSPRT1,UNITYPE=PRINTER
LINE    ADDR=0B8
TERMINAL
NAME    PRTSYS
LINEGRP DDNAME=SYSTAPE,UNITYPE=TAPE
LINE    ADDR=0B5,BUFSIZE=220
TERMINAL
NAME    TAPESYS
LINEGRP UNITYPE=SPOOL,DDNAME=(SYS1,SYS2,SYS3,SYS4)
LINE    BUFSIZE=880
TERMINAL FEAT=AUTOSCH
NAME    SP1,SP2
```

Macro statements to define VTAM communication devices

Examples of system definition macro statements to define VTAM communication devices are provided in these topics.

The following sets of VTAM macro statements describe the VTAM communication devices. A graphic representation of each of these is presented, along with the macro statements required to define it.

COMM macro statement

A sample COMM macro statement is provided.

The COMM macro statement shown in "Sample COMM macro statement" can only appear once in an IMS system definition. When VTAM is included in a system, the COMM macro statement is required.

Sample COMM macro statement

```
Macro statement set                    Column
                                                  72
COMM  RECANY=(4,500),APPLID=IMSD,                  X
      OPTIONS=(TIMESTAMP,MFSTEST,FMTMAST,PAGING)
```

Macro statements to define local 3270 VTAM terminals

Sample macro statements to define local 3270 VTAM terminals are provided.

Figure 19 on page 260 illustrates macro statements for a local 3270 VTAM terminal.

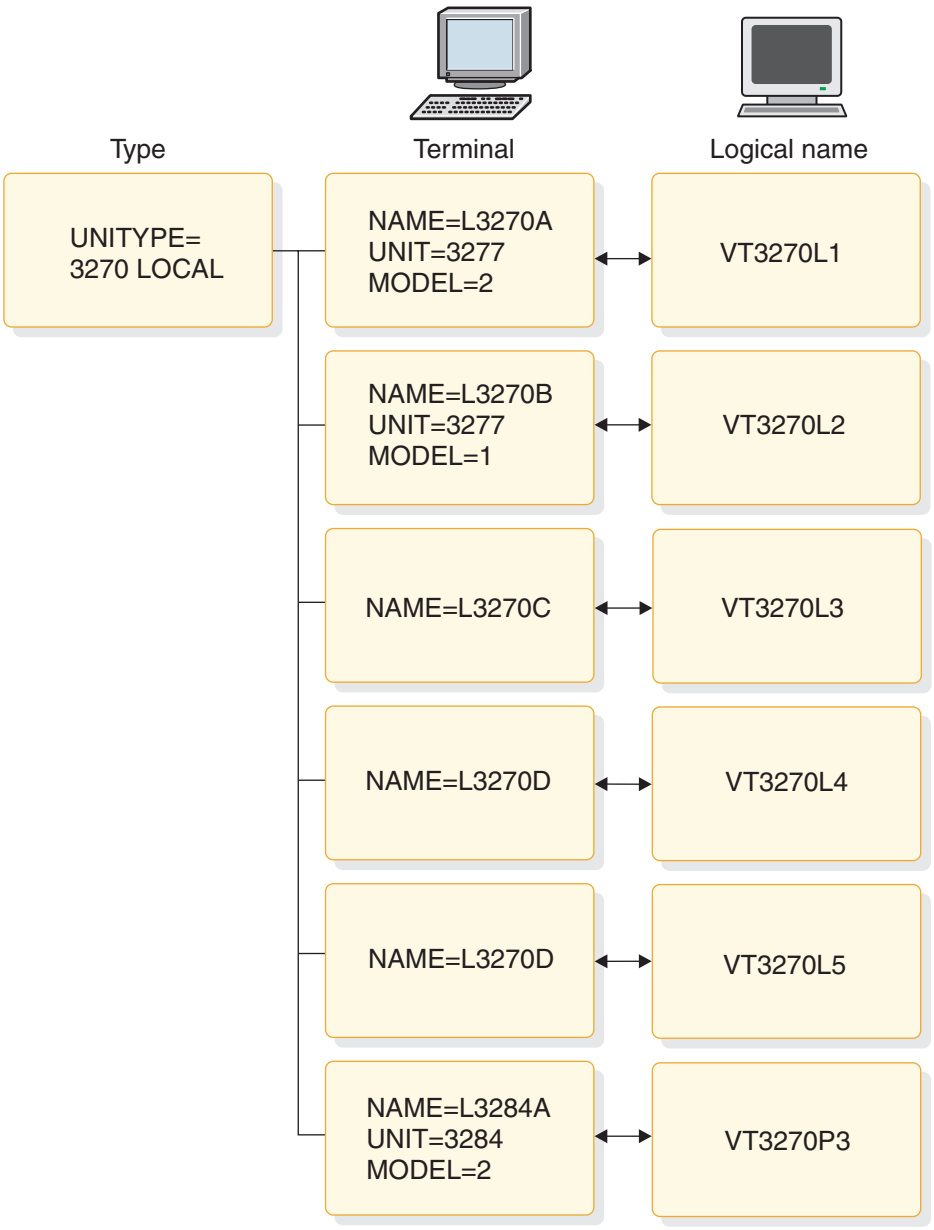


Figure 19. Local 3270 VTAM terminal

"Sample macro statements to define 3270 VTAM terminals" shows the macro statements to define local 3270 VTAM terminals.

Sample macro statements to define 3270 VTAM terminals

```
TYPE  UNITYPE=(3270,LOCAL),MODEL=2
TERMINAL NAME=L3270A
NAME  (VT3270L1,MASTER)
TERMINAL NAME=L3270B,MODEL=1,OPTIONS=FORCRESP
NAME  VT3270L2
TERMINAL NAME=L3270C,TYPE=3270-A2,SIZE=(24,80)
NAME  VT3270L3
TERMINAL NAME=L3270D,TYPE=3270-A3,SIZE=(32,80)
NAME  VT3270L4
```

```
TERMINAL NAME=L3270E,TYPE=3270-A4,SIZE=(43,80)
NAME  VT3270L5
TERMINAL NAME=L3284A,UNIT=3284,PTRSIZE=132
NAME  (VT3270P3,SECONDARY)
```

Macro statements to define remote 3270 VTAM terminals

Examples of macro statements used to define remote 3270 VTAM terminals are provided.

Figure 20 on page 262 illustrates macro statements for remote 3270 VTAM terminals.

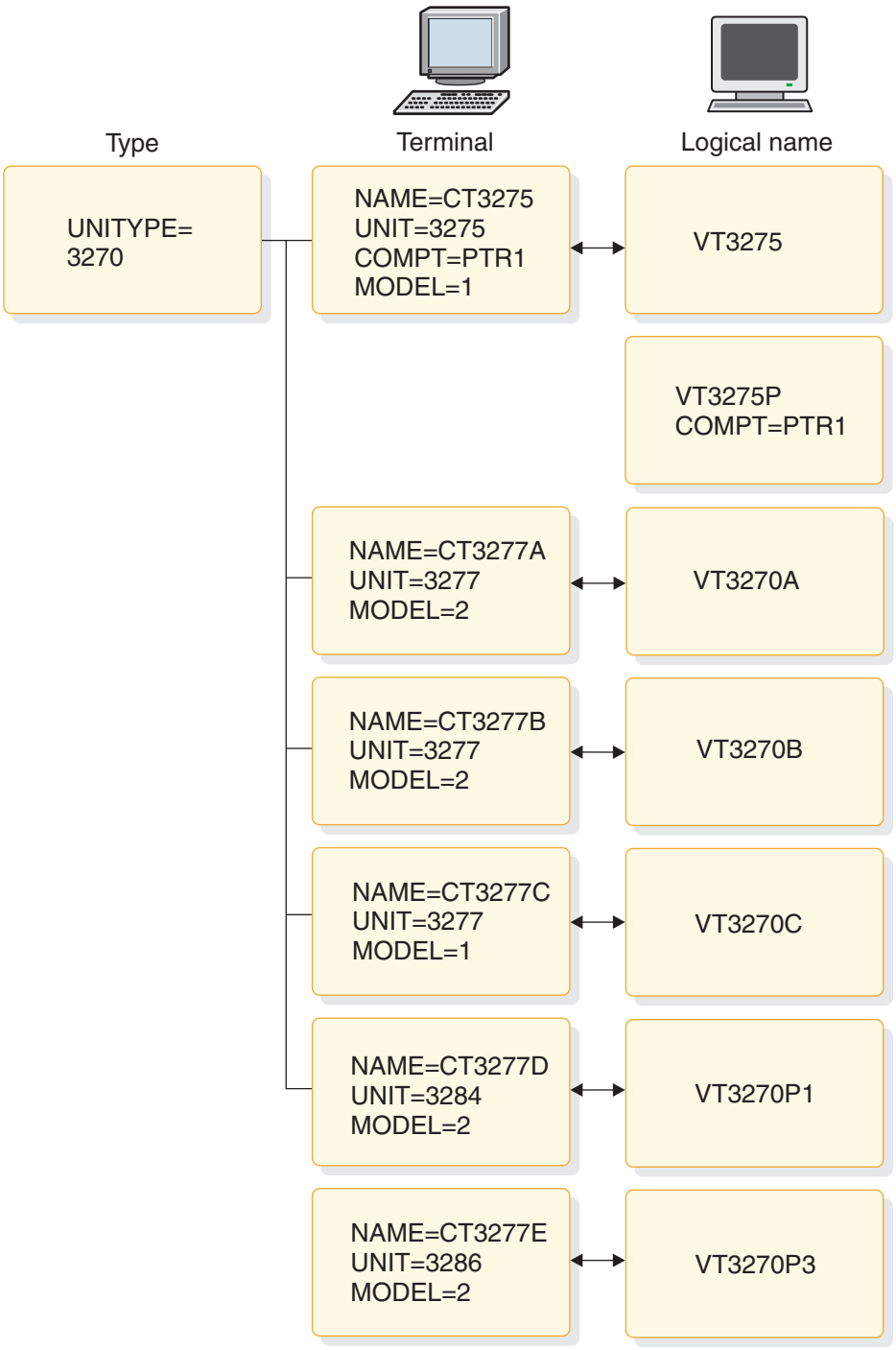


Figure 20. Remote 3270 VTAM terminals

"Sample macro statements to define remote 3270 VTAM terminals" shows the macro statements to define remote 3270 VTAM terminals.

Sample macro statements to define remote 3270 VTAM terminals

```
TYPE UNITYPE=3270,MODEL=2,PTRSIZE=132,OPTIONS=COPY
TERMINAL NAME=CT3275,UNIT=3275,COMPT=PTR1,MODEL=1
NAME VT3275
NAME VT3275P,COMPT=PTR1
TERMINAL NAME=CT3277A
NAME VT3270A
```

```
TERMINAL NAME=CT3277B
NAME VT3270B
TERMINAL NAME=CT3277C,MODEL=1
NAME VT3270C
TERMINAL NAME=CT3277D,UNIT=3284
NAME VT3270P1
SPACE
TERMINAL NAME=CT3277E,UNIT=3286
NAME VT3270P2
```

Macro statements to define a finance communication system

Examples of macro statements that can be used to define a finance communication system are provided.

Figure 21 on page 264 illustrates macro statements for a finance communication system.

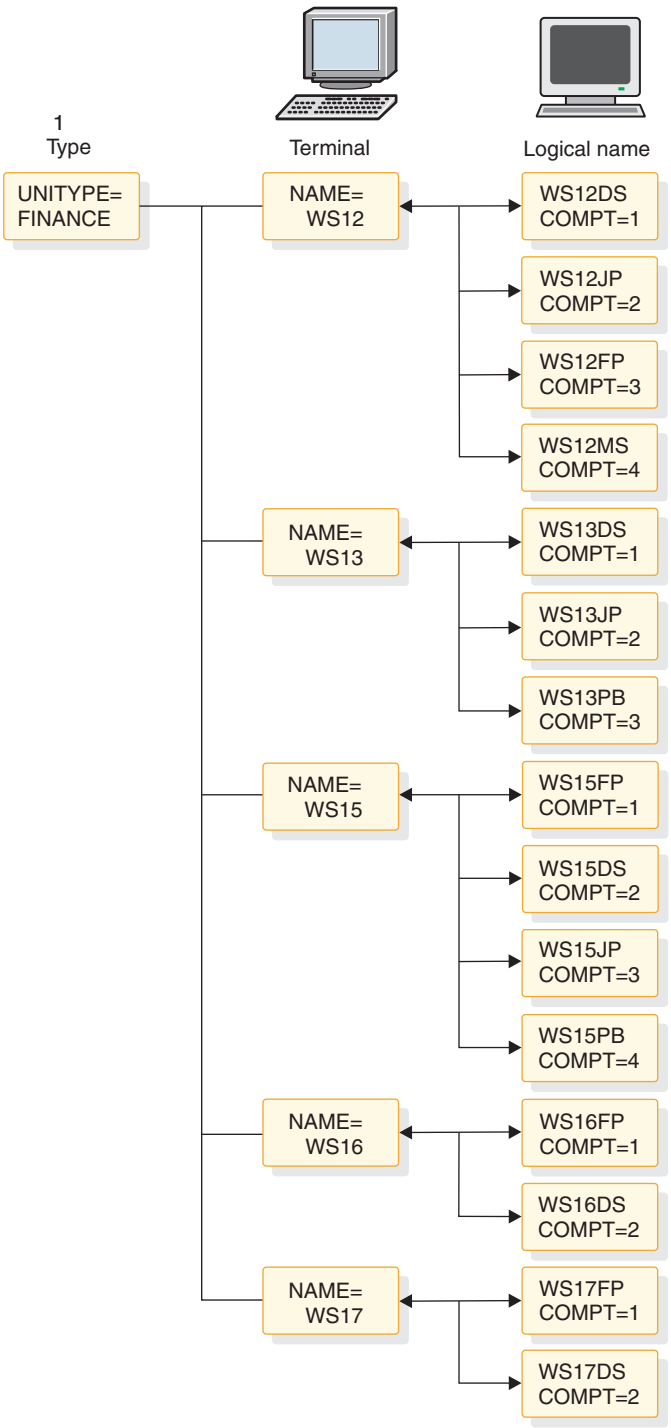


Figure 21. Finance Communication system. [1] Can also specify UNITYPE=3601.

"Sample macro statements to define a finance communication system" shows the macro statements to define a finance communication system.

Sample macro statements to define a finance communication system

```
Macro statement set                                    Column
                                                          72

TYPE UNITYPE=FINANCE,OUTBUF=256
TERMINAL NAME=WS12,OPTIONS=MFS
      COMPT=(36DS3,36JP,36FP,36MS),FEAT=(DUAL,132)
```

```
NAME  WS12DS,COMPT=1
NAME  WS12JP,COMPT=2
NAME  WS12FP,COMPT=3
NAME  WS12MS,COMPT=4
TERMINAL NAME=WS13,OPTIONS=(MFS,NOPNDST,OPTACK),     X
      COMPT=(36DS4,36JP,36PB)
NAME  WS13DS,COMPT=1
NAME  WS13JP,COMPT=2
NAME  WS13PB,COMPT=3
TERMINAL NAME=WS15,COMPT=(36FP,36DS,36JP,36PB),      X
      OPTIONS=MFS
NAME  WS15FP,COMPT=1
NAME  WS15DS,COMPT=2
NAME  WS15JP,COMPT=3
NAME  WS15PB,COMPT=4
TERMINAL NAME=WS16,COMPT=(36FP,36DS4),               X
      OPTIONS=(FORCRESP,MFS,OPTACK,BID,OPNDST)
NAME  WS16FP,COMPT=1
NAME  WS16DS,COMPT=2
TERMINAL NAME=WS17,COMPT=(36FP,36DS),                X
      OPTIONS=(ACK,NOBID,TRANRESP,NOPNDST)
NAME  WS17FP,COMPT=1
NAME  WS17DS,COMPT=2
```

Macro statements to define secondary logical unit type 1 VTAM terminals

Examples of macro statements that can be used to define secondary logical unit type 1 VTAM terminals are provided.

Figure 22 illustrates the macro statements for a secondary logical unit type 1 VTAM terminal.

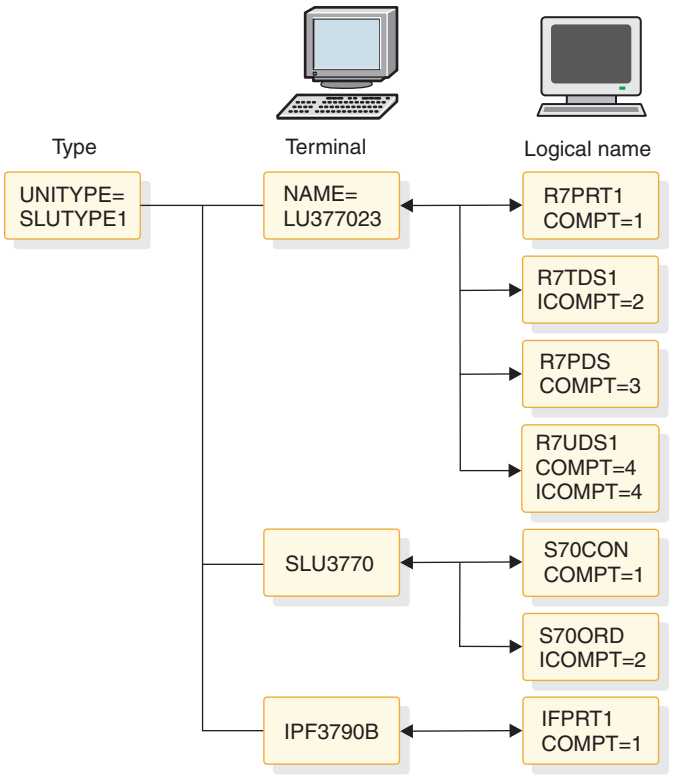


Figure 22. Secondary logical unit type 1 VTAM terminals

"Sample macro statements to define SLU type 1 VTAM terminals" shows the macro statements to define SLU Type 1 VTAM terminals.

Sample macro statements to define SLU type 1 VTAM terminals

```
Macro statement set                                    Column
                                                          72
TYPE UNITYPE=SLUTYPE1
TERMINAL NAME=LU377023,                                    X
      COMPT1=(PRINTER1,BASIC-SCS1),                        X
      COMPT2=(TRANSDS1,BASIC-SCS2),                        X
      COMPT3=(PRINTDS1,BASIC-SCS1),                        X
      COMPT4=(USERDS1,BASIC)
NAME R7PRT1,COMPT=1
NAME R7TDS1,ICOMPT=2
NAME R7PDS1,COMPT=3
NAME R7UDS1,COMPT=4,ICOMPT=4
TERMINAL NAME=SLU3770,COMPT1=(CONSOLE,MFS-SCS1),           X
      COMPT2=(READER1,MFS-SCS2,3)
NAME S70CON,COMPT=1
NAME S70RDR,ICOMPT=2
TERMINAL NAME=IFP3790B,                                    X
      COMPT1=(PRINTER1,BASIC-SCS1),                        X
      OPTIONS=(OPNDST,NBSELM)
NAME IFPRT1,COMPT=1
```

Macro statements to define NTO devices

Examples of macro statements that can be used to define Network Terminal Option (NTO) devices are provided.

Figure 23 illustrates macro statements for NTO devices.

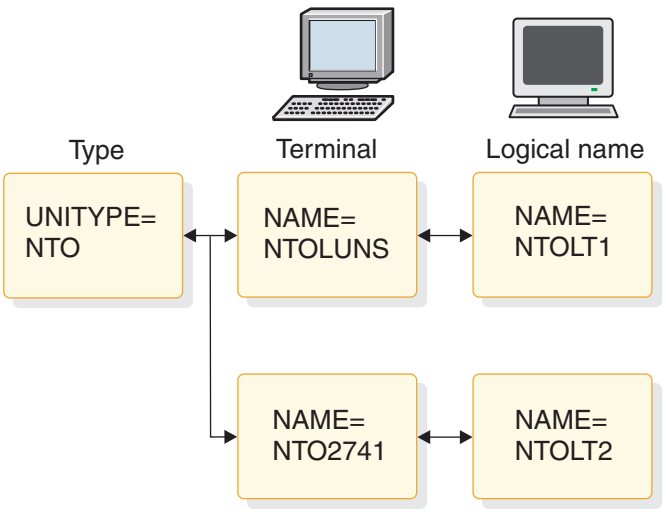


Figure 23. NTO devices

"Sample macro statements to define NTO devices" shows the macro statements to define NTO devices.

Sample macro statements to define NTO devices

```
Macro statement set                                    Column
                                                          72
TYPE UNITYPE=NTO,EDIT=(EDIT1,EDIT2)
TERMINAL NAME=NTOLUNS,EDIT=(YES,YES),                      X
      OUTBUF=300,SEGSIZE=300,PU=LUNS,OPTIONS=OPNDST
```

```
NAME NTOLT1
TERMINAL NAME=NTO2741,PU=2741,                          X
      MSGDEL=NONIOPCB,OPTIONS=(NORESP,MFS)
NAME NTOLT2
```

Macro statements to define secondary logical unit type 2 VTAM terminals

Examples of macro statements that can be used to define secondary logical unit type 2 VTAM terminals are provided.

Figure 24 illustrates macro statements for secondary logical unit Type 2 VTAM terminals.

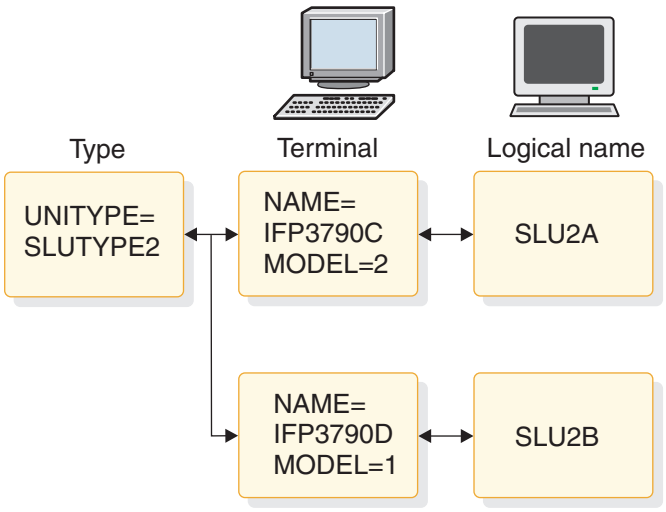


Figure 24. Secondary logical unit type 2 VTAM terminals

"Macro statements to define SLU type 2 VTAM terminals" shows the macro statements to define SLU type 2 VTAM terminals.

Macro statements to define SLU type 2 VTAM terminals

```
TYPE UNITYPE=SLUTYPE2,MODETBL=AAA
TERMINAL NAME=IFP3790C,MODEL=2,MODETBL=BBB
NAME SLU2A
TERMINAL NAME=IFP3790D,MODEL=1,OUTBUF=500
NAME SLU2B
```

Macro statements to define secondary logical unit type P VTAM terminals

Examples of macro statements that can be used to define secondary logical unit type P VTAM terminals are provided.

Figure 25 on page 268 illustrates the macro statements for secondary logical unit type P VTAM terminals.

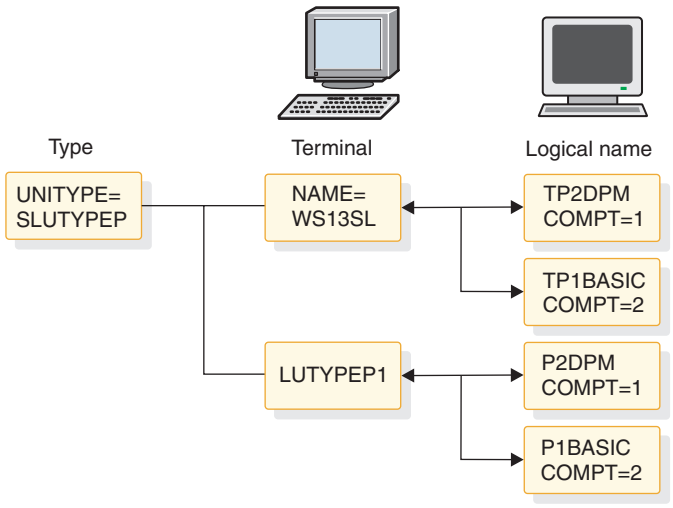


Figure 25. Secondary logical unit type P VTAM terminal

"Sample macro statements to define SLU type P VTAM terminals" shows the macro statements to define SLU Type P VTAM terminals.

Sample macro statements to define SLU type P VTAM terminals

```
Macro statement set                                    Column
                                                          72
TYPE  UNITYPE=SLUTYPEP,OUTBUF=256,                        X
      OPTIONS=(OPTACK,NOBID)
TERMINAL NAME=WS13SL,                                     X
      COMPT1=(PROGRAM2,DPM-A1,4),                         X
      COMPT2=(PROGRAM1,BASIC)
NAME TP2DPM,COMPT=1
NAME TP1BASIC,COMPT=2
TERMINAL NAME=LUTYPEP1,                                   X
      COMPT1=(PROGRAM2,DPM-A1,IGNORE),                    X
      COMPT2=(PROGRAM1,DPM-A3,2),                         X
      COMPT3=PROGRAM1
NAME P2DPM,COMPT=1
NAME P1BASIC,COMPT=2
```

Macro statements to define logical unit type 6 VTAM terminals

Examples of macro statements that can be used to define logical unit type 6 VTAM terminals are provided.

Figure 26 on page 269 shows the macro statements for Logical Unit Type 6 VTAM terminals.

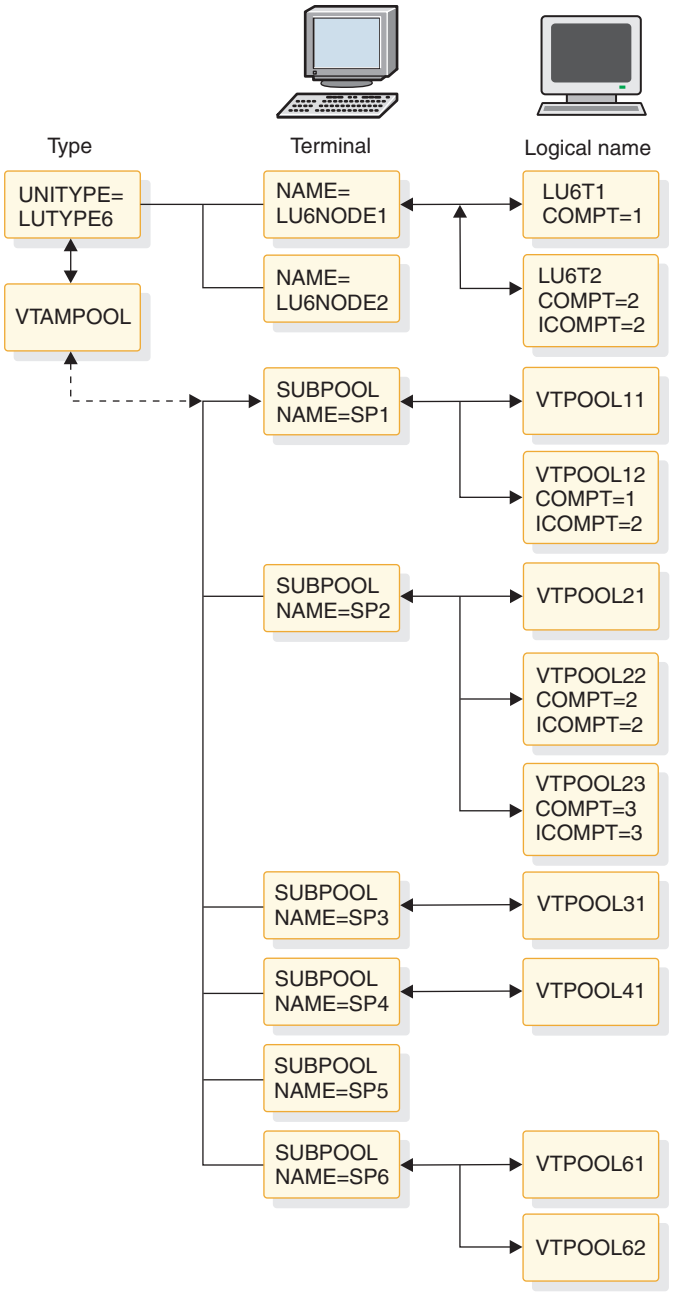


Figure 26. Logical unit type 6 VTAM terminals

"Sample macro statements to define LU type 6 VTAM terminals" shows the macro statements to define LU type 6 VTAM terminals.

Sample macro statements to define LU type 6 VTAM terminals

```
Macro statement set                                    Column
                                                         72
         TYPE  UNITYPE=LUTYPE6

*LU 6 NODE DEFINITION WITH FIXED LTERM ALLOCATION

         TERMINAL NAME=LU6NODE1,OPTIONS=NORESP,      X
               COMPT1=(MULT2,DPM-B1,4),              X
               COMPT2=(SINGLE2,VLVB)
         NAME LU6T1,COMPT=1
```

```
         NAME LU6T2,COMPT=2,ICOMPT=2

*LU 6 NODE DEFINITION -- PARALLEL SESSIONS WITH
 DYNAMIC LTERM ALLOCATION

         TERMINAL NAME=LU6NODE2,OPTIONS=TRANRESP,    X
               COMPT1=(SINGLE1,DPM-B1,IGNORE),       X
               COMPT2=(SINGLE2,DPM-B3,2),            X
               COMPT3=MULT1,                         X
               SESSION=3

               VTAMPOOL
               SUBPOOL NAME=SP1
               NAME VTPOOL11
               NAME VTPOOL12,COMPT=1,ICOMPT=2
               SUBPOOL NAME=SP2
               NAME VTPOOL21
               NAME VTPOOL22,COMPT=2,ICOMPT=2
               NAME VTPOOL23,COMPT=3,ICOMPT=3
               SUBPOOL NAME=SP3
               NAME VTPOOL31
               SUBPOOL NAME=SP4
               NAME VTPOOL41
               SUBPOOL NAME=SP5
               SUBPOOL NAME=SP6
               NAME VTPOOL61
               NAME VTPOOL62
```

System configuration macro statement

An example of IMSGEN, a macro statement that can be used to configure an IMS system, is provided.

In this example, the IMSGEN macro statement must be last in the IMS system definition input stream. The IMSGEN macro statement for that sample system is shown in "Sample system configuration macro statement."

Sample system configuration macro statement

```
Characteristics                                  Column
to define        Macro statements prepared         72


IMS              IMSGEN ASM=HLASM,                  X
Configuration           ASMPRT=ON,                  X
                        LKPRT=(XREF,LIST),          X
                        LKSIZE=(200K,28K),          X
                        LKRGN=200K,                 X
                        SUFFIX=0,                   X
                        OBJDSET=IMS.OBJDSET,        X
                        USERLIB=IMS.SDFSRESL,       X
                        PROCLIB=(YES,8),            X
                        NODE=IMS,                   X
                        SCL=4,                      X
                        JCL=(SYSDEF,                X
                        (A100,B20,M7044L21,'T=MAC''), X
                        MACKAY,0,(MSGLEVEL=(1,1),   X
                        CLASS=N))
```

IMS Multiple Systems Coupling

Examples of system definitions for IMS Multiple Systems Coupling (MSC) are provided in this topic.

The Multiple Systems Coupling (MSC) sample shown in Figure 27 includes the relevant macros from the system definition of each of the three systems in a sample multiple-IMS system configuration. The multiple-systems configuration is shown first, followed by extractions from the three system definitions in "System A IMS system definition example" on page 272, "System B IMS system definition example" on page 273, and "System C IMS system definition example" on page 274. These system definitions do not include Intersystem Communication (ISC).

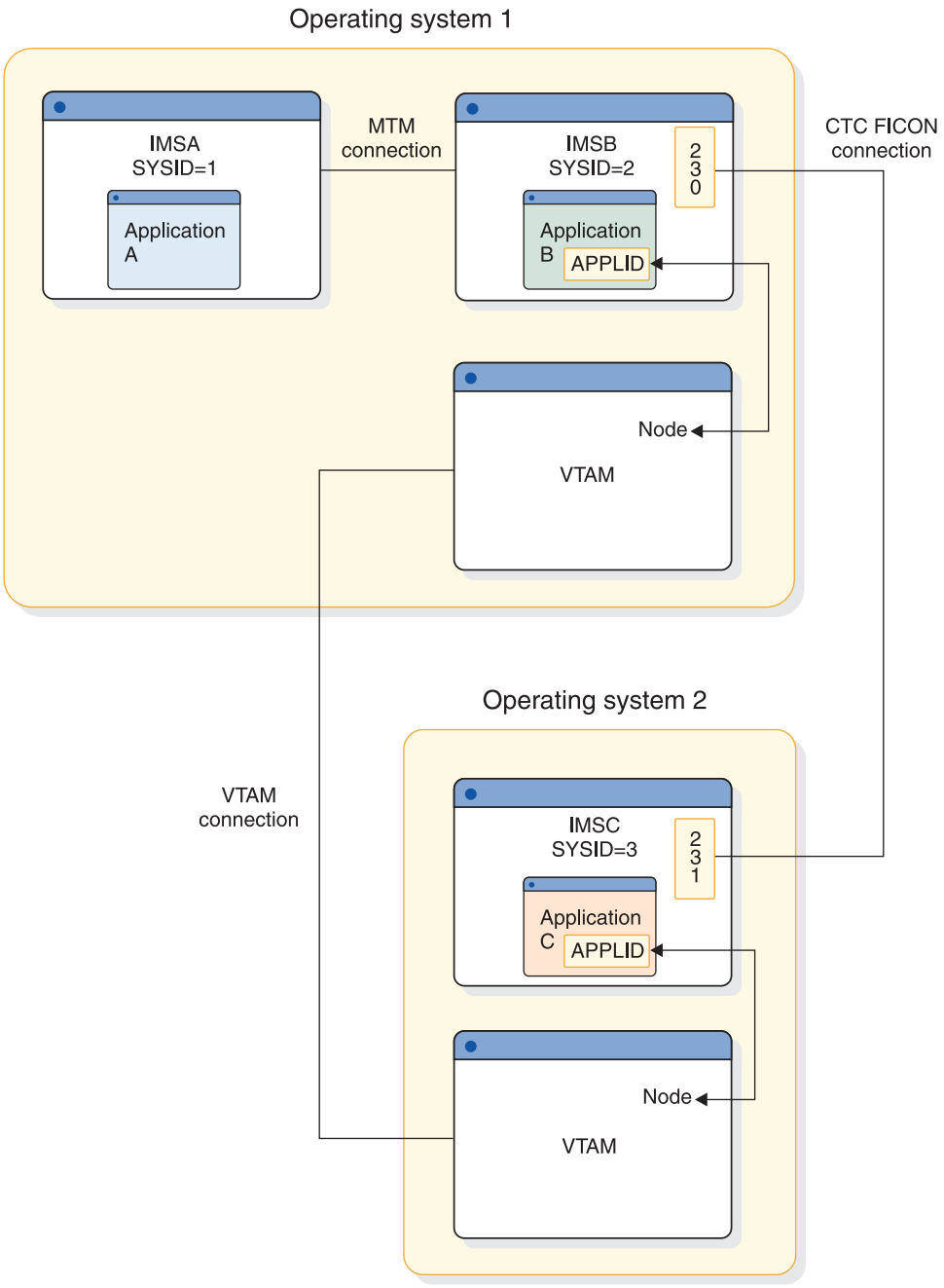


Figure 27. Multiple Systems Coupling example

System A IMS system definition example

```
Characteristics                                          Col.
to define          Label  Macro statements prepared     72


IMS System                IMSCTRL MSVID=1,IMSID=SYSA
Configuration             IMSCTF

Databases and             DATABASE
Application
Macros                    APPLCTN
                          TRANSACT
(Application              APPLCTN PSB=A
programs that             TRANSACT CODE=A
execute in                APPLCTN PSB=B,SYSID=(2,1)
other systems             TRANSACT CODE=B
but process               APPLCTN PSB=C,SYSID=(3,1)
transactions              TRANSACT CODE=C
entered by
System A                  APPLCTN
terminals are             TRANSACT
defined as
remote.)

Data
Communication             COMM
                          TYPE
(These logical            TERMINAL
terminals are
defined in                TYPE
Systems B and C           TERMINAL
as remote
logical
terminals.)               TYPE
                          TERMINAL
                          NAME IMSATRM1
                          TERMINAL
                          NAME IMSATRM2
                          TERMINAL
                          NAME IMSATRM3
Multiple          PLINK1  MSPLINK DDNAME=PLAB1,          X
Systems                         ADDR=230,TYPE=CTC,       X
Coupling                        BUFSIZE=4096
                  PLINK2  MSPLINK TYPE=MTM,              X
                                  BUFSIZE=4096
                  MSCLK1  MSLINK PARTNER=AB,             X
                                  MSPLINK=PLINK1

                          END
(Terminals in
Systems B and C
that can enter
transaction
code A are
defined as
remote
terminals.)
                   MSC12  MSNAME SYSID=(2,1)
                          NAME IMSBTRM1
                          NAME IMSBTRM2
                          NAME IMSBTRM3
                   MSC13  MSNAME SYSID=(3,1)
                          NAME IMSCTRM1
                          NAME IMSCTRM2
                          NAME IMSCTRM3
                          IMSGEN
```

System B IMS system definition example

```
Characteristics                                    Col.
to define          Label  Macro statements prepared  72


IMS System                IMSCTRL MSVID=2,IMSID=SYSB
Configuration             IMSCTF  APNDG=(,,Z6)

Databases and             DATABASE
Applications              APPLCTN
                          TRANSACT
(Application              APPLCTN PSB=B
programs that             TRANSACT CODE=B
execute in                APPLCTN PSB=A,SYSID=(1,2)
other systems             TRANSACT CODE=A
but process               APPLCTN PSB=C,SYSID=(3,2)
transactions              TRANSACT CODE=C
entered by
System B                  APPLCTN
terminals are             TRANSACT
defined as
remote.)

Data
Communication             COMM
                          TYPE
(These logical            TERMINAL
terminals are
defined in                TYPE
Systems A and C           TERMINAL
as remote
logical                   TYPE
terminals.)               TERMINAL
                          NAME IMSBTRM1
                          TERMINAL
                          NAME IMSBTRM2
                          TERMINAL
                          NAME IMSBTRM3
                          TERMINAL
                          NAME IMSVBT1
                          TERMINAL
                          NAME IMSVBT2
                          TERMINAL
                          NAME IMSVBT3
                          TERMINAL
                          NAME IMSVBT4
Multiple          PLINK1  MSPLINK DDNAME=PLBA1,          X
Systems                         ADDR=231,TYPE=CTC,       X
Coupling                        BUFSIZE=4096
                  PLINK2  MSPLINK TYPE=MTM,              X
                                  BUFSIZE=1024
                  PLINK4  MSPLINK TYPE=VTAM,             X
                                  NAME=SANFRAN,          X
                                  SESSION=2,             X
                                  BUFSIZE=8192

                  MSCLK2  MSLINK PARTNER=AB,             X
                                  MSPLINK=PLINK1
                   MSC21  MSNAME SYSID=(1,2)
                          NAME IMSATRM1
                          NAME IMSATRM2
                          NAME IMSATRM3

                  MSCLK3  MSLINK PARTNER=BC,             X
                                  MSPLINK=PLINK2
                   MSC23  MSNAME SYSID=(3,2)
```

```
                NAME IMSCTRM1
                NAME IMSCTRM2
                NAME IMSCTRM3

MSCLK4  MSLINK PARTNER=CB,           X
                MSPLINK=PLINK4
 MSC24  MSNAME SYSID=(4,6)
        NAME IMSVCT1
        NAME IMSVCT2

MSCLK5  MSLINK PARTNER=CC,           X
                MSPLINK=PLINK4,      X
                OPTIONS=FORCSESS
 MSC25  MSNAME SYSID=(5,7)
        NAME IMSVCT3
        NAME IMSVCT4

        IMSGEN

        END
```

System C IMS system definition example

Characteristics to define	Label	Macro statements prepared	Col. 72
IMS System Configuration		IMSCTRL MSVID=3,IMSID=SYSC IMSCTF APNDG=(,,Z6)	
Databases and Applications (Application programs that execute in other systems but process transactions entered by System C terminals are defined as remote.)		DATABASE APPLCTN TRANSACT APPLCTN PSB=C TRANSACT CODE=C APPLCTN PSB=A,SYSID=(1,3) TRANSACT CODE=A APPLCTN PSB=B,SYSID=(2,3) TRANSACT CODE=B APPLCTN TRANSACT	
Data Communication (These logical terminals are defined in Systems A and B as remote logical terminals.)		COMM TYPE TERMINAL TYPE TERMINAL TYPE TERMINAL NAME IMSCTRM1 TERMINAL NAME IMSCTRM2 TERMINAL NAME IMSCTRM3 TERMINAL NAME IMSVCT1 TERMINAL NAME IMSVCT2	

```
                        TERMINAL
                        NAME IMSVCT3
                        TERMINAL
                        NAME IMSVCT4
Multiple        PLINK1  MSPLINK DDNAME=PLCB1,        X
Systems                       ADDR=0BB,TYPE=BSC,     X
Coupling                      BUFSIZE=4096,CONTROL=NO
                PLINK4  MSPLINK TYPE=VTAM,           X
                              NAME=SANJOSE,          X
                              SESSION=2,             X
                              BUFSIZE=8196

                MSCLK6  MSLINK PARTNER=BC,           X
                              MSPLINK=PLINK1
                  MSC31  MSNAME SYSID=(1,3)
                         NAME IMSATRM1
                         NAME IMSATRM2
                         NAME IMSATRM3
                  MSC32  MSNAME SYSID=(2,3)
                         NAME IMSBTRM1
                         NAME IMSBTRM2
                         NAME IMSBTRM3
                 MSCLK7  MSLINK PARTNER=CB,           X
                               MSPLINK=PLINK4
                  MSC34  MSNAME SYSID=(6,4)
                         NAME IMSVBT1
                         NAME IMSVBT2
                 MSCLK8  MSLINK PARTNER=CC,           X
                               MSPLINK=PLINK4,        X
                               OPTIONS=FORCSESS
                  MSC35  MSNAME SYSID=(7,5)
                         NAME IMSVBT3
                         NAME IMSVBT4
                           IMSGEN
                           END
```

Data-sharing system configuration

Four different system definition examples of IMS data-sharing systems are shown, including PSB definition statements.

These examples show four possible data-sharing system configurations. The IMS system definition macros that must be defined to support these configurations and the associated PSB definition statements are shown for each configuration. The illustrations in these examples assume that the database is registered with DBRC.

Data sharing at the database-level with update access

An example of data sharing at the database-level with update access, including IMS and PSB definition statements, is provided.

Figure 28 on page 276 shows a possible configuration for data sharing.

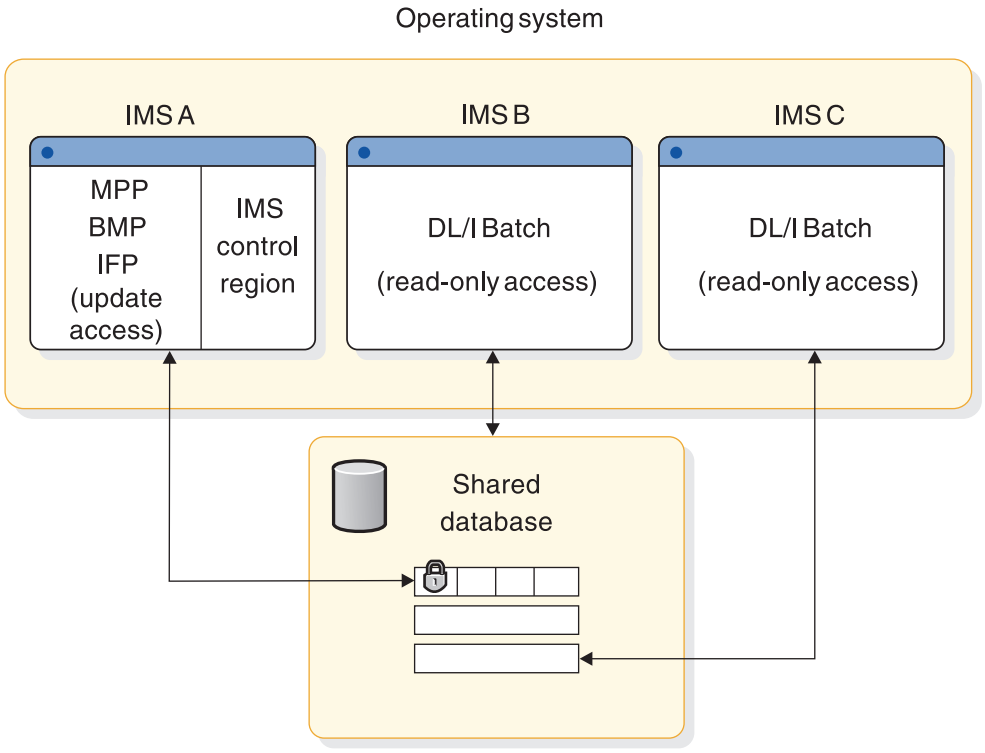


Figure 28. Data sharing at the database level with update access

System A

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSA
DATABASE  DBD=SHRDB,ACCESS=UP
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=A
```

System B

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSB,SYSTEM=(VS2,BATCH)
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=GO
```

System C

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSC,SYSTEM=(VS2,BATCH)
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=GO
```

Data sharing at the database level with read access

An example of data sharing at the database level with read access, including IMS and PSB definition statements, is provided.

Figure 29 shows a possible configuration for data sharing.

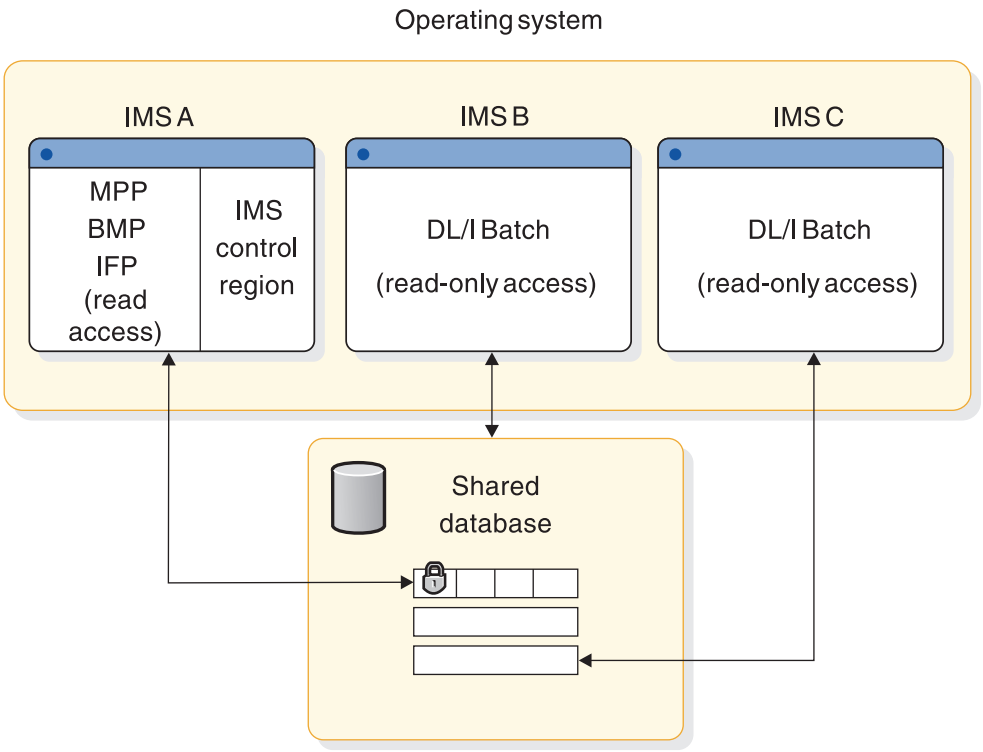


Figure 29. Data sharing at the database level with read access

System A

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSA
DATABASE  DBD=SHRDB,ACCESS=RD
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=G
```

System B

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSB,SYSTEM=(VS2,BATCH)
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=GO
```

System C

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSC,SYSTEM=(VS2,BATCH)
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=G
```

Intra-CPC block-level data sharing

An example of intra-CPC block-level data sharing, including IMS and PSB definition statements, is provided.

Figure 30 shows a possible configuration for data sharing.

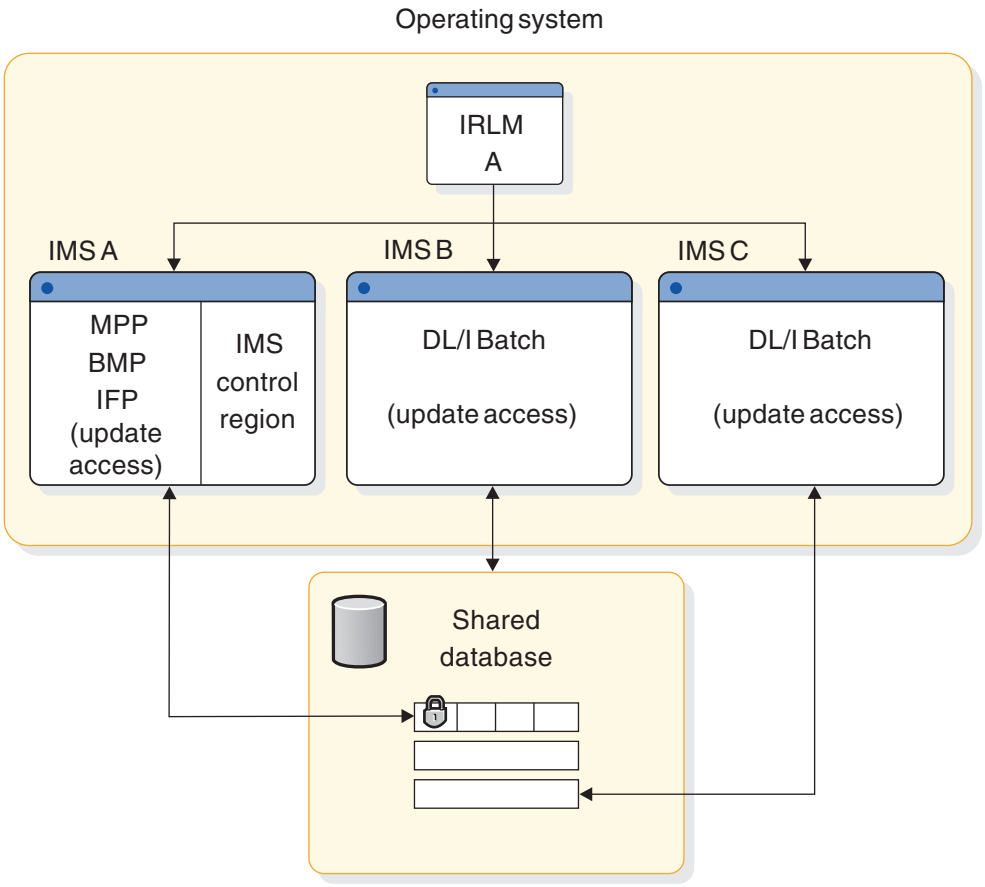


Figure 30. Intra-CPC block-level data sharing

System A

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSA,IRLMNM=RLMA
DATABASE  DBD=SHRDB,ACCESS=RD
```

PSB definition statements include:

- `PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=A`

System B

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSB,IRLMNM=RLMA,SYSTEM=(VS2,BATCH)
```

PSB definition statements include:

```
PCB         TYPE=DB,DBDNAME=SHRDB,PROCOPT=GO
```

System C

IMS system definition statements include:

```
IMSCTRL     IMSID=IMSC,SYSTEM=(VS2,BATCH)
```

PSB definition statements include:

```
PCB         TYPE=DB,DBDNAME=SHRDB,PROCOPT=A
```

Inter-CPC block-level data sharing

An example of inter-CPC block-level data sharing, including IMS and PSB definition statements, is provided.

Figure 31 shows a possible configuration for data sharing.

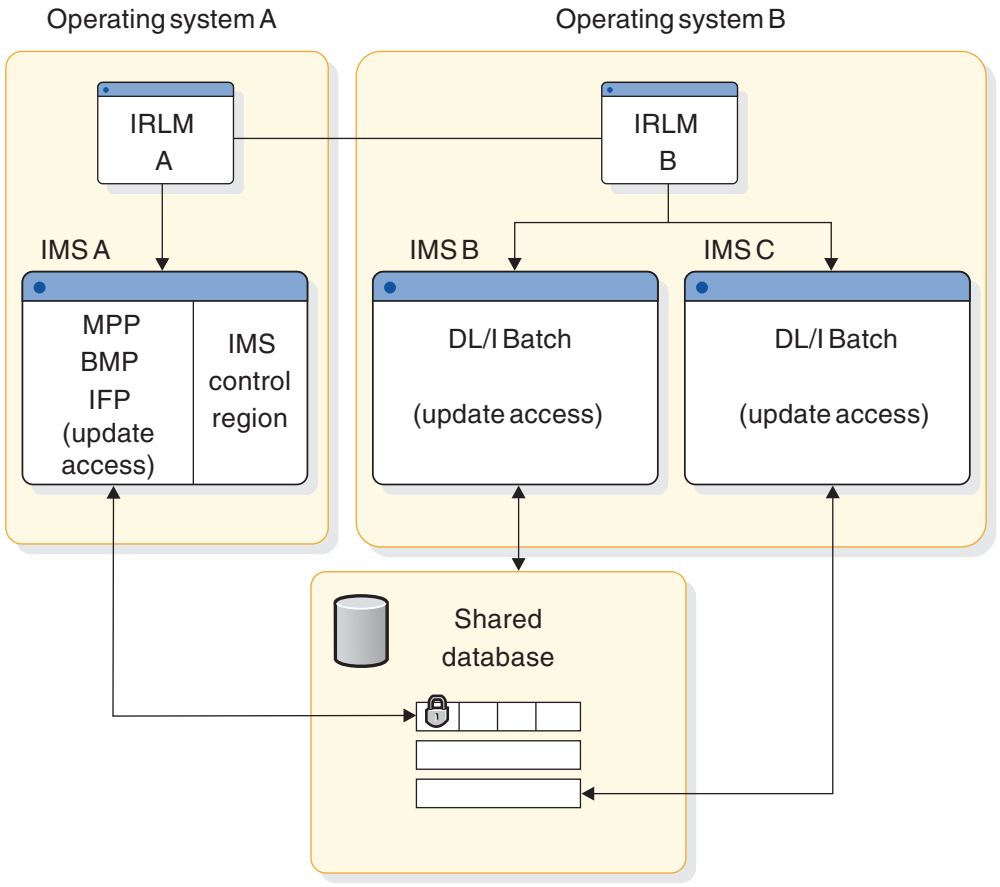


Figure 31. Inter-CPC block-level data sharing

System A

IMS system definition statements include:

```
IMSCTRL     IMSID=IMSA,IRLMNM=RLMA
DATABASE    DBD=SHRDB,ACCESS=UP
```

PSB definition statements include:

- `PCB         TYPE=DB,DBDNAME=SHRDB,PROCOPT=A`

System B

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSB,IRLMNM=RLMB,SYSTEM=(VS2,BATCH)
DATABASE  DBD=SHRDB,ACCESS=UP
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=A
```

System C

IMS system definition statements include:

```
IMSCTRL   IMSID=IMSC,IRLMNM=RLMB,SYSTEM=(VS2,BATCH)
```

PSB definition statements include:

```
PCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=APCB       TYPE=DB,DBDNAME=SHRDB,PROCOPT=A
```

IMS DBCTL environment

Sample stage 1 system definition input for an IMS DBCTL environment is provided in this topic.

Sample stage 1 input specifications for the DBCTL environment

The following shows sample stage 1 input specifications for the DBCTL environment:

```
                                                      Column
                                                        72
         IMSCTRL SYSTEM=(VS/2,(ON-LINE,DBCTL),390),      X
               MAXREGN=(20,52K,A,A),                     X
               MCS=(2,7),DESC=7,MAXCLAS=1,IMSID=IMSA,    X
               CMDCHAR=4
*
         IMSCTF SVCNO=(,203,202),                        X
               LOG=(DUAL,MONITOR),                       X
               RDS=(3380,4096),                          X
               CPLOG=500000
*
*        DEFINE SYSTEM BUFFERS
*
         BUFPOOLS PSBW=60000,DMB=10000,                  X
               SASPSB=(20000,80000)
*
*        DEFINE FAST PATH OPTIONS (IGNORED BY IMS)
         FPCTRL OTHREAD=10,BFALLOC=(5,20,1024)
*
*        DEFINE DL/I DATABASES
*
         DATABASE RESIDENT,DBD=DI21PART
*
*        DEFINE SAMPLE APPLICATIONS
*
         APPLCTN PSB=DFHSAM04,SCHDTYP=PARALLEL
         APPLCTN PSB=DFHSAM05,SCHDTYP=PARALLEL
*
         IMSGEN ASM=(HLASM,SYSLIN),                      X
               ASMPRT=ON,                                X
               LKPRT=(XREF,LIST),                        X
               LKSIZE=(880K,64K),                        X
               LKRGN=4096K,                              X
```

```
      SUFFIX=1,                                    X
      SURVEY=NO,                                   X
      SYSMSG=TIMESTAMP,                            X
      OBJDSET=IMS.OBJDSET,                         X
      USERLIB=IMS.SDFSRESL,                        X
      PROCLIB=(YES,),                              X
      NODE=(IMS,IMS,IMS),                          X
      JCL=(GENJOB,                                 X
      (1),                                         X
      PGMERID,                                     X
      A,                                           X
      (TIME=5,CLASS=K,NOTIFY=PGMERID)),            X
      SCL=(99)
END
```

Chapter 14. IMS Syntax Checker

The Syntax Checker is an ISPF application that helps you define, verify, and validate parameters and their values in the members of the IMS PROCLIB data set. Use the IMS Syntax Checker to avoid typographical and syntactical errors when modifying the parameter values for the IMS PROCLIB data set members.

The Syntax Checker:

- Reads in the IMS PROCLIB data set member.
- Displays the parameters and values.
- Allows modification of the parameter values.
- Helps migrate supported IMS PROCLIB data set members from one version of IMS to another.
- Displays default values for parameters.
- Verifies the validity of parameters and values.
- Saves changes back to the IMS PROCLIB data set member.

The following table lists the members of the IMS PROCLIB data set that Syntax Checker supports, along with the version for that support.

Table 43. IMS PROCLIB data set members and their versions supported by the Syntax Checker

IMS PROCLIB data set member	Version 9	Version 10	Version 11
BPE exit list member	yes	yes	yes
CSLDCxxx	no	no	yes
CSLDIxxx	no	no	yes
CSLOIxxx	yes	yes	yes
CSLRIxxx	yes	yes	yes
CSLSIxxx	yes	yes	yes
CQSIPxxx	yes	yes	yes
CQSSGxxx	yes	yes	yes
CQSSLxxx	yes	yes	yes
DFSCGxxx	yes	yes	yes
DFSDCxxx	yes	yes	yes
DFSDFxxx	no	yes	yes
DFSPBxxx	yes	yes	yes
DFSSQxxx	yes	yes	yes
DSPBIxxx	no	no	yes
HWSCFGxx	yes	yes	yes

The Syntax Checker provides an ISPF panel where you can add or change parameter values in the members. Online help is available at the parameter level and provides assistance in moving to a new IMS release by identifying new parameters as well as any obsolete parameters from the previous release.

© Copyright IBM Corp. 1974, 2015

The Syntax Checker checks parameters for valid values and accepts only valid values. It saves the parameters to appropriate members of the IMS PROCLIB data set in the correct format. The next time the control region is started, it will use the new values.

Requirement: The IMS control region must be restarted to activate the new parameter values.

The IMS IVP contains a Syntax Checker sample application that demonstrates how to use the IMS Syntax Checker. It demonstrates how to migrate a DFSPBxxx member of the IMS PROCLIB data set from an earlier version of IMS to a later version.

Recommendations:

- The IMS Syntax Checker does not perform security checking or bypass system security. You should protect your IMS data sets from unauthorized access by using RACF or an equivalent security product.
- To avoid unpredictable results with the Syntax Checker, use a screen size of 43 X 80.

Related concepts:

IMS installation verification program (IVP) overview (Installation)

"Controlling the IMS PROCLIB data set" on page 174

Related reference:

"The specification of IMS procedures" on page 445

Function keys and the Syntax Checker

Under ISPF, use the `FKA ON` command to display function keys, and use the `KEYS` command to set function keys.

The Syntax Checker panels and Help panels have function keys assigned through an ISPF keylist. Although ISPF allows you to turn off the use of keylists, you should leave keylists on if you use the Help panels. Some function keys continue to be active, but are not shown when FKA ON is used. Table 44 describes the function keys used for the Syntax Checker.

Table 44. Syntax Checker function key descriptions

Key	Description
F1=Help	Display Help information for panels and keywords.
F2=Split	ISPF split screen.
F3=Exit	Exit the current function. Returns back to the previous panel. If modifications have been made to the current member, the user is prompted to SAVE the member before returning back to the previous panel.
F6=Defaults	Display default values for keywords.
F7=Backward	Scroll backward. Pages backward through the list of parameters when the complete list of parameters cannot be displayed on one screen.
F8=Forward	Scroll forward. Pages forward through the list of parameters when the complete list of parameters cannot be displayed on one screen.
F9=Swap	ISPF swap screen command.
F10= Edit	Move cursor to Edit action bar.

Table 44. Syntax Checker function key descriptions (continued)

Key	Description
F12=Cancel	Cancel action and return to previous panel.

Starting the Syntax Checker

You can start the IMS Syntax Checker using an EXEC command or from the IMS Application Menu.

Start the Syntax Checker by issuing the following command in ISPF option 6:

```
EXEC 'HLQ.SDFSEXEC(DFSSCSRT)' 'hlq(HLQ)'
```

You can also start the Syntax Checker from the IMS Application Menu. The IMS Application Menu provides a common interface to IBM-supplied IMS applications that run on TSO and ISPF. The Syntax Checker remains running until you exit it manually.

Related Reading: For more information about the IMS Application Menu, see *IMS Version 11 Installation*.

Syntax Checker online help

Help for the IMS Syntax Checker is available by pressing the Help key (typically the F1 key) or using the help pull-down menu whenever you are using the Syntax Checker.

Help is available by pressing the Help key (typically the F1 key) or using the help pull-down menu whenever you are using the Syntax Checker.

- For any panel: Press the Help key to display a description of the panel.
- For the Syntax Checker Keyword Display Panel: To get specific keyword help, position the cursor anywhere on the keyword line and press the Help key.

Help information is provided in a scrollable panel. The help information describes the Syntax Checker panel or selected keyword parameter.

Using the Syntax Checker

On the main panel of the IMS Syntax Checker, you can enter the IMS PROCLIB data set name and the name of the member to be processed.

After you start the Syntax Checker, the main panel (Syntax Checker Member and Data Set Name panel) shown in Figure 32 on page 286 is displayed.

Important: The panels shown in this document are samples and, as such, might not completely match the actual panels that appear on your screen. The purpose of the panels here is to help guide your use of the Syntax Checker.

```
 File  Help
---------------------------------------------------------------
                 IMS Parameter Syntax Checker
Command ===>

Enter the name of the IMS proclib dataset and press enter.


 ISPF Library:
    Project  . . ________
    Group  . . . ________
    Type . . . . ________
    Member . . .          (Blank for member list)



 Other Partitioned Data Set: 'IMS
    Data Set Name  . . 'HLQ.PROCLIB(DFSPBSYN)'
    Volume Serial  . .           (If not cataloged)
```

Figure 32. Syntax Checker Member and Data Set Name panel

You use the Syntax Checker Member and Data Set Name panel to enter the IMS PROCLIB data set name and the name of the member to be processed. To enter the names, use the standard TSO and ISPF formats:

- Enter the data set and member names using the ISPF Library: Project, Group, Type, and Member fields.
- Enter the data set and member name using the *Other Partitioned Data Set: Data Set Name* field.

If you do not enter the member name, a member list is displayed. Select the member you want to process from the list.

If the data set is not cataloged, enter the data set's volume in the Volume Serial field.

In this example, the IMS PROCLIB data set name HLQ.PROCLIB (where HLQ is the high-level qualifier) and the member name DFSPBSYN are entered.

When you press **Enter** from the main panel, the Syntax Checker reads the input file and tries to determine the IMS release and type of control region. If the Syntax Checker cannot determine this information, one of the following panels displays:

- IMS Release and Control Region panel shown in Figure 33 on page 287.
- IMS Release panel shown in Figure 34 on page 287.

If the Syntax Checker is able to determine the information it requires from comments in the member, then the Syntax Checker Keyword Display panel shown in Figure 35 on page 288 displays.

IMS Release and Control Region panel

The IMS Release and Control Region panel provides the Syntax Checker with IMS release and control region information that is required to process the member correctly.

An example of the IMS Release and Control Region panel is shown in the following figure.

```
  File  Help
---------------------------------------------------------------------
                          IMS Parameter Syntax Checker
Command ===>

Enter the following information and press enter.

IMS Release  . . . . . . . . 1  1. IMS 12.1
                                2. IMS 11.1
                                3. IMS 10.1


Type of Control Region . . . 3  1. DBCTL Control Region
                                2. DCCTL Control Region
                                3. DB/DC Control Region
                                4. FDBR Region
                                5. DLI/DBB Batch
```

Figure 33. IMS Release and Control Region panel

When the Syntax Checker saves the member, it adds comment lines to the top of the member saving this information. The next time the Syntax Checker processes the member, this panel does not display.

The input fields for the IMS Release and Control Region panel include:

Field Description

IMS Release
Enter 1, 2, or 3 to specify the IMS release.

This is a required field.

Type of Control Region
Enter 1, 2, 3, 4, or 5 to specify the type of control region.

This is a required field.

After you type in the data and press **Enter**, the Syntax Checker Keyword Display panel is displayed. "Keyword Display panel" on page 288 shows the parameters that are currently specified for the member provided.

Related reference:

"Syntax Checker online help" on page 285

IMS Release panel

The IMS Release panel provides the Syntax Checker with IMS release information that is required to process the member correctly.

The IMS Release panel is shown in the following figure.

```
File  Help
-------------------------------------------------------------
                  IMS Syntax Checker
Command ===>

Enter the following information and press enter.

IMS Release  . . . . . . . .    1. IMS 12.1
                                2. IMS 11.1
                                3. IMS 10.1
```

Figure 34. IMS Release panel

Enter 1, 2, or 3 to specify the correct IMS release information. This is a required field.

After you type in the data and press **Enter**, the Syntax Checker Keyword Display panel is displayed. "Keyword Display panel" shows the parameters that are currently specified for the member.

Related reference:

"Syntax Checker online help" on page 285

Keyword Display panel

The Keyword Display panel of the IMS Syntax Checker displays the keywords and their values and indicates whether syntax errors exist.

The Keyword Display panel, shown in the following figure, is the main working panel. It displays the keywords and their values and indicates any errors.

```
  File  Edit  View  Help
------------------------------------------------------------------------------
                   IMS 12.1 Parameters for DB/DC
Command ===>
Member DFSPBSYN processed under IMS 12.1 (DB/DC )
Press enter (without other input) to check for errors.

Data Set Name  . . : HLQ.PROCLIB(DFSPBSYN)
IMS Release  . . . : 12.1

  Sel Codes: C = Comment D = Delete I = Insert P = Process / = Select


 Sel Keyword    Value             Description
  _  ALOT     = 9                 ETO Auto Logon Off Time
  _  AOIS     = A                 ICMD Security Option
  _  APPC     = Y                 Activate APPC/IMS (Y|N)
  _  APPLID2  = IMS2              VTAM Applid of XRF Alternate System
  _  ARC      = 99                Automatic Archive: 1-99, 0 - NOT Autom.
  _  ASOT     = 1000              ETO Auto Signoff Time
  _  AUTO     = Y                 Automatic Restart Desired (Y|N)
  _  CHTS     = 1000              Obsolete keyword
  _  CMDMCS   = N                 MCS/EMCS Command Option
  _  DBBF     = 1000              Number of Database Buffers
  _  DBFX     = 10                Num. DB Buffs available at FP Reg Start
  _  DBRCNM   = DBCPROC           DBRC Problib Member Name
```

Figure 35. Keyword Display panel

Related reference:

"Syntax Checker online help" on page 285

Display options for the Keyword Display panel

The Syntax Checker provides several display options that you can select from the View pull-down menu.

The following are the options:

Select Display
: Display all keywords with values. These keywords are saved when the member is saved.

All Display
: Display all possible keywords. Only keywords with values are saved when the member is saved.

New Display
Display all new keywords in this release of IMS. These keywords are saved only if they have values when the member is saved.

The status of the member determines whether keywords are displayed the first time the panel is displayed:

- If the member is new or empty, a list of all possible keywords for the member is displayed (All Display). The list of keywords is shown in alphabetic order.
- If the member is not empty or new, the list contains the current keywords defined in the member (Select Display). The list is shown in custom order. You can obtain a display of all possible keywords for the member by using the View pull-down Display All option. The customer order display and save feature will be delivered through the IMS service process.

To display the default values for parameters, press the F6 function key to toggle between displaying and not displaying defaults. The default values are displayed in the description field of the parameter. When default values are displayed, the description or comment appears to the right of the field. Shifting the display to the right or left is not supported.

Panel field descriptions for the Keyword Display panel

Help information for the IMS Syntax Checker is provided. This topic describes the fields on the Keyword Display panel.

Field Description

Panel Title
The IMS release and control region.

Data Set Name
The IMS PROCLIB data set and member name.

IMS Release
The release of the IMS system to which the parameters apply.

Sel An input field that allows you to:

- Select with an I to insert a keyword. The insert panel pops up.
- Select with a C to process the keyword using the Comments command.
- Select with a D to process the keyword using the Delete command.
- Select with a P to interrupt the processing of the current member and begin processing the member identified by the selected parameter. For more information about this option, see "Interrupting processing to process other members on the Keyword Display panel" on page 294.
- Select with a / (slash) to process the keyword using the Action pull-down option.
- Select with a + (plus) displays the keyword on multiple lines. When a keyword is expanded, the subparameters are displayed in a predefined order.
- Select with a - (minus) displays the keywords on one line. As much of the keyword as fits on one line is displayed followed by "...".

Keyword
The name of the keyword.

The keywords are color-highlighted as follows:

Green Normal color for a keyword with a value.

Blue A new keyword in this release.

Yellow
Warning: Read the help text.

Red Keyword Error

Turquoise
Keyword is a template. The keyword has no value and is not saved. These keywords are only displayed in the DISPLAY ALL panel.

Value The modifiable field which contains the value assigned to the keyword parameter. You can position the cursor on the input field and press the help key to get more information about the keyword parameter.

The keywords are color-highlighted as follows:

Turquoise
Value is correct.

Red Value error.

Description/Comment
Provides a short description of the keyword parameter or a user-provided comment. You can modify both the description and the comment. A modified description becomes a user comment. If the user comment is set to all blanks, the description is displayed. The user comment is written to the member when the member is saved, but the description provided by the Syntax Checker is not.

The user-provided comment can be a maximum of 42 characters, but sometimes it is much shorter than 42 characters because the keyword and values take more spaces than usual. Therefore, the maximum characters the user can provide depends upon the keyword. Comments are not supported by all members. See "Specifying comments in IMS PROCLIB data set members processed by the Syntax Checker" on page 291 for information about specific members that support comments.

Note: For the DFSPBxxx members, an equal sign (=) is not allowed in a user-provided comment. If an equal sign is found in a user-provided comment, the Syntax Checker changes it to a dash (-) and displays message `DFSI937I Equal sign in same line comment changed to dash`.

Syntax Checker error checking

After modifying the member, press the **Enter** key without making any other modifications. Syntax value checking is performed.

If there are errors, the first keyword with an error moves to the top of the display and an error message is displayed. The error must be corrected before the next error can be displayed.

For example, in Figure 35 on page 288, the Syntax Checker Keyword panel contains deliberate errors labeled as `Unknown Keywords`. On the screen, each error shows up in red on the screen and is displayed at the top of the list of keywords.

The first error (APPLID=IMS1) is a typographical error. Adding a 1 to the end of APPLID makes it a valid parameter (APPLID1).

The second error (Keyword XXXXX) is not a valid keyword. To resolve this error, delete the keyword. In this example, a "D" (delete) select code is typed into the Sel field next to the XXXXX.

The third error is that the AUTO parameter has an erroneous value of X instead of a correct value of Y or N. If you press the **Enter** key with no other input, the AUTO parameter is displayed on the top line of the display and an error message describing the error is displayed.

When all errors are resolved and you have modified the member as required, you can save the member to the originally selected PROCLIB and member or to a different PROCLIB and member.

Specifying comments in IMS PROCLIB data set members processed by the Syntax Checker

IMS allows comments to be included in most IMS PROCLIB data set members. Special considerations exist for comments in IMS PROCLIB data set members processed by the Syntax Checker.

IMS allows comments to be included in most IMS PROCLIB data set members. The comments can be either full-line comments or comments that appear on the same line as the parameters. The style of comment depends upon the specific member being processed.

In general, full-line comments are indicated by an asterisk (*) or a slash-asterisk (/*) in column 1. Each comment is associated with the parameter keyword that follows the comment. When the file is written, the comments precede the keyword. You can add, edit, or delete comments in the following ways:

- Use the select option, C, to select the keyword associated with, or to be associated with, the comment. Press **Enter**.

  The panel redisplays with a blank comment line above the selected keyword. Add your comment to the blank line.
- The comment can be modified at any time.
- To delete a comment, enter the select option, D, next to the comment line. Press **Enter**.

The following table describes considerations for specifying comments for specific members.

Table 45. Special considerations for comments in IMS PROCLIB data set members processed by the Syntax Checker

Members	Considerations when specifying comments
• BPE exit list member • CQSIPxxx • CQSSGxxx • CQSSLxxx • CSLDCxxx • CSLDIxxx • CSLOIxxx • CSLRIxxx • CSLSIxxx • DFSCGxxx • DFSDFxxx • DFSPBxxx • DSPBIxxx • HWSCFGxx	Comments can be specified on a separate line, or they can be specified on the same line as parameters. A comment is separated from the parameter by one or more blanks. Comments can contain any characters except the equal sign (=).
DFSDCxxx	Comments must be specified on a separate line; comments are not allowed on the same line as parameters. You cannot edit the description of the parameter.
DFSSQxxx	Comments are supported in IMS Version 10 and later.

Action pull-down options on the Keyword Display panel

Use the action pull-down options on the Keyword Display panel to select which file, edit, and view action to take.

Action pull-down options are located at the top of the panel. Selecting an Action pull-down option displays the options available for the action pull-down.

File action options:

Save The keywords are stored back into the originally selected PROCLIB member. You can save the keywords in alphabetical order.

Save As
The keywords are stored in a specific data set and member. You can save the keywords in alphabetical order.

Change Release
This option changes the IMS release of the parameter member being processed as follows:

1. If any changes have been made to the current member, you are given an option to save the member before changing the release.
2. The member is reprocessed under the new release, identifying any value errors and invalid or obsolete keywords.

Edit action options:

Comments
Adds a comment to keywords that have been selected with the / operand.

Delete Deletes from the display keywords that have been selected with the / operand.

Delete All
Deletes the ALL keywords from the display.

View action options

Display All
Changes the keyword display to include all possible keywords for the selected IMS release and control region type. These keywords are saved only if they have values when the member is saved.

Display Selected
Changes the keyword display to include only the keywords that have a value. This is the list of parameters that are saved when the save command is issued.

Display New
Changes the keyword display to include only the keywords that are new for the IMS release. These keywords are saved only if they have values when the member is saved.

+ (expand)
Displays the keyword on multiple lines. When a keyword is expanded, the subparameters are displayed in a predefined order.

- (contract)
Displays the keywords on one line. As much of the keyword as fits on one line is displayed followed by "...".

Alphabetical Order
Specifies that the keywords are shown in alphabetical order, sorted by keyword name.

Custom Order
Specifies that the keywords are displayed in custom order.

The type and status of the member determines whether keywords are displayed the first time the panel is displayed:

- If the member is order-dependent, the list of keywords is displayed according to the predefined order.
- If the member is new or empty, a list of all possible keywords for the member is displayed (**Display All**). The list of keywords is shown in alphabetical order, unless the member is order-dependent.
- If the member is not empty or new, the list contains the current keywords defined in the member (**Display Selected**). The list is shown in custom order. You can obtain a display of all possible keywords for the member by using the View pull-down **Display All** option.

To display the default values for parameters, press the F6 function key to toggle between displaying and not displaying defaults. The default values are displayed in the description field of the parameter. When default values are displayed, the description or comment appears to the right of the field. Shifting the display to the right or left is not supported.

Related concepts:

"Saving processed members on the Keyword Display panel" on page 295

Inserting keywords into the Keyword Display panel

To insert a keyword into the display, type an "I" (insert) in the Sel field and press **Enter**. A pop-up window that lists the possible items that can be inserted appears.

New keywords are added in alphabetic order. See the following figure for an example of the insert pop-up window.

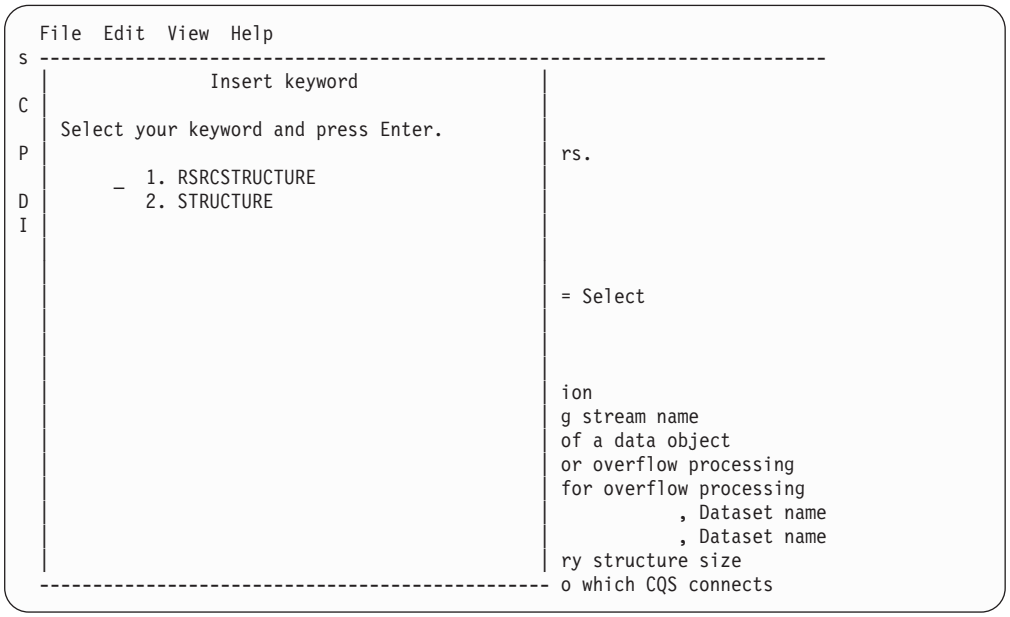


Figure 36. Example of insert pop-up window for Syntax Checker Keyword Display panel

Interrupting processing to process other members on the Keyword Display panel

While using the Keyword Display Panel, you can select the P option to interrupt processing of the current member and begin processing another member.

The P option is valid on any keyword whose value is used to identify a member in the IMS PROCLIB data set. The keyword displays as underlined for easy identification. The keyword must have a value.

The selected member is processed under the current IMS release. When processing of the selected member is complete (F3), processing of the interrupted member returns.

If the member does not exist, the panel shown in Figure 37 is displayed to ask if a new member should be created or if the member should be processed in a different data set. If neither action is appropriate, the user can choose to cancel processing of the member.

```
IMS Syntax Checker
Command ===>

The Member DFSDC001 does not exist in dataset 'xxx.yyy.zzz'
Select one of the following options:

Option: __
 1. Create a new member in current dataset
 2. Cancel processing of the member
 3. Process member in different dataset:
   _____________________________________ (Dataset name)
```

Figure 37. Member Does Not Exist panel

If the member is not one of the members processed by Syntax Checker, the panel shown in Figure 38 is displayed to ask if the user wants to edit or create the member in the current data set, edit or create the member in a different data set, or cancel processing of the member.

```
IMS Syntax Checker
Command ===>

The Member DFSDC001 is not processed by Syntax Checker.
Select one of the following options:

Option: __
1. Edit/Create member in current dataset
2. Cancel processing of the member
3. Edit/Create member in different dataset:
   ____________________________________________ (Dataset name)
```

Figure 38. Member is Not Processed by Syntax Checker panel

Saving processed members on the Keyword Display panel

While using the Keyword Display Panel, the processed member can be saved to the originally selected PROCLIB and member or to a different PROCLIB and member.

To save the member, you use one of the Save or Save As options. The parameters are stored in the PROCLIB member as follows:

- Only the parameters that are displayed in the Display Selected keyword panel are stored. These parameters are either read in from the member or assigned a value during Syntax Checker processing of the member.
- For members that support comments:
  - Comments are stored in front of their keyword. Syntax Checker also supports same-line comments. Same-line comments are stored on the same line as the keyword.
  - Syntax Checker adds the following informational comments to the top of the member:

```
IMS Version:         *<version> 12.1
Last Modified date:  *<date> 2011/09/25
Last Modified time:  *<time> 15:34
Modifying userid:    *<sysuid> GR8ONE
```

Related concepts:

"Save Member prompt panel"

Related reference:

"Action pull-down options on the Keyword Display panel" on page 292

Save Member prompt panel

On the Save Member prompt panel of the IMS Syntax Checker, you are notified of any problems before the Syntax Checker saves the member, and you can either save or not save the member.

The Save Member Prompt panel shown in Figure 39 on page 296, notifies you of an issue or problem before the Syntax Checker saves the member or continues without saving the member. You are prompted to SAVE or NOT SAVE the member currently being processed.

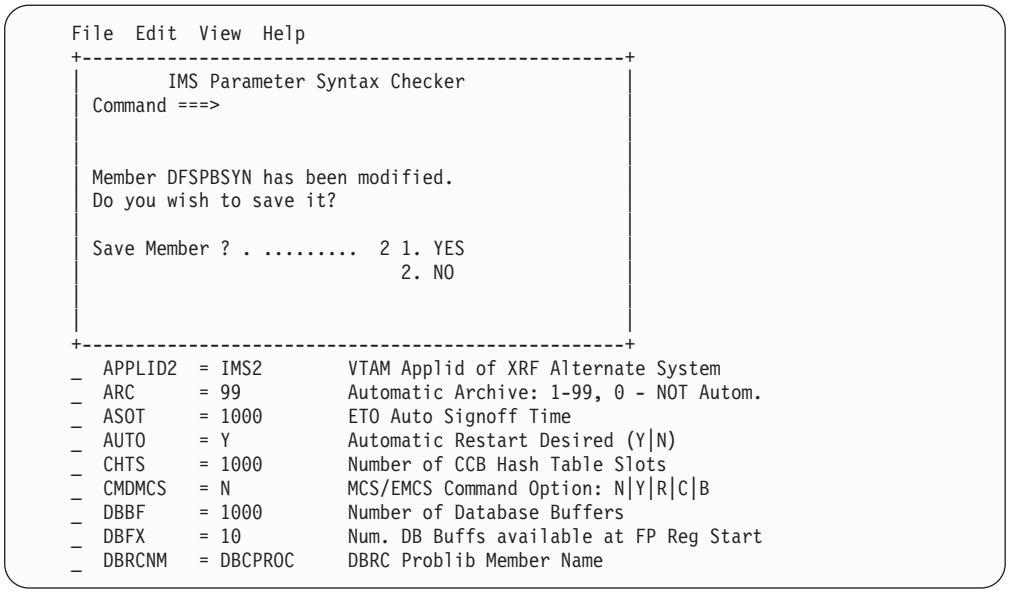


Figure 39. Save Member Prompt panel

The input field for this panel includes:

Save Member?

- Enter 1 to save the member.
- Enter 2 to continue without saving the member.

The **Save Member** field is a required field.

Related concepts:

"Saving processed members on the Keyword Display panel" on page 295

SAVE AS prompt panel

On the SAVE AS prompt panel, you specify the data set name and the member name in which to save the current keywords being processed.

The SAVE AS Prompt panel, shown in Figure 40 on page 297 provides the Syntax Checker with the data set name and the member name in which to save the current keywords being processed. Enter the data set name and member name in the standard TSO and ISPF formats.

```
File  Help
   ------------------------------------------------------------------------
                        Save IMS Parameter
   Command ===>

   Enter the "SAVE AS" dataset and member name then press enter.

    Current Data Set:
      Data Set Name : HLQ.PROCLIB(DFSPBSYN)
      Volume Serial :

    SAVE AS Data set and Member:

    ISPF Library:
      Project  . . _______
      Group  . . . _______
      Type . . . . _______
      Member . . .

    Other Partitioned Data Set:
      Data Set Name  . . 'HLQ.PROCLIB(DFSPB888)'
      Volume Serial  . .
```

Figure 40. SAVE AS Prompt panel

- Enter the data set and member names using the ISPF Library: Project, Group, Type, and Member fields.
- Enter the data set and member names using the Other Partitioned Data Set: Data Set Name field.

If the data set is not cataloged, enter the data set's volume in the Volume Serial field.

For more information about how parameters and comments are stored when you save a member, see "Saving processed members on the Keyword Display panel" on page 295.

IMS Parameter Syntax Checker panel

On the IMS Parameter Syntax Checker panel in the IMS Syntax Checker, you select which release of IMS the Syntax Checker should use.

On the IMS Parameter Syntax Checker panel, shown in the following figure, you can select which release of IMS the Syntax Checker uses to process the keywords.

The Change Release option can be found on the File pull-down menu.

Keywords are processed again under the new release to identify any value errors and invalid or obsolete keywords.

The input field for this panel includes:

IMS Release
Enter 1, 2, or 3 to specify the IMS release.

This is a required field.

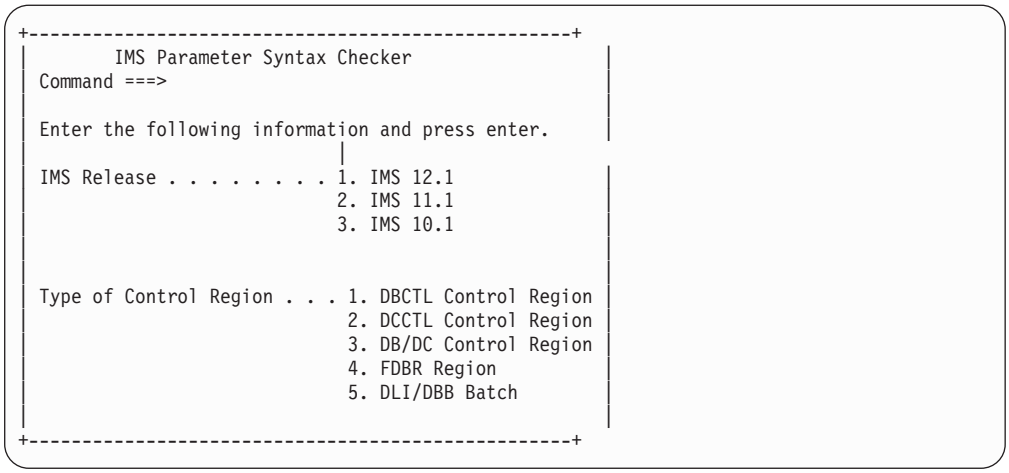


Figure 41. IMS Parameter Syntax Checker panel

Related reference:

"Syntax Checker online help" on page 285

Chapter 15. Macros used in IMS environments

These topics provide general information about using IMS macros.

Related reference:

"Selecting the appropriate macros to define your system" on page 5

Reference information for using IMS macros

Use IMS macro statements to generate a sequence of assembler statements from a single source statement. IMS provides a set of macro statements for system definition.

For example, to describe program resource requirements for your application programs, use the APPLCTN macro. To define characteristics of VTAM nodes and non VTAM communication, use the TERMINAL macro.

The APPLCTN, DATABASE, RTCODE, and TRANSACT macros are used to define application programs, databases, Fast Path route codes, and transactions. These IMS resources can be changed, added, and deleted by using either dynamic resource definition (DRD) or online change. The online change process requires that some system definition tasks be performed offline. The DRD process, which uses CREATE, DELETE, and UPDATE commands to define these resources, does not require any offline tasks. When DRD is enabled (by specifying MODBLKS=DYN in the DFSCGxxx PROCLIB member), online change for resources in the IMS.MODBLKS data set, but not for other resources, is disabled. Likewise, when DRD is disabled (MODBLKS=OLC in the DFSCGxxx PROCLIB member), commands that are used for DRD, except for some types of the UPDATE command, are not supported. The UPDATE commands that change status or runtime values are supported.

System definition macros and their uses

The following table contains a list of IMS system definition macros. "Selecting the appropriate macros to define your system" on page 5 describes which macros and macro keywords can be modified by which type of system definition.

© Copyright IBM Corp. 1974, 2015

Table 46. System definition macros and their uses

Type of macro statement	Macro name	Resources each macro defines
System configuration macros	"BUFPOOLS macro" on page 311	Buffer pool sizes
	"FPCTRL macro" on page 336	Fast Path options
	"IMSCTF macro" on page 337	System control integrity factors
	"IMSCTRL macro" on page 340	Library naming, JCL options, and ETO feature
	"IMSGEN macro" on page 351	z/OS system and control options
	"MSGQUEUE macro" on page 367	Message queue device and message sizes
	"SECURITY macro" on page 386	Security options. IMS Version 12 is the last version of IMS to support the SECURITY macro. You can use initialization parameters to specify most of the SECURITY macro keyword values. For information about using initialization parameters for security, see Controlling security during system startup for DB/DC and DCCTL (System Administration).
Database and application macros	"APPLCTN macro" on page 304	Program characteristics and PSB names
	"DATABASE macro" on page 323	DBD names
	"RTCODE macro" on page 384	Fast Path message routing
	"TRANSACT macro" on page 420	Transaction processing options

Table 46. System definition macros and their uses (continued)

Type of macro statement		Macro name	Resources each macro defines
Data communication macros	Non-VTAM device[1]	"COMM macro" on page 315	General communication options
		"LINE macro" on page 363	Communication line to IMS
		"LINEGRP macro" on page 365	DD names for communication devices
		"NAME macro" on page 380	LTERM definition
		"TERMINAL macro" on page 391	Physical characteristics of terminal
	Multiple System Coupling	"MSLINK macro" on page 372	Logical Link names
		"MSNAME macro" on page 375	Partner systems
		"MSPLINK macro" on page 376	Physical link names
		"NAME macro" on page 380	LTERM definition
	VTAM	"COMM macro" on page 315	General communication options
		"NAME macro" on page 380	LTERM definition
		"SUBPOOL macro" on page 390	Set of LTERMs for a static ISC session
		"TERMINAL macro" on page 391	Physical characteristics of a static terminal
		"TYPE macro" on page 441	VTAM terminal type
		"VTAMPOOL macro" on page 443	Multiple systems with ISC sessions

[1] See the LINEGRP UNITYPE= keyword for detailed information.

Resource naming rules

These rules and restrictions apply to all IMS macros.

- Names cannot include a blank, comma, period, hyphen, or equal sign.
- All PSB and DMB names must begin with an alphabetic character (alphabetic = A through Z, #, $, and @) followed by zero to seven alphanumeric characters.
- The PSB name, DBF#FPU0 is a reserved name.
- For route codes, DBFDFRT1 is a reserved name.
- Logical terminal names and transaction codes must be one to eight alphanumeric characters in length. (Alphanumeric = A through Z, #, $, and @, 0 - 9.) Each name within the set comprising LTERM names and transaction codes must be unique.
- User IDs that are passed to RACF must be one to eight alphanumeric characters, that is, letters A through Z, digits 0 - 9, and the national characters #, $, and @.

For ETO terminals, if the user ID is used to build the LTERM structure, the user ID must be one to eight alphanumeric characters.

- Node names must be unique among themselves, but can duplicate LTERM names, transaction codes, or subpool names.
- Subpool names must be unique among themselves, but can duplicate LTERM names, transaction codes, or node names.
- IMS null words must not be used as resource names (FOR, TO, ON, AFTER, and SECURITY). Also, resource names must not begin with DFS (except for DFSIVPxx and DFSSAMxx, and DFSCONE for transactions) or DBCDM; nor should WTOR, MSDB, SDB, DBRC, BASICEDT, or ISCEDT and its aliases be used as a resource name.
- Command keywords must not be used as resource names.

  Related reading: For more information about command keyword restrictions, see List of reserved words (Commands).
- Symbolic names can be used to define 3270 and SLU Type 2 devices. See the description of the TYPE= keyword of the TERMINAL macro statement for the format of these names. A symbolic name must be used to define the format of any 3270 display device with a screen size other than 480 or 1920 characters.
- Each IMS macro statement can appear in an IMS system definition a limited number of times. The following table shows the maximum number of times each macro statement can occur.
- The maximum number of IMS resources that can be defined is a design limit (that is, a theoretical limit). You should not assume that your installation can successfully define the maximum number of IMS resources that is shown in the following table. The actual limit on this number is influenced by many limiting factors in your hardware and software configuration.
- Resource names must not end with the IMS ID or the CRC character that was chosen by coding the CMDCHAR parameter of the IMSCTRL macro. The last characters of a command are checked during multisegment command processing. If the characters match the IMS ID or the CRC, the command is assumed to be a multisegment command. Command processing waits for the rest of the command.

  For example, if you define a NODE whose name ends with “IMS ID IMSA”, and you issue the command IMSADIS NODE xxxxIMSA, the command does not run. The ending “IMSA” in the NODE name indicates a multisegment command. To run this command, enter:

  ```
  'IMSADIS NODE xxxxIMSA. '
  ```

Table 47. Maximum occurrences of each IMS system definition macro statement

Macro statement	Maximum occurrences
APPLCTN	999,999
BUFPOOLS	1
COMM	1
DATABASE[1]	32,700
FPCTRL	1
IDLIST	1000
IMSCTF	1
IMSCTRL	1
IMSGEN	1
LINE	1000

Table 47. Maximum occurrences of each IMS system definition macro statement (continued)

Macro statement	Maximum occurrences
LINEGRP	676 - (Number of MSPLINK macros specified)
MSGQUEUE	1
MSLINK	676 or 999[6]
MSNAME	676
MSPLINK	676 or 999[6] - (Number of LINEGRP macros specified)
NAME	999,999 - (Number of switched line macros specified)
RTCODE	999,999
SECURITY	1
SUBPOOL	999,999 - (Number of TERMINAL and STATION macros specified)
TERMINAL[2]	200,000
TRANSACT	999,999
Transaction Edit Routines[3]	255
TYPE	200,000
User routines[4]	200
VTAMPOOL[5]	1

Notes:

1. In IMS execution, database allocation is limited to the number of DD names that z/OS allows.
2. Because of the size of the IMS control block, the maximum number of static terminal resources that can be defined is approximately 200,000.

   Recommendation: If your system includes many terminals, use the ETO feature. By using the ETO feature, you can manage terminals more effectively during IMS online activity.
3. Transaction Edit routines are specified in the EDIT operand of the TRANSACT macro. Although the limit of TRANSACT statements is 999,999, only 255 transaction edit routines are accepted.
4. *User Routines*: Refer to the user-supplied physical terminal output and input edit routines (specified in the EDIT operand of the LINEGRP and TYPE macros).
5. VTAMPOOL is used once for each set of LU 6.1 subpools defined. It can be used multiple times within a system definition to define multiple sets of LU 6.1 subpools.
6. The MSLINK and MSPLINK maximum occurrences are 999 for a normal system definition (SYSGEN) and 676 for a large system definition (LGEN).

Coding conventions

All IMS installation information uses the following conventions in coding macros and examples:

- Uppercase letters, stand-alone numbers, and punctuation marks must be coded exactly as shown. The only exceptions are subscripts, which are not coded.
- Lowercase letters, words, and associated numbers represent variables for which specific values can be substituted.

- A space or the lowercase “b” indicates one blank position.
- If an alternative item is underlined, it is the default. IMS assumes that if none of the items is coded, the underlined item is automatically the choice.
- You must specify positional parameters in the order shown.
- You can specify keyword parameters in any order.

APPLCTN macro

The APPLCTN macro allows you to define the program resource requirements for application programs that run under the control of the IMS DB/DC environment, and for application programs that access databases through DBCTL.

An APPLCTN macro combined with one or more TRANSACT macros defines the scheduling and resource requirements for an application program. Using the APPLCTN macro, you only describe programs that operate in message processing regions, Fast Path message-driven program regions, batch message processing regions, or CCTL threads. You do use the APPLCTN macro to describe application programs that operate in batch processing regions.

Requirement: When you define programs or transactions (using the APPLCTN and TRANSACT macros) to an IMSplex using shared queues, you must ensure that the programs and transactions are defined with the same attributes on all IMS systems in the IMSplex. If the attributes are defined differently on different IMS systems in the IMSplex, results can be unpredictable. For example, a transaction defined as conversational on one IMS and non-conversational on another IMS causes unpredictable results when messages for that transaction are put on the shared queues.

The APPLCTN macro is optional in DBCTL, DB/DC, and DCCTL environments. If you do not include this macro during stage 1 system definition, no warning is issued, and it is assumed that you define your application programs dynamically using CREATE PGM and UPDATE PGM commands.

This topic includes the following information:

- “Dynamic definition”
- “Syntax” on page 305
- “Positional parameters” on page 306
- “Keyword Parameters” on page 308

Dynamic definition

To define or change program resource requirements for application programs dynamically, you can use the CREATE PGM command and the UPDATE PGM type-2 command. The following table compares the APPLCTN macro keywords and the equivalent CREATE and UPDATE command keywords used for dynamic definition. Default values are shown in **bold**.

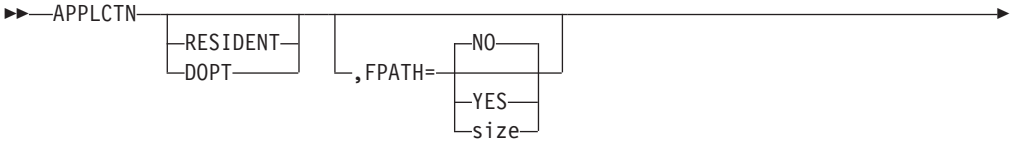
Table 48. APPLCTN macro keywords and the equivalent CREATE and UPDATE command keywords used in dynamic definition

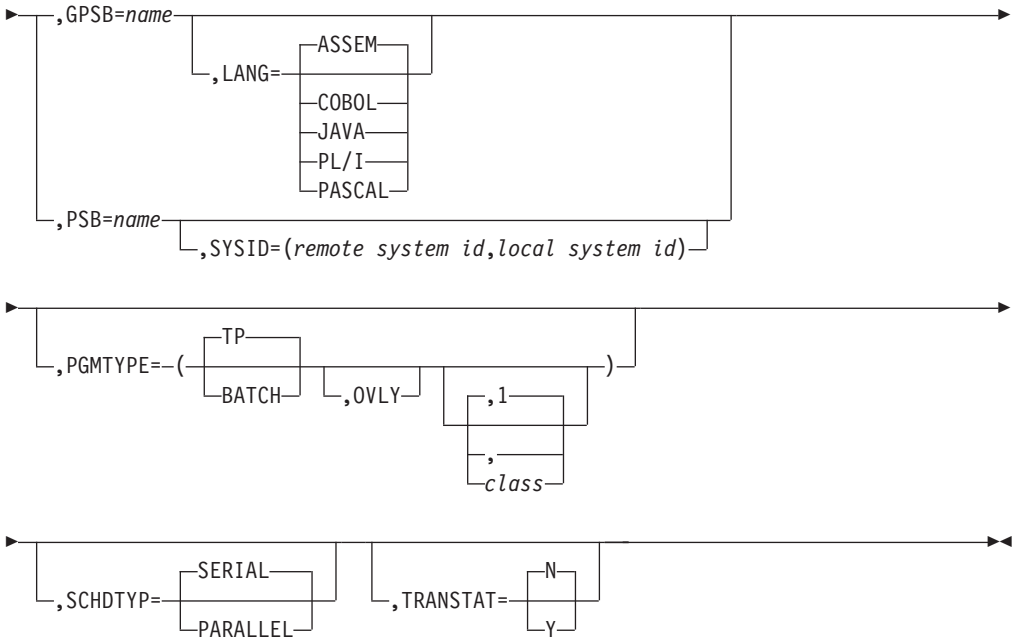
APPLCTN macro keyword	CREATE | UPDATE PGM keyword equivalent	Usage comments
PSB=*name*	NAME(*name*)	

Table 48. APPLCTN macro keywords and the equivalent CREATE and UPDATE command keywords used in dynamic definition (continued)

APPLCTN macro keyword	CREATE | UPDATE PGM keyword equivalent	Usage comments
RESIDENT | DOPT	RESIDENT(**N** | Y) DOPT(**N** | Y)	
FPATH=(**NO** | YES | *size*)	FP(**N** | E)	The size variable is not supported on the CREATE PGM and UPDATE PGM commands. Use the keyword variable EMHBSZ(size) on the CREATE TRAN command instead.
GPSB=*name*	NAME(*name*) GPSB(**N** | Y)	
LANG=(**ASSEM** | COBOL | PL/I | JAVA | PASCAL)	LANG(**ASSEM** | COBOL | PL/I | JAVA | PASCAL)	
PGMTYPE=(**TP** | BATCH, OVLY, **1** | *class*)	BMPTYPE(Y | **N**)	The class variable is used on the CREATE TRAN and UPDATE TRAN commands.
SCHDTYPE=**SERIAL** | PARALLEL	SCHDTYPE(SERIAL | **PARALLEL**)	• SERIAL is the default on the APPLCTN macro. • PARALLEL is the default on the CREATE and UPDATE commands.
SYSID=(*remote system ID*, *local system ID*)		The SYSID keyword is not supported on the CREATE PGM or UPDATE PGM command. Use the keyword variables SIDL(*localsysid*), SIDR(*remotesysid*), or MSNAME(*msname*) on the CREATE TRAN and UPDATE TRAN commands instead.
TRANSTAT=**N** | Y	TRANSTAT(**N** | Y)	The TRANSTAT keyword is optional. If a value is not specified for the TRANSTAT keyword, the system default is used.

Syntax





Positional parameters

The APPLCTN macro includes two positional parameters, RESIDENT and DOPT, which are mutually exclusive.

RESIDENT
Specifies that the PSB associated with this application program is to be made resident during system initialization.

DOPT
Specifies that the PSB associated with this application program is to be located dynamically.

If you specify DOPT, you cannot specify SCHDTYP=PARALLEL. In addition, when the IMS control region executes:

- Initialization does not perform a BLDL on the PSB associated with this application program. Thus the PSB does not need to be in any data set defined by the ACBLIB DD statement until it is required to process a transaction. This is not true, however, of the DBDs that the PSB might reference. All DBDs that are to be used online must be available during initialization in a data set defined by the ACBLIB DD statement. PSBs referencing DBDs that are missing during initialization are not scheduled.
- Each time the program associated with this PSB is scheduled, a BLDL is performed, and the latest copy of the PSB is located. A BLDL is not performed for associated DBDs, so the DBD cannot be modified until the system is initialized again or until online change is used.
- When the program terminates, the PSB is deleted from the PSB pool as part of the termination process.

Usage information

Neither RESIDENT or DOPT is the default parameter. Rather, if neither RESIDENT or DOPT is selected, IMS system initialization causes a BLDL to be performed on

the PSB associated with the application being defined. The PSB is not made resident (that is, loaded from the ACBLIB) until the application is scheduled.

To provide greater flexibility in your control of the library containing the dynamic PSBs, the following restrictions are imposed:

- The PSB must reside in a library other than the primary ACBLIB and must be concatenated to it.
- The concatenated library containing the PSB must be in ACBLIB format.
- The IMS.PSBLIB data set cannot be used.

If the BLDL performed at the scheduling of the PSB determines that the PSB resides in the first concatenation of the IMSACB DD statement set, the PSB is stopped and an error message is sent to the master terminal. The PSB is not scheduled. If the dynamic PSB is added to the concatenation of the IMS.ACBLIB, causing the data set to be expanded to a secondary extent, that PSB is not available to the online system until the ACBLIB is closed and reopened by IMS.

For MSC remote applications, the APPLCTN macro provides documentation and a reference for its transactions. It does not generate a control block. The PSB need not be generated in systems where it is used as a remote reference. The PSB name is defined here for documentation purposes.

To dynamically reassign a transaction from remote to local processing, a control block must be present in the local system.

Thus, if dynamic reassignment is desired, a transaction must be defined as local. That is, it must be under a local application having the same PSB name as the remote application (using the APPLCTN macro without a SYSID parameter). If dynamic reassignment is not desired, local definition is unnecessary.

Online changes to both the RESIDENT and DOPT options can be made using a MODBLKS system definition. However, PSBs associated with new application programs defined as RESIDENT (that is, changed from NONRESIDENT or DOPT to RESIDENT) during a MODBLKS system definition are not made resident until the next restart of IMS. Until that time, they are to be treated as nonresident. Online changes made to already-resident PSBs (using an ACBGEN) cause the PSB to be treated as nonresident until the next IMS restart.

MPPs scheduled against dynamic PSBs are not allowed to go through the quick reschedule process or become pseudo WFIs.

Requirement: If, after IMS initialization, an ACB that was originally generated as LANG=*non-Java* and DOPT, is changed to LANG=JAVA and DOPT (and IMS is not restarted), you must ensure that both of the following regions are available the first time IMS tries to schedule the affected transaction:

- An MPP region that has the same class as the application
- A JMP region that has the same class as the application

The reverse situation is also true. If, after IMS initialization, an ACB that was originally generated as LANG=JAVA and DOPT, is changed to LANG=*non-Java* and DOPT (and IMS is not restarted), you must ensure both of the following regions are available the first time IMS tries to schedule the affected transaction:

- A JMP region that has the same class as the application
- An MPP region that has the same class as the application

Keyword Parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

FPATH=

Specifies whether (YES) or not (NO) this is a Fast Path-exclusive application program. FPATH=size, which determines the EMH buffer size required to run the transaction, overrides the EMHL execution parameter and implies FPATH=YES. The minimum specification for FPATH=size is 12; the maximum is 30720. FPATH=YES implicitly defines a wait-for-input (WFI) application program. The PGMTYPE= parameters that define the overlay structure and class are invalid if FPATH=YES is specified. The SYSID= parameter is also invalid if FPATH is specified.

If FPATH=YES is specified during a MODBLKS system definition, Fast Path must have been previously defined for the online system to which this change is made.

Fast Path-potential transactions must be able to run under two applications. One of these applications must be defined with FPATH=YES; the other with FPATH=NO. The application with FPATH=YES must also be defined with a routing code that can be assigned by the user Input Edit/Routing exit routine. This routing code can have the same name as the Fast Path-potential transaction.

FPATH=NO must be specified if specifying LANG=JAVA.

When Fast Path is included during system definition, the FPBUF parameter on the TERMINAL macro statement is ignored, except to determine the default EMH buffer size. Fast Path buffers are provided by the EMHB pool, which expands and contracts dynamically depending on the number of ETO terminals concurrently entering Fast Path transactions.

GPSB=

Causes the scheduling process of all environments to generate a PSB containing an I/O PCB and an alternate modifiable PCB. With the GPSB= keyword, you do not need to perform the PSBGEN and ACBGEN, thus eliminating I/O to the ACBLIB.

The name is 8 characters in length with blank spaces included.

GPSB= generates an I/O PCB named IOPCB (with 3 blank spaces on the right). The modifiable, alternate PCB is named TPPCB1 (with 2 blank spaces on the right). With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.

You can make an online change to add the GPSB option to an existing application, or add a new application with the GPSB options by using a MODBLKS system definition. However, the GPSB option does not take effect unless the ACBLIB is also changed with online change.

If LANG=JAVA is specified and the Java application is a message processing program, the name specified on GPSB= is the name of the Java application class.

LANG=

Defines the language interface of the application program. LANG= is used only with GPSB=. You can use the following values with the LANG= keyword:

- ASSEM

- COBOL
- JAVA
- PL/I
- PASCAL

ASSEM is the default value.

Requirements:

- If LANG=JAVA is specified, then FPATH=NO must also be specified.
- If, after IMS initialization, an ACB that was originally generated as LANG=*non-Java* and DOPT, is changed to LANG=JAVA and DOPT (and IMS is not restarted), you must ensure that both of the following regions are available the first time IMS tries to schedule the affected transaction:
  - An MPP region that has the same class as the application
  - A JMP region that has the same class as the application

  The reverse situation is also true. If, after IMS initialization, an ACB that was originally generated as LANG=JAVA and DOPT, is changed to LANG=*non-Java* and DOPT (and IMS is not restarted), you must ensure both of the following regions are available the first time IMS tries to schedule the affected transaction:
  - A JMP region that has the same class as the application
  - An MPP region that has the same class as the application

PGMTYPE=

Specifies whether the program executes in a BMP-type region or not. A BMP-type region includes BMP and JBP regions. The following PSBs can be defined with either BATCH or TP:

- PSBs scheduled by DB2 for z/OS stored procedures.
- PSBs scheduled by programs running under WebSphere Application Server for z/OS.
- Other users of the ODBA interface.

TP Specify TP for:

- Programs that execute in IMS TM MPP, JMP, and IFP regions.
- PSBs scheduled by IBM CICS Transaction Server for z/OS programs using DBCTL.
- Other users of the DRA interface.

This is the default.

BATCH

Specify BATCH for programs that execute in BMP-type regions, such as IMS BMP and JBP regions. Any associated transactions are assigned normal and limit priority values of zero (0).

OVLY

No longer used by IMS but is retained for compatibility. An execution time parameter, OVLA on procedure DFSMPR, is available when starting an MPP to indicate whether the overlay supervisor should be preloaded by IMS.

If a program is changed from batch to online using the online change facility, you must enter the `/ASSIGN` command to assign nonzero current, limit, and normal priorities (CPRI, NPRI, and LPRI keywords) to the transactions using that program. This is because the online change facility does not alter attributes

that are changeable through the /ASSIGN command. Regardless of whether you specify new values for the transaction in the MODBLKS system definition, they are ignored during /MODIFY processing.

If FPATH=YES is specified:

- TP specifies a message-driven Fast Path application program
- BATCH cannot be specified. If BATCH is specified, an error message is issued. Fast Path nonmessage-driven application programs are not supported, and should be changed to run as BMPs.
- The PGMTYPE= parameters that define the overlay structure and class are not valid.
- The SYSID= parameter is not valid.

The third parameter of the PGMTYPE= keyword specifies the class to which the transaction codes specified in the following TRANSACT macro statements are to be assigned. This parameter must be a decimal number from 1 to 999. This value must not exceed the value given (by specification or default) on the MAXCLAS= keyword of the IMSCTRL macro. The default is 1. If the transaction code class is to be specified in the individual TRANSACT macro statements, this parameter need not be coded. If the transaction code class is specified in both the APPLCTN and TRANSACT macro statements, the APPLCTN macro specification is ignored, and the TRANSACT macro specification is used.

If the PGMTYPE= (,,class) parameter is to be changed online, the class value that is specified cannot exceed the definition (by specification or default) on the MAXCLAS= keyword of the IMSCTRL macro in the online system to which this change is to be made.

The numeric class subparameter must not be specified if FPATH=YES is specified.

PSB=

Specifies the name of the PSB associated with this application program definition. Each local PSB name must be unique. A remote and a local application can each be defined as having the same PSB name. This is required in order to dynamically reassign a transaction from remote to local processing. The first character of the PSB name must be a letter. If PGMTYPE=TP, the PSB name must also be the program name.

SYSID=

Specifies in the multiple-IMS system configuration, the system identification (SYSID) of the remote system (that system on which the application executes) and the SYSID of the local system (the originating system to which the responses are returned). The values specified must be numbers in the range from 1 to 2036.

The remote SYSID specified must also be defined in an MSNAME macro statement, but the local SYSID can be defined in any or all the MSNAME, TRANSACT, and APPLCTN macro statements. If SYSID is specified, all other keywords except PSB are ignored.

If the SYSID parameter is specified in the APPLCTN macro statement, you need not specify the SYSID in the TRANSACT macro statement. If the SYSID is specified in both the APPLCTN and TRANSACT macro statements, the TRANSACT specification is ignored.

The SYSID parameter is independent of the link type (BSC, CTC, MTM, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Because the values associated with the SYSID= keyword can be changed using an /MSASSIGN command, you should not add SYSID= to an existing APPLCTN macro for a MODBLKS system definition. Doing so causes the local application to be deleted after an online change sequence of commands.

A PSB cannot be initially defined or redefined from remote to local by using the SYSID= parameter on the APPLCTN macro during an online change.

The SYSID keyword parameter is invalid if FPATH=YES is specified.

SCHDTYP=

Specifies whether (PARALLEL) or not (SERIAL) this application program can be scheduled into more than one message region or batch message region simultaneously. The default value is SERIAL.

When the SCHDTYP= parameter is changed by a MODBLKS system definition from serial to parallel or vice versa, and the PSB, defined as resident, is not changed, the PSB is considered non-resident until the next IMS restart.

SCHDTYP=PARALLEL and the positional parameter DOPT are mutually exclusive.

If you specify SCHDTYP=SERIAL, omit the PARLIM parameter in the TRANSACT macro and accept the default (65 535). Also, omit the MAXRGN parameter in the TRANSACT macro or set it equal to 0.

TRANSTAT=

Specifies whether transaction level statistics are logged. If you specify Y, transaction-level statistics are written to the log in an X'56FA' log record.

N Transaction-level statistics are not logged.

Y Transaction-level statistics are logged.

The TRANSTAT keyword is optional. If a value is not specified for the TRANSTAT keyword, the system default (**N**) is used. The system default for the transaction-level statistics parameter is set with the TRANSTAT parameter in the DFSDFxxx PROCLIB member. Use the TRANSTAT keyword on the CREATE PGM or CREATE PGMDESC command to override the system default when creating a program or program descriptor.

BUFPOOLS macro

The BUFPOOLS macro statement specifies default storage buffer pool sizes for the DB/DC and DBCTL environments.

The storage buffer pool sizes that are specified on the BUFPOOLS macro are used unless otherwise expressly stated for that buffer or pool at control program execution time for an online system.

The Work Area Pool (WKAP) cannot be specified during system definition. When the system is initialized, the WKAP is automatically defined with a value of 5000. You can override this at execution time by using the WKAP= operand in the IMS, DBC, or DCC procedures, or in the DFSPBIMS, DFSPBDBC, or DFSPBDCC members of the IMS.PROCLIB data set.

This topic includes the following information:

- "Dynamic definition" on page 312
- "Supported environments" on page 312
- "Syntax" on page 312

- "Positional parameters"
- "Keyword Parameters"

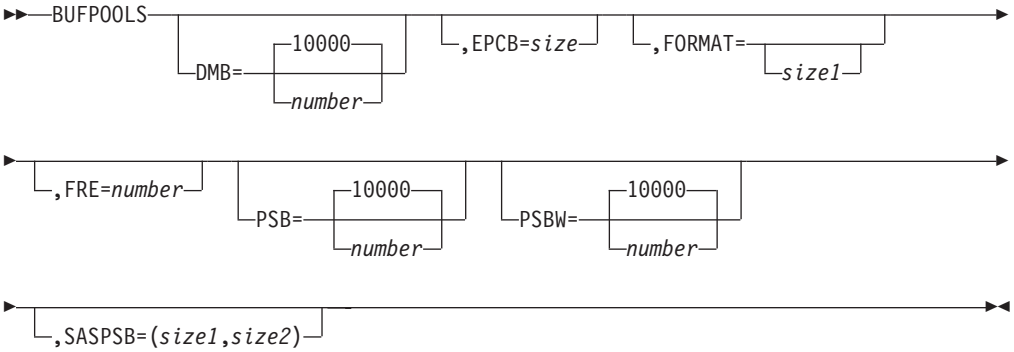
Dynamic definition

Buffer pools cannot be dynamically defined.

Supported environments

The BUFPOOLS macro can be used in the IMS DB/DC and DBCTL environments.

Syntax



Positional parameters

The BUFPOOLS macro does not include any positional parameters.

Keyword Parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

DMB=
Specifies the size of the DMB control block pool. The default is 10,000 bytes. The maximum allowable specification is 9,999,000 bytes. The minimum allowable specification is 8 bytes.

EPCB=
Specifies the size of the EPCB pool. If Fast Path is generated, the default size is 8 KB. Otherwise the default EPCB pool size is 0. The maximum allowable specification is 9,999,000 bytes. The storage for this pool resides in ECSA.

After working this calculation for each PSB, determine the largest PSB that might execute in a region, and add all the results for all regions. This sum is the value for your EPCB pool. You can override this pool specification using the EPCB parameter on the IMS procedure.

See Table 49 on page 314 for more information about calculating EPCB storage for PSBs.

FORMAT=
Specifies sizes of the message format buffer pool.

size1 specifies the size of the message format buffer pool. The minimum specification is 2000 bytes; the maximum specification is 99,999,000 bytes. This parameter is ignored when no unit types requiring MFS are included in the

defined system. If unit types that require MFS are included in the defined system and this parameter is not specified, the default value is calculated. *size2* is no longer supported.

The default value is calculated as follows:

```
1000  +  X  +  2124  +  FRE  x  44
```

where:

X Is the greater of: `((3270V/8) x 2200)` or `14336`

3270V Is the number of 3270 VTAM terminals

FRE is the number of specified or defaulted FREs

FRE=

Specifies the number of fetch request elements (FREs) for loading MFS control blocks into the message format buffer pool. The number specified can range from 10 to 99,999. If the size of the buffer pool is increased, the number of FREs should also be increased. Otherwise, the additional space cannot be used.

The default value is calculated as follows:

```
(10  +  X) or 30, whichever is greater
```

where:

X `(3270V/8) x 4`

3270V Is the number of 3270 VTAM terminals

PSB=

Specifies the size of the PSB control block pool if the DL/I address space option is not used. The SASPSB parameter specifies the size of the PSB control block pool when the DL/I address space option is used. The default is 10000 bytes, with a maximum of 9,999,000 bytes. The minimum allowable specification is 8 bytes.

PSBW=

Specifies the size of the PSB work area pool. The default is 10,000 bytes. The maximum allowable specification is 9,999,000 bytes. The minimum allowable specification is 8 bytes.

SASPSB=

Is used only if the DL/I separate address space option is selected. If you are not using this option, the size of the single PSB control block pool is specified with the PSB parameter.

With the DL/I address space option, two PSB control block pools exist. *Size1* is the size of the pool in the z/OS CSA. *Size2* is the size of the pool in DL/I local storage. The maximum allowable for either is 9,999,000 bytes.

Important: Due to control block size increases in z/OS, PSBs are slightly larger and require more space in the PSB pool.

Both *size1* and *size2* must be specified. Normally, the value of *size2* should be larger than that of *size1*, and neither value can be 0.

The ACBGEN utility output provides information about the relative PSB pool sizes. You should examine the output of ACBGEN before coding SASPSB.

The defaults are:

size1 20% of the PSB specification on BUFPOOLS

size2 80% of the PSB specification on BUFPOOLS

For example, SASPSB=(20000) is invalid, because *size2* is missing. The defaults apply only if the SASPSB parameter is not specified.

Usage information

Table 49 shows how to calculate EPCB storage for PSBs. The calculation is for MPP, IFP, and BMP region types that use Fast Path resources (DEDBs, MSDBs, or EMH).

Table 49. Calculating EPCB storage for PSB

EPCB type	Size	Total virtual storage requirements for this EPCB type (in bytes)	Total
IOPCB	28 bytes	28	_____K
Alt-Resp	28 bytes	28 x (the number of Alt-Resp PCBs)	_____K
MSDB	1 byte [1]	The sum of all MSDB PCBs	_____K
DEDB	1 byte [1]	The sum of all DEDB PCBs	_____K
Total EPCB storage for this PSB			_____K

[1]

Recommendation: This value can vary by release. The size of IMS control blocks can increase from release to release. Any change in the control block size would therefore impact the calculation of the pool size. Verify the control block size associated with the version of IMS that you are using before making these calculations.

Formula for calculating storage required for one MSDB EPCB in the EPCB pool:

```
       4 byte pointer
  +    length of DBFEPCB for MSDBs
```

For example, a calculation to find the storage needed for a single MSDB PCB for IMS might look like the following:

```
        4 bytes
   +   76 bytes
 ------------------------
 Total: 80 bytes needed in the EPCB pool for this single MSDB PCB
```

Formula for calculating storage required for one DEDB EPCB in the EPCB pool:

```
       4-byte pointer
  +    length of DBFEPCB for DEDBs
  +   (length of DBFSNMT) * (number of SENSEG statements + 2)
  +   (length of DBFMLTE) * (number of SENSEG statements)
  +    length of the key feedback area (maximum is 255; for HSSP PSBs only)
```

For example, a calculation to find the storage needed for a single DEDB PCB might look like the following:

```
        4 bytes
   +  192 bytes
   +   56 (8 bytes * 7)
   +  580 (116 bytes * 5)
-------------------------
Total: 832 (X'340') bytes needed in the EPCB pool for this single DEDB PCB
```

Example of JCL to code BUFPOOLS

The following figure shows how to code BUFPOOLS. If the DL/I address space option is not used, the single PSB pool size is 80,000 bytes. If the option is selected, the PSB pool size in the CSA is 16,000 bytes, and the DL/I local storage pool size is 64,000 bytes. This example also specifies a 40,000-byte DMB pool.

```
BUFPOOLS PSB=80000,SASPSB=(16000,64000),
DMB=40000
```

COMM macro

The COMM macro specifies general communication requirements that are not associated with any particular terminal type. COMM is always required for terminal types that are supported by VTAM. It is optional for BSAM, GAM, and ARAM terminal types.

The COMM macro can also be required to specify additional system options, such as support for MFS on the master terminal.

The COMM macro should be placed before the data communication specifications in the stage 1 input sequence.

This topic includes the following information:

- "Dynamic definition"
- "Supported environments"
- "Syntax"
- "Positional parameters" on page 316
- "Keyword parameters" on page 316

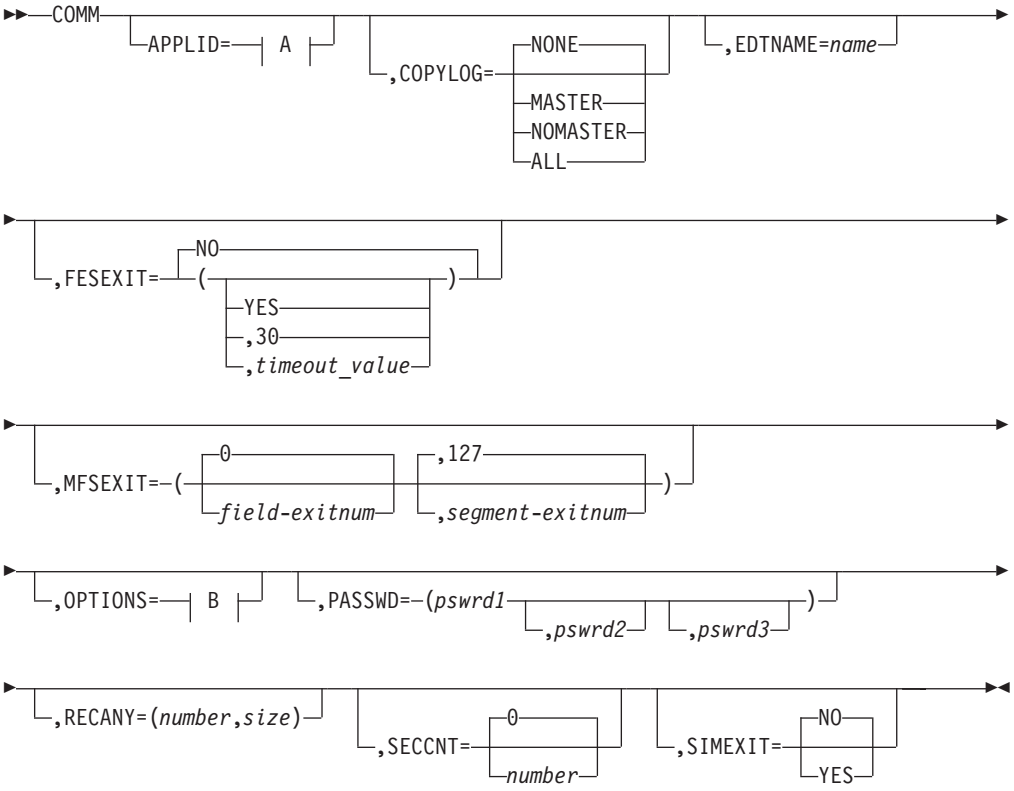
Dynamic definition

General communication requirements cannot be dynamically defined.

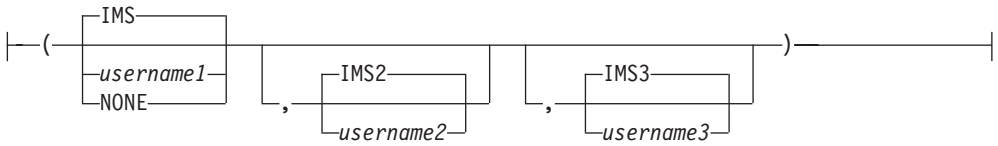
Supported environments

The COMM macro can be used in the IMS DB/DC and DCCTL environments.

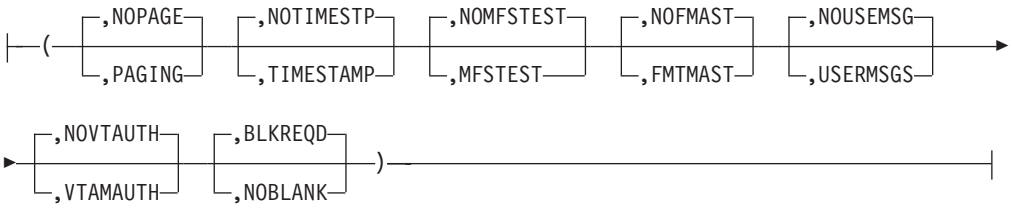
Syntax



A:



B:



Positional parameters

The COMM macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

AOEXIT=
No longer supported.

APPLID=

Specifies the application identification in the VTAM ACBs if VTAM is used. The default value is IMS.

For XRF, specify two VTAM APPLIDs: *username1* and *username2*. These two parameters correspond to the two XRF systems. The defaults are IMS and IMS2, respectively. When MNPS is used for XRF, *username1* and *username2* refer to the APPLID ACBs in the two XRF systems.

For RSR without XRF, you should specify at least two VTAM APPLIDs: *username1* (or *username2*) and *username3*, where *username3* is the application identification for the RSR tracker. In all cases, *username3* can only be used by the RSR tracker.

For RSR with XRF, you use all three VTAM APPLIDs: *username1* is for your active IMS (for both XRF and RSR), *username2* is your XRF alternate (at the RSR active site), and *username3* is your RSR tracker.

All three user name values can be overridden by specifying APPLID1=, and either APPLID2= or APPLID3=, or both, in the IMS or DCC procedures. These keywords can also be specified in the DFSPBIMS or DFSPBDCC members. *Username3* can match either *username1* or *username2*, but no two APPLIDs can be active in the network at the same time.

The defaults are IMS, IMS2, and IMS3, respectively.

If NONE is specified:

- The MSVERIFY program (DFSUMSV0) cannot validate that the APPLID names of the IMS system and its partner match if MSC/VTAM links are defined.
- Either the job step name of the job to bring up the IMS control region is used, or the name of the started task (if a started task is used for the IMS control region) is used.

If you specify APPLID=NONE and you issue a `/DIS ACTIVE DC` command, the output of the command is APPLID=NONE.

APPLID=NONE cannot be specified for XRF or for RSR.

COPYLOG=

Specifies the different categories of command and command response messages for which a copy is to be printed on the secondary master terminal.

If you want hardcopy logging, you must define a secondary master terminal.

The default is NONE.

NONE

Specifies that the commands listed in Table 50 on page 318 and command responses from terminals are not to be printed on the secondary master terminal. The queuing for the secondary master terminal of messages not in the categories described in Table 50 on page 318 is not affected by the COPYLOG statement.

MASTER

Specifies that copies of commands listed in Table 50 on page 318 and command responses from the master terminal are to be printed on the secondary master terminal.

NOMASTER

Specifies that copies of the commands listed in Table 50 on page 318

and command responses from terminals other than the master terminal are to be printed on the secondary master terminal.

ALL Specifies that the commands listed in Table 50 and command responses from all terminals are to be printed on the secondary master terminal.

COPYLOG does not affect system messages, which continue to be logged to the secondary master terminal.

The commands that can be copied to the secondary master terminal are:

Table 50. Commands that can be copied to the secondary master terminal

/ACTIVATE	/DBRECOVERY	/MSASSIGN	/RCOMPT
/ASSIGN	/DELETE	/OPNDST	/RSTART
/CHECKPOINT	/DEQUEUE	/PSTOP	/START
/CLSDST	/DISPLAY	/PURGE	/STOP
/COMPT	/IDLE	/QUIESCE	/TRACE
/DBDUMP	/MONITOR	/RCLSDST	/UNLOCK SYSTEM

Other commands can be logged but are not under the control of COPYLOG as /MSVERIFY is, for example.

The COPYLOG specification can be dynamically changed at execution time by the /SMCOPY operator command.

EDTNAME=

Names the ISC edit process for LU 6.1 nodes. The name specified becomes a synonym for ISCEDT, the default name that can be input and is output on the DPN parameter of the SCHEDULER function management header. When this name is input to IMS on the DPN parameter of the ATTACH or SCHEDULER function management header, it causes IMS to use the ISC edit process. The name used on the EDTNAME parameter cannot be the same as any MFS message input descriptor (MID) name used as an input ATTACH DPN parameter to invoke an MFS process.

FESEXIT=

Specifies whether (YES) or not (NO) the Front End Switch exit routine is to be included in the IMS nucleus.

The default is NO.

The *timeout_value* specifies a timeout value in seconds. A *timeout_value* specifies the time interval in which a reply for an FES message must be received. If no reply is received by the front-end system in time, timeout processing begins. The minimum value is 1, and the maximum value is 300.

If FESEXIT=YES is specified without a timeout value, the default for the *timeout_value* parameter is 30.

The FES exit routine must be placed in the data set defined by the USERLIB= keyword (of the system definition IMSGEN macro) before the execution of stage 2. The name of the exit routine must be DFSFEBJ0.

FESEXIT=(YES) and OPTIONS=NOBLANK are mutually exclusive. If you specify both, stage one assembly does not complete successfully.

MFSEXIT=

The variable *fieldexitnum* specifies the highest-numbered MFS Field Edit exit routine to be included in the generated system. *fieldexitnum* must be a decimal number greater than or equal to 0 and less than the default or specified value for the *segmentexitnum* parameter. The default is 0.

The *segmentexitnum* value specifies the lowest-numbered MFS Segment Edit exit routine to be included in the generated system. The *segmentexitnum* must be a decimal number greater than the default or specified value for the *fieldexitnum* parameter and less than or equal to 127. The default is 127.

Recommendation: Make lower-numbered exit routines be field exit routines, even though the lower-numbered exit routines can be segment exit routines, or can be a mixture of field and segment exit routines. An additional recommendation is that the higher-numbered exit routines be segment exit routines, although the higher-numbered exit routines can be field exit routines, or can be a mixture of field and segment exit routines.

An MFSEXIT=(14,120) specification indicates that user-supplied MFS Field Edit exit routines DFSME001 through DFSME014 and user-supplied MFS Segment Edit exit routines are IMS-supplied routines and are always included in the generated system when MFS requirements exist.

User-supplied MFS exit routines are included in the system definition-generated online control blocks module DFSIBLKx. Therefore, MFS exit routine specifications can be changed by an IMS CTLBLKS, NUCLEUS, ON-LINE, or ALL system definition. The MFSEXIT keyword specification is ignored when MFS exit requirements do not exist in the generated system.

OPTIONS=

The parameters included in this description are not position dependent within the operand sublist. If any terminal or password security options are specified here and also in the IMSGEN or SECURITY macro statement, a warning message is issued.

NOPAGE | PAGING

Defines whether (PAGING) or not (NOPAGE) the terminal paging feature is included in the defined system. The default is NOPAGE. If 3270 or SLU 2 devices are included, or if SLU P, LU 6.1, or 3600 devices defined with MFS are included, the paging capability is automatically included.

NOTIMESTP | TIMESTAMP

Specifies whether (TIMESTAMP) or not (NOTIMESTP) the system message time stamp feature is desired in the defined system. The default is NOTIMESTP.

When TIMESTAMP is specified, the time at which a message was generated is inserted between the message number and the message text for each message in the table.

NOMFSTEST | MFSTEST

Specifies whether (MFSTEST) or not (NOMFSTEST) the Message Format Service test facility, MFSTEST, is to be included in the generated system. Specifying MFSTEST is invalid for systems not containing MFS terminals. The default is NOMFSTEST.

Specifying a numeric value on the COMM macro statement is no longer recommended; for compatibility reasons it is still allowed, but any numeric value is ignored. When a numeric value is specified the NOMFSTEST default is overridden and MFSTEST is set; if MFSTEST=NO is specified on the IMSGEN macro, it is also overridden.

Using MFSTEST can degrade IMS performance due to MFSTEST use of the CIOP pool.

NOFMAST | FMTMAST

Specifies whether (FMTMAST) or not (NOFMAST) the IMS-provided

support for MFS on the master terminal is to be used. This support is available for the following devices used as the master terminal:

- 3277 model 2
- 3270 displays that have a 24 x 80 screen and use symbolic names
- 3277 model 2 when defined as SLUTYPE2

NOUSEMSG | USERMSGS
Specifies whether (USERMSGS) or not (NOUSEMSG) the user-supplied user message tables module DFSCMTU0 is to be included within the generated system.

NOVTAUTH | VTAMAUTH
Specifies whether (VTAMAUTH) or not (NOVTAUTH) IMS is to use the VTAM authorized path for communicating.

Related reading: Refer to *ACF/VTAM Macro Language Guide* for information about the authorized path facility. These parameters can be overridden by the JCL prepared for system execution.

BLKREQD | NOBLANK
Specifies whether the 1- to 8-byte transaction code requires a trailing blank. NOBLANK lets you enter a 1- to 8-byte transaction code without having to specify a trailing blank.

The selection of NOBLANK indicates that a blank does not have to be specifically entered after a one- to eight-byte transaction code, but if data is being included after the transaction code, then a blank must be inserted between the code and the data.

OPTIONS=NOBLANK and FESEXIT=(YES) are mutually exclusive. If you specify both, stage one assembly does not complete successfully.

PASSWD=
Specifies, when neither XRF or RSR is used, that one password is to be specified in the VTAM ACB. This password is checked by VTAM. If no password is specified, either in the VTAM ACB or in the PASSWD1= parameter, and VTAM requires one (during VTAM system definition), the IMS VTAM ACB is not initialized.

For XRF, you should specify two VTAM passwords that correspond to the two XRF systems. If only one password is specified, it is used as the password for both XRF systems.

For RSR without XRF, you should specify at least two VTAM passwords:

- *password1* (or *password2*)
- *password3*

In all cases, *password3* can only be used by the RSR tracker.

For RSR with XRF, you use all three VTAM passwords:

- *password1* for your active IMS (for both XRF and RSR)
- *password2* for your XRF alternate (at the RSR active site)
- *password3* for your RSR tracker

If you specify only one password, it is used for all systems. If you specify only two passwords, *password1* is used for the RSR tracker.

RECANY=
This parameter is required only when defining VTAM terminals. It defines the number and size of the RECEIVE ANY buffers.

This keyword and its parameters are optional and can be either overridden with execution-time parameters or defaults can be used if not specified by either of these methods.

number
Specifies the number of VTAM RECEIVE ANY buffers to be present in the IMS system. Valid values are from 1 to 500. The default is 16 if not specified during system definition or with execution-time parameters. You can override this specification with the EXEC parameter RECA= in the IMS procedure.

When MNPS is used for XRF, this parameter specifies the number of buffers for the MNPS ACB. IMS automatically allocates one additional buffer for the APPLID ACB.

Specifying too many buffers does not improve performance and can result in excessive resource consumption, especially high CPU utilization. You can check current and maximum use by looking in the X'450D' log record.

size
Specifies the size of the largest RECEIVE ANY buffer. This size must be large enough to handle maximum input from any VTAM-attached terminal. The minimum acceptable size is 50; the maximum is 30720. The default is 1920 if not specified during system definition or with execution-time parameters. The resulting usable input buffer size for any terminal can be calculated as follows for the devices shown:

3601: Size is record size plus header. The header size is 2 bytes if MFS is not used and 11 bytes if MFS is used.

3270: Size is the input data stream length. The minimum size is 300 bytes and a reasonable maximum size is 3842 bytes.

SLU 1, SLU 2: Record size should be equal to or greater than the largest input record to be received.

SLU P: Size is record size plus header. The header size is 5 bytes if MFS is not used and variable from 7 bytes to 40 bytes if MFS is used.

LU 6.1: Size is record size plus header. The header size without MFS is variable from 0 bytes to 45 bytes. The header size with MFS is variable from 17 bytes to 52 bytes.

NTO: Record size should be equal to or greater than the largest input record to be received.

For all VTAM terminals, the RECANY buffer size must be a decimal value that can be expressed by the algorithm `X` times 2 to the power of `Y`. `X` must be a value from 8 through 15, and `Y` must be a value from 3 through 11. This is a VTAM restriction. Thus, for example, the value 144 (representing $9x2^4$) or the value 28672 (representing $14x2^{11}$) are acceptable values.

Related reading: Details of this VTAM restriction are in *z/OS V1R2 Communications Server: SNA Programming*.

This restricts the valid VTAM buffer sizes to one of the following decimal values:

Table 51. Valid VTAM buffer sizes

112	320	896	2560	7168	20 480
120	352	960	2816	7680	22 528
128	384	1024	3072	8192	24 576

Table 51. Valid VTAM buffer sizes (continued)

144	416	1152	3328	9216	26 624
160	448	1280	3584	10 240	28 672
176	480	1408	3840	11 264	30 720
192	512	1536	4096	12 288	
208	576	1664	4608	13 312	
224	640	1792	5120	14 336	
240	704	1920	5632	15 360	
256	768	2048	6144	16 384	
288	832	2304	6656	18 432	

IMS converts the specified decimal value to the format required for the bind parameter fields and places the value into the appropriate field in the bind parameter list. If the maximum request unit (RU) value specified cannot be converted exactly into the bind format, the value is rounded down to the next-lower value bind format for inbound RUs and rounded up to the next-higher value bind format for outbound RUs.

SECCNT=

Specifies the maximum number of terminal and password security violations to be accepted per physical terminal before master terminal notification of such violation. The default is 0, which nullifies notification of the master terminal. The number specified can be 0, 1, 2, or 3. This value is reset to 0 upon successful signon and does not continue across transactions.

If SECCNT is not 0, the master terminal is notified for every violation.

SIMEXIT=

Specifies whether (YES) or not (NO) the Shared Printer Message Router exit routine is to be included in the IMS nucleus.

This exit routine must be placed in the data set defined by the USERLIB= keyword of the system definition IMSGEN macro statement before the execution of stage 2. The name of the exit routine must be DFSSIML0. A default exit routine is not provided.

The exit routine is bypassed unless SIMEXIT=YES is specified. If SIMEXIT=YES is specified, the exit routine DFSSIML0 is called when a message is queued for terminals defined with OPTIONS=SHARE on the TERMINAL macro statement.

When SIMEXIT=NO, the default, the router module assumes that `/OPN` is always simulated when output is enqueued for a terminal defined as OPTIONS=SHARE on the TERMINAL macro statement. Even if SIMEXIT=NO is specified, the module DFSSIML0 can be manually bound into the nucleus.

Usage information

The keyword parameters that follow are no longer used by the COMM macro; if they are specified, they are ignored:

- NOPSWD, PASSWD, FORPSW
- NOTERMNL, TERMINAL, FORCTERM
- NOMSPEX, MSPEXIT
- NOMSLEX, MSLEXIT

DATABASE macro

Use the DATABASE macro statement to define the set of physical databases that IMS is to manage.

The DATABASE macro is optional, and IMS issues no warning if it is not included during Stage 1 processing. Databases can be added later to the online system using the commands CREATE DB and UPDATE DB.

One DATABASE macro instruction must be specified for each HSAM, HISAM, and HDAM database. Two DATABASE macro instructions are required for a HIDAM database: one for the INDEX DBD and one for the HIDAM DBD. One DATABASE macro instruction must be included for each secondary index database that refers to any database defined to the online system.

For Fast Path, a DATABASE macro statement must be included for each Main Storage Database (MSDB) and Data Entry Database (DEDB) to be processed.

A database that is defined in an ALL or MODBLKS system definition cannot be converted into a HALDB partition without a cold start of IMS. A cold start is required even if the database is deleted online. After the cold start, the database must be redefined as a HALDB partition.

A database that is defined as a HALDB partition to DBRC and IMS cannot be redefined in an ALL or MODBLKS system definition without a cold start of IMS. A cold start is required even if the database is deleted online.

This topic includes the following information:

- "Dynamic definition"
- "Supported environments" on page 324
- "Syntax" on page 324
- "Positional parameters" on page 324
- "Keyword parameters " on page 324

Dynamic definition

To dynamically define the set of physical databases that IMS is to manage, you can use the CREATE DB and UPDATE DB type-2 commands. The following table compares the DATABASE macro keywords and the equivalent CREATE and UPDATE command keywords used for dynamic definition. Default values are shown in **bold**.

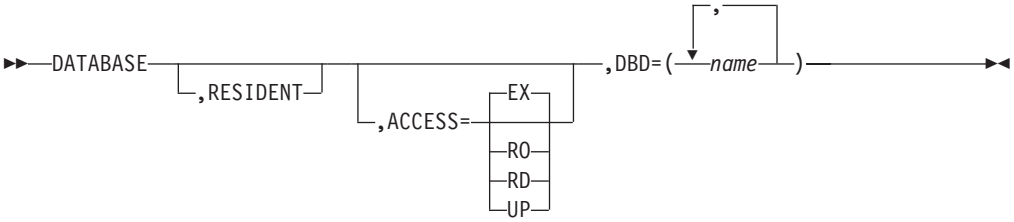
Table 52. DATABASE macro keywords and the equivalent CREATE and UPDATE command keywords used in dynamic definition

DATABASE macro keyword	**CREATE | UPDATE DB keyword equivalent**
DBD=*name*	NAME(*name*)
ACCESS=**EX** | RO | RD | UP	ACCTYPE(**EXCL** | BRWS | READ | UPD)
RESIDENT	RESIDENT (**N** | Y)

Supported environments

The DATABASE macro is optional and can be used in the IMS DBCTL and IMS DB/DC environments.

Syntax



Positional parameters

RESIDENT=

Indicates that the DMB directory control block created for this DATABASE statement should reside in storage during system and DBCTL initialization.

Fast Path makes the database control blocks resident in storage during system initialization regardless of whether the RESIDENT option is specified.

Online changes to the RESIDENT option are permitted. However, DMBs associated with new databases defined as RESIDENT are not made resident until the next IMS restart. Until that time, they are treated as nonresident. Changes to a RESIDENT DMB (from an ACBGEN) cause that DMB to be treated as nonresident until the next restart of IMS.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

ACCESS=

When used with the database-sharing level declared to database recovery control (DBRC), this parameter specifies the access for the defined database, that is, how the subsystem requesting access plans to use the database.

The ACCESS parameter can be modified by a /START command during IMS execution. Because this parameter can be changed using a /START DATABASE command, online changes to the ACCESS keyword have no effect until the next cold start.

You cannot specify an ACCESS value for MSDBs and GSAM. The possible ACCESS values are:

EX (exclusive)

Indicates that the named database can be accessed exclusively by this IMS subsystem. This is the default for ACCESS. Concurrent access to this database by other subsystems is prohibited by DBRC. Unless this is a DEDB, PCBs that refer to this database are scheduled, regardless of the PROCOPT values specified in PSBGEN.

Concurrent image copy cannot be run against a database with ACCESS=EX. If you are running with high-speed sequential processing (HSSP), you must specify ACCESS=EX or ACCESS=UP.

RD (read)
Indicates that the named database can only be read by this subsystem. The database is opened for input only. No application programs are allowed to physically update this database.

RO (read only)
Indicates that the named database is read-only in this IMS subsystem. This database is opened for input only. PCBs that refer to this database are allowed only if PROCOPT=GO is specified in PSBGEN.

UP (update)
Indicates that the named database can be updated and read. PCBs that refer to this database can be scheduled regardless of their PROCOPT values.

If you are running with high-speed sequential processing (HSSP), you must specify ACCESS=UP or ACCESS=EX.

DBD=
Specifies the name of one or more database descriptions (DBDs). If more than one DBD name is specified, each is assumed to have the same characteristics as any positional parameter specified or used by default. The first character of the name must be a valid alphabetic character (A through Z, #, $, or @). At execution time, the DBD must have been processed by the block builder utility program. It must exist as a member in the partitioned data set named in the IMSACB DD statement (IMS.ACBLIB). If the DBD has not been processed by the block builder utility program, and therefore, is not present in the ACBLIB data set, the database is locked at execution time. This parameter is required.

Example:

```
DATABASE DBD=(N1,N2,N3)
```

DFSMDA macro

Use the Dynamic Allocation macro (DFSMDA) to build a member (that is, one or more parameter lists) for naming data sets that can participate in dynamic allocation and deallocation. Members include databases, DEDBs, and data sets.

To use DFSMDA you must catalog all specified database data sets. However, you do not need to initially allocate them through control region JCL.

IMS users and IBM CICS Transaction Server for z/OS users can dynamically allocate IMS databases. For Fast Path databases, if the database data sets to be allocated are registered in DBRC, the information required to dynamically allocate the data sets is obtained from DBRC. You do not need to supply DFSMDA members for them. When the dynamic allocation information is obtained from DBRC, the DISP= used to allocate the data sets is either DISP=OLD or DISP=SHARE depending on the following:

- If SHARELVL=0, DISP=OLD is used.
- If SHARELVL=1, 2, or 3, DISP=SHARE is used.

DFSMDA is not a system definition macro.

The priority of allocation information is shown in the following table.

Table 53. Allocation information priorities

	DBRC	DD statement	DFSMDA member
DEDBs and MSDBs	1	2	3
All others	N/A	1	2

Database data sets specified in DFSMDA are allocated at different times depending on whether you are running in an IMS DB/DC, IMS batch, or CICS environment. The environment requirements are:

- **IMS DB/DC** database data sets are allocated either when a /START command is issued for the database or when an IMS application program is scheduled. You deallocate the data set by the /DBR command. If a database data set is specified in the JCL, it is allocated by z/OS during control region startup. You can deallocate it with the /DBR command and reallocate it with the /START command.
- **IMS batch** database data sets are allocated near the beginning of the job step, before the batch application starts execution.

  Dynamic allocation is always attempted for all non-JCL allocated databases defined in the PSB being executed. This is performed by searching the JOBLIB/STEPLIB concatenation for DFSMDA members, unless dynamic allocation is disabled (for batch only) by the presence of the NODYNALLOC statement in your DFSVSMxx member.

  If a batch job uses a PSB with more database PCBs than are necessary for a particular job, you can avoid dynamic allocation of the unnecessary databases while still maintaining a library of DFSMDA members for all databases belonging to the PSB. You have two methods of doing this:

  – You can include the NODYNALLOC statement in your DFSVSMxx member and include DD statements for only the necessary databases in your job JCL. The library of DFSMDA members does not need to be removed from the JOBLIB/STEPLIB concatenation because the NODYNALLOC statement disables batch dynamic allocation.
  – You can maintain separate libraries of DFSMDA members, which can be included or excluded from the JOBLIB/STEPLIB concatenation as needed. DFSMDA members need not be kept in your IMSVS.SDFSRESL.

    For example, you can maintain one main library of DFSMDA members for all the databases for a PSB and maintain several subset libraries. You concatenate only the library that is appropriate for the job being run. Dynamic allocation searches the entire JOBLIB/STEPLIB concatenation for DFSMDA members, so you must remove or alter all libraries that contain undesired members.

  If the databases for which your program has update intent have logical relationships or secondary indexes, those additional databases containing the logical relationships or secondary indexes can also be allocated, whether by JCL or DFSMDA members. To cause dynamic allocation of a logically related database, change the PROCOPT to indicate update intent. To dynamically allocate a secondary index, change the PROCOPT to indicate update intent or include a PCB with PROCSEQ= for the secondary index.

  If the PCB specifies a PROCOPT that does not indicate update intent, no intent is propagated to a logically related database or to a secondary index, and dynamic allocation is not attempted for either of these related databases.
- **CICS** database data sets are allocated when an application program issues a schedule call for the PSB. Deallocation occurs, for example, during the processing of STOP and RECOVERDB commands issued against the database.

You can dynamically allocate online log data sets (OLDS), write ahead data sets (WADS), and system log data sets (SLDS) if they are named in the DFSMDA macro. The DFSMDA macro must be defined to permit SLDS input to IMS to restart in z/OS.

When you start an OLDS using the /START command, the OLDS must be defined in the DFSMDA macro, even if it is allocated in JCL.

The IMS Monitor data set can also participate in dynamic allocation and deallocation. The IMS Monitor data set is allocated when it is started with the /TRACE ON command and deallocated when it is stopped with the /TRACE OFF command. It need not be initially allocated through JCL. It must not be cataloged if residing on tape; it must be cataloged if on DASD.

Recommendation: If you use the multiple DEDB area data set facility, register all data sets belonging to that area in either DBRC or DFSMDA.

The specified areas are allocated either when a /START command is issued for the area or when an application program attempts to use the area. The area is deallocated by /STOP AREA. Multiple areas can be deallocated by /STOP ADS.

In an XRF environment, all database and area data sets must be dynamically allocated.

Supported environments

The IMSDALOC macro can be used in the IMS DB/DC, BATCH, and DBCTL environments.

Syntax

The DFSMDA macro is coded as a z/OS macro. The statement label is optional, the macro "DFSMDA" is coded after one or more blanks, and additional parameters are separated by blanks. z/OS continuation rules apply.

The DFSMDA macro has several statement types (as indicated by the TYPE= parameter), each of which uses different additional parameters. Code the statements types as follows:

1. One TYPE=INITIAL statement to start the parameter list build.
2. As many of the other TYPE= statements as necessary. The maximum number of TYPE=DATABASE statements allowed is 250.
3. One TYPE=FINAL to end the list.

TYPE=INITIAL statement

This statement indicates the start of a parameter list build and is required. No other parameters are valid on a TYPE=INITIAL statement. The format of this statement is:

TYPE=DATABASE statement

This statement specifies the start of the definition for a database to participate in dynamic allocation and deallocation: one or more TYPE=DATASET statements should follow. (Do not use this statement for a DEDB area.) The format of the statement is:

DBNAME=
Specifies the DBD name of a database whose data sets are to be dynamically allocated. This name is used as a member name in IMS.SDFSRESL to identify this database parameter list. Care should be taken to ensure that this name does not conflict with existing members in IMS.SDFSRESL. This includes, but is not limited to, IMS modules and user-supplied exit routines.

TYPE=DATASET statement

This statement defines either a data set within the database specified in the previous TYPE=DATABASE statement or a DEDB area. One complete TYPE=DATASET is used for each data set or area data set defined. Every data set within a database to be dynamically allocated and deallocated must be named in a TYPE=DATASET statement. When defining dynamic allocation of IMSACBA or IMSACBB, the TYPE=IMSACBA or TYPE=IMSACBB statement is followed by one or more TYPE=DATASET statements. When defining DEDB areas, a TYPE=FPDEDB statement must precede each TYPE=DATASET statement. If the data set within a database identifies a secondary index data set shared with another database, the DFSMDA members for the two databases must be generated in separate assemblies. The format of this statement is:

DSNAME=
Specifies the name of the data set. The name can be any combination of simple and compound names valid in JCL, except the name cannot contain special characters.

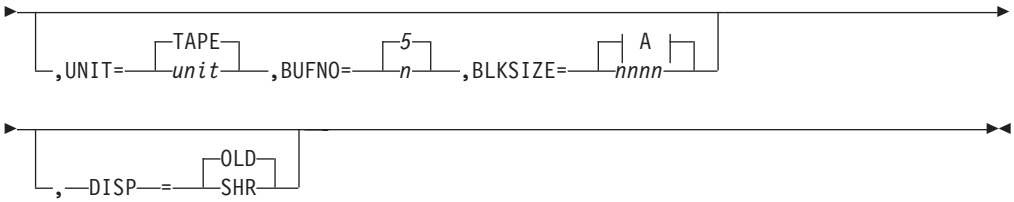
DDNAME=
Specifies the name of the DD statement defining this data set. This name is the same as that used in the DATASET or AREA statement of the DBDGEN.

For multiple ADSs, this name is the same as the ADDN name registered in the ADS RECON data set.

DISP=
Specifies the disposition of this data set when allocated. The default is OLD.

TYPE=DFSDCMON statement

Include one TYPE=DFSDCMON statement to define the dynamic allocation parameter list for the IMS Monitor data set. The format of this statement is:



A:

DSNAME=
Specifies the name of the data set, which must not be cataloged if the unit defines a TAPE device. However, if UNIT=DASD is specified, then the data set must be cataloged and available. The name can be any combination of simple and compound names valid in JCL, but must not contain special characters.

DDNAME=IMSMON
Is the required value for DDNAME.

UNIT=
Specifies the unit for the DC Monitor data set. If the data set resides on a direct access device, UNIT=DASD must be specified and the data set must be cataloged. Otherwise, the value of UNIT= can be the name of any tape device valid to the installation. The default is UNIT=TAPE.

BUFNO=
Specifies the number of buffers for the IMS Monitor data set. Valid numbers range from 2 to:

- a maximum of 99 for DFP
- 255 for DFSMS

The default is 5.

BLKSIZE=
Specifies the block size for the IMS Monitor data set. For a tape device (UNIT≠DASD), the default is 32 KB. If UNIT=DASD, the default is 27 992.

DISP=
Specifies the disposition for the IMS Monitor data set for a UNIT=DASD data set definition. Valid values are OLD and SHR. OLD is the default if this parameter is not supplied. A warning message is issued if any other value is supplied, and a DISP=OLD value overrides the value specified.

TYPE=FPDEDB statement

This statement defines an area within a data entry database (DEDB). One TYPE=FPDEDB statement is required for each area to be specified. The format of this statement is:

DBNAME=
Specifies the DBD name of the DEDB in which the specified area resides. This

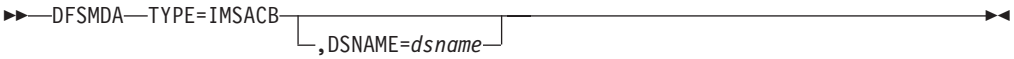
parameter is optional, and is used for documentation purposes only. For DEDB areas, the IMS.SDFSRESL parameter list is not named with the database name, but rather with the area's ddname.

TYPE=IMSACB statement

IMS uses the TYPE=IMSACB statement to dynamically create the staging library during the ACB library member online change process.

The ACB library member online change function works only in an IMSplex environment, where the global online change process is supported. You can optionally include the IMSACB DD statement in the "IMS procedure" on page 560 rather than the DFSMDA macro to dynamically allocate the staging ACBLIB. See this topic for an overview of dynamically allocating the staging ACBLIB: "Dynamically allocating the ACB staging library for ACBLIB member online change" on page 118.

The format of this statement is:



DSNAME=
Specifies the name of the data set. The name can be any combination of simple and compound names valid in JCL, except that it cannot contain special characters.

TYPE=IMSACBA statement

IMS uses the TYPE=IMSACBA statement to create the dynamic allocation member to allocate the IMS IMSACBA library data sets. Using DFSMDA to dynamically allocate IMSACBA allows you to resize the inactive ACBLIB data sets, add data sets to the concatenation, or change data sets in the concatenation without bringing IMS down. See the following topic for an overview of dynamically allocating the IMSACBA data set: "Dynamically allocating the IMSACBA and IMSACBB library data sets" on page 118.

The format of this statement is:

The TYPE=IMSACBA statement is followed by one or more TYPE=DATASET statements.

Dynamic allocation statements for the IMSACBA and IMSACBB data sets can be combined in the same job. They cannot be combined with other statements to dynamically allocate any other IMS data set.

TYPE=IMSACBB statement

IMS uses the TYPE=IMSACB statement to create the dynamic allocation member to allocate the IMS IMSACBB library data sets. Using DFSMDA to dynamically allocate IMSACBB allows you to resize the inactive ACBLIB data sets, add data sets to the concatenation, or change data sets in the concatenation without bringing

IMS down. See the following topic for an overview of dynamically allocating the IMSACBB data set: "Dynamically allocating the IMSACBA and IMSACBB library data sets" on page 118.

The format of this statement is:

The TYPE=IMSACBB statement is followed by one or more TYPE=DATASET statements.

Dynamic allocation statements for the IMSACBA and IMSACBB data sets can be combined in the same job. They cannot be combined with other statements to dynamically allocate any other IMS data set.

TYPE=RECON statement

This statement defines the dynamic allocation parameter list for database recovery control (DBRC). The format of this statement is:

DSNAME=
Specifies the name of the data set. The name can be any combination of simple and compound names valid in JCL, except that it cannot contain special characters.

DDNAME=
Specifies the name of the DD statement defining this data set. If an application uses alternate DD names for the RECON data sets, the dynamic allocation members must be created for the alternate DD names before the application can use these DD names.

WAIT=
If YES is specified on any of the TYPE=RECON statements, a wait is issued for any of the RECONs found to be offline during DBRC initialization. WAIT=NO is the default. Omitting the WAIT= parameter or specifying WAIT=NO causes dynamic allocation to fail if a RECON data set is offline during DBRC initialization.

TYPE=OLCSTAT statement

This statement defines the dynamic allocation parameter list for the global online change OLCSTAT data set. The format of this statement is:

DSNAME
Specifies the name of the OLCSTAT data set. The name can be any combination of simple and compound names valid in JCL, except that it cannot contain special characters.

TYPE=OLDS statement

This statement defines the dynamic allocation parameter list for the online log data set (OLDS). There must be as many DFSMDA macros as there are OLDS.

Requirement: If you use dual logging, DFSMDA member names are required for both the primary and secondary OLDS.

The format of this statement is:

```
►►—DFSMDA—TYPE=OLDS,DSNAME=dsname,DDNAME=DFSOLxnn————————————————————►◄
```

DSNAME=
Specifies the name of the data set. The name can be any combination of simple and compound names valid in JCL, except that it cannot contain special characters.

DDNAME=
Specifies the OLDSs to be allocated. If the OLDSs are dual, there must be a pair of macros, one with the ddname of the primary OLDS and the other with the ddname of the secondary OLDS (for example, DFSOLP01 and DFSOLS01). The data set must be cataloged. Substitute P for x when declaring a primary data set. Substitute S for x when declaring a secondary data set. Values from 00 through 99 can be specified for nn.

TYPE=SLDS statement

This statement defines the dynamic allocation parameter list for the SLDS. SLDSs are dynamically allocated when required as input for restart. A single DFSMDA member with name IMSLOGR must be created to specify the UNIT information required for allocation. All SLDSs to be used as input to restart must reside on the same device type. The format of this statement is:

```
►►—DFSMDA—TYPE=SLDS,UNIT=device type,DDNAME=IMSLOGR————————————————————►◄
```

UNIT=
Specifies the device required for allocation. All SLDSs used as input for restart must reside on the same device type. This applies to both the primary and secondary data sets when dual logging is used. The device type can be tape or DASD.

DDNAME=IMSLOGR
Is the required value for DDNAME.

TYPE=TRACE statement

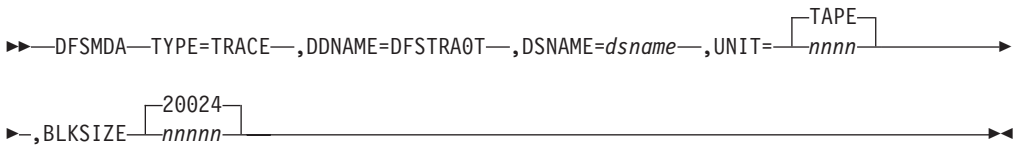
This statement defines the dynamic allocation parameter for external trace data sets. External trace data can be written to disk storage or to a tape unit. The statement differs with the type of storage chosen. The format of the statement for disk allocation is:

```
►►—DFSMDA TYPE=TRACE,DDNAME=DFSTRA0n,DSNAME=dsname————————————————————►◄
```

DDNAME
Specifies the ddname of the data set located on the disk. **n** specifies the number of the data set, and must be either 1 or 2. Each data set must be cataloged. Use two data sets to ensure that trace data is available at EOV.

DSNAME
Specifies the data set name. The name can be any combination of alphanumeric characters that is valid for IMS, except for special characters such as @, $, or #. DSNAME can be up to 44 bytes long.

The format for data set allocation to tape is:



DDNAME
Specifies the ddname of the data set located on the tape. DFSTRA0T is the required ddname if you allocate the external trace data set to a tape.

DSNAME
Specifies the data set name. DSNAME must not be cataloged. The name and can be any combination of alphanumeric characters that is valid for IMS, except for special characters such as @, $, or #. The name you specify can be up to 44 bytes long.

UNIT
Specifies the unit for the external trace data set. The unit must be a tape device, but can be any name valid to the installation. The default value is TAPE.

BLKSIZE
Is the block size of the external trace data set. The minimum value is 4008. Any other value chosen must be a multiple of 4004 (the LRECL) plus 4. The default is 16384.

Recommendation: Use a block size of 20 024 because it is 1/2 track.

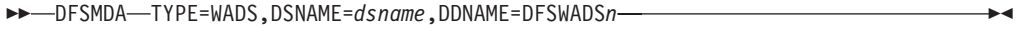
Future DASD might change the track size, and older DASD might have different track sizes.

TYPE=WADS statement

This statement defines the dynamic allocation parameter list for the write ahead data set (WADS). There must be as many DFSMDA macros as there are WADS data sets. The DFSMDA member name must be the same as the ddname of the WADS that it defines.

Requirement: If dual logging is used, DFSMDA member names are required for both the primary and secondary WADS.

The format of this statement is:



DSNAME=
Specifies the name of the data set. The name can be any combination of simple and compound names valid in JCL, except that it cannot contain special characters. The data set must be cataloged.

DDNAME=

Specifies the WADS to be allocated. Values 0 through 9 can be specified for n. When dual logging for the WADS is requested using the WADS=D execution time parameter, there must be at least two WADS provided.

TYPE=FINAL statement

This statement indicates the end of a parameter list build and is required. No other parameters are valid on a TYPE=FINAL statement. The format of this statement is:

JCL examples

The examples in this topic contain the following comment line above the SYSIN statement, for reference only, to aid in column alignment.

```
//* +----1----+----2----+----3----+----4----+----5----+----6----+----7---
```

Example 1: Specify three databases and the IMS Monitor data set to participate in dynamic allocation and deallocation.

```
//DALOC    JOB
//*
//STEP    EXEC IMSDALOC
//* +----1----+----2----+----3----+----4----+----5----+----6----+----7---
//SYSIN     DD  *
   DFSMDA TYPE=INITIAL
   DFSMDA TYPE=DATABASE,DBNAME=DI41M101
   DFSMDA TYPE=DATASET,DSNAME=IMSQA.M1I3I1,DDNAME=M1I3I1
   DFSMDA TYPE=DATASET,DSNAME=IMSQA.M1I3O1,DDNAME=M1I3O1
   DFSMDA TYPE=DATABASE,DBNAME=DX41SK03
   DFSMDA TYPE=DATASET,DSNAME=IMSQA.DB5H111,DDNAME=DXSK0301,           X
               DISP=SHR
   DFSMDA TYPE=DATASET,DSNAME=IMSQA.DB5H222,DDNAME=DXSK0302,           X
               DISP=SHR
   DFSMDA TYPE=DATASET,DSNAME=IMSQA.DB5H333,DDNAME=DHSK0301,           X
               DISP=SHR
   DFSMDA TYPE=DATABASE,DBNAME=DH41SK03
   DFSMDA TYPE=DATASET,DSNAME=IMSQA.DB4D111,DDNAME=DDSK0101,           X
               DISP=SHR
   DFSMDA TYPE=DATASET,DSNAME=IMSQA.DB4D222,DDNAME=DDSK0102,           X
               DISP=SHR
   DFSMDA TYPE=DFSDCMON,DDNAME=IMSMON,DSNAME=I115T237.IMSMON
   DFSMDA TYPE=FINAL
   END
/*
```

Example 2: Specify three DEDB areas to participate in dynamic allocation and deallocation

```
//DALOC    JOB
//*
//STEP    EXEC IMSDALOC
//* +----1----+----2----+----3----+----4----+----5----+----6----+----7---
//SYSIN     DD  *
   DFSMDA TYPE=INITIAL
   DFSMDA TYPE=FPDEDB,DD41SK02
   DFSMDA TYPE=DATASET,DSNAME=DB9AREA0,DDNAME=DB9AREA0
   DFSMDA TYPE=FPDEDB
   DFSMDA TYPE=DATASET,DSNAME=DB22AR0,DDNAME=DB22AR0,                  X
               DISP=SHR
   DFSMDA TYPE=FPDEDB,DEDBJN22
```

```
   DFSMDA TYPE=DATASET,DSNAME=DB22AR1,DDNAME=DB22AR1,                  X
               DISP=OLD
   DFSMDA TYPE=FINAL
   END
/*
```

Example 3: Specify the JCL and macro statements for SLDS to participate in dynamic allocation and reallocation

```
//ASSMBLY EXEC IMSDALOC
//*
//SYSLIB    DD  DSN=RNC.SDFSMAC,DISP=SHR
//          DD  DSN=I13OTS13.SDFSMAC,DISP=SHR
//          DD  DSN=IMS.&SYS2.SDFSMAC,DISP=SHR
//* +----1----+----2----+----3----+----4----+----5----+----6----+----7---
//SYSIN     DD  *
   DFSMDA TYPE=INITIAL
   DFSMDA TYPE=SLDS,UNIT=SYSDA,DDNAME=IMSLOGR
   DFSMDA TYPE=FINAL
   END
/*
//LNKEDT.SYSLMOD DD DSNAME=IMSQA.TNUC2,DISP=SHR,
//             UNIT=SYSDA,VOL=SER=USER01
```

Example 4: Build allocation members for RECON data sets RECON1, RCONSS, and RCONPW, and to trace data sets on DASD

```
//DYNALL   JOB
//*
//STEP    EXEC IMSDALOC
//SYSIN     DD  *
   DFSMDA TYPE=INITIAL
   DFSMDA TYPE=RECON,DSNAME=IMSV41.RECON01,DDNAME=RECON1,WAIT=YES
   DFSMDA TYPE=RECON,DSNAME=IMSV41.RECON02,DDNAME=RCONSS,WAIT=YES
   DFSMDA TYPE=RECON,DSNAME=IMSV41.RECON03,DDNAME=RCONPW,WAIT=YES
   DFSMDA TYPE=TRACE,DDNAME=DFSTRA01,DSN=IMS41.DFSTRA01
   DFSMDA TYPE=TRACE,DDNAME=DFSTRA02,DSN=IMS41.DFSTRA02
   DFSMDA TYPE=FINAL
   END
/*
```

Example 5: Build allocation members for RECON data sets and to trace data sets on tape

```
//DYNALL   JOB
//*
//STEP    EXEC IMSDALOC
//SYSIN     DD  *
   DFSMDA TYPE=INITIAL
   DFSMDA TYPE=RECON,DSNAME=IMSV41.RECON01,DDNAME=RECON1,WAIT=YES
   DFSMDA TYPE=RECON,DSNAME=IMSV41.RECON02,DDNAME=RECON2,WAIT=YES
   DFSMDA TYPE=RECON,DSNAME=IMSV41.RECON03,DDNAME=RECON3,WAIT=YES
   DFSMDA TYPE=TRACE,DDNAME=DFSTRA0T,DSNAME=TAPEDS1,UNIT=TAPE,BLKSIZE=20024
   DFSMDA TYPE=FINAL
   END
/*
```

Example 6: Define three data sets in the IMSACBA concatenation

```
//DYNALL   JOB
//*
//STEP    EXEC IMSDALOC
//SYSIN     DD  *
   DFSMDA TYPE=INITIAL
   DFSMDA TYPE=IMSACBA
   DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB1
   DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB2
```

```
   DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB3
   DFSMDA TYPE=FINAL
   END
/*
```

Example 7: Define three data sets in the IMSACBB concatenation

```
//DYNALL   JOB
//*
//STEP    EXEC IMSDALOC
//SYSIN     DD  *
   DFSMDA TYPE=INITIAL
   DFSMDA TYPE=IMSACBB
   DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB4
   DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB5
   DFSMDA TYPE=DATASET,DSNAME=IMS.ACBLIB6
   DFSMDA TYPE=FINAL
   END
/*
```

Restrictions

The following restrictions apply when using the Dynamic Allocation macro:

- HALDBs are dynamically allocated and do not need the Dynamic Allocation macro.
- If you are going to dynamically allocate a database, all DD statements referenced in the DMB for the database must be defined in the TYPE=DATASET, DDNAME= parameter. A database cannot be partially allocated by JCL and partially allocated by a dynamic allocate member.
- Because dynamic allocation cannot resolve logical relationships between DBDs, you must define a dynamic allocation member for each DBD in a logically related database. For example, a HIDAM database is composed of two logically related DBDs, the index DBD and the data area DBD. Each DBD in this example must have a dynamic allocation member with the same name as the DBD.
- The Batch Backout utility (DFSBBO00) is the only IMS utility that is supported for dynamic allocation.
- A database that is generated as a DFSMDA member cannot be given a name that is a duplicate of any label name that is generated during the assembly step of the DFSMDA job. IMS generates a label using the database name during this step, and an error occurs if that label name exists in code invoked by DFSMDA. This restriction does not apply to data set names.
- A database that is generated as a DFSMDA member cannot be defined with a ddname that is identical to the ddname defined for another database during the same assembly step of the DFSMDA job. If more than one database must be defined with the same ddname (as in the case of secondary indexes), the DFSMDA job must be run separately for each required occurrence of the ddname.
- Unless it is a dynamic allocation member, no member that has the same name as a database should be bound into IMS.SDFSRESL.

FPCTRL macro

The FPCTRL macro is ignored during system definition; however, you can include it for compatibility purposes.

To activate Fast Path, use the Fast Path parameter (FP=Y|N) on the IMS procedure. All values previously supported by FPCTRL are supported parameters on the IMS procedure.

Related Reading: See "IMS procedure" on page 560 for more information.

IMSCTF macro

Use the IMSCTF macro statement to define the SVCs to be used by DBCTL, logging options, and the device type for DBCTL's restart data set.

On the IMSCTF macro, the DYLOG and DISKLOG keywords and the third parameter on the RDS keyword (2|number) are no longer used and are not included in the macro format described in this information. If either of these are specified, they are ignored. They can remain in your system definition to provide compatibility with earlier IMS releases.

- "Dynamic definition"
- "Supported environments"
- "Syntax"
- "Positional parameters" on page 338
- "Keyword parameters" on page 338

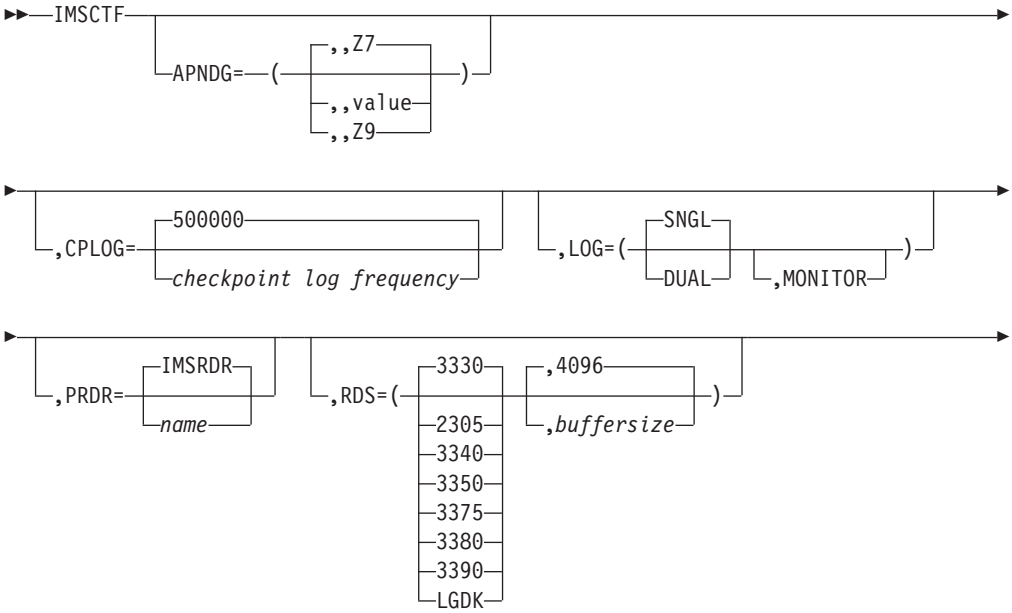
Dynamic definition

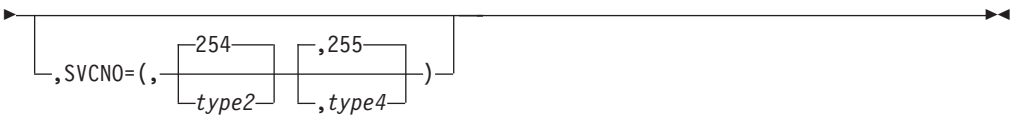
The parameters defined with IMSCTF cannot be defined dynamically.

Supported environments

The IMSCTF macro can be used in the DB/DC, DBCTL, and DCCTL environments.

Syntax





Positional parameters

The IMSCTF macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

APNDG=
The first operand and second operands are no longer used. The comma is retained to maintain compatibility with previous releases.

The third operand specifies the Multiple Systems Coupling channel-to-channel appendage suffix for z/OS systems. The value specified can range from WA to Z9 (the default is Z7). This appendage module's IMS name is DFSCMC10. However, for it to be loaded from the LPA or MLPA library, it must be named IGG019*xx*, where *xx* is the value selected for APNDG=. See the topic Installing required IMS links to z/OS (System Administration) in *IMS Version 11 System Administration* for more information about linking this module into z/OS LPA or MLPA libraries.

User appendages IGG019Z7 or IGG019Z8 can exist. If any of these appendages do exist, you should do one of the following:

- Rename the existing appendages
- Do not permit the APNDG parameter to default to Z7, but instead choose other suffixes
- Each IMS release running on the same z/OS should have a unique version of IGG019xx linked into the LPA or MLPA library using that release's version of DFSCMC10

  For example, if IMS Version 10 and IMS Version 11 both have IMS systems with CTC links on the same z/OS, then you can use IGG019Z6 for IMS Version 10 and IGG019Z7 for IMS Version 11. When IMS Version 10 is no longer being used, then IGG019Z6 can be removed or used for another future release (after it is relinked).

CORE=
No longer used. CORE= has been replaced by the PIINCR and PIMAX execution parameters for DB/DC and DBCTL IMS online systems, respectively.

CPLOG=
Specifies the number of system log records between system-generated checkpoints. The permitted value ranges from 500 to 16,777,215. The default is 500,000.

LOG=
The first operand, SNGL|DUAL, is no longer used for an online execution. If specified in an online configuration, it is ignored.

When specified for a batch execution, this parameter determines whether 1 (SNGL) or 2 (DUAL) DD statements are to be generated in the IMS batch procedures for the system log.

The second operand, MONITOR, specifies that the IMS Monitor DD statement is to be included in the procedure for the online system.

PRDR=
Specifies an optional IMS reader procedure name. The name must be 1 to 8-alphanumeric or national characters, with the first being alphabetic or national. The default is IMSRDR. You can also specify the reader procedure name in the control region JCL and in the default parameter module, DFSPBIMS. DBCTL users use DFSPBDBC.

RDS=
The first parameter specifies the device type on which the restart data set (IMS.RDS) is to reside.

You can specify one of the following device types: 3330 (the default), 2305, 3340, 3350, 3375, 3380, 3390 or LGDK.

When using a 3350, specify the drive format used—3330 or 3350.

LGDK is a generic definition for disk drives that have a track size greater than 32,760 bytes. The 3375, 3380, 3390, and future devices can also be specified as LGDK.

The second parameter specifies the buffer size to be used for the data set. The minimum buffer size is 1024. The maximum allowable size is the track size for the device type specified or 32,760, whichever is smaller. The default is 4096.

The third parameter is no longer used. If specified, it is ignored.

SVCNO=
Specifies supervisor call instruction (SVC) numbers reserved for use by the generated system.

The first parameter specifies the type 2 SVC number. The entered value can range from 200 to 255. The default is 254. The type 2 SVC number is required for batch, DBCTL, DCCTL and DB/DC IMS control program functions. One type 2 SVC number can be used for any number of IMS systems.

The second parameter specifies the type 4 SVC number reserved for use by database recovery control (DBRC). The entered value can range from 200 to 255. The default is 255. One type 4 SVC number can be used for any number of IMS systems.

For reasons of compatibility, the SVC numbers, if specified, must be immediately preceded by a comma and then enclosed in parentheses. Also, the type 2 and type 4 SVC number cannot be the same.

If you are installing different levels of IMS in the same processor, the Type 2 and Type 4 SVCs are downward compatible.

JCL example for IMSCTF macro

Following is an example of the IMSCTF macro instruction for the following conditions:
- Type 2 SVC is 254.
- DBRC Type 4 SVC is 255.
- Checkpoint frequency is every 2000 log records.
- The maximum dynamic storage allowed is 16 KB.
- The increment is 4 KB.

```
IMSCTF SVCNO=(,254,255),APNDG=(,Z0),CPLOG=2K
```

IMSCTRL macro

Use the IMSCTRL macro statement to describe the basic IMS control program options, the z/OS system configuration under which IMS is to execute, and the type and class of IMS system definition to be performed.

The IMSCTRL macro is required. This macro instruction must be the first statement of the system definition control statements that are entered into the system definition stage 1 for an IMS or a DBCTL system. You can have only one IMSCTRL macro instruction within an IMS system definition.

The only other macro instruction required for a batch environment is IMSGEN. The IMSCTF macro instruction is optional.

Important: If you run stage 1 and stage 2 processing using the IMSCTRL macro without specifying MODBLKS, a cold start is required. Failing to specify MODBLKS causes the IMS.SDFSRESL data set to be updated and copied, which requires you to perform a cold start.

- "Dynamic definition"
- "Supported environments"
- "Syntax"
- "Positional parameters" on page 341
- "Keyword parameters" on page 341

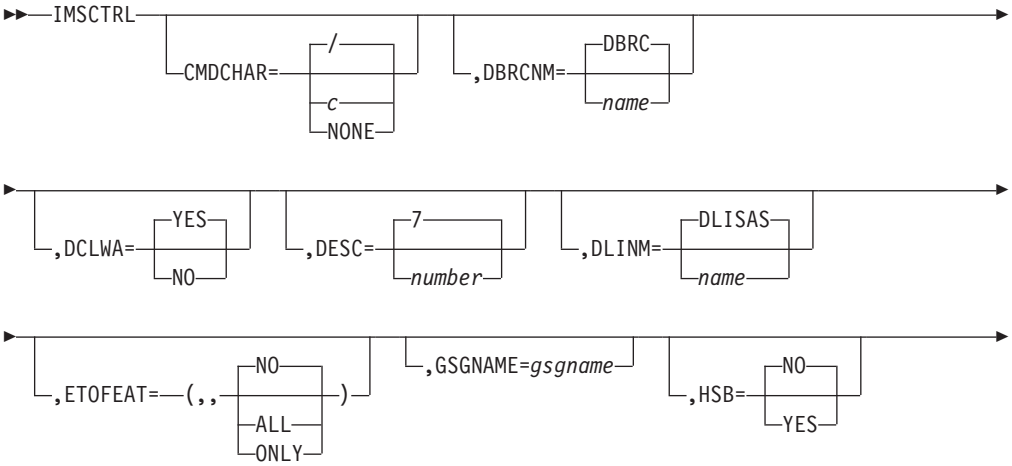
Dynamic definition

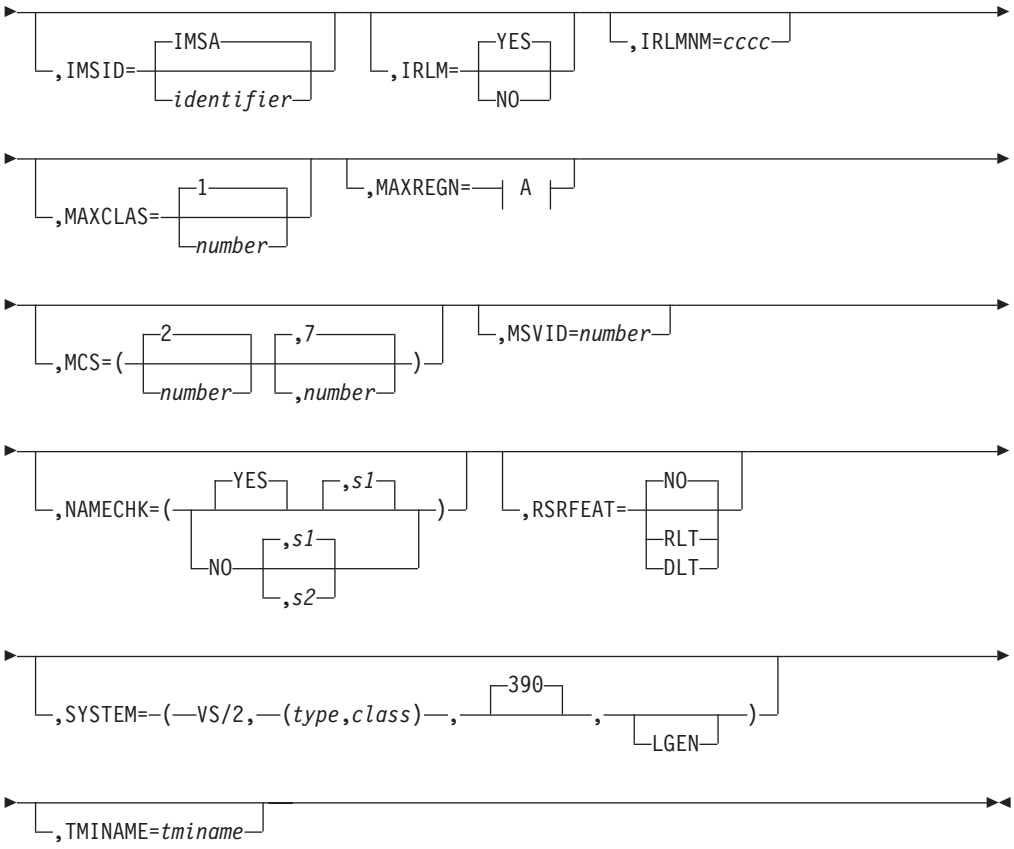
The IMS control region cannot be dynamically defined. However, you can export the MODBLKS resource definitions that are generated by the environment created using this IMSCTRL macro for subsequent dynamic definition of applications, databases, transactions, and routing codes.

Supported environments

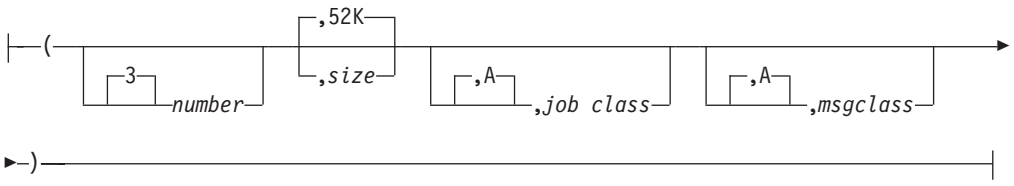
The IMSCTRL macro must be the first macro defined in the DB/DC, DBCTL, and DCCTL environments.

Syntax





A:



Positional parameters

The IMSCTRL macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

CMDCHAR=

/ The default value.

c Any character except comma (,), less than (<), or greater than (>). Ampersand (&) must be entered as && to define it as the command character.

NONE
No command recognition character (CRC) is defined.

CMDCHAR can be specified for DBCTL, DB/DC, or DCCTL. The value of CMDCHAR can be specified at execution time by specifying execution parameter CRC=.

With a DBCTL system, CMDCHAR specifies the command recognition character used by the MTO to enter IMS commands for this DBCTL environment. If CMDCHAR=NONE is specified and the CRC is not specified as an execution parameter (CRC=), then commands are entered for this DBCTL system with a prefix of the IMSID.

DB/DC and DCCTL systems allow commands to be entered at MCS/E-MCS consoles with a CRC prefix by specifying execution parameter keyword CMDMCS with a value other than N.

The CRC can be specified with CMDCHAR or with CRC at execution time.

If CMDCHAR=NONE is specified and the CRC is not specified as an execution parameter (CRC=), commands are entered for the DB/DC or DCCTL system with a prefix of the IMSID.

Although other products and subsystems might use certain command recognition characters, the CRC is not required to be unique among all IMS systems. Choose an unreserved character that is appropriate for your particular system.

Do not use a character for the CRC that is the same as the beginning character of a z/OS command. If the CRC is the same as the beginning character of a z/OS command, that command does not work after you start IMS. For example, if you start IMS with CRC=D, z/OS does not respond to any of the z/OS display commands such as `D A,L`.

The CRC value you select might affect multisegment command processing. The last character of the command is checked during multisegment command processing. If that character matches the CRC, the command is assumed to be a multisegment command. Processing of the command waits for the rest of the command. For example, if you use B for the command recognition character, and issue the command BMODIFY PREPARE ACBLIB, the command does not run. The ending 'B' in 'ACBLIB' is interpreted to indicate a multisegment command. To enter this command correctly, enter:

```
'BMODIFY PREPARE ACBLIB.  '
```

The character you choose for a DB/DC system becomes the command recognition for the IMS system. If your installation allows commands from MCS or E-MCS consoles, you need a CRC only if both of the following statements are true:

- IMS provides DBCTL service to a CCTL.
- CCTL broadcasts an IMS SWITCH command preceded by a CRC.

CMDCHAR does not affect the normal DB/DC MTO's use of the / character when entering IMS commands.

DBRC=

The system definition process ignores this parameter, though for compatibility purposes, it can still be coded.

DBRCNM=

Specifies for z/OS the IMS.PROCLIB member containing the skeletal procedure for the DBRC address space, produced by IMS system definition. The IMS control region automatically initiates the DBRC address space, using the z/OS start command and specifying this name. To rebuild the DBRC procedure, you must use the IMS.PROCLIB member containing the skeletal procedure for the

DBRC address space produced during system definition. This procedure contains the name you specify in stage 1. You must first change DFSPBIMS, DFSPBDBC, or DFSPBDCC to reflect the specified DBRCNM, and then perform another system definition.

Consequently, the DBRC procedure produced by IMS system definition must be copied from IMS.PROCLIB to SYS1.PROCLIB. For more information, see "Making IMS and IMSRDR procedures accessible to z/OS" on page 183.

You can specify up to an eight-character name, the first character being alphabetic. The default name is DBRC.

Specifying DBRCNM=xxxxxxxx results in the following:

A DBRC address space cataloged procedure is created with the name xxxxxxxx.

To make the DBRC address space cataloged procedure name of xxxxxxxx operational, the following must be true:

A DBRC address space must be cataloged as xxxxxxxx in SYS1.PROCLIB.

Simply specifying DBRCNM=xxxxxxxx in your IMS system definition is not sufficient.

DCLWA=

Establishes a default value that IMS applies to any transactions defined by subsequent TRANSACT macro statements. DCLWA= specifies whether (YES) or not (NO) IMS should perform log write-ahead for recoverable nonresponse-mode input messages and transaction output messages. The default is YES. If no value is specified for DCLWA= on the IMSCTRL macro, the value for DCLWA= on any subsequent TRANSACT macros accepts the default of YES, or you can specify YES or NO on the individual TRANSACT macros.

Specify or accept the default of YES to ensure that:

- A nonresponse-input transaction is made recoverable across IMS failures, before IMS acknowledging receipt of the input.
- Database changes are made recoverable before IMS sending associated output reply messages.

YES ensures that information in the log buffers is written to the IMS log before the associated input acknowledgment or output reply is sent to the terminal.

Specify or accept the default of YES for all VTAM terminal types.

Specify NO if input message integrity and the consistency of output messages with associated database updates is not required. DCLWA does not apply to response mode or Fast Path input processing, and, if specified in these situations, is ignored during IMS execution.

DESC=

Specifies the message descriptor codes to be assigned to the IMS system console messages if MCS support is included in the z/OS system definition. You can specify up to 16 values. If DESC is not specified, the default value is 7.

The IMS MCS= and DESC= keywords should be defined as is required for the ROUTCDE and DESC keywords of the z/OS WTO macro.

DLINM=

Specifies for z/OS the IMS.PROCLIB member to contain the skeletal procedure for the optional DL/I address space produced by IMS system definition. The

IMS control region automatically initiates the DL/I address space using the z/OS start command and specifying this name.

Consequently, the DL/I procedure produced by IMS system definition must be copied from IMS.PROCLIB to SYS1.PROCLIB. You probably need to modify the skeletal procedure.

For more information, refer to "Member Name DLISAS" in "Environments that support IMS system definition-supplied procedures" on page 446.

You can specify up to an eight-character name, the first character being alphabetic. The default name is DLISAS.

Specifying DLINM=xxxxxxxx results in the following:

A DL/I address space cataloged procedure is created with the name xxxxxxxx.

ETOFEAT=

Specifies whether the system definition process should build ETO descriptors. The default is not to build the ETO descriptors.

The first two parameters of the ETOFEAT keyword are no longer valid and are ignored if specified. Commas appear in place of these parameters in the syntax diagram.

The third parameter indicates whether the system definition process should produce ETO descriptors only (ONLY), produce descriptors along with a normal system definition (ALL), or produce a normal system definition without descriptors (NO). The default is NO.

If you have not installed ETO, the ETO option is forced off.

GSGNAME=

Specifies the global service group (GSG) name to be used for the RSR complex. No default value exists. If you have the RSR feature (RLT, DLT, or both) installed and you specify a GSG name, RSR is enabled for this system definition.

If you do not specify the GSG name in the IMSCTRL macro and you want RSR enabled, the GSG name must be specified in the DFSRSRxx member (or the JCL for batch or utility jobs) to enable RSR. If you do specify the GSG name in the IMSCTRL macro, you might have to change batch JCL to allow batch jobs to run with RSR not enabled.

Recommendation: If RSR is not installed or you do not want RSR enabled for this system definition, do not specify a value for GSGNAME.

If you do specify a value for GSGNAME and you do not want RSR enabled, you must override the RSR enablement by specifying RSR(NO) in the DFSRSRxx PROCLIB member. For more information, see "DFSRSRxx member of the IMS PROCLIB data set" on page 720.

HSB=

Specifies whether or not an XRF-capable system is generated. The default is NO.

If you specify NO, an XRF-capable system is not generated.

If you specify YES, an XRF-capable system is generated and ISC has duplexed blocks. If you plan to run XRF or an alternate subsystem before the next planned cold start of IMS, specify HSB=YES on the IMSCTRL macro to eliminate the need for a subsequent IMS system definition and cold start.

XRF is disabled if no value is specified for HSBID in the DFSPBIMS or DFSPBDCC members, or in the IMS execution JCL. Informational message DFS3899I is issued.

IMSID=

Specifies a one- to four-character alphanumeric identifier for the IMS system.

For IMS control region: Specifies a one- to four-character identifier that is a valid subsystem identifier to the operating system being used. It can be overridden by the IMSID= keyword specified in the startup procedures.

For IMS dependent regions: Specifies a one- to four-character identifier that is a valid subsystem identifier to the operating system to which this dependent region connects. It can be overridden by the IMSID= keyword specified in the startup procedures.

For IMS batch region: Specifies a one- to four-character IMS identifier that is used in IMS messages that are written to the system log. It can be overridden by the IMSID= keyword specified in the startup procedures.

Recommendation: Specify a unique IMSID for each batch region. This helps to avoid confusion as to which region has issued a console message.

Do not use characters for the IMSID that match the beginning characters of a z/OS command. If the IMSID is the same as the beginning characters of a z/OS command, that command does not work after you start IMS. For example, if you start IMS with IMSID=D, z/OS does not respond to any of the z/OS display commands such as `D A,L`.

For online control regions, the IMSID must be different from any other IMS subsystem identifier or non-IMS subsystem identifier defined to the operating system under which IMS is running. This identifier is also used to relate messages that are routed to the z/OS system console with the corresponding IMS system. To avoid confusion as to which region has issued a console message, specify a unique IMSID for each batch region. This is however, not a requirement.

The IMSID name must not be the same as the procedure name that starts IMS unless one of the following is true:

- All DD statements in the startup procedure are cataloged in the master catalog.
- The unit and volume are specified on each DD statement.

IRLM=

IRLM=NO specifies that the IRLM is not used by online or batch systems unless overridden at execution time.

IRLM=YES specifies that the IRLM is used by online and batch systems unless overridden at execution time.

If the IRLM= parameter is omitted, and IRLMNM= is specified, the IRLM default is set to YES, but can be overridden by the IRLM=N parameter at execution time.

If the both the IRLM= parameter and the IRLMNM= parameter are omitted, the IRLM default is set to NO but can be overridden at execution time. If it is overridden at execution time, a default value of IRLM is used for the IRLMNM= parameter.

If the IRLMNM= parameter is omitted and the IRLM= parameter is specified, a warning assembly error message is issued stating that the IRLM= parameter is

invalid, and NULL is assumed. A default value of IRLM is used for the IRLMNM= parameter if IRLM=Y is specified in the execution JCL.

IRLMNM=
Specifies a one- to four-character alphanumeric name assigned to the Internal Resource Lock Manager (IRLM) included in your system.

If the IRLMNM= parameter is not specified, the IRLM is not invoked by batch or online systems by default. To invoke the use of the IRLM by a batch or online system, you must code IRLM=Y in the execution JCL.

When the IRLMNM= parameter is not specified in the IMSCTRL macro or in the execution JCL, and IRLM=Y is specified in the execution JCL, the IRLM name that is used is IRLM.

The IRLM must be active in your system if your IMS system is to share data at the block level with other IMS systems. The IRLM is optional for systems that do not share data at the block level.

Only one IRLM can be specified for any IMS system (and thus, only one IRLMNM), but multiple IRLMs can exist within a particular z/OS system. Each IRLM must be defined as a z/OS subsystem.

The name specified on this IRLMNM= parameter must be the z/OS subsystem name assigned to the IRLM with which this IMS system is to connect. If IMS is connected to an executing copy of IRLM, DB2 for z/OS cannot connect to it. If DB2 for z/OS is connected to an executing copy of IRLM, IMS cannot connect to it.

If this parameter is specified and not overridden with the IRLM=NO parameter on the online or batch startup procedures, IMS only completes initialization if the named IRLM is available at execution time. Also, IMS uses the named IRLM for all locking services, both local and sharing. Program isolation locking is not used.

It is important to understand the interaction of this IRLMNM parameter with the DBRC parameter and with the ACCESS parameter on the DATABASE macro statement.

MAXCLAS=
Specifies the range of active (started) classes after IMS initialization. Valid values are decimal numbers from 1 to 999. The default is 1. Any transaction code class specified on either the APPLCTN or TRANSACT macro statement must not exceed this value. If the class is not active at the time of scheduling, the schedule request is treated as if the class were stopped.

MAXIO=
Is no longer used. If MAXIO is specified the value is ignored.

MAXREGN=
Specifies the maximum number of IMS control block sets that are permanently allocated. A control block set is required for each dependent region type (batch message processing, message processing, or CCTL threads) that can be started. (Note that this does not imply that the same number of dependent regions are always active.)

This value must be a decimal number from 1 through 999. When the count of dependent regions started exceeds the value specified by the MAXREGN= parameter, control block sets are dynamically allocated, but the storage for these sets is released as the dependent regions terminate. The default is 3.

For an RSR tracking subsystem, the value specified on the MAXREGN= parameter determines the maximum number of PSTs that are to be used for DL/I database tracking. This number should be twice the number of DL/I PSTs used in all the active subsystems tracked by the tracking subsystem. If many PSTs are non-updating, set this number to twice the number of updating DL/I PSTs used in all active subsystems. A minimum of two PSTs is used.

The second through fourth fields specify region size (default value is 52 KB), region job class (default class is A), and job message class (default class is A).

Region size (which must be expressed in terms of xxK), region job class, and job message class must conform to the operating system JCL specifications. This operand is ignored for BATCH system definitions.

MCS=

Specifies the z/OS routing code to be assigned to the IMS system console if multiple console support (MCS) is included in the operating system. If MCS is not specified, the default is (2,7).

Although z/OS supports more than 16 route codes, IMS uses only route codes 1 through 16.

In a DBCTL environment, MCS= defines which consoles are to receive unsolicited DBCTL messages.

MSVID=

Specifies a one- to three-digit decimal number 1 - 676 that is used to complete the eight-character name of the generated IMS Multiple Systems Verification utility control blocks module generated for this system (DFSMSxxx). If fewer than three decimal digits are specified, the value is right-justified and filled to the left with zeros. (For example, specifying 3 creates the name DFSMS003.)

This control blocks module is assembled and bound during IMS system definition stage 2 processing. The Multiple Systems Verification utility procedure (IMSMSV) is, as the user directs, placed into the IMS.PROCLIB data set.

The MSVID= keyword:

- Is applicable only to resources in the IMS.MODBLKS data set that are defined by system definition.
- Is required when MSVERIFY is specified in the SYSTEM keyword.
- Is ignored if no multiple systems definition statements are present.
- Is ignored for DBCTL.
- Must be present to generate the IMS Multiple Systems Verification utility control blocks and procedure.

The MSC Verify utility (DFSUMSV0) can verify resources defined by system definition. It cannot verify resources defined dynamically, because those resources are no longer in MODBLKS. However, you can use the /MSVERIFY command to verify resources that are defined dynamically.

The specification of MSVID on the IMSCTRL macro statement is not related to the specification of SYSID on the MSNAME macro statement.

NAMECHK=

Specifies whether (YES) or not (NO) resource name checking and cross-checking are to be performed, and, if NAMECHK=NO, whether sorting is to be performed in stage 1 (S1) or stage 2 (S2).

If duplicate resource name checking and resource name cross-checking are performed (that is, NAMECHK is specified as or uses the default of YES),

sorting must be performed in stage 1 (S1 specified or used by default). If resource name checking is to be bypassed, sorting can occur in either stage 1 (S1 specified or by default) or stage 2 (S2 specified).

It is suggested that you use the default values for NAMECHK unless:

- You have run the preprocessor before your stage 1 run and you want to omit resource name checking here (by specifying NAMECHK=NO).
- You have a specific performance reason for having IMS sort in stage 2 (for example, you generally run stage 1 multiple times per stage 2).

Related reading: For information about the options available to you by including or omitting the preprocessor in combination with options on the NAMECHK parameter, see "Verifying the stage 1 input" on page 22.

This keyword is ignored if LGEN is specified in the SYSTEM keyword.

RSRFEAT=

Specifies RLT or DLT for RSR. This keyword is ignored by IMS, but is left in for compatibility with prior releases. For information about enabling RSR, see GSGNAME=.

SYSTEM=

Specifies the operating system configuration and the type of IMS system definition to be performed.

The first SYSTEM= parameter must be VS/2, indicating the operating system is z/OS.

The second SYSTEM= parameter consists of two subparameters. The first subparameter defines *type* of system definition. The definition types are: ALL, BATCH, CTLBLKS, MODBLKS, MSVERIFY, NUCLEUS, and ON-LINE.

The second SYSTEM= subparameter defines the system definition environment *class*. The three system definition environment classes are:

DB/DC
Builds a standards IMS system. DB/DC is the default.

DBCTL
Builds only the DBCTL environment.

DCCTL
Builds only the DCCTL environment.

If you do not specify DBCTL on the second subparameter, the following functions are always included:

- All enhanced security codes and commands
- All MFS code, commands, procedures, and default formats
- VTAM code and commands, whenever VTAM is requested

The MFS format library is a required data set (regardless of whether MFS is used), except for DBCTL systems.

The MSVERIFY,DBCTL type/class combination is invalid. An error message is generated for invalid combinations.

The third SYSTEM= parameter indicates the operating system; the default is 390 (z/OS).

The fourth SYSTEM= parameter defines the large system definition configuration. Use the LGEN parameter if a stage 1 or any of the stage 2 job steps of a system definition requires more than 4 MB of private storage for execution.

The LGEN parameter is only valid when stage 1 source statements are input to the preprocessor. The preprocessor passes a group of up to 10 000 resources to the assembler for stage 1 processing. The assembler is invoked as many times as is necessary to process all the stage 1 resources.

When you first install IMS, you must perform a full system definition. Thereafter, you can perform a subset of the full system definition to add or change a function. The parameters used in these subset system definitions are described in this section.

The SMP JCLIN process should be run after the stage 1 of all definition types.

ALL
Combines the BATCH and ON-LINE options.

BATCH
Moves required modules from the IMS distribution libraries to the user's libraries; it also generates system procedures and a database system.

A BATCH definition should not be run to add maintenance or to add or delete database functions or features to a residence library that contains a nucleus with the Data Communication feature.

CTLBLKS
Generates control block modules of all IMS control blocks for use within an IMS nucleus in the IMS control program region. The blocks built in the MODBLKS option are also included in this definition. This subparameter should be specified to replace the control blocks of an existing nucleus. If you are performing a CTLBLKS definition, the new nucleus must have the same suffix as the old nucleus. SMP maintenance integrity is lost if you rename the nucleus before using it in a CTLBLKS definition because SMP does not recognize the contents of the renamed nucleus. To change the control blocks in an existing nucleus, do a CTLBLKS definition. To generate a new nucleus, do a NUCLEUS definition.

The SMP JCLIN process should be run after the stage 1 of a CTLBLKS definition.

A CTLBLKS specification generates the MSC Verification Utility control blocks, if MSC definitions are present. When you specify CTLBLKS, you need not perform a separate system definition with MSVERIFY.

If any device or option is to be deleted during a CTLBLKS definition, unresolved references may exist when the nucleus is bound. This can also happen if logical terminal names are added or deleted before the occurrence of master or secondary master logical terminal names.

MODBLKS
Generates the control blocks members for resources to be added or changed online. These control blocks are used by the IMS control region and the Multiple Systems Coupling Verification utility.

The MODBLKS system definition requires that the stage 1 input contains all source statements for the existing system to which the online changes are to be made, except for communication specifications. If you use the Multiple Systems Coupling MSVERIFY utility, **all** source statements, including communication specifications, are required input to your stage 1 run. However, if communication specifications are omitted from a MODBLKS system definition, the following stage 1 error checking is not performed:

- Transaction names are not cross-checked against LTERM names to locate duplicates.
- For MSC users, the remote system ID (SYSID) on the TRANSACT macro statement is not checked against the remote system ID (SYSID) on the MSNAME macro statement to ensure that they are not duplicates.

Thus, omitting communication statements can result in undetected system definition errors. You must therefore consider your requirements for improved stage 1 performance against the possibility of undetected errors in your stage 1 input to determine whether you want to include or omit communication statements from your MODBLKS system definition input.

Only APPLCTN, DATABASE, TRANSACT, and RTCODE statements can be added, deleted, or modified. The restrictions pertaining to these modifications are found in the descriptions of these macro statements. IMS does not enforce these restrictions; rather, you are responsible for adhering to the directions provided there.

The effects of a MODBLKS system definition are made known to the IMS online system with a master terminal operator `/MODIFY` command sequence.

Stage 2 assembler output for a MODBLKS system definition is placed in IMS.OBJDSET. Binder output for the control blocks members affected by running an online change is placed in IMS.MODBLKS.

Certain types of changes, when made across a MODBLKS system definition, do not affect an existing resource except across a cold start. That is, the effects of prior commands are not negated by online change procedures. For example, resources that have been modified by commands such as `/START`, `/STOP`, `/ASSIGN`, `/MSASSIGN` are not altered, nor are resources whose status has been altered because of changes in the operating configuration (for example, a resource stopped because of an error condition).

Important: If you run stage 1 and stage 2 processing using the IMSCTRL macro without specifying MODBLKS, a cold start is required. Failing to specify MODBLKS causes updates to IMS.SDFSRESL, which requires the new IMS.SDFSRESL to be copied and, therefore, a cold start.

MSVERIFY
Generates a subset of the online control blocks required for execution in a multiple-IMS system configuration. MSVERIFY also generates the bind JCL and control statements required to place the Multiple Systems Verification utility into the IMS.SDFSRESL data set. This subparameter should be specified to generate a test system to verify, before attempting online execution, the consistency of definitions in an MSC configuration. This keyword causes an assembly error if the multiple systems definition statements are not present. If MSVERIFY is specified, MSVID is required.

NUCLEUS
Generates an IMS nucleus for the IMS control program region and control block modules for all IMS control blocks. This subparameter should be specified when generating an alternate nucleus that includes new or deleted old system features, such as additional terminal support or conversational support. A NUCLEUS specification generates the MSC Verification utility control blocks if MSC definitions are present. If NUCLEUS is specified, you do not need to perform a separate system definition, with MSVERIFY specified, to obtain the blocks.

To change the control blocks in an existing nucleus, do a CTLBLKS definition. To generate a new nucleus, do a NUCLEUS definition.

The SMP JCLIN process should be run after the stage 1 of all definition types.

ON-LINE
Moves required modules from the IMS distribution libraries to the user's libraries, generates system procedures, and generates those modules that contain the CTLBLKS and NUCLEUS options for an IMS system. It generates the assembler language and bind statements for all operating systems and VTAM-sensitive modules that are supplied in source form.

You must specify ON-LINE with the hyphen.

TMINAME=
Specifies the one- to four-character name that becomes the instance name for the Transport Manager Subsystem (TMS) and the subsystem name (SSN) for an RSR active or tracking subsystem. If TMINAME= is not specified, IMS uses a series of blanks as the default, which associates IMS with the TMS that used a default instance name.

Usage information

Guidance information is provided to help you select the CTLBLKS, NUCLEUS, ON-LINE, or MODBLKS system definition type to use when modifying an existing ALL type of system definition. The guidelines can help you determine whether modification of parameters at execution can preclude the need for a new system definition.

Sample IMSCTRL macro JCL

The JCL below is an example of the IMSCTRL macro with:
- Three message regions
- Message region job class A (default)
- Job message class C
- Online and batch

```
IMSCTRL SYSTEM=(VS/2,(ALL,DB/DC),390),                          X
     MAXREGN=(3,52K,,C)
```

IMSGEN macro

Use the IMSGEN macro to specify the assembler and binder data sets and options, and the system definition output options and features.

IMSGEN must be the last IMS system definition macro in the stage 1 input stream, and it must be followed by an assembler END statement.

- "Dynamic definition" on page 352
- "Supported environments" on page 352
- "Syntax" on page 352
- "Positional parameters" on page 354
- "Keyword parameters" on page 355
- "IMSGEN macro statement - example 1" on page 362
- "IMSGEN macro statement - example 2" on page 362

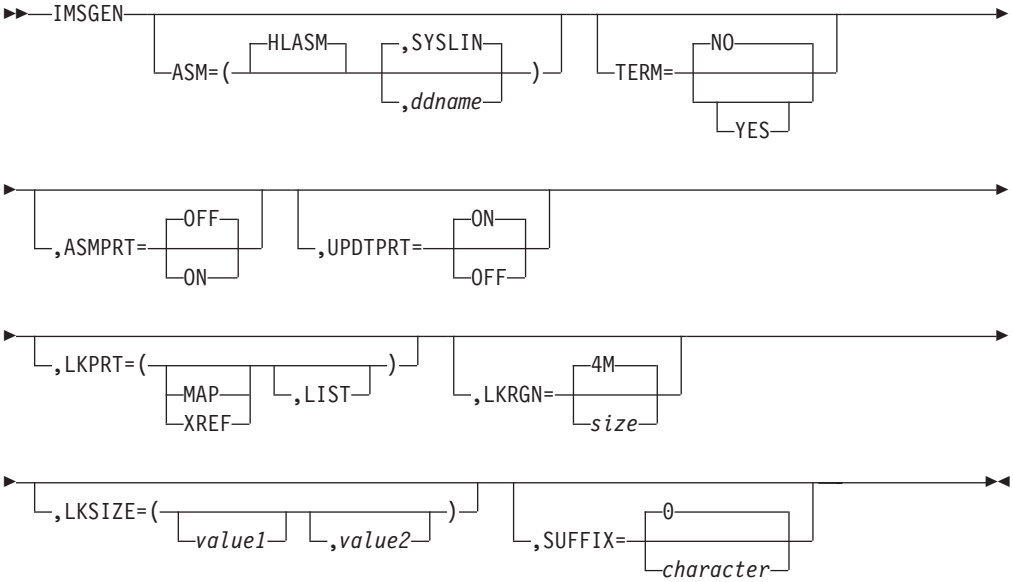
Dynamic definition

Assembler and binder data sets and options cannot be dynamically defined.

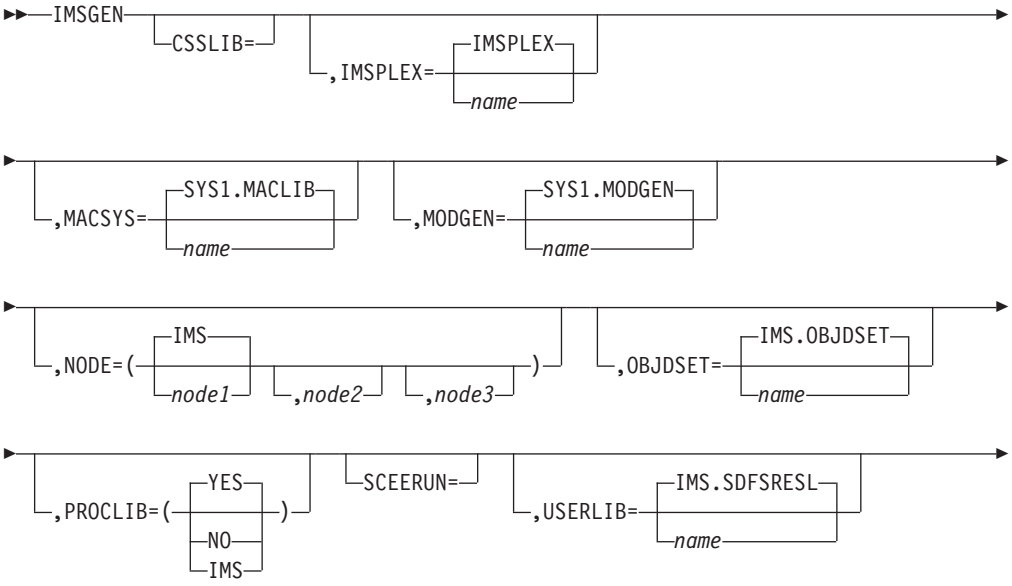
Supported environments

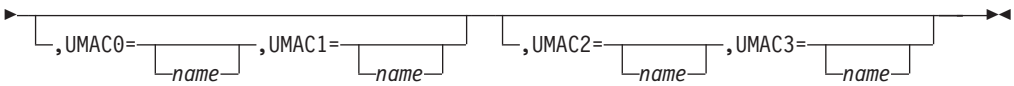
The IMSGEN macro can be used in the DB/DC, DBCTL, and DCCTL environments.

Syntax

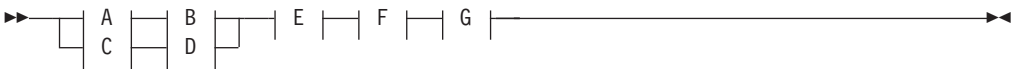


IMS data set options

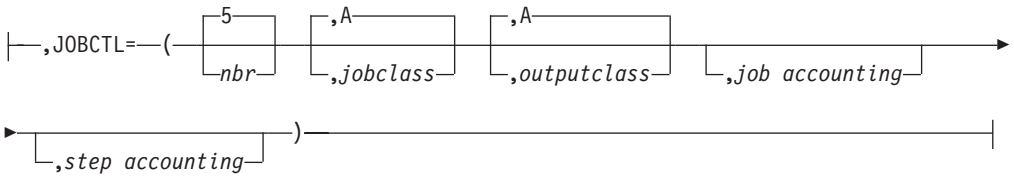




Job Control Language options

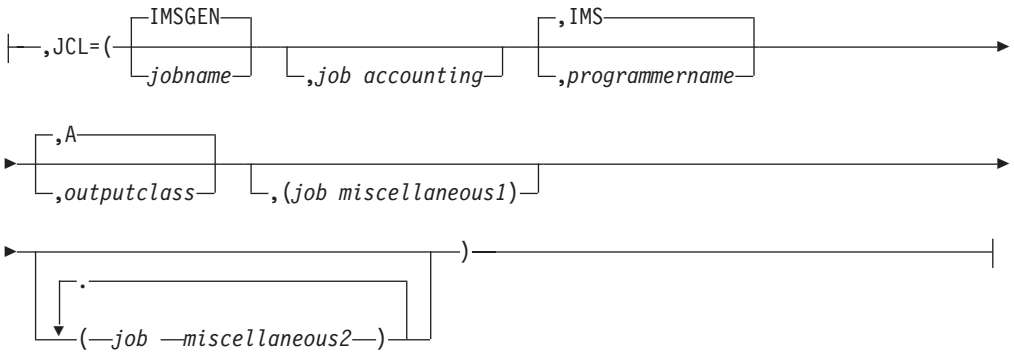


A:

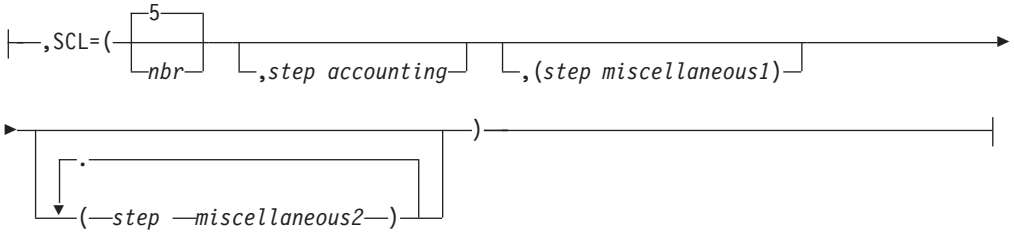


B:

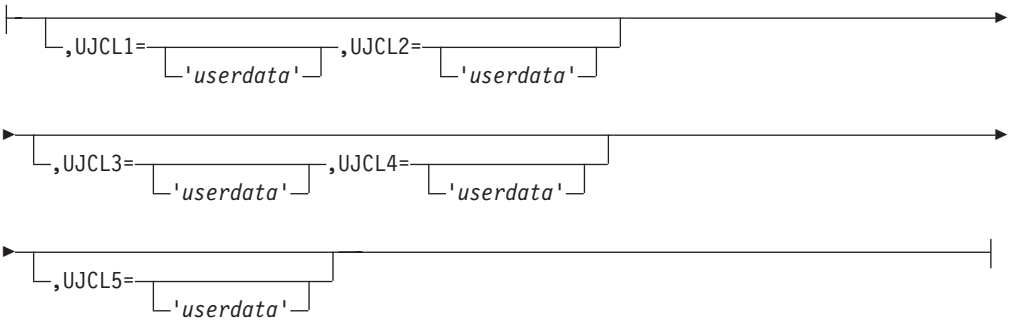
C:



D:



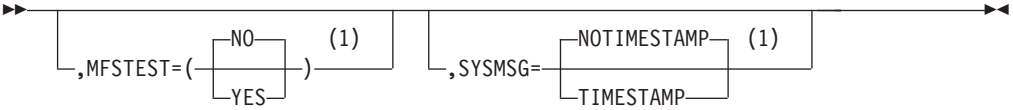
E:



F:

G:

General communication options



Notes:

1 If you specify the COMM macro, the general communication option parameters are overridden.

Security options

Notes:

1 If you specify the COMM or SECURITY macro, the security option parameters are overridden.

Positional parameters

The IMSGEN macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

The keyword parameters for the IMSGEN macro are divided by the possible options:

- Assembler and binder
- IMS data set
- JCL
- General communication
- Security

Assembler and binder options

ASM=

The first parameter specifies the assembler JCL to be produced for the stage 2 assembly steps. High Level Assembler, the high-level assembler, is the only valid first parameter.

For a large system assembly, the high-level assembler is used in the stage one assembly cycles.

The second parameter can be used to specify the ddname to be used for the assembler object output. If the ddname is not specified, the ddname is SYSLIN.

Avoid the use of DD names other than SYSLIN, SYSGO, and SYSPUNCH if you are using the z/OS System Modification Program (SMP) to apply maintenance to your IMS libraries. SMP does not recognize other DD names during JCLIN processing.

The option of changing the object output ddname is intended to accommodate those users who have installed an assembler and altered its ddname defaults. The stage two assembler execution statement options are PARM='OBJECT,NODECK,NODBCS'. The object file output ddname is the one that must be coded in place of the default value of SYSLIN. Also, you must use the default assembler option ALIGN for assemblies of IMS modules.

Recommendation: Use NOUSING and FLAG(NOPUSH) options to avoid assembler errors. Because the assembler code that is created might contain lowercase characters, you should not specify the assembler option COMPAT(CASE).

ASMPRT=

Specifies the assembler print options for those assembler job steps produced by IMS system definition.

ON specifies that assembly listings are to be generated.

OFF specifies that assembly listings for inline assemblies are not to be generated. The default is OFF.

LKPRT=

Specifies binder print options for those binder job steps produced by IMS system definition. These print options include:

Value	Print Option
MAP	Module map
XREF	Cross-reference table (XREF includes the MAP option)
LIST	List of control statements in statement image format

If this parameter is omitted, only binder error messages, if any, are printed.

LKRGN=
Specifies a region size value to be placed on the generated EXEC statements for execution of the binder. This parameter can be specified in kilobytes or megabytes.

In kilobytes, the value can be 1 through 7 decimal numbers (1 through 2096128).

In megabytes, the value can be 1 through 4 decimal numbers (1 through 2047).

LKSIZE=
Specifies the values to be placed in the SIZE parameter of the EXEC statements for the binder job steps.

value1
Specifies the maximum number of bytes of main storage and available virtual storage. This value can be specified in the form *n* (where *n* represents the actual number of bytes of virtual storage, not to exceed 16384000) or *n*K (where *n* represents the number of 1 KB blocks of virtual storage, not to exceed 16000 KB).

value2
Specifies the number of bytes of storage to be allocated for the load module buffer. It is expressed in the form *n* (where *n* cannot exceed 65520) or *n*K (which cannot exceed 63 KB).

Default values are not supplied in LKSIZE for the binder SIZE parameter. If *value1* and *value2* are omitted, the Binder SIZE defaults are used.

Related reading: For a more detailed description of the SIZE option, see *DFSMS/MVS Program Management*.

SUFFIX=
Specifies the alphanumeric suffix character appended to the generated composite control block, nucleus, MFS device characteristics table, and security directory module names when they are put into the IMS.SDFSRESL or IMS.MODBLKS data sets. The IMS online nucleus name always starts with DFSVNUC. The suffix character supplies the eighth character of the nucleus names. If the suffix character equals 0, the online nucleus name is DFSVNUC0. This concept allows the system user to generate multiple IMS systems for use in one environment where the characteristics of each system vary. The default is 0.

The modules that receive the suffix are shown in the table below. If no IMSGEN macro is coded with the default SUFFIX of 0, subsequent functions might fail due to the lack of the default, DFSUDT0.

Table 54. Modules that receive the suffix character

DBFSCD0x, DBFSCD1x	DFSCLVxx	DFSRCTEx
DFSBLK0x	DFSDDIRx	DFSRSR0x
DFSCLCxx	DFSFEDBx	DFSSCD0x, DFSSCD1x
DFSCLIDx	DFSFRB0x, DFSFRB1x	DFSSMB0x
DFSCLL0x	DFSISDBx	DFSSYS0x
DFSCLL1x	DFSISDCx	DFSUDT0x
DFSCLRxx	DFSOPL0x	DFSVNUCx
DFSCLSxx	DFSPDIRx	

The suffix is also appended to the intermediate control block assemblies for SMP/E purposes.

TERM=
Specifies the assembler TERM/NOTERM option for stage 2 assembler job steps produced by the IMS system definition.

YES specifies that the TERM option is in the PARM= field for each of the stage 2 assembler job steps. YES also specifies that the //SYSTERM DD statement is to be built in each of the stage 2 assembler job steps.

NO specifies that the NOTERM option is included in the PARM= for each of the stage 2 assembler job steps. The default in NO.

UPDTPRT=
Specifies whether the IEBUPDTE SYSPRINT is to be produced by IMS system definition stage 2.

ON specifies that the IEBUPDTE SYSPRINT is to be generated. The default is ON.

OFF specifies that the IEBUPDTE SYSPRINT is to be suppressed. This includes the suppression of any errors or warnings issued by the IEBUPDTE utility.

IMS data set options

CSSLIB=
Specifies the name of the z/OS (R2V10 or above) callable services library data set that is part of the concatenation of the STEPLIB DD statement of the DFSJMP and DFSJBP procedures. The specified data set name cannot be longer than 44 alphanumeric characters. If the keyword is not specified, a default value of SYS1.CSSLIB is used as the data set name.

No system definition is needed unless you are using the DFSJMP or DFSJBP procedure library members created by system definition.

IMSPLEX=
Specifies the data set qualifier (or qualifiers) for the OLCSTAT data set. This name should be the same for all IMS systems running in the same IMSplex because they all share the same OLCSTAT data set.

The name specified on the IMSPLEX option can be from one to 35 characters in length (including embedded periods). The first character of each level of qualification must be a letter, @, $, #, or a number (zero through 9). The IMSPLEX name must conform to z/OS data set naming conventions. The default name is IMSPLEX.

MACLIB=
This keyword is obsolete.

MACSYS=
Specifies the name for the MACLIB data set to be used in the stage 2 assemblies. The specified data set name cannot be longer than 44 alphanumeric characters.

MODGEN=
Specifies the name for the MODGEN data set to be used in the stage 2 assemblies. The specified data set name cannot be longer than 44 alphanumeric characters.

NODE=
Specifies the node assigned to IMS data set names.

node1
Specifies the node to be assigned to all IMS data set names, as related to the use and definition of the data set names, by IMS system definition. The specified node can contain from 1 to 26 characters, including embedded periods. The first character of each level of qualification must be a letter, @, $, or #. The remaining seven characters can be any of the preceding characters and the numerals zero through nine. The node name must conform to z/OS data set naming conventions. The default node generated is IMS.

node2
Specifies the node to be assigned to the IMS data set names listed in Table 55. This node overrides the *node1* assignment for these specific data sets. The character set restrictions are as defined for *node1*.

node3
Specifies the node to be assigned to the IMS data set names listed in Table 55. This node overrides the *node1* assignments for these specific data sets. The character set restrictions are as defined for *node1*.

Table 55. Node assignments for IMS data set names

node1		node2	node3
• MSDBCP1 • MSDBCP2 • MSDSCP3 • MSDBCP4 • MSDBDUMP • MSDBINIT • QBLKS • QBLKSL • SHMSG • SHMSGL • LGMSG • LGMSGL • MODSTAT • MODSTAT2 • IMSMON • IMSLOG	• IMSLOG2 • RDS • RDS2 • PGMLIB • PSBLIB • DBDLIB • ACBLIB • ACBLIBA • ACBLIBB • REFERAL • FORMAT • FORMATA • FORMATB • TFORMAT • SYSOnnn	• PROCLIB • SDFSRESL • SDFSMAC • JOBS • OPTIONS • MODBLKS • MODBLKSA • MODBLKSB	• ADFSMAC • ADFSLOAD • ADFSSRC • LGENIN • LGENOUT

OBJDSET=
Specifies the name of a cataloged partitioned data set into which assembler object modules are assembled during stage 2 of IMS system definition. The specified data set name must not be longer than 44 alphanumeric characters. If this parameter is not supplied, these modules are placed in IMS.OBJDSET.

PROCLIB=
Specifies whether system procedures are to be generated.

YES
Generates all system procedures that become the system definition-supplied cataloged procedures in IMS.PROCLIB. This is the default.

NO Does not generate system procedures. Use caution when specifying PROCLIB=NO. A new system definition does not reflect the proper defaults because the updated procedures are not created.

IMS
Generates only the IMS procedure if SYSTEM=CTLBLKS, NUCLEUS, or

ON-LINE in the IMSCTRL macro. If SYSTEM=BATCH or ALL in the IMSCTRL macro, and PROCLIB=IMS is specified, it is ignored, and all system procedures are generated.

SCEERUN=
Specifies the name of the System C Runtime Library data set that is part of the concatenation of the STEPLIB DD statement of the DFSJMP and DFSJBP procedures. The specified data set name cannot be longer than 44 alphanumeric characters. If the keyword is not specified, a default value of CEE.SCEERUN is used as the data set name.

USERLIB=
Specifies the name of the library in which user-written routines, such as message edit routines, are to be included in the generated IMS nucleus. If this operand is omitted, the library containing the routines is assumed to be node(2).SDFSRESL data set. The name of the data set, which must be cataloged, must not be longer than 44 characters.

UMAC0=
Specifies the name of a data set to be put at the top of the SYSLIB concatenation of the stage 2 assembly job steps. The specified data set name cannot be longer than 44 alphanumeric characters.

UMAC1=, UMAC2=, and UMAC3=
Specifies the name of a data set to be concatenated to the SYSLIB DD statement of the stage 2 assembly job steps. The specified data set name cannot be longer than 44 alphanumeric characters.

Typically MACSYS is used for either SYS1.MACLIB or SYS1.AMACLIB, and MODGEN is used for either SYS1.MODGEN or SYS1.AMODGEN. If the SYS1.AMACLIB and SYS1.AMODGEN data sets are specified, the UMAC1= keyword must specify SYS1.ATSOMAC. This ensures that all macros that might be referenced are at the same level.

If no value is provided, the corresponding statement is dropped from the SYSLIB DD.

JCL options

Use the first two parameters (JOBCTL and PRTY) or the last two (JCL and SCL).

JOBCTL=
The following parameters are valid:

nbr
Specifies the number of steps per JOB statement to be produced by system definition stage 1 for execution of stage 2. The maximum allowable value is 255; the default is 5. Regardless of the value specified, a JOB statement is produced for the beginning of the bind steps and for the nucleus bind step (if applicable).

jobclass
Specifies the job class to be generated on the stage 2 JOB statement. The default is A.

outputclass
Specifies the output class to be generated for the stage 2 JCL. The default is A.

job accounting, step accounting, or both
Specifies that job accounting data, step accounting data, or both, be placed

in the stage 2 JCL. The length of each set of (job and step) accounting data cannot exceed 50 bytes. If job accounting data is specified, a programmer name of IMS is provided.

When the accounting information fields contain special characters (except hyphens), special considerations apply to the manner in which this field is specified. See the examples for the JCL.

PRTY=
Specifies the priority placed on the JOB statements for IMS system definition stage 2 jobs. The default is priority 0.

JCL=
The following parameters are valid:

jobname
Specifies a maximum of six alphanumeric characters to be used as the first portion of the generated job names. The last two characters of the job names are internally generated, sequentially incremented, hexadecimal numeric values representing the relative position of each job in the stage 2 stream. The default is IMSGEN.

The character set that can be used is described on the node keyword.

job accounting
Specifies job accounting data to be placed in the stage 2 JCL. The length of the accounting data cannot exceed 50 bytes.

When the accounting information fields contain special characters (except hyphens), special considerations apply to the manner in which this field is specified. See the examples for the JCL.

programmername
Specifies the programmer name to be placed in the stage 2 JCL. The default is IMS.

outputclass
Specifies the output class to be generated for the stage 2 JCL. The default is A.

job miscellaneous 1 & 2
Specifies any additional parameters the user can desire to have placed in the stage 2 JOB statements. Length of this parameter cannot exceed 60 bytes. If you code this parameter, do not code the MSGCLASS parameter.

MFSDFMT=
Specifies whether (YES) or not (NO) the default message format screens are built in stage 2 of IMS system definition. If MFSDFMT=NO, stage 2 does not have the default MFS screens generated in stage 2. The default is YES.

ONEJOB=
Subparameter1 specifies whether (YES) or not (NO) a single job is to be built for stage 2. The default is NO. If ONEJOB=NO, stage 2 has a multiple of job steps as specified in JOBCTL= subparameter 1.

Subparameter 2 specifies whether (YES) or not (NO) condition code checking for the bind steps is to be performed. If ONEJOB=(YES,YES) is specified, condition code checking is done in the bind steps. These steps are bypassed if any previous step did not complete with a condition code of 0. The default for subparameter 2 is YES if subparameter 1 is YES.

When ONEJOB= is specified for a single JOB statement, JOBCTL= and SCL= subparameter 1 are ignored.

If condition code checking is being specified for SCL= and ONEJOB= subparameter 2 is YES, a JCL error occurs due to the specification of duplicate COND= parameters.

SCL=
The following parameters are valid:

nbr
Specifies the number of steps per JOB statement to be produced by system definition stage 1 for execution of stage 2. The maximum allowable value is 255; the default is 5. Regardless of the value specified, a JOB statement is produced for the beginning of the bind steps and for the nucleus bind step (if applicable). This parameter is ignored in two cases:

- The first job that updates the IMS options data set or the IMS PROCLIB has one or two job steps for system definitions (one job step for each applicable update). IMS requires this for compatibility with the SMP/E GENERATE function.
- The last job is a multi-step job with two or four steps (for system definitions that create MFS libraries. This is caused by steps 2 and 4 (if present) being dependent on the respective preceding step.

step accounting
Specifies step accounting data to be placed in the stage 2 JCL. The length of the accounting data cannot exceed 50 bytes.

When the accounting information fields contain special characters (except hyphens), special considerations apply to the manner in which this field is specified. See the examples for the JCL.

step miscellaneous
Specifies any additional parameters that you might want to place in the stage 2 EXEC statements. The length of this parameter cannot exceed 50 bytes. Do not code REGION= here, because internal IMS processing automatically generates this parameter. Do not code COND= if the MFSDFMT=NO keyword is not coded. Some of the default MFS job steps have COND= already coded.

UJCL1=,...,UJCL5=
Specifies the contents of a JCL statement that is placed between the JOB statement and the first EXEC statement in the stage 2 job stream. The specified statement cannot be longer than 60 characters. No other syntax checking of the parameter contents is done.

The intent of these keywords is to provide comment or JES control statements in the stage 2 job stream. For example, these parameters can be used to add ROUTE statements, or JOBLIB DDs.

If no value is provided, the corresponding statement is dropped from the SYSLIB DD.

General communication options

All communication operands specified are ignored and a warning message is issued if the COMM macro is specified. For information about which security specifications on macros will override other macros, see “Usage information” in SECURITY macro (System Definition).

MFSTEST=
Specifies whether (YES) or not (NO) the Message Format Service test facility,

MFSTEST, is to be included in the generated system. YES is invalid for systems not containing MFS terminals. The default is NO.

The use of MFSTEST can degrade IMS performance due to MFSTEST use of the communication line buffer pool.

SYSMSG=
Specifies whether (TIMESTAMP) or not (NOTIMESTAMP) the system message time-stamp facility is to be included for the generated system. The default is NOTIMESTAMP.

When TIMESTAMP is specified, the time that a message was generated is inserted between the message number and the message text for each message in the table.

Security options

All security options are ignored or overridden if the COMM or the SECURITY macro is specified.

SECCNT=
Specifies the maximum number of terminal and password security violations to be accepted per physical terminal and transaction command violations per transaction before master terminal notification of such violation. The default is 0, which nullifies notification to the master terminal. The number specified can range from 0 through 3.

If SECCNT is not 0, the master terminal is notified for every violation.

IMSGEN macro statement - example 1

The following figure is one example of an IMSGEN macro statement.

```
                                                    72
IMSGEN    ASMPRT=ON,LKPRT=(MAP,LIST),SUFFIX=9,      X
          OBJDSET=IMSXXX.OBJ,                       X
          USERLIB=IMSXXX.USER,                      X
          PROCLIB=YES,NODE=IMSXXX
```

The example JCL indicates that:

- An assembly listing is requested.
- Binder print options are MAP and LIST.
- Composite control block, nucleus, map module, nucleus and security directory blocks module suffix is 9.
- The cataloged partitioned data set into which assembler load modules are to be placed is IMSXXX.OBJ.
- The library containing user routines is IMSXXX.USER.
- System procedures are requested.
- All IMS data set names are to be prefixed by the node IMSXXX (that is, IMSXXX.SDFSRESL).

IMSGEN macro statement - example 2

The examples of the IMSGEN macro statement shown here compare the JOB and EXEC statements produced using the JOBCTL and PRTY keywords to the statements produced using the JCL and SCL keywords.

```
                                                                   72
IMSGEN JOBCTL=(4,D,A,(P01,9987)),PRTY=8

 //IMSGEN1 JOB (P01,9987),IMS,MSGLEVEL=1,MSGCLASS=A,
         CLASS=D,PRTY=8

IMSGEN JCL=(OSCAR,(P01,9987),FELIX,,(MSGLEVEL=1,       X
         PRTY=3,TYPRUN=HOLD))

 //OSCAR01 JOB (P01,9987),FELIX,MSGLEVEL=1,MSGCLASS=A,
         PRTY=3,TYPRUN=HOLD

IMSGEN JCL=(GEORG,(P01,"9/12/99"),IMS,D,(PRTY=3)),     X
       SCL=(3,("/83468"),(PARM=123))

 //GEOG01 JOB (P01,"9/12/99"),
 // IMS,
 // PRTY=3,
 // MSGCLASS=D
 //STEP1 EXEC PGM=IEBUPDTE,PARM=NEW,
 //            ACCT=("/83468"),
 //            PARM=123
```

When the accounting information consists of more than one subparameter, you must enclose the information in parentheses, not apostrophes. For example:
`(5438,GROUP6).`

If any of the subparameters contains special characters (except hyphens), enclose that subparameter in double apostrophes, not quotation marks. For example:
`(5438,''12/15/99:'').`

Related reference:

"DFSDSCMx member of the IMS PROCLIB data set" on page 689

LINE macro

Use the LINE macro to describe both switched and nonswitched communication lines to IMS. The LINE macro provides the address and characteristics of one line in the line group specified by the LINEGRP statement.

The LINE macro statement describes communication lines. If the line described has only one terminal attached, only one TERMINAL macro instruction is included after the LINE macro instruction. Multiple TERMINAL macro instructions are included if the description is for a multidrop line. For nonswitched lines, multiple NAME macro instructions can be included after each TERMINAL macro instruction that follows a LINE macro instruction. Each LINE macro instruction must be followed by at least one TERMINAL macro instruction.

Dynamic definition

Communication lines cannot be dynamically defined.

Supported environments

The LINE macro can be used in the DB/DC and DCCTL environments.

Syntax

All lines

All lines except spool

Disk, tape, and spool lines only

Positional parameters

The LINE macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

ADDR=
Specifies the address of the communication line as defined in the Transmission Control Unit. The address value is three or four hexadecimal digits between 0000 and FFFF. All line addresses specified in the system definition must be unique values. This operand is used only to generate IMS DD statements for cataloged procedures. It is not allowed if the terminal is a 3270 local. This operand is not required for SPOOL line groups.

This operand is ignored for UNITYPE=PRINTER when a JES spool data set (SYSOUT=X) is to be used rather than a local printer. When XRF is used, a second optional device address can be specified with the ADDR keyword.

BACKUP=
Specifies (for XRF only) the control of the automatic restart after takeover. Use only when HSB=YES is specified on the IMSCTRL macro.

X is a numeric integer from 1 (lowest) to 7 (highest), inclusive, that specifies the priorities for reestablishing the session. If the entire keyword or the operand is omitted, the default is 4. Specifying BACKUP=NO suppresses the automatic restart of the devices after a takeover.

BUFSIZE=
This operand is only valid for DISK, TAPE, and SPOOL lines.

For DISK, TAPE, or SPOOL lines, this operand is used to specify, in bytes, the maximum output buffer size to be used for the line. The operand is required. The minimum specification is 16; the maximum specification is 32 760.

Use of SYSIN local card reader: When the DD statement for a line (specified in the LINEGRP macro statement as UNITYPE=READER) does not allocate a card reader device, the following conditions apply:

- If IMS is running as a batch program, the DD statement referring to the line must contain DCB=BLKSIZE=80.
- If IMS is running as a system task, the SYSIN test stream must be placed in a sequential data set with the DCB attributes RECFM=F and BLKSIZE=80. The SYSIN test stream must be pointed to by a DD statement for a reader line within the IMS procedure.

Although VTAM requests are priority-ordered by IMS, the active requests can be completed in any order because of internal VTAM conflicts and pacing.

LINEGRP macro

The LINEGRP macro defines the beginning of a set of macro instructions that describe your telecommunications system.

The LINEGRP macro statement is the first in a series of macros. The IMS macros that can be part of a LINEGRP macro set are LINE, TERMINAL, and NAME.

Dynamic definition

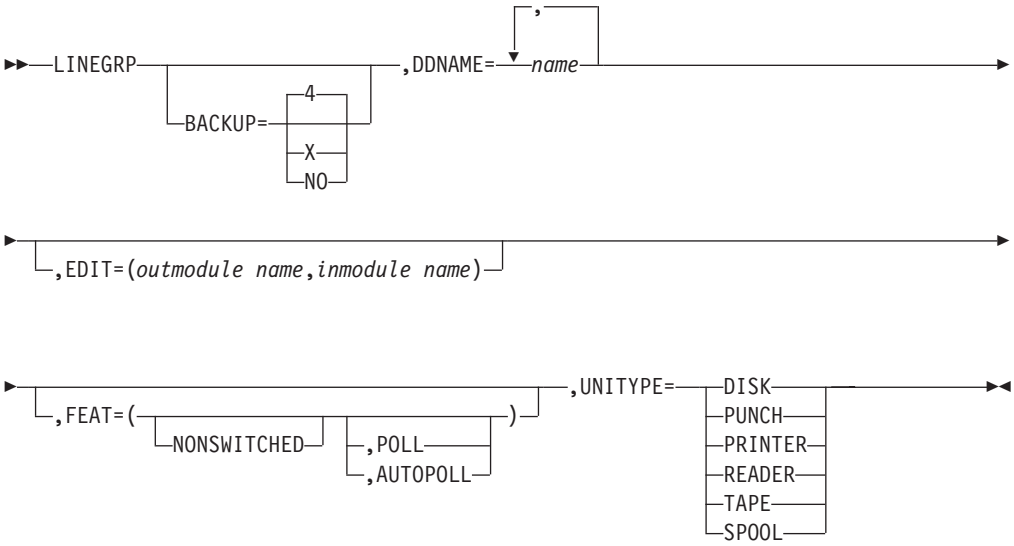
Your telecommunications system cannot be defined dynamically.

Supported environments

The LINEGRP macro can be used in the DB/DC and DCCTL environments.

Syntax

All line groups



Positional parameters

The LINEGRP macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

BACKUP=

Specifies (for XRF only) the control of automatic restart after takeover. Use only when HSB=YES is specified on the IMSCTRL macro.

X is a numeric integer from 1 to 7, inclusive, which specifies the priority assigned to reestablishing the session. When either the keyword or the operand is omitted, the default is 4. Specifying BACKUP=NO suppresses the automatic restart of the devices after a takeover.

DDNAME=

Specifies a one- to eight-character name that associates the generated DCB for this line group, within the IMS nucleus control block, with the generated JCL DD statements identifying the lines within the line group. If the line group being defined is a SPOOL line group, a list of up to 20 names can be specified: 255 SPOOL data sets, but no more than 20 names per line group can be specified. Each SPOOL name specification is associated, in the order in which it was specified, with the sequentially generated SPOOL data set names produced for the SPOOL print procedures. These procedures are generated and named sequentially for each SPOOL line group.

The names must begin with an alphabetic character, and the operand is required. The following names cannot be used as LINEGRP ddnames:

• DFSRESLB • DUMP • IEFRDER • IEFRDER2 • IMSACB • IMSDBL • IMSDILIB	• IMSLOG • IMSLOGR • IMSLOGR2 • IMSLOG2 • IMSMON • IMSRDS • IMSSPA	• IMSTFMT • IMSUDUMP • LGMSG • MSDBCP1 • MSDBCP2 • MSDBDUMP	• MSDBINIT • PRINTDD • PROCLIB • QBLKS • SHMSG

Recommendation: Allocate at least two data sets.

EDIT=

Specifies a one- to eight-character, user-supplied physical terminal output edit routine and a one- to eight-character, user-supplied physical terminal input edit routine for the terminal type in this line group.

This routine cannot be the same as the one that is used on a TRANSACT EDIT= parameter.

FEAT=

Specifies that this group of telecommunication lines is leased (NONSWITCHED). This is the default.

Polling features are ignored, and no polling list is generated if any of the following values are specified for UNITYPE= keyword: READER, PRINTER, PUNCH, TAPE, DISK, and SPOOL.

Direct SYSOUT specifications (PRINTER, PUNCH, TAPE, or DISK) do not require that a specific device type is assigned at IMS execution time. All specifications except PRINTER affect only default BUFSIZE or generated JCL. Specifying PRINTER results in execution time being translated to the 48-character set, lowercase is translated to uppercase, and all other codes translated to periods(.). If a specification other than PRINTER is used, no

translation occurs. If a line not generated as a PRINTER is allocated to a printer having the universal character set feature (UCS), and fold mode operation is used, unprintable characters print as extraneous alphanumerics.

UNITYPE=
Specifies the terminal device type contained in this line group.

Valid UNITYPE parameters are DISK, PUNCH, PRINTER, READER, TAPE, and SPOOL.

When a JES SPOOL DATASET (SYSOUT) is used, UNITYPE = PRINTER must be specified.

Sample LINEGRP macro JCL

The JCL example below shows the LINEGRP macro and the resulting DDNAME/DSNAME and print procedure associations for a SPOOL line group.

```
LINEGRP DDNAME=(NAME1,NAME2),UNITYPE=SPOOL

LINEGRP DDNAME=(NAMEA,NAMEB),UNITYPE=SPOOL
```

The SPOOL print procedure for line group 1 is named IMSWT000. The data set names produced for this print procedure are IMS.SYSO1 and IMS.SYSO2.

The SPOOL print procedure for line group 2 is named IMSWT001. The data set names produced for this print procedure are IMS.SYSO3 and IMS.SYSO4.

The corresponding JCL statements generated for the IMS online execution procedure are as follows:

```
 //NAME1  DD  DISP=SHR,DSNAME=IMS.SYSO1
 //NAME2  DD  DISP=SHR,DSNAME=IMS.SYSO2
 //NAMEA  DD  DISP=SHR,DSNAME=IMS.SYSO3
 //NAMEB  DD  DISP=SHR,DSNAME=IMS.SYSO4
```

If UNITYPE=SPOOL on the LINEGRP macro statement, DD names used in the preceding JCL must agree with the actual DD names specified on this statement.

MSGQUEUE macro

Use the MSGQUEUE macro to define the characteristics of the three message queue data sets: QBLKS, SHMSG, and LGMSG. The information you specify in this macro is also used in a shared-queues environment. The MSGQUEUE macro is required for all DB/DC and DCCTL systems.

You must include the MSGQUEUE macro statement when the type of definition specified in the IMSCTRL macro statement is ALL, ON-LINE, CTLBLKS, or NUCLEUS.

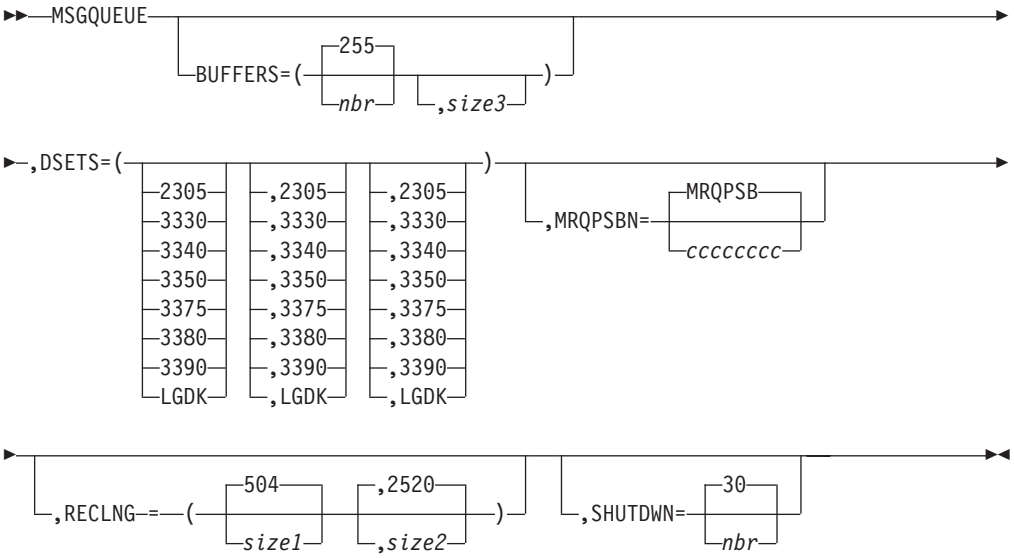
Dynamic definition

The characteristics of the three message queue data sets cannot be dynamically defined.

Supported environments

The MSGQUEUE macro can be used in the DB/DC, DBCTL, and DCCTL environments.

Syntax



Positional parameters

The MSGQUEUE macro statement does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

BUFFERS=

Specifies the number of buffers to allocate for message queue management, and the block size to be assigned for all three message queue data sets. If *nbr* is not specified, a default of 255 is used. The minimum *nbr* specification is 3; the maximum is 9999. At execution time, the minimum value for buffers in a shared queues environment is increased to 200. If *size3* is not specified, it is calculated by the following formula:

```
SIZE3 = ((SIZE1+SIZE2-1)/SIZE1) x ((SIZE1+55)/56) x 56
```

For the division operations in this formula, only the integer portion of the result is used; the fractional portion of the result is truncated.

This calculation can leave null space at the end of the buffer block for short and long message records.

size1 and *size2* should be an even multiple of the QBLKS LRECL of 56.

If the RECLNG defaults of *size1* (504) and *size2* (2520) are used, then *size3* of BUFFERS equals 2520. The maximum specification for *size3* is 30632, or the track length of the device on which the data set resides, whichever is smaller. All sizes specified are rounded up to multiples of 4.

RECLNG can be changed before a cold start of IMS, but if a smaller value is specified before a BUILDQ restart, the restart might abend.

You should not allocate less space for a restart of IMS than was allocated for the prior execution. If a smaller logical record length has been allocated, or if less space for the Message Queue data sets has been allocated for restart with BUILDQ, the restart process may terminate abnormally.

DSETS=

Specifies the device types on which the three message queue data sets (IMS.QBLKS, IMS.SHMSG, and IMS.LGMSG, respectively) are to reside. If all three data sets are to reside on the same device type, you can specify just the first parameter.

You can specify the following device types: 2305, 3330, 3340, 3350, 3375, 3380, 3390, or LGDK.

When using a 3350, specify the drive format used—3330 or 3350.

LGDK is a generic definition for disk drives that have a track size equal to or greater than 32767 bytes. The 3375, 3380, 3390, and future devices can also be defined as LGDK.

MRQPSBN=

Specifies the one- to eight-character alphanumeric name of the IBM IMS Queue Control Facility for z/OS (QCF) program specification block. This parameter allows you to override the default MRQ PSB name. If not specified, the default name of MRQPSB is used by the QCF licensed program. The MRQ program can be either defined in system definition with the APPLCTN macro, or, if DRD is enabled, defined dynamically with the `CREATE PGM NAME(MRQPSB) SET(BMPTYPE(Y),SCHDTYPE(PARALLEL))`. The default name is MRQPSB. Regardless of whether you use the default name, you must include a PSB for the message region in the stage 1 system definition. If you do not use the default name, you must specify a name for the MRQPSBN block with the MRQPSBN keyword. If an error is made in specifying the MRQPSBN PSB name, a warning message is issued, and the default name is used.

Note: The MRQPSBN block is for the exclusive use of MRQ and QCF programs. It cannot be used by other user application programs running as MPPs, BMPs, or IFPs. If an attempt is made to use the PSB block by some other user application program, IMS returns a status code of MR and an AIBRETRN code of 000000F0 on the call.

RECLNG=

Specifies the logical record lengths for the short and long message queue data sets, respectively.

The minimum value for *size1* is 392. This size includes user data length of 64 bytes plus minimum prefix size of 328. The value specified for *size1* must be large enough to accommodate the ISC/LU6.1 (type 84) prefix and the Conversation Extension (type 8D) prefix. If shared queues are used, it must also include an additional X'28' bytes. If *size1* is not specified, a default of 504 is assigned.

The minimum value for *size2* is 1176. If *size2* is not specified, a default of 2520 is assigned. The value specified for *size2*:

- Must be equal to or greater than *size1*.
- Cannot exceed the track length of the device on which the data set resides.
- Must be divisible by 4. If it is not, IMS uses the next greater value that is divisible by four as the value of *size2*.
- Should be even multiples of the QBLKS LRECL of 56.
- Should not exceed the size of the OLDS data set block minus the sum of the block descriptor word length (4 bytes) and prefix item.

If a message does not fit in a large queue record (*size2*), IMS either spans it to other queue records, or rejects the message with an error.

Choose values for *size1* and *size2* based on the sizes of the messages processed by the queue manager. When determining which size message queue record to use, the queue manager calculates the size of the message prefix, subtracts that value from the size of the short message queue buffer, and then doubles the remaining value. The queue manager then compares that value to the average user data length for the destination. If the remainder buffer size is greater than or equal to the average user data size, then the short message queue buffer is used. If the remainder buffer size is less than the average user data size, then the large message queue buffer is used. The length of the message prefix varies based on the IMS system options specified.

SHUTDWN=

Specifies the number of records to be reserved in each data set to allow the system to automatically shut down if the data set becomes filled with unprocessed messages. The maximum valid specification is 32767. The default is 30. In an XRF system, this parameter also reserves the number of records in each of the local message queue data sets (IMS.LGMSGL, IMS.SHMSGL, or IMS.QBLKSL) for shutdown.

The number of records specified should provide enough space in each data set to allow for an orderly shutdown, which depends on message throughput and the number of regions that are scheduled. In general, the following calculation applies:

Maximum number of output messages between synchronization points per application **x** number of regions scheduled + 1 input message per logical terminal active at the same time.

Usage information

Some messages (such as MFS) from an input terminal can span message queue records. If messages span message queue records, the size of the input segment has no limit. When messages do not span message queue records, the maximum allowable size for input segments is *size2* minus the amount for various prefix items included with the data portion of the message queue record.

The maximum size allowed for output segments is *size2* minus the amount for various prefix items included with the data portion of the LRECL (message queue record).

Attention: When an output message segment that spans queue records is formatted by MFS, the message might be truncated or formatted incorrectly. To avoid such problems, increase *size2* to prevent spanning.

Prefix items and sizes for the versions of IMS currently in support are displayed in Table 56, along with usage comments.

Table 56. Message prefix size

Prefix section	IMS Version 10 and later - size (in bytes)	Comments
Basic [1]	64	All messages
System Segment (81)	56	All messages
	64	All messages with MFS
ISC/LU6.1 Prefix (84) [3]	22	All ISC/LU6.1 messages
Extended Prefix (86) [4]	16	All messages

Table 56. Message prefix size (continued)

Prefix section	IMS Version 10 and later - size (in bytes)	Comments
APPC (LU6.2) (87)	128-512	All APPC messages
OTMA (87)	128-4096	All OTMA messages
Security Prefix (88) [2]	22	All messages if RACF or ETO is defined
Workload Manager (89)	24	All messages
System Extension (8A)	24	All messages
MSC Extension (8B)	120	All messages if MSC is defined
TMR (8C)	144	All messages
	144	All messages with shared queues
Conversation Ext. (8D) [3,4]	40	All conversational messages
TM/MSC User Prefix (8E) [6]	5-512	Dependent on using DFSMSCE0 User Prefix exit routine
IMS Internal Prefix (8F) [7]	5-512	Reserved for IMS
User Data Segments [5]	V	All messages

Note:

1. All messages begin with a basic prefix segment that has no prefix code. If the message is a first or only queue buffer message, it has a basic prefix and some or all the prefixes type 81 through 8D. User data segments can follow the IMS prefixes.

   Prefix segments 81 through 8D begin with a 3-byte LLC field where LL is the 2-byte length of the prefix segment (including the LL field) and C is the 1-byte prefix code.
2. The security prefix is included in only the first message buffer if RACF or ETO is used.
3. Because they are not system definition values, the ISC/LU6.1 (type 84) prefix and the conversation extension (type 8D) prefix are always included in the maximum length of the IMS prefix calculated at IMS initialization. If shared queues are used, an additional 40 bytes is added.
4. All segments following the extended prefix header (type 86) are extended prefix segments. The IMS release level determines the size of the extended prefix segments created for a message. When MSC or shared queues are used, a message can be processed by an IMS that is of a different release level than the originating IMS. If the extended prefix segments lengths are greater for the target IMS, IMS can create or increase the size of the extended prefix segments. Extended prefix segments are never shortened.

   Recommendation: In an environment where MSC and shared queues are used and IMS systems are of different release levels, use the highest calculated values for *size1* and *size2* for each IMS.
5. User data segments follow the IMS prefix segments.
6. You can request that a user prefix be inserted into the message prefix by using the MSC Routing exit routine (DFSMSCE0). The size of the user prefix can be from 5 bytes to 512 bytes long.
7. The Workload Router Tool (product number 5697-B87) or another IMS tool can request that an internal prefix be inserted into the message prefix. The size of this user prefix can be between 5 bytes to 512 bytes long.

For messages that span message queue records, the first item (basic) is included in each record of the message. Also, if RACF is generated, the RACF item plus a small (4-byte) system segment is included. All other prefix items, if applicable, appear only in the first record of the message.

The values for *size1* and *size2* should be tuned for your system.

Related reading: For information about how to tune the values of *size1* and *size2* after IMS is operational, refer to "Message queue data set allocation in DB/DC and DCCTL environments" on page 126, and to "Initial Optimizing of IMS Buffer Pools" in *IMS Version 11 System Administration*.

The result of this tuning should yield a value for *size1* such that the I/O activity is equally split between the IMS.SHMSG data set, containing input and output message segments up to the length of *size1*, and the IMS.LGMSG data set, containing all input and output message segments larger than *size1*.

RECLNG can be changed before a cold start of IMS, but if a smaller value is specified before a BUILDQ restart, the restart may abnormally terminate.

If emergency restart procedures using BUILDQ are to be used, reallocation of logical record and data set spaces must be done carefully. The BUILDQ procedure always restores the message queue entries to the relative position in the respective queue data sets they had at the time they were saved. If the logical record or data set size has been decreased, it may be impossible to perform the restart.

Sample MSGQUEUE macro JCL

The following figure shows an example of the MSGQUEUE macro statement with all data sets on a 3380:

```
MSGQUEUE DSETS=(LGDK),BUFFERS=(40,2520),SHUTDWN=200
```

This example indicates that the default sizes for the short and long message queue data sets are used. The size of each buffer block is 2520, allowing 1 long message record, 5 short message records or 45 queue block records to fit into a block.

For non-shared queues, 40 buffers are allocated and 200 records in each data set are reserved to allow the system to automatically shut down. For shared queues, the minimum buffer value of 200 is used, and the SHUTDOWN value is ignored.

MSLINK macro

Use the MSLINK macro statement to define a Multiple Systems Coupling (MSC) logical link to another system. The MSLINK macro can be followed by one or more macro statements that define the logical link paths if any are to be defined for this logical link.

Logical links are assigned to physical links during system definition (using the MSPLINK keyword) or dynamically with `/MSASSIGN` or `UPDATE MSLINK` commands.

You can define a maximum of 1018 logical links (MSLINK macros) in an IMS system, unless the system is defined as a large system by SYSTEM=LGEN in the IMSCTL macro, in which case, you can define only 676 MSLINK macros.

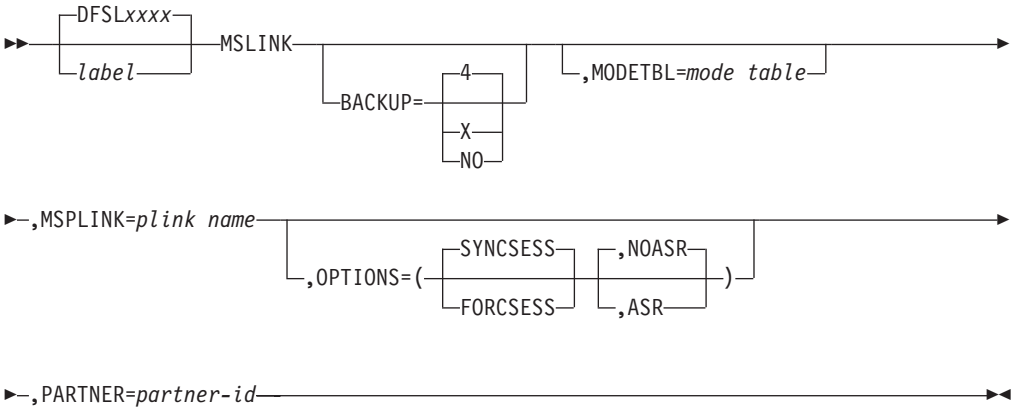
Dynamic definition

Logical links cannot be dynamically defined.

Supported environments

The MSLINK macro can be used in the DB/DC and DCCTL environments.

Syntax



Label field

The label field specifies a 1- to 8-character link name that is used by the type-1 UPDATE and type-2 QUERY and UPDATE commands to identify the logical link. If you do not specify a name, a default name is assigned, DFSL*xxxx*, where *xxxx* is the logical link number that system generation assigns to the link. That name is used in the QUERY and UPDATE commands. The logical link name can be changed with the UPDATE MSLINK command.

Positional parameters

The MSLINK macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

BACKUP=

For XRF-capable IMS systems, controls the automatic restart of VTAM links after an XRF takeover.

When specified on the MSLINK macro, BACKUP overrides switching options specified on the MSPLINK macro. Use only if HSB=YES is specified on the IMSCTRL macro.
X is a numeric integer from 1 to 7, inclusive. It sets priorities for reestablishing the session. When either the BACKUP keyword is omitted or no value is specified for BACKUP on either the MSPLINK macro or the MSLINK macro, the default is 4. NO suppresses the session recovery of the MSC physical link at takeover.

Although requests to VTAM are prioritized by IMS, the active requests can be completed in any order because of internal VTAM conflicts and pacing.

MODETBL=
Specifies the name of the VTAM logon mode table entry (logon mode name) containing the SNA bind parameters to be used when a session is established for this terminal. The maximum is 255 unique names for each IMS system.

This function allows a system definition specification for referencing an entry other than the default entry in the user's VTAM logon mode table.

With this function, if MODETBL= is not specified at system definition, no functional or operational change affects the user.

If MODETBL= is specified at system definition, the specified entry name is used.

You can display the current MODETBL name using either the `/DISPLAY` command or the `QUERY MSLINK` command. You can override the MODETBL name by using any of the following commands:

- The `VARY ACT, LOGON=` command by the network terminal operator
- The `/RST` or the `/CHANGE` command by the master terminal operator
- The type-2 command `UPDATE MSLINK NAME(`*`linkname`*`) SET(MODETBL(`*`modetablename`*`))` through the OM API

If the VTAM-node=name being defined is in another domain (that is, a cross-domain resource), the MODETBL parameter need not be specified.

Related reading: See *z/OS Communications Server: SNA Programming* for additional information.

MSPLINK=
This operand identifies the physical link to which this logical link is assigned. If you do not specify this operand, no assignment to a physical link is made during system definition. You can assign this logical link to a physical link later by using either the /MSASSIGN command or the type-2 command UPDATE MSLINK NAME(*linkname*) SET(MSPLINK(*msplinkname*)). You must assign the logical link to a physical link before communication can be established between the two systems.

When specified, the MSPLINK macro must be defined before the MSLINK macro.

OPTIONS=
This parameter allows you to specify certain options for session initiation and restart. The specifications made on this OPTIONS parameter must be consistent with the MSLINK OPTIONS specification in the partner systems.

The OPTIONS parameter is valid for VTAM links only.

You can specify the following values on the OPTIONS= keyword:

SYNCSESS|FORCSESS
SYNCSESS indicates that session initiation is to be completed only if session resynchronization is successful. Successful session resynchronization occurs when the message sequence numbers of the two logical units in session agree, or when the sequence number of the sender is not less than the sequence number of the receiver. The default is SYNCSESS.

FORCSESS forces the session to be completed regardless of whether session resynchronization is successful.

During IMS execution, this option can be overridden by either the type-1 command /CHANGE LINK *nn* FORCSESS|SYNCSESS|COLDSESS or the type-2 command UPDATE MSLINK NAME(*linkname*) SET(SYNCOPT(FORCSESS|SYNCSESS|COLDSESS)).

ASR|NOASR
When using the Session Outage Notification facility, specifies whether automatic session restart (ASR) processing is enabled for a defined node. NOASR is the default.

ASR works only if both sides of the link are using the ASR option.

Specifying ASR or NOASR on the MSLINK macro overrides ASR specifications on the TYPE and MSPLINK macros.

To display the current ASR option you can use either of the following commands:

- The type-1 command /DISPLAY
- The type-2 command QUERY MSLINK NAME(*linkname*) SHOW(STATUS)

To change the current ASR option you can use either

- The type-1 command /CHANGE
- The type-2 command UPDATE MSLINK NAME(*linkname*) SET(ASR(OFF|ON))

PARTNER=
The value for *partner-id* is a two-character alphanumeric identification. It ensures that the two related logical links in two systems are always logically and physically connected. Both systems must have MSLINK macro statements with the same value for *partner-id*. A logical link can be assigned to a different physical link; the two systems still communicate through the logical link, which remains as defined.

You can modify the partner ID by using the type-2 command `UPDATE MSLINK NAME(linkname) SET(PARTNER(partner-id))`.

MSNAME macro

The MSNAME macro provides a name for the remote and local system identifications that it represents. The MSNAME macro can be followed by one or more NAME macros that define remote logical terminals. MSNAMEs are also referred to as logical link paths.

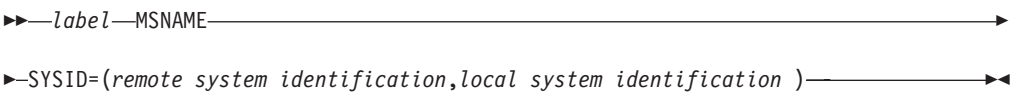
Dynamic definition

You cannot dynamically define logical link path names for remote and local MSC systems.

Supported environments

The MSNAME macro can be used in the DB/DC and DCCTL environments.

Syntax



Label field

The label field *msname* is a one- to eight-character alphanumeric name used externally in commands. Link names, msnames, transaction codes, and logical terminal names, collectively, cannot contain duplicates. Specifying the *msname*label field is required.

Positional parameters

The MSNAME macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

SYSID=

remote system identification identifies the remote system that is represented by this name.

remote system identification specification must not have been previously defined as:

- A remote system identification within a previous MSNAME macro statement.
- A local system identification within an APPLCTN, TRANSACT, or MSNAME macro statement.

The *local system identification* specifies the system identification that is to be used for routing messages back to this system.

local system identification specification must not have been previously defined as a *remote system identification* within a previous MSNAME macro statement.

Values from 1 through 2036 are valid.

You can change the local or remote system IDs of the SYSID= keyword by using the type-2 command UPDATE MSNAME NAME(*msname*) SET(SIDR(*remote_SID*), SIDL(*local_SID*)).

A one-to-one relationship exists between remote SYSIDs and MSNAMEs. APPLCTN and TRANSACT macro statements must refer to remote SYSIDs specified in MSNAME statements in the same system definition when defining remote applications.

MSPLINK macro

The MSPLINK macro defines an MSC physical link.

You can define an MSC physical link to use any one of the following types of connection between two systems:

- Channel-to-channel connection (CTC)

- Main-storage to main-storage connection (MTM)
- Virtual Telecommunications Access Method (VTAM)

You can define a maximum of 1018 physical links (MSPLINK macros) in an IMS system, unless the system is defined as a large system by SYSTEM=LGEN in the IMSCTL macro, in which case, you can define only 676 MSPLINK macros.

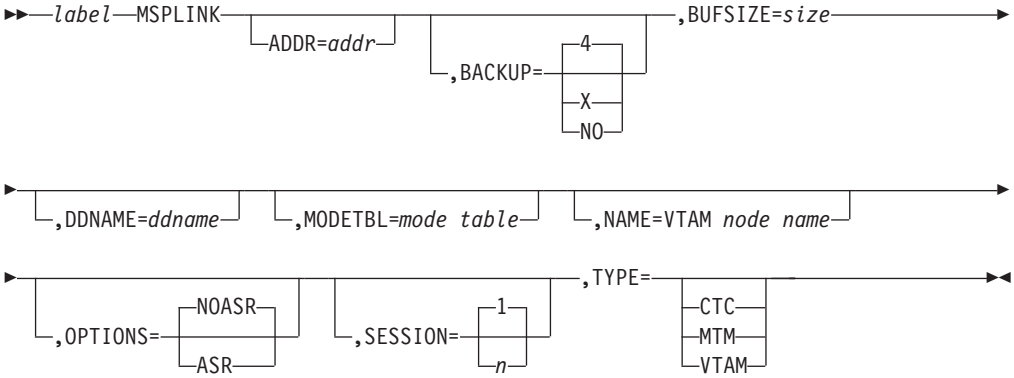
Dynamic definition

Physical links cannot be dynamically defined.

Supported environments

The MSPLINK macro can be used in the DB/DC and DCCTL environments.

Syntax



Label field

The MSPLINK name specifies a 1- to 8-character alphanumeric name for this physical link. You must specify this name. This name is used (in the MSPLINK= keyword operand of the MSLINK macro) to assign a logical link to this physical link. You can change this name using the UPDATE MSPLINK command.

Positional parameters

The MSPLINK macro does not include positional parameters.

MSPLINK macro keyword parameters

To find which parameters apply to your IMS configuration, see "Selecting the appropriate macros to define your system" on page 5.

ADDR=

Specifies the address of the communication line or the address of the channel-to-channel adapter. The address value can be three or four hexadecimal digits. This operand is required for CTC link types and invalid for other link types.

All devices attached to the same channel as the CTC adapter must be accessible through an alternative channel. In addition, do not attach system resources (for example, paging devices) to the same channel as the CTC adapter.

BACKUP=

For XRF-capable IMS systems, controls the automatic restart of VTAM links after an XRF takeover.

X is a numeric integer from 1 to 7, inclusive, that specifies the priority for reestablishing the session. The default is 4. Specifying BACKUP=NO suppresses the automatic restart of the MSC physical link.
Although requests to VTAM are prioritized by IMS, the active requests can be completed in any order because of internal VTAM conflicts and pacing.

BUFSIZE=

Specifies the input and output buffer sizes for each logical link assigned to this physical link.

Buffer sizes for all link types can range from 1024 bytes to 65536 bytes.

The same buffer size must be specified by the IMS systems at each end of a physical link. If the IMS system at the other end of this physical link is an IMS Version 9 or earlier system, the maximum buffer size that can be specified on this MSPLINK macro is also 30720, because Version 9 and earlier IMS systems have a maximum allowable buffer size of 30720 bytes.

IMS initializes the MSC link buffers to the size specified on the BUFSIZE parameter. For links with IMS Version 9 systems, to ensure that the buffer sizes are compatible, IMS automatically increases the buffer size of the IMS Version 9 links at link restart by adding 296 bytes to the size for VTAM and 106 bytes for CTC/MTM. These additional bytes are not displayed on the `/DIS LINK OPTION BUFSIZE` or `QUERY MSLINK SHOW(BUFSIZE)` commands.

If you are using bandwidth mode, a BUFSIZE value of 1024 is too small to send multiple messages with one buffer. A value of at least 4096 is recommended.

After system definition, you can specify a different buffer size for individual logical links by using the type-2 command `UPDATE MSLINK NAME(`*`linkname`*`) SET(BUFSIZE(`*`new_bufsize`*`))`. The buffer sizes for individual logical links follow the same specification requirements as buffer sizes specified on the MSPLINK macro.

DDNAME=

The ddname of the JCL statement that describes this physical connection. This operand is required CTC link types and is invalid for MTM and VTAM link types.

Each name must begin with an alphanumeric character. The following names cannot be used as DD names:

```
DFSRESLB   IMSLOG     IMSTFMT    MSDBINIT
DUMP       IMSLOGR    IMSUDUMP   PRINTDD
IEFRDER    IMSLOGR2   LGMSG      PROCLIB
IEFRDER2   IMSLOG2    MSDBCP1     QBLKS
IMSACB     IMSMON     MSDBCP2    SHMSG
IMSDBL     IMSRDS     MSDBDUMP
IMSDILIB   IMSSPA
```

MODETBL=

Specifies the name of the VTAM logon mode table entry (logon mode name) containing the SNA bind parameters to be used when a session is established for this terminal.

With MODETBL=, you can have a system definition specification for referencing an entry other than the default entry in the user VTAM logon mode table. Normally, the terminal operator specifies this mode table entry

name when logging on to a terminal; this result was not possible, however, from the IMS master terminal because IMS initiated the session.

If MODETBL= is not specified at system definition, no functional or operational change affects the user.

If MODETBL= is specified at system definition, the specified entry name is used.

You can display the current MODETBL name for each assigned logical link by using either the `/DISPLAY LINK` command or the `QUERY MSLINK` command. You can override the MODETBL name by using:

- The LOGON APPLID entry by the remote terminal operator
- The `VARY ACT, LOGON=` command by the network terminal operator
- The `/RST` or the `/CHANGE` command by the master terminal operator
- Either the type-2 command `UPDATE MSPLINK` or `UPDATE MSLINK` through the Operations Manager API

The MODETBL= parameter is required for the IMS master terminal if the VTAM default mode table has not been configured specifically for the device to be used as the IMS master terminal.

Specifying MODETBL on the MSPLINK macro circumvents the requirement to specify the mode table entry when logging on to terminals that always require specification of the same mode table entry name.

If the terminal being defined is in another domain (that is, a cross-domain resource), the MODETBL parameter does not have to be specified.

Related reading: See z/OS Communications Server: SNA Programming for additional information.

NAME=

Specifies the VTAM node name of the remote system at the other end of the link. NAME= must be the same as the label on the VTAM APPL statement for the remote system (that is, the minor node name). For an MSC VTAM link communicating with an XRF complex, the node name must be the VTAM USERVAR associated with the partner IMS/XRF complex. If the ACBNAME parameter of the VTAM APPL statement is not specified, NAME= is the same as the application identification (APPLID=) specified on the IMS COMM macro statement in the remote system. This operand is required for MSC VTAM links only and is invalid for other link types.

You can change the VTAM node name that is defined by the NAME= keyword by using the type-2 IMS command `UPDATE MSPLINK NAME(`*`msplinkname`*`) SET(NODE(`*`VTAM_nodename`*`))`..

OPTIONS=

The Session Outage Notification facility only, specifies automatic session restart on all logical links associated with physical links. The default is NOASR. ASR or NOASR in the MSLINK macro overrides ASR definitions on the MSPLINK macro.

To display the current ASR option you can use the type-1 command `/DISPLAY`. To change the current ASR option you can use either of the following commands:

- The type-1 command `/CHANGE`
- The type-2 command `UPDATE MSPLINK NAME(`*`msplinkname`*`) SET(ASR(OFF|ON))`

ASR works only if both sides of the link are using the ASR option.

SESSION=

For physical links of the VTAM type only, specifies the number of parallel sessions that can be active for the physical link. Valid values are from 1 to 999. The default is 1.

With a large SESSION value, you can dynamically assign more logical links to the physical link than were originally assigned during system definition. The SESSION value can be increased at system startup by using JCL. For instructions, see the keyword NLXB described in "Environments that support IMS system definition-supplied procedures" on page 446.

Because a given number of logical parallel sessions uses the same amount of storage for control blocks and buffers as would the same number of physical links, allocating many parallel sessions can use an excessive amount of common storage area. Predetermine how much common storage area you want to use.

TYPE=

Defines the type of physical link being described. Valid parameters are CTC, MTM, and VTAM.

With the MTM type link, you can have more than one IMS system running in the same z/OS system without a hardware link.

NAME macro

The NAME macro statement defines a logical terminal name (LTERM) that is associated with a physical terminal.

The LTERM can be specified as the IMS master terminal if the associated physical terminal is a 3270 display terminal, SLU 1, or SLU 2. The terminal must be nonswitched. If the terminal is a SLU 1, both the input (ICOMPT) and the output (COMPT) component designations must refer to the first physical component. The first physical component of a SLU 1 terminal must be a console if it is to be defined as the IMS master terminal.

For SLU 1 terminals, the NAME macro statement also defines the data set at the terminal to receive output for corresponding components defined in the TERMINAL macro statement as USERDS1.

For LU 6.1 nodes, the position of the NAME macro within the system definition macro set determines whether LTERM allocation is fixed or dynamic.

If the allocation of LU 6.1 logical terminals is to be predefined (fixed), the NAME macro statement must follow the TYPE and TERMINAL macro statements, and the SESSION parameter must be equal to 1 (by specification or default).

If the allocation of LU 6.1 logical terminals is to be dynamic, the NAME macro statement must follow the SUBPOOL macro statement, and the SESSION parameter must specify a valid value in the range 1 through 255. The default is 1.

- "Dynamic definition" on page 381
- "Supported environments" on page 381
- "Syntax" on page 381
- "Positional parameters" on page 381
- "Keyword Parameters" on page 383
- "Sample NAME macro JCL" on page 384

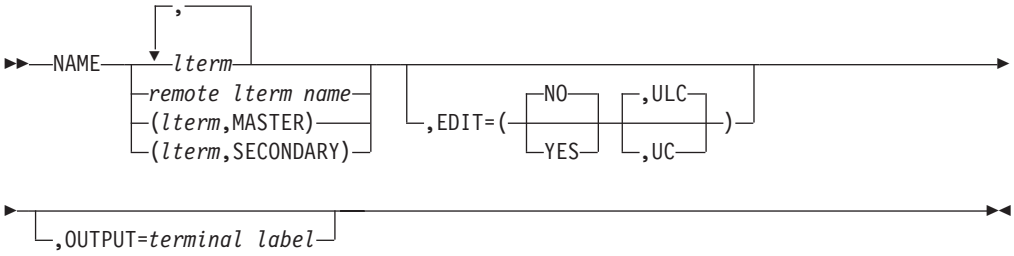
Dynamic definition

You cannot dynamically define a logical terminal name (LTERM) that is associated with a physical terminal.

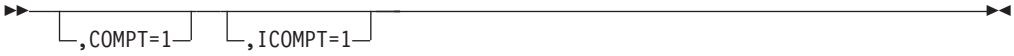
Supported environments

The NAME macro can be used in the DB/DC and DCCTL environments.

Syntax



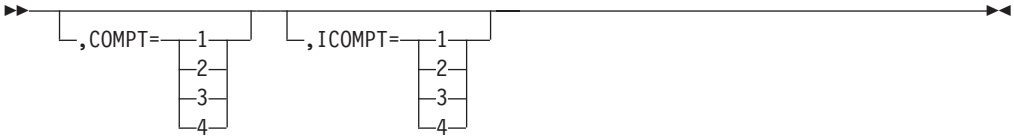
Master terminal



3601 workstation

3275 terminal

SLUTYPE1, SLUTYPEP, LUTYPE6 terminals



Positional parameters

lterm
Specifies one or more 1- to 8-character names for logical terminals associated

with previously defined physical terminals. The name can contain only alphanumeric characters, that is, the letters A through Z, digits 0 through 9, and national characters #, $, and @. WTOR and DFSMTCNT are invalid *lterm* names. The entered name cannot begin with the character sequence INQU. Logical terminal names, transaction codes, and MSNAME link names, collectively, cannot contain duplicates. This operand is required. For further restrictions, refer to "Resource naming rules" on page 301.

If you enter the operand as a sublist with the keyword MASTER as a second parameter, the entered name is the identifier of the IMS master terminal.

If you enter the operand as a sublist with the keyword SECONDARY as the second parameter, the entered name is the identifier of the IMS secondary master terminal.

The IMSID is a reserved keyword and cannot be used as a master or secondary master LTERM name.

One logical terminal in the system must be specified as the master terminal. The logical terminal chosen as the master terminal cannot be on a switched LINEGRP. It must be one of the following types:

- 3270 display terminal
- Terminal defined as either SLUTYPE1 or SLUTYPE2

It must also be the console component if the terminal is defined as SLUTYPE1.

One logical terminal can be specified as the secondary master terminal. This logical terminal cannot be on a switched line group. It must be one of the following types:

- A 328x physical terminal
- A terminal defined as SLUTYPE1
- A terminal in a line group defined as a SPOOL

You can specify multiple LTERMs per TERMINAL macro. For example:

```
NAME  LTERMA,LTERMZ
NAME  LTERMA,LTERMX,LTERMZ
```

For IMS to designate data and commands as having originated at a master terminal, you must specify the first LTERM as MASTER. For example, for IMS to use master terminal default command security for commands entered from the master terminal, specify the first LTERM as MASTER:

```
NAME  (LTERMA,MASTER),LTERMX,LTERMZ
```

However, to specify a master terminal and to prevent IMS from designating data and commands as having originated at a master terminal, such as to bypass default command security and use some other form of security such as RACF command security, specify an LTERM other than the first LTERM for the terminal as MASTER:

```
NAME  LTERMA,(LTERMX,MASTER),LTERMZ
```

For terminals with multiple NAME LTERMs defined, system error message responses are sent to the logical terminal name that is lowest in the collating sequence, unless the terminal is an input-only terminal. For input-only terminals, the system responses go to the first assigned logical terminal OUTPUT= terminal.

remote lterm name
Specifies a one- to eight-character name for a logical terminal associated with a physical terminal defined in a remote IMS system. The name can contain only

alphanumeric characters. WTOR and DFSMTCNT are invalid logical terminal names. The name cannot begin with the character sequence INQU. For further restrictions, refer to “Resource naming rules” on page 301. No other operands are meaningful for a remote logical terminal; if provided, they are interpreted as comments only.

Usage information

If the IMS master terminal is specified as a SLU 1, the designation of a secondary master terminal is optional. The master secondary terminal can be specified if the logical terminal's associated physical terminal is SLU 1. The terminal must be nonswitched. If the terminal is a SLU 1, both the input (ICOMPT) and the output (COMPT) component designations must refer to the first physical component. The first physical component of a SLU 1 can be either a console or a printer if it is to be defined as the IMS secondary master terminal.

If the IMS master terminal is specified as a 3270 display terminal, a secondary master terminal is required, and must be specified for a 328x, associated physical terminal.

If the IMS master terminal is specified as SLU 2, you must specify a secondary master terminal in a preceding NAME macro for a 328x, SLUTYPE1, or SPOOL associated physical terminal.

If other physical terminals are defined on the same communication line as the primary or secondary master terminal, then special restrictions apply to the use of the `/CHECKPOINT` and `/IDLE` commands in the IMS system.

The Message Format Service (MFS) special master terminal formatting is only available for 3270 display terminals specified with a screen size of 24 x 80.

For output purposes, the COMPT operand associates the specified logical terminal with a specific component in a 3601, SLU 1, or SLU P terminals, or LU 6.1 nodes. The COMPT operand is valid only for these terminal types.

If two or more sessions are defined on the SESSION parameter, dynamic allocation must be used.

Consider using dynamic rather than fixed allocation for one parallel session (SESSION=1). This permits all subpools to be available for dynamic allocation to that session.

Keyword Parameters

To find which parameters apply to your IMS configuration, refer to “Selecting the appropriate macros to define your system” on page 5.

COMPT=
Specifies the output component associated with this logical terminal. If this is the master terminal, the COMPT= operand value must be either not specified or 1.

For 3601 work stations, you can specify 1, 2, 3, or 4.

COMPT= cannot be specified for NTO devices.

EDIT=
Specifies whether the logical terminal user-supplied edit routine DFSCNTE0 is

to be used when routing a message to this logical terminal. The logical terminal edit routine is only included within the IMS control program nucleus if one of the NAME statements specifies EDIT=YES. If ULC is specified, output is transmitted as received. If UC is specified, output is translated to uppercase before transmission.

The logical terminal edit routine DFSCNTE0 is not supported for ETO logical terminals.

ICOMPT=
Specifies the input component associated with the terminal defined in the previous TERMINAL macro. If this is the master terminal, the ICOMPT= operand value must be either not specified or 1.

For SLU 1, SLU P, or LU 6.1 terminal systems, you can specify 1, 2, 3, or 4. These values relate to the components specified in the preceding TERMINAL macro statement.

For secondary logical units defined as SLUTYPE1, if the first component is a console, the default is 1; if the first component is a printer, no default exists. For terminals defined as SLUTYPEP or LUTYPE6, the default is 1.

ICOMPT= cannot be specified for NTO devices.

OUTPUT=
Specifies a terminal that is to be used as the output terminal for this logical terminal name. This is specified by entering the label of the TERMINAL macro defining the desired terminal. The referenced TERMINAL statement must occur before this NAME statement. OUTPUT= cannot be specified for VTAM terminals. OUTPUT= cannot be used with response mode, Fast Path, or conversational transactions.

A split input/output LTERM is not applicable to ACF/VTAM terminals.

Sample NAME macro JCL

The following figure is an example of the order of the macros for fixed and dynamically allocated subpools:

```
TYPE     UNITYPE=LUTYPE6TYPE     UNITYPE=LUTYPE6
TERMINAL SESSION=1               TERMINAL SESSION=(1-255)
NAME     ltrmname                 .
                                  .
                                  .
                                  VTAMPOOL
                                  SUBPOOL  NAME=SBPLNAME
                                  NAME     ltrmname
```

RTCODE macro

The RTCODE macro specifies the routing codes that identify the application program that is named in the preceding APPLCTN macro statement.

The RTCODE macro statement can be used one or more times with the APPLCTN macro statement that defines an IMS Fast Path application program. A TRANSACT macro statement that specifies an IMS Fast Path-exclusive transaction automatically generates an RTCODE macro statement with a routing code identical to the transaction code.

If you do not include this macro during stage 1 system definition, no warning is issued, and it is assumed that you define your routing codes dynamically using CREATE RTC and UPDATE RTC commands.

A routing code that is specified on an RTCODE macro must not be a duplicate of a Fast Path-exclusive transaction code, but can be a duplicate of a Fast Path potential transaction code. Invalid duplicate routing codes can be identified by using the NAMECHK option of the IMSCTRL macro.

The Fast Path Input Edit/Routing exit routine (DBFHAGU0) must be used to route the transaction to the correct Fast Path application program.

Routing codes can be added, changed, or deleted by a MODBLKS system definition. However, new routing codes can only be added if the online system to which they are to be added already has Fast Path defined for it.

Dynamic definition

To dynamically define the routing codes that identify application programs previously defined, you can use the CREATE RTC command and the UPDATE RTC type-2 commands. The following table compares the RTCODE macro keywords and the equivalent CREATE and UPDATE command keywords used for dynamic definition. Default values are shown in **bold**.

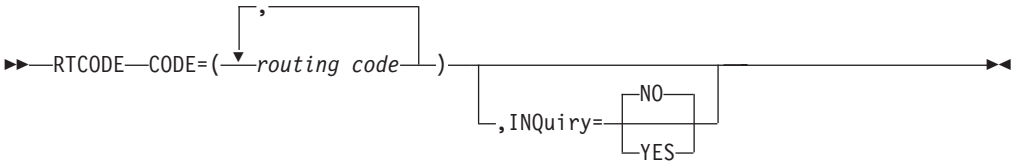
Table 57. RTCODE macro keywords and the equivalent CREATE and UPDATE command keywords used in dynamic definition

RTCODE macro keyword	CREATE | UPDATE RTC keyword equivalent
Previous APPLCTN macro statement	PGM(*name*)
CODE=routing code	NAME(*name*)
INQUIRY=**NO** | YES	INQ(**N** | Y)

Supported environments

The RTCODE macro can be optionally used in an IMS DB/DC and IMS DBCTL environment.

Syntax



Positional parameters

The RTCODE macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

CODE=
Specifies the one- to eight-character alphanumeric routing code, or list of routing codes. The first character must be either a letter or a digit. Routing codes can be duplicates of either transaction codes or logical terminal names, but each must be unique within the set of routing codes.

INQUIRY=
Specifies whether (YES) or not (NO) any message associated with the routing code specified on the same RTCODE macro statement is an inquiry transaction. The default is NO. INQ=YES should be specified only for transactions that do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database when processing a transaction defined as INQ=YES.

Usage information

If you do not include this macro during stage 1 system definition, it is assumed that you dynamically define your route codes using the CREATE RTC and UPDATE RTC commands. See Table 57 on page 385 for more information.

An APPLCTN macro statement that defines a message-driven Fast Path application program should be followed by at least a TRANSACT macro statement or an RTCODE statement.

SECURITY macro

Use the SECURITY macro statement to specify that optional security features are in effect during IMS execution unless they are overridden during system initialization.

IMS Version 12 is the last version to support the SECURITY macro. You can use initialization parameters to specify most of the SECURITY macro keyword values. The SIGNEXIT and TRANEXIT keywords do not have corresponding initialization parameters.

The corresponding initialization parameters for the RCLASS and SECCNT keywords were delivered through the following APARs/PTFs:

- IMS Version 11 - PM48203/UK74050
- IMS Version 12 - PM48204/UK74051

For information about using initialization parameters for security, see Controlling security during system startup for DB/DC and DCCTL (System Administration).

The IMS system can be defined to use the Resource Access Control Facility (RACF) licensed program (or equivalent), an exit routine, or both to perform the following types of security protection:

- Transaction authorization
- Command authorization
- Signon verification
- PSB authorization verification
- Application resource, or DBCTL, access authority checking

Authority to issue specific IMS commands can be restricted to certain transactions, and password and terminal security requirements can be imposed.

If the SECURITY macro is included, any values specified or accepted by default override options specified in the COMM or IMSGEN macro.

- "Dynamic definition"
- "Supported environments"
- "Syntax"
- "Keyword Parameters"

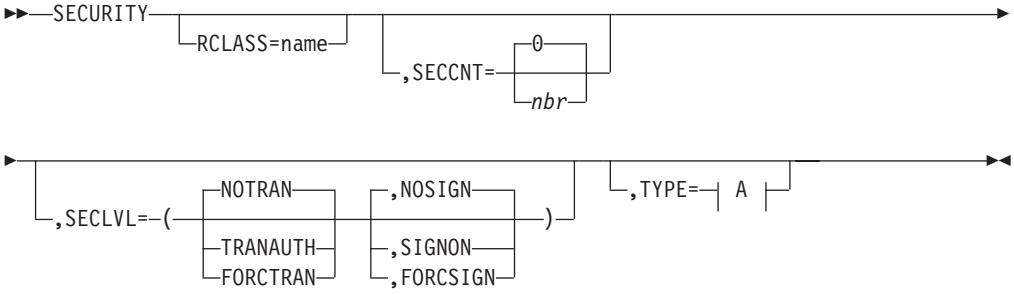
Dynamic definition

You cannot specify optional security features dynamically.

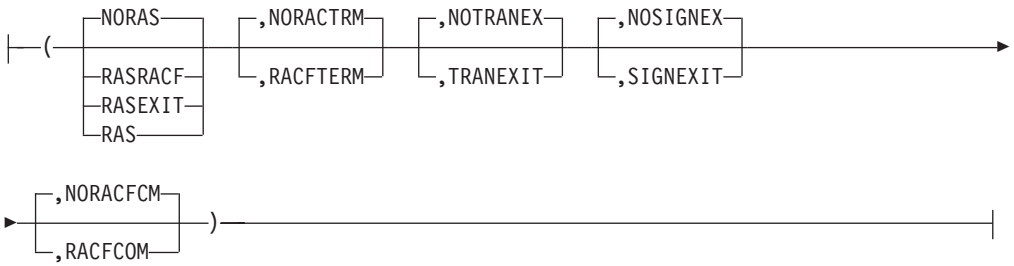
Supported environments

The SECURITY macro can be used in the DB/DC, DBCTL, and DCCTL environments.

Syntax



A:



Keyword Parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

RCLASS=*name*

Specifies an identifier of 1 to 7 alphanumeric characters that is to be used to identity the IMS system as a resource class to RACF for transaction authorization and user ID verification.

The specification is valid only if TYPE=RACFTERM is specified on the system definition SECURITY macro; or RCF=A,T, or Y is specified on the IMS procedure.

IMS class names do not need to be unique. If you do make them unique, you can, for example, define the same transaction name on a production subsystem and a test subsystem, with different access lists for each subsystem, with no ambiguity.

Except for a DBCTL system, if an RCLASS= value is specified on the DFSDCxxx PROCLIB data set member, it overrides the RCLASS= value that is specified on the system definition SECURITY macro. A DBCTL system is an IMS system that is started either with the DBC procedure, or with the system definition macro IMSCTRL defined with SYSTEM=DBCTL.

SECCNT=

Specifies the maximum number of terminal and password security violations to be accepted per physical terminal and the number of transaction command violations per transaction before master terminal notification of such violations. The default is 0, which nullifies notification to the master terminal. The number specified must be 0, 1, 2, or 3.

If ETO is enabled and SECCNT is not 0, the master terminal is notified for every violation.

SECLVL=

Specifies the user ID verification and transaction authorization security enforcement level options available to the MTO when restarting IMS.

NOTRAN | TRANAUTH | FORCTRAN

Specifies whether (FORCTRAN or TRANAUTH) or not (NOTRAN) transaction authorization is to be performed by the online IMS system.

If TRANAUTH or NOTRAN is specified, this option can be overridden by the MTO at restart. The default is NOTRAN.

A specification of FORCTRAN or TRANAUTH requires that TYPE=RACFTERM or TRANEXIT, so that the processing can be performed.

If FORCTRAN or TRANAUTH is specified, the requirements for user ID verification must also be considered.

NOSIGN | SIGNON | FORCSIGN

If SIGNON or NOSIGN is specified, this option can be overridden by the MTO at restart. Specification of FORCSIGN or SIGNON requires that TYPE=RACFTERM or SIGNEXIT, so that processing can be performed.

Transaction authorization requires a validated user ID in order to verify that the user is authorized to invoke the transaction. For this reason, the security level of user ID verification must be at least as great as that of transaction authorization. The requirements are shown in the table that follows.

Table 58. Required user ID verification levels for transaction authorization

Parameter	Default	Invalid
NOTRAN (specified or default)	NOSIGN	- -
TRANAUTH	SIGNON	NOSIGN
FORCTRAN	FORCSIGN	SIGNON
FORCTRAN	FORCSIGN	NOSIGN

TYPE=

Specifies whether application group name authorization, application resource access authorization, transaction authorization, or signon user ID verification is to be performed.

NORAS | RASRACF | RASEXIT | RAS

Specifies whether (RASRACF, RASEXIT, RAS) or not (NORAS) application resource access security authorization is to be performed by IMS at execution time. Resource access security checking can be performed by the RACF licensed program (RASRACF), by an exit routine named DFSRAS00 (RASEXIT), or by both RACF and an exit routine (RAS). The default is NORAS.

NORACTRM | RACFTERM

Specifies whether (RACFTERM) or not (NORACTRM) linkage to the RACF program product is to be provided for transaction authorization or `/SIGN ON` user ID verification. If RACFTERM is specified, RACF is invoked whenever either of the two functions is required. The default is NORACTRM.

NOTRANEX | TRANEXIT

Specifies whether (TRANEXIT) or not (NOTRANEX) the user transaction authorization exit (DFSCTRN0) is called by IMS to process transaction authorization requests at execution time.

When TRANEXIT is specified, the SECLVL keyword must also be specified with valid combinations of TRANAUTH | FORCTRAN and SIGNON | FORCSIGN.

If both RACFTERM and TRANEXIT are specified, RACF is called first. If TYPE=TRANEXIT is specified in the SECURITY macro, either SECLVL=TRANAUTH or SECLVL=FORCTRAN is also required to enforce transaction authorization.

The default is NOTRANEX.

NOSIGNEX | SIGNEXIT

Specifies whether (SIGNEXIT) or not (NOSIGNEX) a user `/SIGN ON` exit (DFSCSGN0) is to be called by IMS to validate a user ID when entered at execution time. If both RACFTERM and SIGNEXIT are specified, RACF is called first. If TYPE=SIGNEXIT is specified in the SECURITY macro, SECLVL=SIGNON is also required to enforce user ID verification.

The default is NOSIGNEX; however, SIGNEXIT is assumed and forced if TRANEXIT is specified and NORACTRM is specified or accepted by default, because user ID verification is a prerequisite for transaction authorization checking.

NORACFCM | RACFCOM

Specifies whether (RACFCOM) or not (NORACFCM) RACF is to be called to verify command authorization. This applies to commands from ETO terminals. The default is NORACFCM.

To enable RACF for both ETO and STATIC terminals, specify RCF=S in the DFSPBxxx PROCLIB data set member. You cannot enable RACFCOM for only static terminals unless you have ETO RACF command security.

Usage information

The SECURITY macro is provided to enhance IMS system security. It can replace the security specifications on the IMSGEN and COMM macros, both explained

elsewhere in this manual. This section describes the relationship that exists between these three macros and suggests a possible installation strategy.

All security specifications have now been consolidated in the SECURITY macro. However, for compatibility with previous releases, the COMM or IMSGEN keywords related to security specifications can still be used.

The security-related keywords from these three macros are accepted hierarchically in the order SECURITY, COMM, and then IMSGEN; that is, if the SECURITY macro is used, the specifications and defaults from that macro take precedence over any security specifications coded on either COMM or IMSGEN.

If security specifications are coded on more than one of these macros, a warning message is issued, indicating that the specifications and defaults from the highest macro in the hierarchy present in the stage 1 input is used. Likewise, if the COMM macro is used, and any of the IMSGEN macro's general communication options are specified, a warning message is issued indicating that the general communication specifications and defaults from the COMM macro are used.

SUBPOOL macro

The SUBPOOL macro statement, when used in a VTAM macro set, is a delimiter between groups of NAME macro statements to create LU 6.1 LTERM subpools.

At least one NAME macro statement must be defined within the VTAMPOOL; however, it is valid to define one or more subpools with no NAME statements. This definition creates a reserved subpool to which no terminals are allocated until reassigned there with the `/ASSIGN` command.

The subpools must be named to be used in IMS commands or session initiation requests. The LU 6.1 subpool names, specified on the required NAME keyword operand of the SUBPOOL macro statement, must follow standard z/OS naming conventions and must be unique with respect to other LTERM subpool names within the VTAMPOOL. The subpool names must be unique among themselves, but need not be unique to other names used in an IMS system, such as LTERM names and transaction names. These subpools can only be used with statically defined ISC terminals. They cannot be used with ETO terminals.

If parallel sessions are defined in your system, at least one subpool must be defined.

Recommendation: Define at least one subpool for each parallel session defined in your system. Additional subpools can be defined if you desire.
For a session to be brought up, a subpool must be allocated to it. Thus, if you have defined two parallel sessions in your system and only one subpool, only one session can be brought up at any one time.

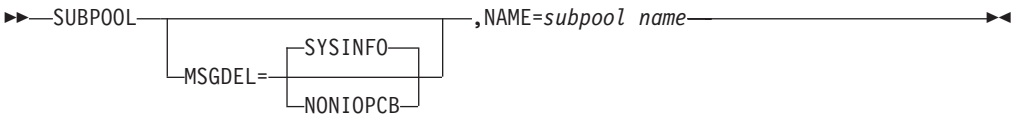
Dynamic definition

You cannot dynamically define LU 6.1 LTERM subpools.

Supported environments

The SUBPOOL macro can be used in the DB/DC and DCCTL environments.

Syntax



Positional parameters

The SUBPOOL macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

MSGDEL=
Used for VTAM LU 6.1 devices and specifies which message types IMS should discard for this terminal. The default is SYSINFO.

SYSINFO | NONIOPCB

SYSINFO, the default, specifies that IMS should discard DFS059 TERMINAL status messages for this terminal and also DFS3650 for SLU P devices.

NONIOPCB specifies that IMS should discard the following message types destined for this terminal:

- Message switches
- Messages inserted by an application program to an alternate PCB
- /BROADCAST messages
- DFS059 TERMINAL status messages

NAME=*subpoolname*
Used for VTAM LU 6.1 devices and names the LU 6 subpool.

TERMINAL macro

Use the TERMINAL macro statement to define physical and logical characteristics of VTAM nodes and non-VTAM communication terminals.

The NAME macro statements that follow a TERMINAL macro statement supply the logical terminal names that are associated with the physical terminal at system definition. Whichever terminal name is designated in the first NAME macro statement that follows a TERMINAL macro statement becomes the response or input/output logical terminal.

A TERMINAL macro statement that defines terminals connected to a switched communication line must not be immediately followed by a NAME macro statement.

All Fast Path-eligible terminals must operate in response (forced or transaction) mode. Where applicable, the PAGDEL option must be specified (or used by default) for these terminals.

Message Format Service (MFS) can be used in an IMS Fast Path configuration.

The TERMINAL macro statement can be specified without operands if the terminal is a printer, punch, tape, or disk.

All non-VTAM data communication specifications must precede the VTAM macro set in your IMS system definition stage 1 input deck. You receive a stage 1 output warning message if the VTAM macro set is not the last physical set. If an MSC macro set is part of your system definition, it must precede the VTAM macro set, or your system definition does not complete. To add VTAM support, you must specify either an ON-LINE or ALL system definition on the IMSCTRL macro statement.

For VTAM terminals, the default values shown for operands of the TERMINAL macro statement are ignored if these operands are specified on the TYPE macro statement.

If system definition statements from previous releases of IMS are being used, TERMINAL macro definitions from the previous releases are valid for this release. The TERMINAL keywords in this topic must be used for terminals newly supported with this release of IMS and for terminals that use new IMS terminal functions.

The TERMINAL macro statement can be used to define secondary logical units type 1, type 2, type 4, and type P, logical unit type 6.1, and NTO devices. These are designated on the TYPE macro as UNITYPE=SLUTYPE1, SLUTYPE2, SLUTYPEP, LUTYPE6, and NTO, respectively.

By specifying SLUTYPE1 on the TYPE macro, you can define an appropriate configuration of console, bulk printer, disk, and card reader/punch. The 3767 and 3770 nonprogrammable terminals must be defined to IMS as SLUTYPE1.

SLUTYPE2 must also be specified for display devices attached to a 3274 or 3276 Control Unit that is operating in SNA mode.

Related reading: For 3270/SLUTYPE2 dynamic terminals that are not defined by the TERMINAL macro, see MFS Device Characteristics Table utility (DFSUTB00) (System Utilities).

By specifying SLUTYPEP on the TYPE macro, you can define 3600 terminals and 3790 or later programs using the Host Communication Facility. This support extends full IMS functional capabilities to user-written programs within the controller and includes the MFS Distributed Presentation Management (DPM) function. The support is in addition to current support for 3600 and 3790 user-written programs.

By specifying LUTYPE6 on the TYPE macro, you can define a logical unit type 6 node such as IBM CICS Transaction Server for z/OS, another IMS, or a user-written program. This node can communicate with IMS as an SNA primary or secondary half-session.

For logical units type 6, the log write-ahead option, as specified by the LTWA and NLTWA parameters on the TERMINAL macro statement, need no longer be specified, because log write-ahead occurs automatically. Therefore, although this parameter can be retained for compatibility with previous releases, if it is specified, it is ignored by IMS.

By specifying NTO on the TYPE macro, you can define NTO devices. NTO support provides a non-SNA start/stop terminal interface to VTAM for 3101, TTY, and TTY-compatible devices. The TERMINAL macro statement keyword PU= permits you to designate the type of terminal being defined to use NTO support.

Related reading: For further information about SLU P and LU 6.1 (ISC), see *IMS Version 11 Communications and Connections*.

The TERMINAL macro statement can also be used to define terminals belonging to a Finance Communication System such as the 3600 Finance Communication System or the 4700 Finance Communication System. Finance Communication System components for devices specified as UNITYPE=3601 or as UNITYPE=FINANCE on the TYPE macro statement can be defined on the TERMINAL macro statement using either specific 3600 definitions or generic FINANCE terminal definitions. The code generated to support the terminal is identical regardless of whether the terminal is specified as UNITYPE=3601 or UNITYPE=FINANCE.

By specifying SLUTYPE2 on the TYPE macro, you can define 3277 terminals, Models 1 and 2, operating under control of the 3790/3270 Data Stream Compatibility. Under this support, the 3277 Model 2 can be defined as the IMS master terminal; the Model 1 is not supported as an IMS master terminal. This support is in addition to current support for the 3600 and 3790 user-written programs. The 3790 can be defined appropriately as SLUTYPE1, SLUTYPE2 or as a subset of FINANCE.

- "Dynamic definition"
- "Supported environments"
- "Syntax"
- "Positional parameters" on page 401
- "Keyword parameters" on page 402

Dynamic definition

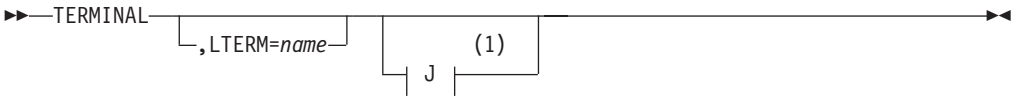
You cannot dynamically define the physical and logical characteristics of VTAM nodes and non-VTAM terminals.

Supported environments

The TERMINAL macro can be used in the DB/DC and DCCTL environments.

Syntax

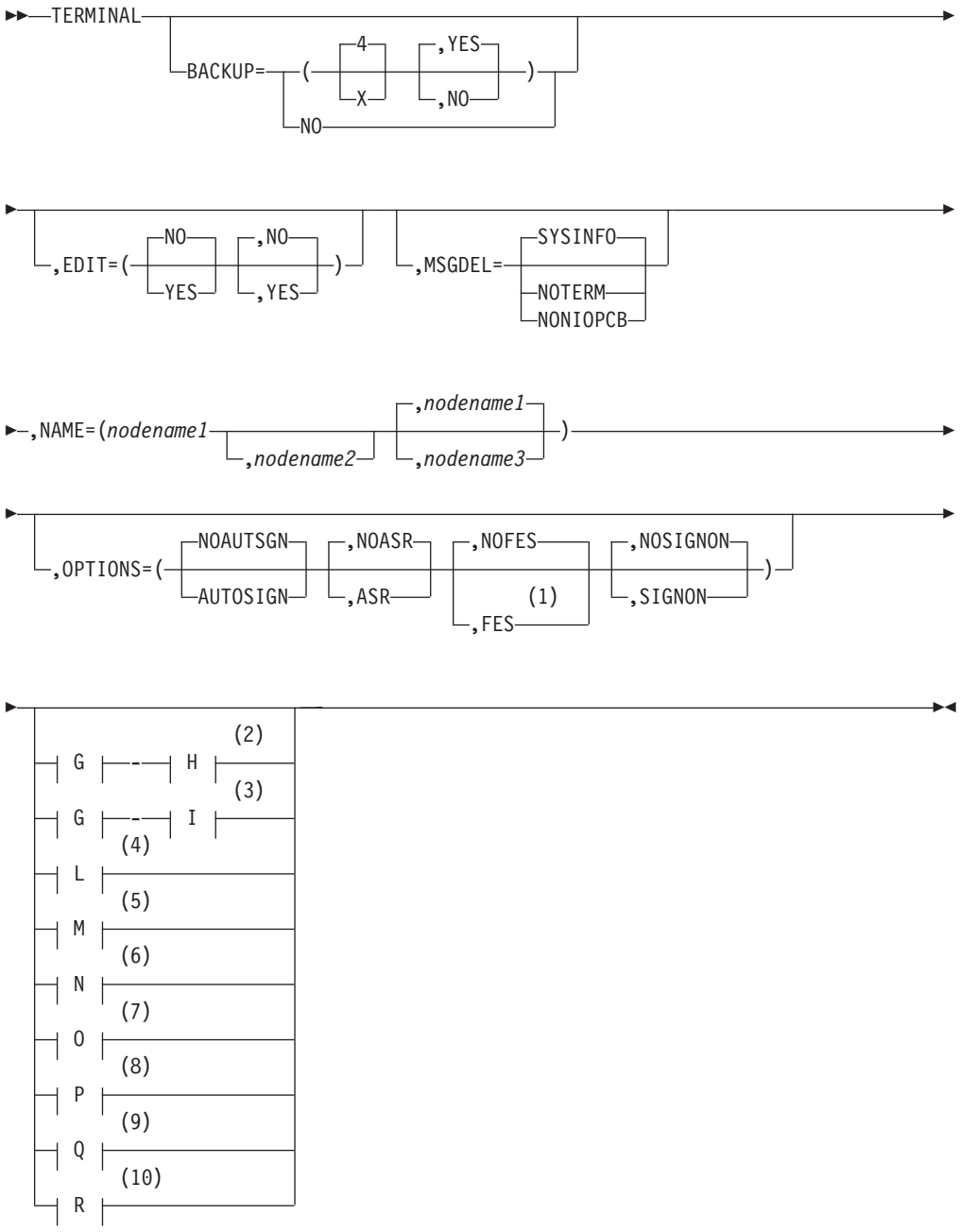
All Non-VTAM terminals



Notes:

1 Spool terminal

All VTAM terminals



Notes:

1 FES not valid for LU 6 terminals

2 3270 local terminal

3 3270 remote terminal

4 Finance workstation (If UNITYPE=FINANCE on the TYPE macro)

5 3600 workstation (If UNITYPE=3601 on the TYPE macro)

6 SLU 1 terminal

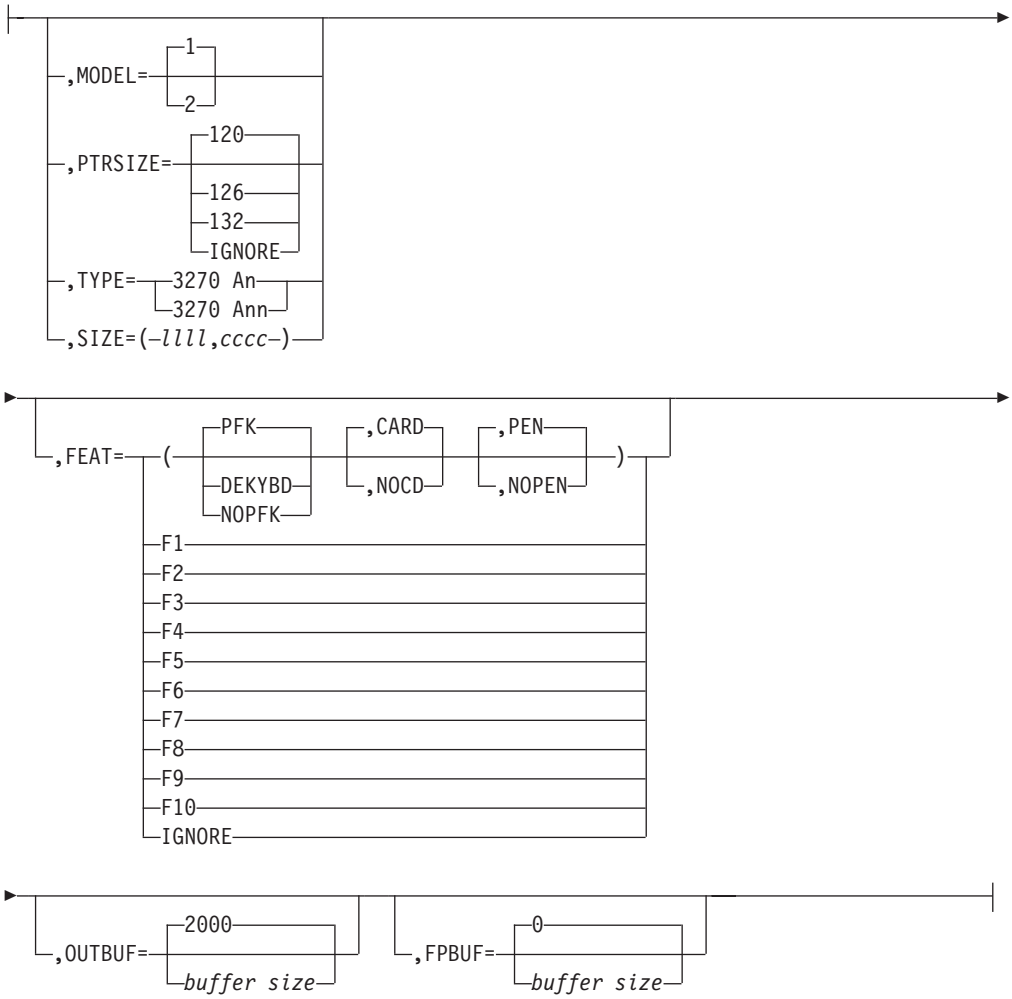
7 NTO device

8 SLU 2 terminal

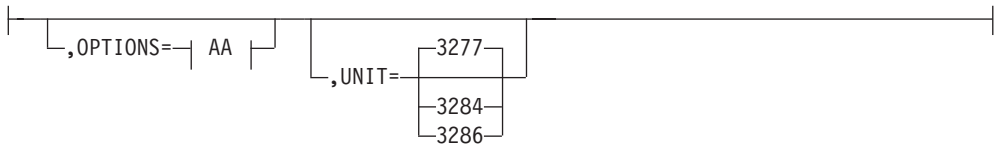
9 SLU P terminal

10 LU 6 terminal

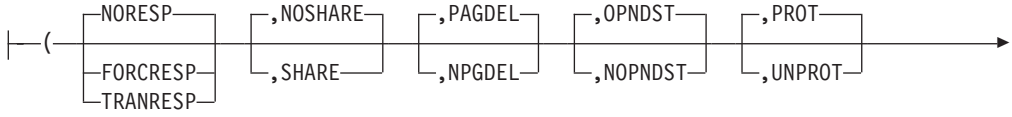
G (all 3270 terminals):



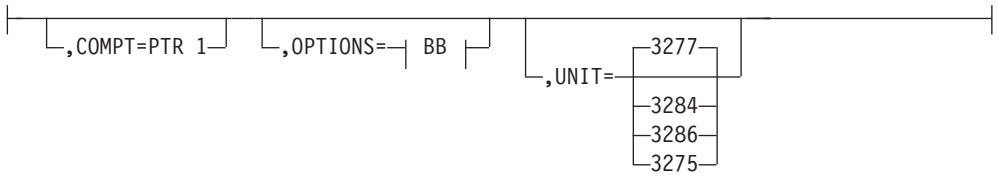
H (3270 local terminals):



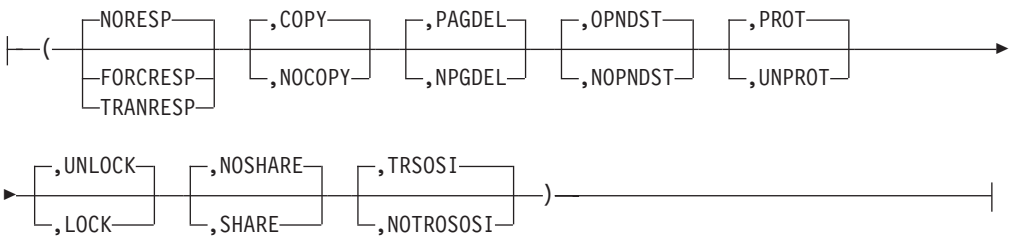
AA:



I (3270 remote terminal):

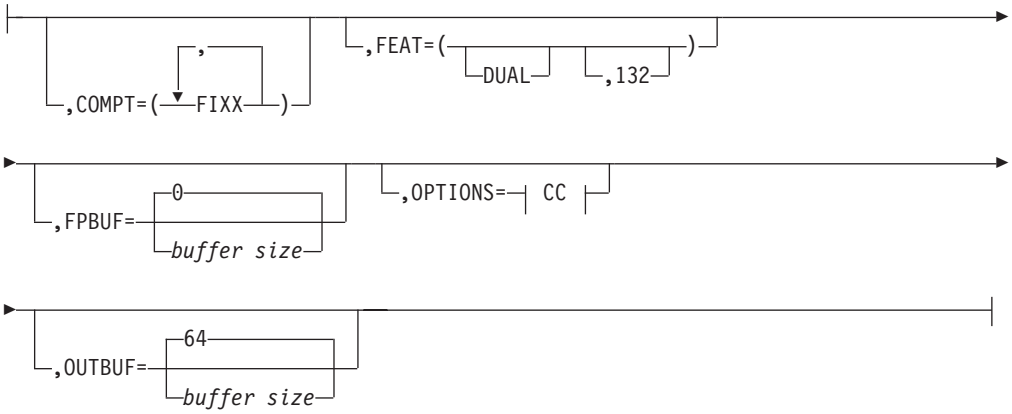


BB:

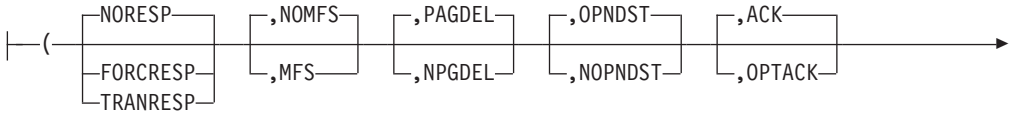


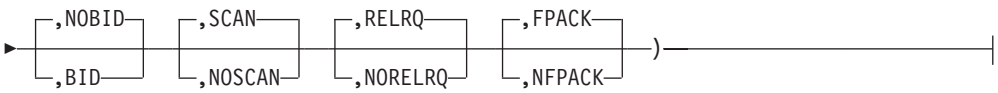
J (spool terminal):

L (finance workstation (UNITYPE=FINANCE on the TYPE macro)):

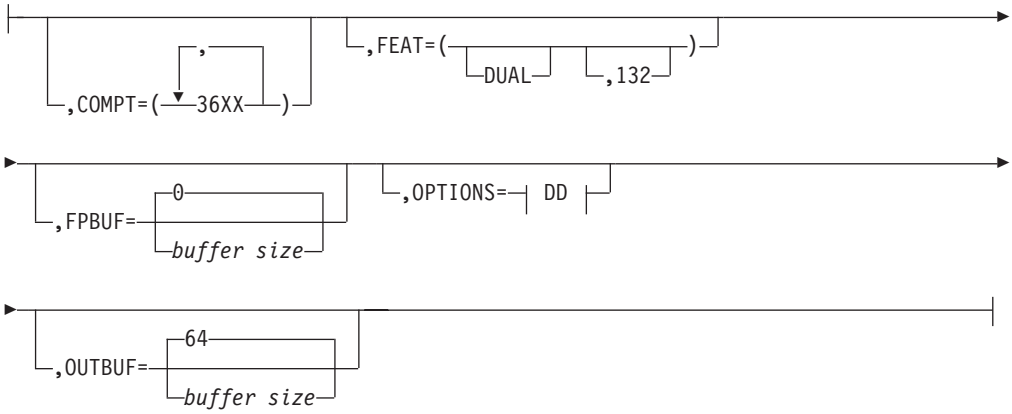


CC:

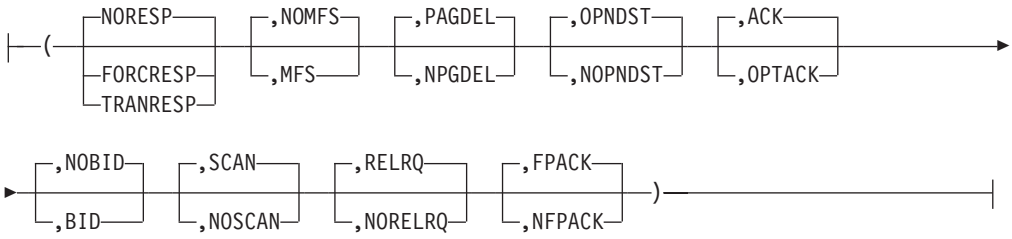




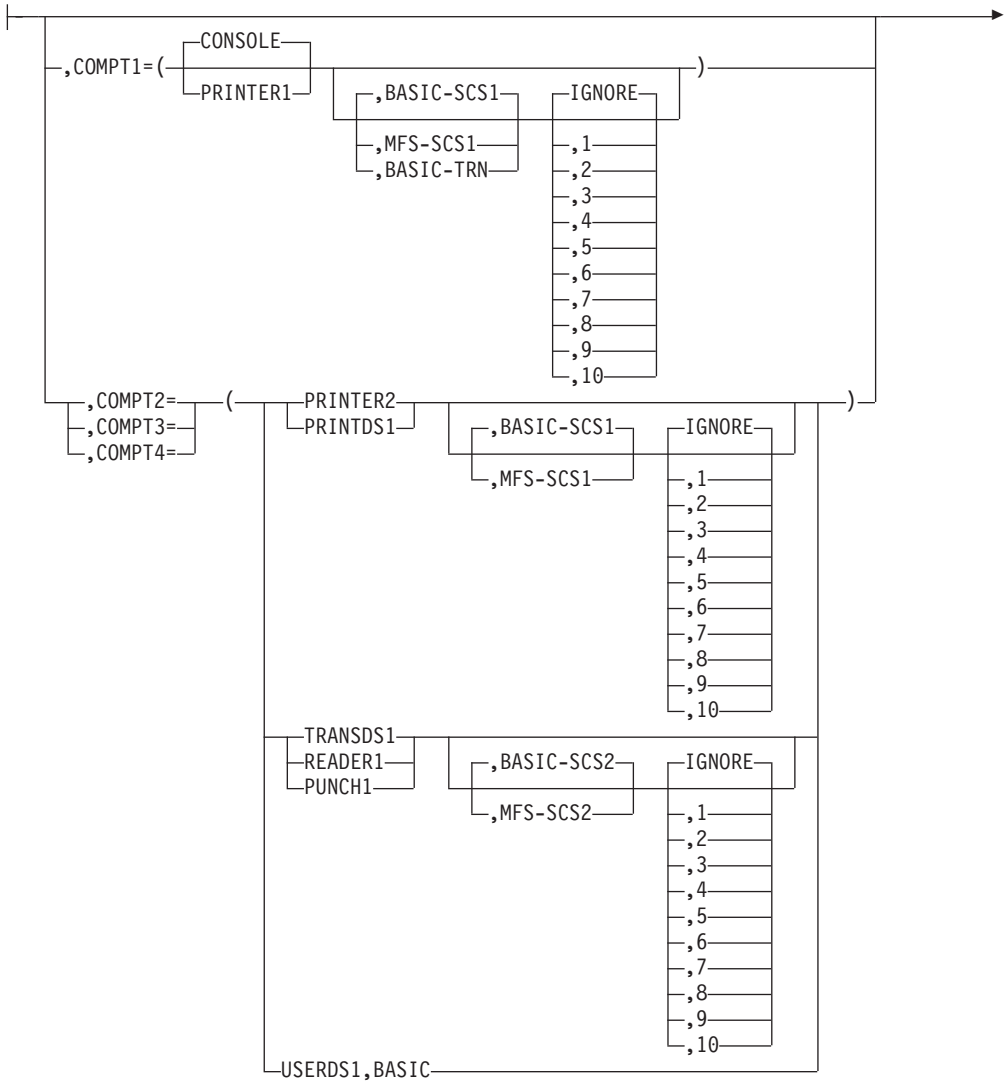
M (3600 workstation (if UNITYPE=3601 on the TYPE macro)):

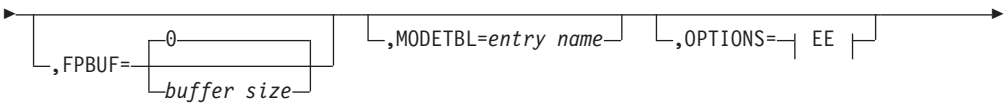


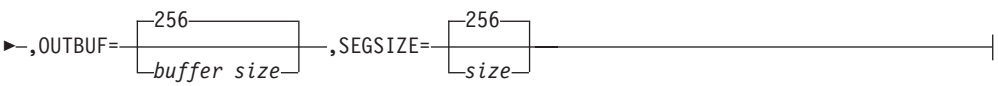
DD:



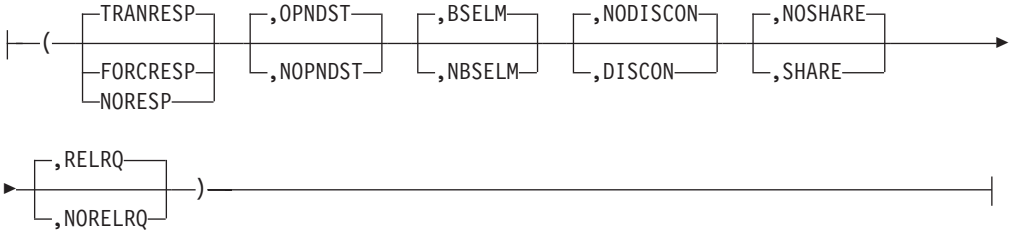
N (SLU 1 terminal):


,COMPT1=(
CONSOLE
PRINTER1
,BASIC-SCS1
,MFS-SCS1
,BASIC-TRN
IGNORE
,1
,2
,3
,4
,5
,6
,7
,8
,9
,10
)
,COMPT2=
,COMPT3=
,COMPT4=
(
PRINTER2
PRINTDS1
,BASIC-SCS1
,MFS-SCS1
IGNORE
,1
,2
,3
,4
,5
,6
,7
,8
,9
,10
TRANSDS1
READER1
PUNCH1
,BASIC-SCS2
,MFS-SCS2
IGNORE
,1
,2
,3
,4
,5
,6
,7
,8
,9
,10
USERDS1,BASIC
)

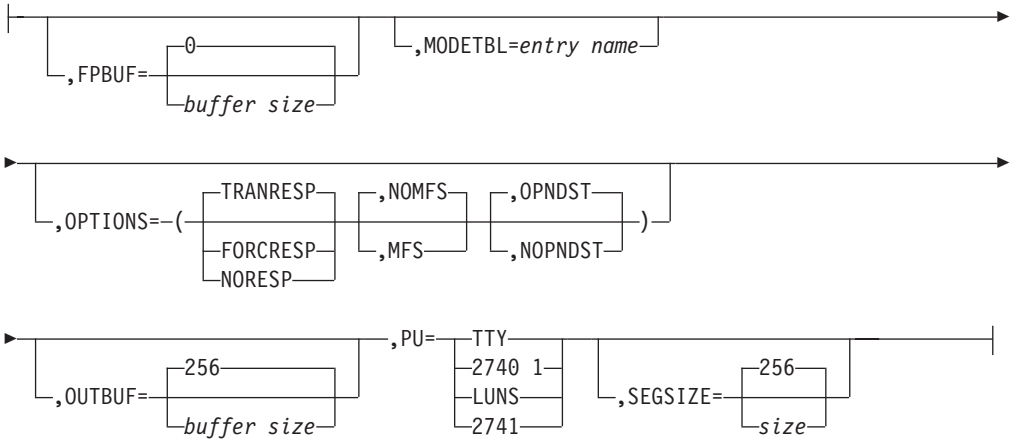

,FPBUF=
0
buffer size
,MODETBL=entry name
,OPTIONS=
EE


,OUTBUF=
256
buffer size
,SEGSIZE=
256
size

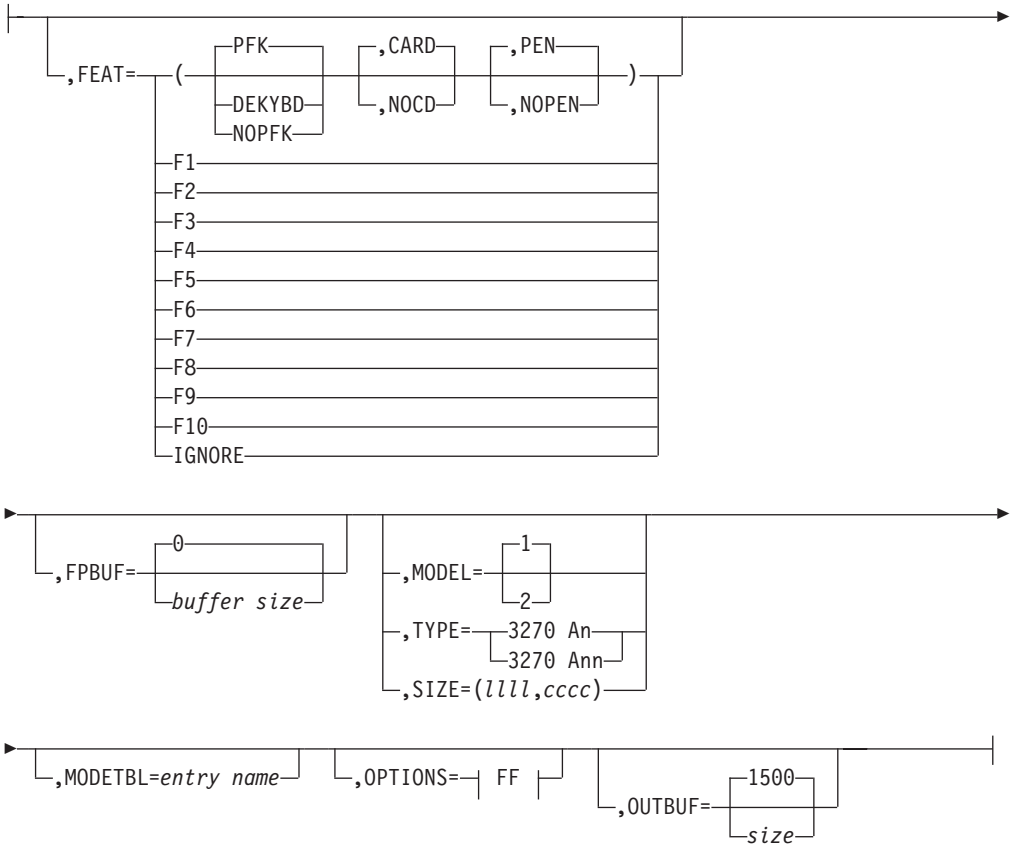
EE:


(
TRANRESP
FORCRESP
NORESP
,OPNDST
,NOPNDST
,BSELM
,NBSELM
,NODISCON
,DISCON
,NOSHARE
,SHARE
,RELRQ
,NORELRQ
)

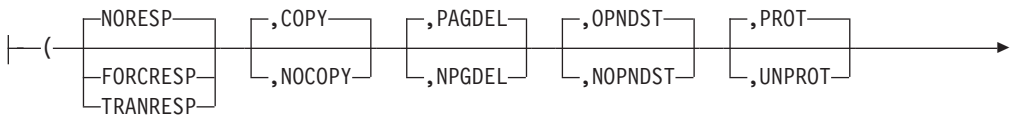
O (NTO device):



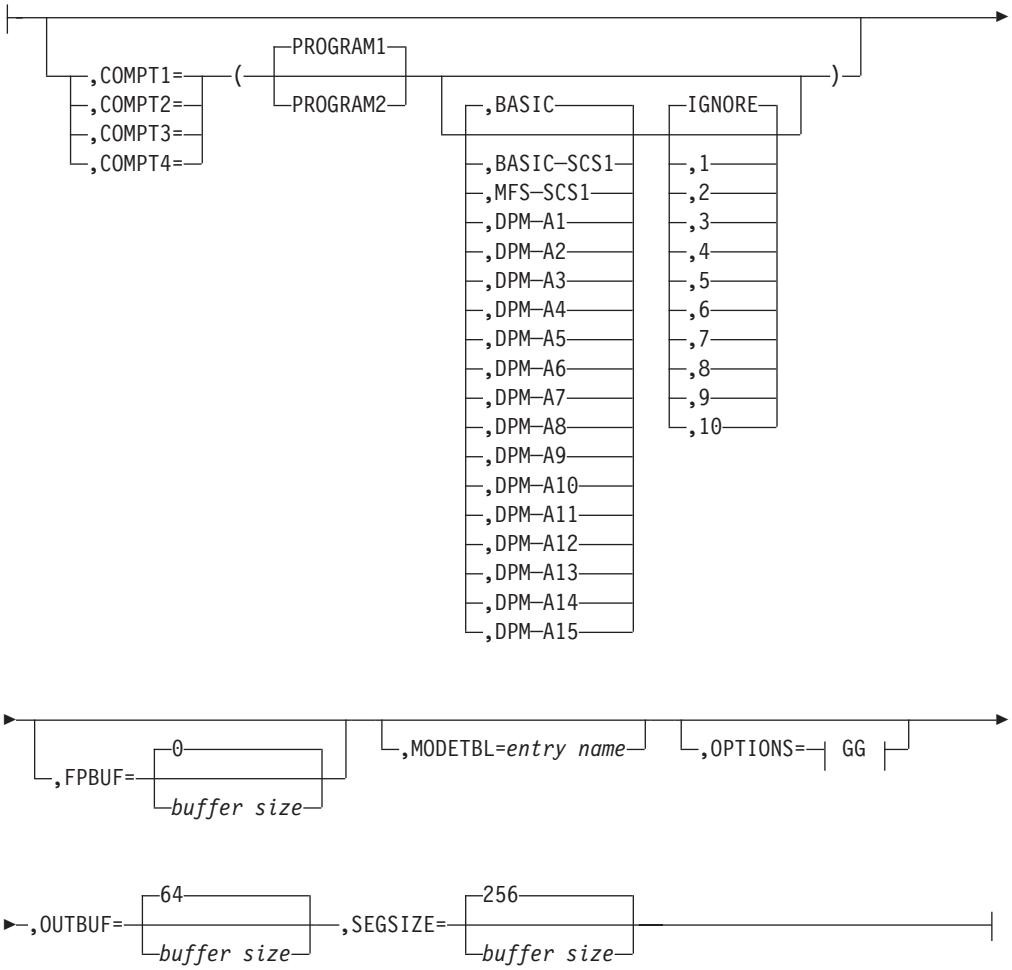
P (SLU 2 terminal):



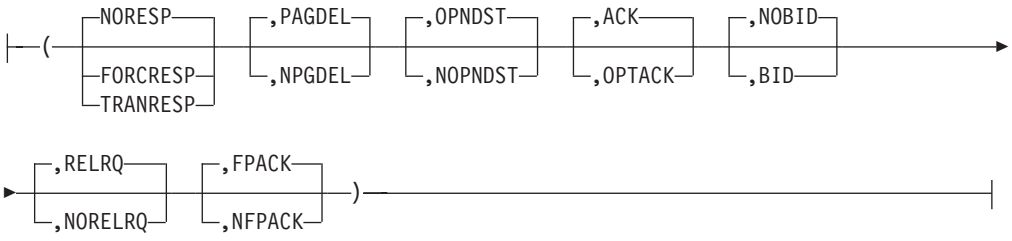
FF:



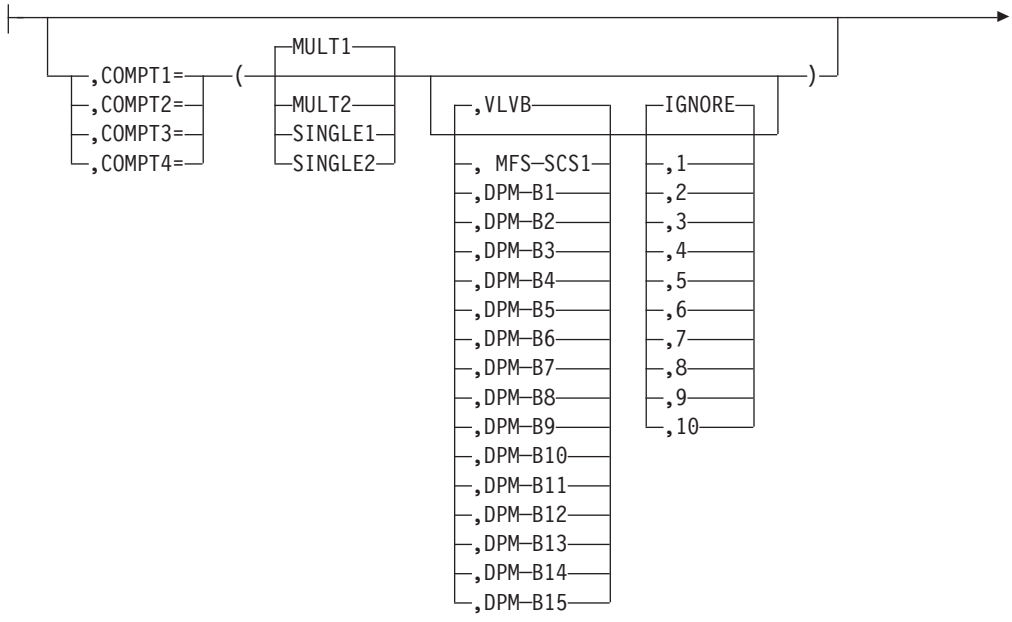
Q (SLU P terminal):

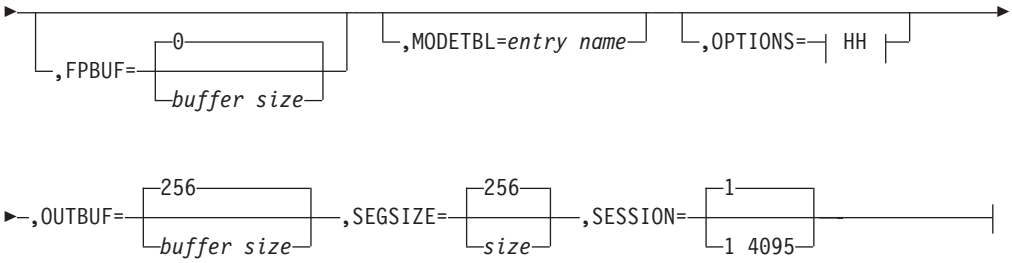


GG:

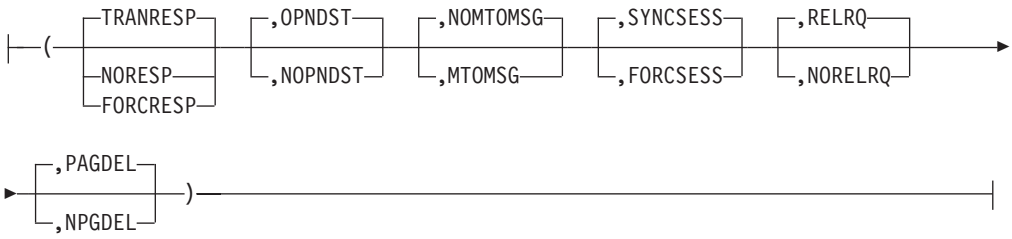


R (LU 6.1 terminal):





HH:



Label Field

If any subsequent NAME macro statement refers to this TERMINAL macro statement in its OUTPUT keyword, a label must be specified for the TERMINAL macro statement.

Positional parameters

The TERMINAL macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

BACKUP=

Specifies (for an XRF complex) control of the automatic restart or session switching (VTAM) after takeover. Use only when HSB=YES is specified on the IMSCTRL macro.

X is a numeric integer from 1 to 7, inclusive, that specifies the priority for reestablishing the session. The default is 4 if either the keyword or the parameter is omitted. NO suppresses session recovery of the terminal at takeover.

Although requests to VTAM are prioritized by IMS, the active requests can be completed in any order because of internal VTAM conflicts and pacing.

The second parameter is only used with VTAM terminals. It determines whether the backup system is to attempt to establish a backup session to the terminal when the active request establishes a session. Backup sessions are not valid for UNITYPEs 3270, LU6, and NTO. If this parameter is specified as YES for these UNITYPEs, it is ignored. If it is not specified, the default is NO. The default for this second parameter is YES when either the keyword or the parameter is omitted. Specifying NO for this parameter suppresses the establishment of a backup session.

BUFSIZE=

Specifies the size of the output buffer. The value specified here cannot exceed the value specified on the preceding LINE macro statement. The default is 120.

COMPT=

Specifies components of this terminal. This operand is valid only for 3600 terminals. For terminals specified as UNITYPE=3601 or FINANCE on the TYPE macro, COMPT= describes the components that compose this workstation. A maximum of four terminal cans can be defined. Acceptable values for components are:

36DS or FIDS
3604 display or 4704 keyboard/display

36DS3 of FIDS3
3604 display, model 3

36DS4 or FIDS4
3604 display, model 4

36DS7 or FIDS7
3604 display, model 7

36JP or FIJP
3610/3612 journal printer or 4710 receipt validation printer

36PB or FIPB
3611/3612 passbook printer

36FP or FIFP
3618 line printer

36MS or FIMS
Magnetic strip encoder

36CT or FICT
3614 customer transaction facility (a component of FINANCE)

One 36DS, 36DS3, 36DS4, 36DS7, or 36CT (or equivalent components specified as FIDS, FIDS3, FIDS4, FIDS7, or FICT) can be specified per workstation. Multiple components of the same type (excluding 36DS, 36DS3, 36DS4, and 36DS7 or equivalent FIxx components) can be specified. The order in which components are specified determines the value placed in the COMPT field of the Finance Communication System output message header.

COMPT1=, COMPT2=, COMPT3=, COMPT4=

- For SLU 1 terminals:

  Defines the component types and processing associated with terminals defined as SLUTYPE1. A maximum of four components can be defined. For each component, three subparameters can be defined: media, edit processing, and features.

  You can assign a number to a set of user-defined features (such as printer size, vertical forms control, horizontal tabs on a printer, and logical record length on a diskette) and tell MFS what these features are. IGNORE can be used to specify that all features (or any designated feature) are to be ignored.

  Component 1 must be either CONSOLE or PRINTER1. Components 2, 3, and 4 must be PRINTER2, PRINTDS1, TRANSDS1, READER1, PUNCH1, or USERDS1.

  If no components are specified, the default is COMPT1=(CONSOLE,BASIC-SCS1).

  The first subparameter (media) of the second, third, or fourth components has no default.

  If BASIC-SCS1, BASIC-SCS2, BASIC, or BASIC-TRN is specified as the second subparameter (editing), the third subparameter (feature) cannot be specified. If MFS-SCS1 or MFS-SCS2 editing is specified as the second subparameter, the default is IGNORE for the third subparameter.

  The following values can be defined:

  CONSOLE (BASIC-SCS1 or MFS-SCS1)
  Identifies the presence of a console keyboard and a printer. Either this component or PRINTER1 must be (and can only be) specified first in a sublist, because all error messages are routed to this destination. This component uses SCS1 data streams. This can be the only component specified if the secondary logical unit type 1 is a 3767.

  CONSOLE (BASIC-TRN)
  Identifies a SLU 1 terminal whose input data does not conform to SCS (SCS1 or SCS2) protocols. IMS processes the data without editing, translating, or deleting characters following the destination and password fields. No MFS editing is supported.

  This form of transparency support differs from SCS2 transparency support in that BINDPSB1=BINTRNDS in the SLU 1 bind image need not be set (and has no effect if set), and in that IMS does not scan the input data stream for transparency control characters (X'35') to identify transparent fields.

  PRINTDS1
  Identifies the presence of a print data set or the SYS.INTR data set for the SLU 1 programmable model. This component uses SCS1 data streams.

PRINTER1
Identifies the presence of 3784, 3284, 3286, or similar printers. This component or console must be (and can only be) specified first in a sublist, because all error messages are routed to these destinations. This component uses SCS1 data streams.

PRINTER2
Identifies the presence of a SLU 1 printer. This component uses SCS1 data streams.

PUNCH1
Identifies the presence of a 3770 card output device. This component uses SCS2 data streams. Card records do not span RU boundaries.

READER1
Identifies the presence of a 3770 card input device. This component uses SCS2 data streams.

TRANSDS1
Identifies the presence of an input component consisting of the SLUTYPE1 transmit data set. This component uses SCS2 data streams. TRANSDS1 can be specified only once as a component of a particular terminal.

USERDS1
Identifies the presence of the user's sequential data sets for output and input. MFS support is not available for this component. The data stream on output is undefined; the inbound data stream is the same as the transmit data set data stream.

If multiple USERDS1 components are defined on the same TERMINAL macro statement, all user data set (UDS) input appears to be from the first UDS defined. Also, unless the CHNG call is used by the IMS Message Processing Program, all output is sent to the output component associated with the first UDS defined. (See the COMPT parameter of the NAME macro statement.)

If MFS-SCS1 or MFS-SCS2 data streams are specified, MFS-SCS1 or MFS-SCS2 formatting is used for messages processed by MFS.

- For SLU P nodes:

  Defines the component types and processing (including distributed presentation management for MFS) associated with a node defined as SLUTYPEP, and indicates that the component is represented by a user-written program.

  Up to four components can be specified; for each component, up to three subparameters can be specified. If the component is not specified, the default is COMPT1=(PROGRAM1,BASIC). Components 1, 2, 3, and 4 must be either PROGRAM1 or PROGRAM2. If component 2, 3, or 4 is present and takes all defaults (PROGRAM1,BASIC), at least one of the default subparameters must be specified so that the presence of the component can be recognized.

  The first subparameter is the media type coded as PROGRAM1 or PROGRAM2.

  If PROGRAM1 is specified, IMS does not assume component protection for this component and might send consecutive messages without waiting for intervening input requests. When using MFS, component protection can be

accomplished for messages which use MFS with paging options in MFS control blocks defined as DEV TYPE=DPM-A01...DPM-A15 or DPM-A1...DPM-A15.

If PROGRAM2 is specified, IMS assumes component protection and does not send consecutive messages without intervening input requests.

The second subparameter specifies the type of editing to be provided by IMS for input to and output from user-written programs.

If BASIC is specified, no deblocking occurs on input, and MFS is not used on output or input.

If DPM-A01...DPM-A15 or DPM-A1...DPM-A15 is specified, no deblocking occurs on input, and the MFS DPM function can be used on output or input. To define DPM for this component, code DPM-An, where "n" is a decimal number from 1 to 15. When MFS is to be used, this operand must match an MFS DEV TYPE=DPM-An statement within the MFS definitions.

If BASIC-SCS1 is specified, SCS1 data streams are used. MFS is not available on output or input.

If MFS-SCS1 is specified, SCS1 data streams are used, and MFS is available.

The third subparameter specifies a user-defined feature code. The designated feature is used to select an MFS format—MFS-SCS1, DPM-A01...DPM-A15, or DPM-A1...DPM-A15—with matching feature specification. IGNORE can be used to specify that an MFS format (SCS1 or DPM-An) with the FEAT=IGNORE operand of the DEV statement is to be selected.

The third subparameter cannot be specified if BASIC or BASIC-SCS1 is specified for the type of editing (second subparameter).

- For LU 6.1 nodes:

  Defines the SNA send/receive and bracket protocols to be used during asynchronous IMS output to LU 6.1 nodes. The MFS distributed presentation management (DPM) facility can be defined as available selectively by component.

  The first, second, and third subparameter are specified independently of one another.

  The first subparameter (MULT1, MULT2, SINGLE1, SINGLE2) indicates the output protocol to be used and the number of messages to be sent on output. These subparameters are defined as follows:

  – MULT1—Multiple messages or empty queue ends the bracket
  – MULT2—Multiple messages or empty queue causes the flow to be returned to the other half-session
  – SINGLE1—A single message or empty queue ends the bracket
  – SINGLE2—A single message or empty queue causes the flow to be returned to the other half-session

  The default is MULT1.

  Related reading: Refer to *IMS Version 11 Communications and Connections* for information about whether to specify a component as SINGLE1, SINGLE2, MULT1, or MULT2. *IMS Version 11 Communications and Connections* also includes information about the characteristics of SINGLE1, SINGLE2, MULT1, MULT2.

  The second subparameter (VLVB/DPM-Bn) indicates whether MFS is available for the component and whether the data blocking algorithm is to be used. VLVB indicates variable-length, variable-blocked format is to be used in place of MFS for both input and output. DPM-Bn indicates that MFS DPM is available for both input and output from that component on a

message-by-message basis. The default is VLVB. You should not specify the DPM-B(xx) operand for the XRF/ISC link. If DPM-B(xx) is specified, a G732 warning message is issued, and the default value of VLVB is assumed.

The third subparameter, a number 1 - 10, specifies a user-defined feature code. The designated feature is used to select an MFS format. IGNORE can be used to specify that an MFS format with the FEAT=IGNORE operand of the DEV statement is to be selected. The default is IGNORE. (If the second subparameter is specified as VLVB, the third subparameter cannot be specified.)

EDIT=

The first parameter specifies whether (YES) or not (NO) the user-supplied physical terminal output edit routine specified in the TYPE or LINEGRP macro statement is to be used for this workstation. The default is NO.

The second parameter specifies whether (YES) or not (NO) the user-supplied physical terminal input edit routine specified in the TYPE or LINEGRP macro statement is to be used for this workstation. The default is NO.

FEAT=

(If the system definition statements from previous releases of IMS are used, TERMINAL macro statements from previous releases are valid for this release.)

- For nonswitched or local 3270 terminals:

  The first subparameter specifies whether a DEKYBD (Data Entry Keyboard) or PFK (Program Function Keys) is available on this 3275 or 3277 terminal. PFK and DEKYBD are not valid, and NOPFK (no Program Function Keys) is forced for 3284 and 3286 terminals. The default is PFK.

  The second subparameter specifies whether the Operator Identification Card Reader is available (CARD) or not (NOCD) on this 3275 or 3277 terminal. CARD specification is not valid, and NOCD is forced for 3284 and 3286 terminals. The default is CARD.

  The third subparameter specifies whether (PEN) or not (NOPEN) the Selector Pen is available on this 3275 or 3277 terminal. PEN specification is not valid and NOPEN is forced for 3284 and 3286 terminals. The default is PEN.

- For terminals defined as SLUTYPE2:

  The comments for nonswitched and local 3270 terminals apply equally to terminals defined as SLUTYPE2.

- For all 3270 Terminals:

  IGNORE can be specified if the presence or absence of the value does not affect terminal operation within IMS.

  If IGNORE is not specified, the default values assumed for any omitted subparameters are PFK, CARD, and PEN, respectively.

- For all 3270 terminals (except for the printer component of the 3275) and secondary logical units type 2 (SLU 2):

  F1, F2...F10 specify terminal features that the user designates to correspond to each integer. The same number is used for any given feature during both system definition and MFS processing. This specified integer is used instead of, and not in addition to, other feature specifications.

  If you are using the same MFS format for the 3180 and 3290, Fn must have different values for the 3180 and 3290. These same values must be coded on the corresponding FEAT=Fn parameter of the MFS DEV statements.

- For 3275 printer components:

FEAT=F1...F10 is not a valid specification for 3275 printer components (that is, devices specified as UNIT=3275 and COMPT=PTR1).

- For 3284 and 3286 units:

  F1...F10 specify terminal features that the user designates to correspond to each integer. The same number is used for any given feature during both system definition and MFS processing. This specified integer is used instead of, and not in addition to, other feature specifications.

  Specifications of FEAT=F1...F10 and PTRSIZE are mutually exclusive for the 3284 and 3286 devices. If PTRSIZE= is specified, FEAT= is invalid. If FEAT= is specified, no default PTRSIZE is assumed.

  Also, FEAT=IGNORE is invalid for these devices. PTRSIZE=IGNORE is used instead.

  The use of a particular specification for F1...F10 should be directly related to a particular printer line size and should be consistently used for printers having that line size and identical attributes throughout your system definition.

  Example: FEAT=F5 might be used to define printers with a line size of 126 print positions and color, FEAT=F6 for printers with a line size of 126 print positions but without color, and FEAT=F7 for printers with a line size of 120 or 132.

- For UNITYPE=3601 or FINANCE and COMPT=36FP:

  DUAL indicates the presence of the DUAL forms control feature, and 132 indicates the presence of the extended print line feature.

- For SPOOL Terminals:

  If FEAT=AUTOSCH, the print utility is automatically scheduled when the spool lines are stopped, when a spool data set fills, or when IMS is initialized. If FEAT=NOAUTOSCH (the default), the MTO must schedule the print utility.

FPBUF=

Specifies the Fast Path terminal buffer size for the following terminal types: 3270 (VTAM), 3601, SLU 1, SLU 2, NTO, SLU P, and LU 6.1. FPBUF provides the system default for the EMH buffer if the EMHL parameter is not specified. FPBUF=0 means that the terminal is not eligible for Fast Path processing. FPBUF=0 is the default. This specification can also be used to override an FPBUF= default on the TYPE macro statement. If the terminal is eligible for Fast Path transaction processing, the minimum value that can be specified is 12. The maximum value is 30 720. The value specified must not be greater than the logical record length of the long message queue data set specified in the RECLNG= parameter of the MSGQUEUE macro.

This keyword is optional and cannot be specified on ETO terminal descriptors. See the RECLNG parameter of the MSGQUEUE macro for further information.

LTERM=

Specifies a logical terminal name for this TERMINAL statement. This name is used to check input terminal security for this terminal. The NAME macro referred to must precede this TERMINAL statement. When this operand is present, the TERMINAL statement cannot be followed by a NAME statement.

This operand is required for READER terminals. For example, a previously defined TERMINAL statement defined a 3270 remote terminal and was followed by the name ABLE. In addition, assume that this TERMINAL

statement refers to a READER terminal that is an input-only device. All input security for the READER terminal is checked against the security characteristics assigned to ABLE.

This operand cannot be specified for NTO devices and is not supported for ACF/VTAM terminals.

When a list of LTERMs is defined under the previous TERMINAL statement to which reference is made, only the first LTERM name is selected from that list. If an attempt is made to select other than the first, IMS ignores the name chosen and assigns the value of the first name listed.

Example:

```
TERMINAL    ADDR=E2
  NAME      LTERM1
  NAME      LTERM2
  NAME      LTERM3

TERMINAL    ADDR=E4,LTERM=LTERM2
```

In this example, IMS treats the second TERMINAL statement as though it reads LTERM=LTERM1.

MODEL=

Specifies the terminal model number for terminals with screen sizes of 480 (Model 1) or 1920 (Model 2) characters. Terminals with screen sizes other than 480 or 1920 characters are defined using the TYPE and SIZE keyword parameters. The default is 1. The model number specified for a 3277 for which OPTIONS=COPY is specified must be less than or equal to the model number specified for any candidate printer. The MODEL and TYPE/SIZE keyword parameters are mutually exclusive.

MODETBL=

Specifies the name of the VTAM logon mode table entry (logon mode name) that contains the SNA bind parameters to be used when a session is initiated by the MTO or through the /OPNDST command.

This function allows a system definition specification for referencing an entry other than the default entry in the user's VTAM logon mode table. Normally, the terminal operator specifies this mode table entry name when logging on to a terminal; this is not possible, however, from the IMS master terminal because IMS initiates the session.

With this function, if MODETBL= is not specified at system definition, no functional or operational change affects the user.

If MODETBL= is specified at system definition, the specified entry name is used. The MODETBL can be overridden by:

- The LOGON APPLID entry by the remote terminal operator
- VARY ACT, LOGON= command by the network terminal operator
- /OPNDST, /RST, or /CHANGE command by the MTO

The MODETBL= parameter is required for the IMS master terminal if the VTAM default mode table is not configured specifically for the device to be used as the IMS master terminal.

With MODETBL=, you do not need to specify the mode table entry when logging on to terminals that always require specification of the same mode table entry name.

MSGDEL=

Specifies which message types IMS should discard for this terminal, if it is not

the master terminal. The default is NONE for non-VTAM terminals and SYSINFO for VTAM terminals. NONE is ignored for VTAM. The MSGDEL option applies to a switched terminal only while that terminal is connected to IMS; any message queued while an LTERM is not connected to the switched line is not deleted.

NONIOPCB | NOTERM | SYSINFO | NONE

NONIOPCB specifies that IMS should discard the following message types destined for this terminal:

- Message switches
- Messages inserted by an application program to an alternate PCB
- /BROADCAST messages
- DFS059 TERMINAL status messages

When a switched terminal defined as MSGDEL=NONIOPCB is not connected to IMS, any message normally discarded as a function of specifying NONIOPCB is queued for an LTERM that might become associated with the switched terminal at signon (/IAM) time.

NOTERM specifies that IMS should discard DFS059 TERMINAL status messages and DFS3650 SESSION status messages for this terminal. Do not specify NOTERM for 3601, SLU P, or LU 6.1 VTAM terminals. If NOTERM is specified, a warning message is issued and the specification default is SYSINFO.

SYSINFO specifies that IMS should discard DFS059 TERMINAL status messages for this terminal.

NONE specifies that IMS should not discard any message types for this terminal.

When MSGDEL is specified, the following additional restrictions apply to the reassignment of logical terminals:

- Assignment of a logical terminal to a physical terminal defined as MSGDEL=NONIOPCB can only take place if no messages are in the Q3 (logical terminal's system) or Q4 (other than system) queues, unless that logical terminal is currently assigned to a physical terminal also defined as MSGDEL=NONIOPCB.
- Assignment of a logical terminal to a physical terminal defined as MSGDEL=SYSINFO or MSGDEL= NOTERM can only take place if no messages are in the system queue of the logical terminal, unless that logical terminal is currently assigned to a physical terminal defined as MSGDEL=SYSINFO, MSGDEL=NOTERM, or MSGDEL=NONIOPCB.

An LU 6.1 session can be initiated only if the MSGDEL specifications on both the TERMINAL macro statement and the SUBPOOL macro statement match. An LU 6.1 session is not established, and an error message is issued if these specifications are not identical.

NAME=

Must be the VTAM node name specified during VTAM/NCP definition.

When using XRF and VTAM-supported master terminals, a second node name should be specified for the master terminals. The two node names specify nodes in the two IMS XRF systems. The first is used by the system with HSBID=1 and the second by the system with HSBID=2. If a secondary master is used, it also should have two specifications. For more information about the

HSBID parameter, see "Environments that support IMS system definition-supplied procedures" on page 446.

When using RSR and VTAM-supported master terminals, a third node name can be specified for the master terminals. The three node names specify nodes in the IMS RSR subsystems. The first is used by the subsystem with MTOID=1, the second by the subsystem with MTOID=2, and the third by the subsystem with MTOID=3. If a secondary master is used, it also should have at least two specifications (for RSR without XRF) or three specifications (for RSR with XRF at the active site). The third node name can only be used for the tracking subsystem. For more information see the description of the MTOID parameter for "DFSRSRxx member of the IMS PROCLIB data set" on page 720.

The specification of *nodename* corresponds to the HSBID= specification at IMS start up. This means that HSBID=2 uses *nodename2* as the master or secondary master terminal, and uses PASSWD2 (from COMM macro). RSR tracking subsystems must select *nodename* and PASSWDn using the MTOID= parameter in the DFSRSRxx member. Tracking subsystems can override the APPLID for the subsystem by using the APPLID3= startup parameter.

When you define an ISC surveillance link between the XRF active and alternate systems, the two node names in the NAME keyword must match the two VTAM application names in the APPLID keyword of the COMM macro.

For an ISC node communicating with an XRF complex, the node name must be the VTAM USERVAR associated with the XRF complex. Defining additional nodes for the actual VTAM APPLIDs of the IMS systems which comprise the XRF complex results in errors during session initiation.

The recommended use of the *nodename* keyword is:

- Use *nodename1* only for DB/TM or DCCTL systems that are non-XRF and non-RSR
- Use *nodename1* and *nodename2* for DB/TM or DCCTL systems using XRF
- Use *nodename1* and *nodename3* for DB/TM or DCCTL systems using RSR
- Use *nodename1*, *nodename2*, and *nodename3* for DB/TM or DCCTL systems using both XRF and RSR

The default for *nodename3* is the value of *nodename1*.

OPTIONS=

Specifies certain communication options associated with this terminal. The OPTIONS= subparameters are not position dependent.

ACK | OPTACK

ACK indicates that recoverable and update transactions must be acknowledged with a definite response 1 (FME) or definite response 2 (RRN). OPTACK allows input messages containing only a begin bracket indicator to be acknowledged by an end bracket indicator on the next output message. A definite response must be requested for recoverable input if ACK is specified, and is optional if OPTACK is specified. The use of OPTACK also affects the sequence number recording for message resynchronization.

Related reading: Additional information about ACK and OPTACK is found in *IMS Version 11 Communications and Connections*.

OPTACK is forced for Fast Path-eligible terminals if the value specified for FPBUF is greater than 0. ACK can be specified if BID is not also used. ACK allows a 3600/LUP terminal to add Fast Path without initially recoding the remote terminal program.

BSELM | NBSELM
Specifies whether (BSELM) or not (NBSELM) the backspace character elimination is required for input from a SLU 1 terminal.

COPY | NOCOPY
Specifies whether (COPY) or not (NOCOPY) the copy function is requested for this terminal. Terminals defined with the copy function must have appropriate candidate printers.

Related reading: For a detailed discussion of candidate printers for 3270 terminals, refer to the discussion of controlling output in *IMS Version 11 Communications and Connections*.

This subparameter is supported for dynamic SLU 2 terminals, but not for dynamic non-SNA VTAM 3270s.

FPACK | NFPACK
Applies only to 3601/Finance and SLU P terminal types and specifies whether (FPACK) or not (NFPACK) Fast Path output message protocol is to be used, as defined in *IMS Version 11 Communications and Connections*.

The Fast Path output message protocol for FPACK normally requires another input (transaction or SNA RTR command), following transmission of the output Fast Path reply message, before any asynchronous output message can be sent. The NFPACK option indicates that standard non-Fast Path protocols are to be used, and allows any asynchronous output message to be sent immediately following acknowledgment of the Fast Path output reply message.

NOASR | ASR
For the Session Outage Notification facility only; specifies automatic session restart (ASR) on a VTAM node. The default is NOASR. Specifying ASR or NOASR on the TERMINAL macro overrides ASR definitions on the TYPE macro for a defined node.

Use the `/DIS` command to display the current ASR option, and the `/CHANGE` command to change it.

The ASR option is not supported for nodes that can have XRF backup sessions.

For LUTYPE6 nodes, the ASR option works only if IMS is acting as the primary logical unit (PLU).

NOAUTSGN | AUTOSIGN
AUTOSIGN indicates that IMS automatically signs on to static terminals with the first LTERM name as the user ID, bypassing a password check. The user ID needs to be defined in RACF.

For ISC static terminals, the SUBPOOL name is used as the user ID instead of the LTERM name.

NOAUTSGN indicates that IMS does not automatically sign on to static terminals with the first LTERM name as the user ID. This is the default.

The AUTOSIGN and NOAUTSGN definitions in this macro override those definitions on the TYPE macro.

If you specify AUTOSIGN on the TYPE macro, every subsequent TERMINAL macro after that TYPE macro also specifies AUTOSIGN. You can specify exceptions to the AUTOSIGN specification by specifying NOAUTSGN for particular TERMINAL macros.

NOBID | BID
NOBID indicates that the VTAM BID command is never to be used. BID indicates that the VTAM BID command always precedes output messages that occur while between brackets, regardless of response mode option used.

Before the availability of this option, IMS systems forced the equivalent of NOBID if NORESP was specified, and forced the equivalent of BID if FORCRESP is specified.

NODISCON | DISCON
Determines whether IMS should automatically terminate a session for a terminal defined as SLUTYPE1 when operating in unattended mode. (ATTEND or UNATTEND mode is determined from the logon mode table entry specified to be used during session initiation.) DISCON specifies that IMS should automatically terminate the communication session with the terminal after receiving all immediately available input or sending all available output. NODISCON suppresses that automatic termination. The logon mode table is described in the MODETBL= keyword description.

NOFES | FES
Specifies whether (FES) or not (NOFES) this terminal is capable of front—end switching. The default is NOFES.

To define a terminal as capable of front-end switching, define the front end switch exit routine with the FESEXIT keyword of the COMM macro. See "COMM macro" on page 315 for more information about the FESEXIT keyword.

If FES is specified, the FES exit gets control for each input from this terminal.

NOMFS | MFS
Specifies whether (MFS) or not (NOMFS) the Message Format Service (MFS) is to be provided for this terminal. The default is NOMFS.

NOMTOMSG | MTOMSG
Specifies whether (MTOMSG) or not (NOMTOMSG) normal session initiation and termination messages are to be sent to the IMS MTO. The default is NOMTOMSG.

NOSHARE | SHARE
Specifies whether (SHARE) or not (NOSHARE) this printer can be shared between VTAM subsystems. If SHARE is specified, then IMS can issue a SIMLOGON VTAM macro when either of the following occurs:

- Data is queued for this printer.
- The VTAM 3270 PF12 key invokes the internal IMS copy function that selects this terminal, and the terminal is not connected to IMS.

If SHARE is specified, the OPNDST option must also have been selected for this terminal.

If SHARE is specified, a MODETBL entry cannot be specified for cross-domain resources. An entry of ALL X'00' in the VTAM MODETBL indicates to ACF/VTAM that the default session parameters are to be used for this terminal and that the terminal can be defined in any VTAM domain.

If the default option, NOSHARE, is specified, and if IMS releases this terminal to another VTAM application, the IMS MTO must issue an /OPN command, a VTAM VARY command, or a UNIX System Services logon for IMS to reacquire the terminal.

This option is applicable to SLU 1, 3284/86 local, and VTAM 3270 devices only. If UNITYPE=3270 is specified on the TYPE macro statement, UNIT must be a 3284 or 3286 device.

NOSIGNON | SIGNON
For VTAM terminals only, specifies signon verification security processing (SIGNON). The default is NOSIGNON.

The SIGNON and NOSIGNON definitions in this macro override those definitions on the TYPE macro.

Instead of specifying OPTIONS=SIGNON in more than one system definition macro, you can require all static terminals to sign on by specifying a single parameter, SIGNON=ALL, in the DFSDCxxx PROCLIB member.

Related reading: For more information about the SIGNON keyword, see "DFSDCxxx member of the IMS PROCLIB data set" on page 656.

OPNDST | NOPNDST
Applies only to VTAM terminals and specifies whether (OPNDST) or not (NOPNDST) the /OPNDST command is valid for this node. The default for all terminal types except NTO is that /OPNDST is valid.

The default value for NTO devices is NOPNDST. Either NOPNDST, the default value, or OPNDST is valid for nonswitched NTO devices. When using switched NTO devices, NOPNDST must be specified or accepted by default, because these devices must be logged on by an operator. An error results if OPNDST is specified for switched NTO devices.

The NOPNDST option is used to prevent the /OPNDST command from being used to start a session between IMS and a VTAM terminal.

This option does not apply to a VTAM terminal specified in the NAME macro as MASTER (the IMS master terminal), because that terminal is automatically opened.

PAGDEL | NPGDEL
Specifies whether (PAGDEL) or not (NPGDEL) automatic page deletion is to be specified for this terminal. This option applies to 3270 terminals, SLU 2 terminals, SLU P nodes, and to 3601 workstations using MFS. PAGDEL is the default option, but the LU 6.1 nodes default is NPGDEL.

When the last physical page of the last logical page of the message is sent, the message is dequeued no matter which parameter is specified.

PAGDEL should be specified for terminals used in an interactive or conversational manner. If input from the terminal or node is normally followed by a response to the terminal or node, PAGDEL should be specified.

NPGDEL is provided for terminals that produce many input messages to which no response is made. When NPGDEL is specified, input from a paged message is responded to by IMS with the first page of that same message. Any application program response is queued.

PROT | UNPROT
Specifies whether (PROT) or not (UNPROT) the terminal display is

protected after each page of a message. If PROT is specified, the protected status of the terminal depends upon the unprotected screen mode option on the DSCA keyword of the MFS DEV statement or the SCA keyword of the MFS MFLD statement. This keyword applies only to 3270 display terminals and secondary logical units type 2 (SLU 2).

RELRQ | NORELRQ
Specifies whether (RELRQ) or not (NORELRQ) IMS is to release this terminal upon request from other VTAM subsystems. If NORELRQ is specified, requests by other VTAM subsystems for this terminal are not honored. The MTO must intervene to reassign this terminal to another VTAM subsystem. If RELRQ, the default, is specified, then IMS releases this terminal to other VTAM subsystems upon request.

This option is applicable to devices specified on the TYPE macro statement as UNITYPE=3601, FINANCE, SLUTYPE1, SLUTYPEP, and LUTYPE6, in addition to 3284 and 3286 printers.

NORELRQ is forced for 3277, SLU2, and NTO terminals.

SCAN | NOSCAN
Specifies whether (SCAN) or not (NOSCAN) IMS should edit 3601 device-control sequences (such as SELECT, POSITION, and WRITE TRANSPARENT) that would otherwise be split across two transmissions when a non-MFS formatted output segment must be split into multiple transmissions.

If an MFS-formatted output segment must be split across two transmissions, IMS edits 3601 device-control sequences (SELECT and POSITION) that would otherwise be split across two transmissions even if the NOSCAN option is specified.

Related reading: See *IMS Version 11 Communications and Connections* for additional information.

SYNCSESS | FORCSESS
Specifies when session initiation is to be completed.

- SYNCSESS specifies that session initiation is to be completed only when session restart modes (message sequence numbers) agree.
- FORCSESS specifies that session initiation is to be completed regardless of the agreement of session restart modes.

The default is SYNCSESS.

TRANRESP | NORESP | FORCRESP
TRANRESP specifies that the transaction code definition should be referred to determine whether response mode should be used. All Fast Path-eligible terminals operate in response mode—either TRANRESP or FORCRESP.

- TRANRESP is the default value for 3767, 3770 switched and station-control terminals, and for terminals defined as SLUTYPE1, NTO, or LUTYPE6.
- NORESP specifies that response mode is not to be allowed for this terminal. NORESP is the default for 3270, and 3601 terminals, and terminals defined as SLUTYPE2 and SLUTYPEP. NORESP is not compatible with Fast Path.
- FORCRESP specifies that response mode is to be forced for all transactions entered from this terminal.

For VTAM terminals, the combined specification of FORCRESP and NPGDEL is not recommended. Also, if MSGTYPE=RESPONSE is specified

on the TRANSACT macro statement for these terminals, the combined specification of TRANRESP and NPGDEL is not recommended.

The keyboard of a display terminal defined with no automatic page deletion (NPGDEL) can become locked during execution if operating in terminal response mode. NPGDEL specifies that the current output message is not to be dequeued at the time of input. However, with terminal response mode, input is inhibited until the current output message is dequeued and the terminal response mode is reset. Thus, a terminal receiving multiple pages of output and defined with the combined specification of FORCRESP and NPGDEL might require master terminal intervention to reset the terminal response mode.

TRSOSI | NOTRSOSI
Specifies whether IMS translates the shift-out/shift-in (SO/SI) characters for MFS editing. The SO/SI characters indicate which characters are used for EGCS terminals. The default of TRSOSI should be used for non-EGCS terminals. This keyword applies only to 3270 terminals and type 2 secondary logical units (SLU2).

UNLOCK | LOCK
Specifies whether IMS is to unlock the terminal keyboard and reset the modified data tags after an MFS bypass.

- UNLOCK specifies that MODNAME=DFS.EDTN unlocks the terminal keyboard and resets the modified data tags
- LOCK specifies that an application program assumes responsibility for unlocking the terminal keyboard and resetting the modified data tags.

This keyword applies only to 3270 terminals and type 2 secondary logical units (SLU2).

OUTBUF=
Specifies the size of the IMS output buffer to be used for the workstations that are identified in this parameter description. The OUTBUF value indicates the maximum size of IMS message segments that are sent from IMS to the workstation in a single transmission.

For VTAM SNA devices, a single transmission represents a transmission from IMS to VTAM. For these devices, the output message is blocked (chained) based on the OUTBUF value and might result in multiple transmissions for a single message.

For VTAM non-SNA devices, IMS sends the message in a single transmission. No blocking or chaining is performed.

If MFS bypass is used, the OUTBUF value must be large enough to contain the entire output message. If the message that is inserted by the IMS application exceeds the OUTBUF value, the message is rejected by IMS. If MFS is used to edit the output message, the OUTBUF value is ignored and the message is sent regardless of its length. When sending to a printer, the OUTBUF value must be equal to or greater than the segment size, but less than 4096 bytes.

IMS also uses the OUTBUF value to set the transmission services usage field (byte 11) in the bind parameter. If this value is specified incorrectly, it might cause the device to reject the bind parameter. The value specified, plus the size of any routing information later appended by VTAM, must be less than the maximum message size specified for VTAM and its related NCPs. The output buffer size is calculated as follows for:

3270: Record size. The range of values that can be specified is 256 through 30720 bytes. The default is 2000 bytes. The actual hardware buffer capacity should be considered when creating the OUTBUF calculation. The hardware buffer maximum might be less than the default of 2000 bytes. For a device that is not a SNA 3270, OUTBUF is used only if you are using MFS bypass. Otherwise, OUTBUF is ignored.

Related reading: For information about devices attached to a local 3274 control unit, see *3274 Control Unit Description and Programmer's Guide*.

3601 or FINANCE: Record size plus header. (Header is 3 bytes if MFS is not used or 29 bytes if MFS is used.) The range of values that can be specified is 64 bytes to 30720 bytes. The default is 64 bytes.

SLU 1: Record size plus header. (The header is 25 bytes.) The range of values that can be specified is 128 bytes to 30720 bytes. The default is 256 bytes.

NTO: The range of values that can be specified is 256 bytes to 30720 bytes. The default is 256 bytes.

Output message segments are not blocked. Each segment is sent as an RU. Segments greater than the RU size are spanned, but an RU never contains more than one segment.

SLU 2: The range of values that can be specified is 256 bytes to 30720 bytes. The default is 1500 bytes. A 3790 defined as SLUTYPE2 has a maximum value of 1536.

SLU P: Record size plus header. (The header is 5 bytes if MFS is not used and up to 42 bytes if MFS is used.) The range of values that can be specified is 64 bytes to 30720 bytes. The default is 64 bytes.

LU 6.1: Record size plus header. The range of values that can be specified is 256 bytes to 30720 bytes. The default is 256 bytes. The header size without MFS is variable from 8 bytes to 45 bytes; with MFS, the header size is variable from 8 bytes to 62 bytes.

Related reading: For more information about LU 6.1 record size, see *IMS Version 11 Communications and Connections*.

For all logical units that require a bind from IMS, the OUTBUF buffer size must be a decimal value that can be expressed by the algorithm X times 2 to the power of Y. X must be a value from 8 through 15 and Y must be a value from 3 through 11. This is a VTAM restriction. Thus, for example, the value 144 (representing $9x2^4$) and the value 28672 (representing $14x2^{11}$) are acceptable values. Details of this VTAM restriction are described in *z/OS V1R2 Communications Server: SNA Programming*, with information about requesting unit sizes.

For an IMS Version 11 system partnering with another IMS Version 11 system or an IMS Version 11 or IMS Version 10 system, any size between 1024 and 65536 is valid. For IMS Version 11 systems partnering with versions of IMS before Version 10, valid VTAM buffer sizes are restricted to one of the following decimal values:

Table 59. Valid VTAM buffer sizes for IMS Version 11 systems partnering with versions of IMS before Version 10

112	320	896	2560	7168	20 480
120	352	960	2816	7680	22 528
128	384	1024	3072	8192	24 576

Table 59. Valid VTAM buffer sizes for IMS Version 11 systems partnering with versions of IMS before Version 10 (continued)

144	416	1152	3328	9216	26 624
160	448	1280	3584	10 240	28 672
176	480	1408	3840	11 264	30 720
192	512	1536	4096	12 288	
208	576	1664	4608	13 312	
224	640	1792	5120	14 336	
240	704	1920	5632	15 360	
256	768	2048	6144	16 384	
288	832	2304	6656	18 432	

IMS converts the value given here to the format required for the bind parameter fields and places the value into the appropriate field in the bind parameter list. If the maximum RU value specified cannot be converted exactly into the bind format, the value is rounded down to the next-lower value bind format for inbound RUs and rounded up to the next-higher value bind format for outbound RUs.

Note: For IMS to CICS ISC connections, IMS checks the ISC link buffer definitions during the bind to avoid potential storage corruption that could occur by receiving a message that is larger than the available buffer. If a mismatch is detected, IMS issues message `DFS2066I`.

To avoid this conflict, ensure that:

1. If CICS is the primary half session, the CICS RECEIVESIZE is greater than or equal to the IMS OUTBUF size.
2. If IMS is the primary half session, the CICS SENDSIZE is greater than or equal to the OUTBUF size.
3. VTAM buffer sizes are a valid decimal value (expressed by the algorithm x times 2 to the power of y, with x = 8 through 15, and y = 3 through 11). If valid VTAM buffer sizes are not specified, unwanted rounding of those values might occur, and a mismatch in bind RU sizes might result.

PTRSIZE=

Specifies the number of print positions of the 3284 or 3286 printer. The default is 120.

PTRSIZE=IGNORE causes MFS to always use the device output format (DOF) with FEAT=IGNORE in its DEV statement when editing output for this device. The existing parameters and control block values, including default parameters, are unchanged.

PU=

Specifies the type of physical device that is connected by means of the Network Terminal Option (NTO) licensed program. PU= is a required keyword for NTO devices; an error message results if it is omitted.

Each keyword represents a device type.

- TTY—TTY devices, such as a 33/35, or TTY compatible devices, such as a 3101 running in TTY mode.
- LUNS—These devices are non-SNA logical units, such as a 3101 not running in TTY mode.

SEGSIZE=

Specifies the maximum size of IMS message segments that are sent to IMS. Acceptable values are from 256 bytes to 32000 bytes. The default is 256.

If an APPC logical record exceeds the SEGSIZE value, IMS breaks the record into multiple IMS segments. The IMS application must issue get Next (GN) DL/I calls to receive the second through nth segments.

Calculate the segment size as follows:

- SLU 1, SLU P, and NTO devices:

  Size is the maximum input segment size that can result from blocking or deblocking one or more SCS1 input records. This parameter is ignored for non-SCS1 devices.

- LU 6.1:

  Size equals the size of the largest input record that can result from blocking or deblocking one or more VLVB or chain records.

  IMS spans queue buffers on an ISC input message if the data is sent in VLVB format. This allows IMS to handle large ISC input messages without having to increase the LGMSG buffer size. IMS obtains an input buffer based on SEGSIZE and fills this buffer with VLVB input data before enqueuing as a message segment. For example, if the LGMSG buffer size is 8 KB and the SEGSIZE is 8 KB, a 32-KB message received from CICS in VLVB format would be enqueued as four 8 KB message segments spanning four 8-KB LGMSG queue buffers. The IMS application would issue a Get Unique (GU) call followed by 3 Get Next (GN) calls to get the entire message.

  A DFS2056 message is issued if the input segment exceeds the value for SEGSIZE.

SESSION=

For LU 6.1 sessions, specifies the maximum number of parallel sessions for the logical unit defined in the TERMINAL macro statement. From 1 to 4095 parallel sessions can be established. The default is 1.

When defining the XRF ISC SURVEILLANCE link, set the number of parallel sessions equal to one, or eliminate the SESSION= parameter.

SIZE=

Specifies the physical screen size of this 3270 display terminal having the device type symbolic name in the TYPE= operand. MFS uses the specified screen size to verify format specification. The first and second positional parameters of the operand relate to screen lines and columns, respectively. Both numbers must be specified as decimal numbers having the following range:

```
llll = 1 through 16384
cccc = 2 through 16384
product of llll x cccc = 80 through 16384
```

where *llll* is the number of lines and *cccc* the number of columns available on the screen.

For 3270 devices defined as SLUTYPE2, the maximum number of lines and the maximum number of columns that can be defined for each is 255. Any symbolic name can be associated with any size screen. However, after this association is defined, the same device type symbolic name must be used. That is, it must duplicate a previously specified TYPE= specification of the LINEGRP, TYPE, or TERMINAL statement.

Specification of the physical screen size is required for the first (or only) definition of the specific device type symbolic name and can be omitted for subsequent identical device type symbolic name specifications. If omitted, the SIZE keyword operand originally specified with the device type symbolic name is used when the symbolic name is repeated. If the SIZE keyword is specified, the screen size defined must be the same as that defined for the original device type symbolic name specification. You should establish a standard for relating device type symbolic names to the physical screen sizes.

The standards listed in Table 60 are recommended:

Table 60. Standards to relate device names to screen sizes

User-defined symbolic name	Screen size
3270-A1 or 3270-A01	12x80
3270-A2 or 3270-A02	24x80
3270-A3 or 3270-A03	32x80
3270-A4 or 3270-A04	43x80
3270-A5 or 3270-A05	12x40
3270-A6 or 3270-A06	6x40
3270-A7 or 3270-A07	27x132
3270-A8 or 3270-A08	62x160

For the 3270 display devices defined with the device type symbolic name, the system definition produces an MFS Device Characteristics Table (DFSUDT0x) where x is the suffix specified in the SUFFIX keyword of the IMSGEN macro statement. You can also use the MFS Device Characteristics Table (DCT) utility (DFSUTB00) to build and update this table. This table is stored in IMS.SDFSRESL and is used by the MFS Language Utility and the IMS online to extract the information associated with the specific device type symbolic name.

Important: The MFS DCT must include each combination of device type symbolic name, features, and screen size for all 3270 and SLU-2 terminals that are to be connected to IMS. Because ETO terminals are not defined in the IMS system definition, if DCT entries for ETO 3270 and SLU-2 terminals are not already in the DCT as the result of system definition statements for static terminals, use the MFS DCT utility to add the entries.

If the MFS Language utility (DFSUPAA0) cannot find the correct entry in the MFS DCT, based upon device type symbolic name (3270-An) and features, then the MFS Language utility produces error message `DFS1862` or `DFS1863`. When a user logs on to an ETO 3270 or SLU-2 terminal that is connected to IMS, and VTAM provides the screen size to IMS, IMS uses the screen size and the features that are specified on the logon descriptor to search the MFS DCT. IMS searches the MFS DCT to determine the device type symbolic name. If the search is unsuccessful, the logon attempt is rejected with messages `DFS3646` and `DFS3672`.

Related reading:

- Refer to *IMS Version 11 Database Utilities* for additional information about the MFS Language utility and the MFS DCT utility.
- Refer to *IMS Version 11 Application Programming APIs* for more information about the Device Characteristics Table

MNOTE statements in the system definition stage 1 output listing denote each device type symbolic name specified for 3270 and SLU type 2 devices with their corresponding screen size values.

Example:

Display Device Symbolic Names and Screen Sizes

```
                              Screen size
   Symbolic Name           Lines        Columns

     3270-A01                12            80
     3270-A02                24            80
     3270-A03                32            80
     3270-A04                43            80
```

After a device type symbolic name (for example, 3270-A03) is associated with a screen size (for example, 32x80), the size need not be specified and must not be changed for any TYPE or TERMINAL macro statement for which the same device type symbolic name is used. During a subsequent system definition, the MFS Device Characteristics table and terminal control blocks can change if terminal specifications are altered. You should examine your MFS formats to determine whether they are affected by the changes and recompile them if necessary.

TYPE=

Specifies that an alphanumeric symbolic name for a device type is to be associated with this IMS-supported VTAM 3270 physical display terminal. The symbolic name can be of the form 3270-A*nn*, where *nn* is a value from 01 through 15. The leading zero can be omitted. The physical screen size of the 3270 display device associated with the device type symbolic name is specified in the SIZE= operand. Any 3270 display device can be defined with a device type symbolic name. 3270 display devices having 480-character or 1920-character screen sizes can be defined using the MODEL keyword parameter rather than the TYPE and SIZE keyword parameters. However, 3270 display devices with other than 480 or 1920 characters must be defined using the device type symbolic name and, therefore, TYPE and SIZE. The TYPE/SIZE and MODEL keyword parameters are mutually exclusive.

UNIT=

Specifies the terminal for the previously defined line group. For 3270 line groups, UNIT= is used to define this terminal as a 3284 or 3286 printer or 3277 display unit. 3277 display units cannot be defined in the same local line group as 3284 or 3286 printers.

TRANSACT macro

Use the TRANSACT macro statement to specify the transaction codes that cause the application program named in the preceding APPLCTN macro to be scheduled for execution in an IMS message processing region.

The TRANSACT macro statement is used one or more times with each APPLCTN macro statement to identify transactions as IMS exclusive, IMS Fast Path potential, or IMS Fast Path exclusive. It also provides the IMS control program with information that influences the application program scheduling algorithm. It can define a message editing routine.

If you do not include this macro during stage 1 system definition, no warning is issued, and it is assumed that you define your transaction codes dynamically. You can define them dynamically using either CREATE TRAN and UPDATE TRAN

commands, or the Destination Creation exit routine (DFSINSX0). See "Dynamic definition" on page 422 for more information.

Attention: When you define programs or transactions (using the APPLCTN and TRANSACT macros) to an IMSplex using shared queues, you must ensure that the programs and transactions are defined with the same attributes on all IMS systems in the IMSplex. If the attributes are defined differently on different IMS systems in the IMSplex, results can be unpredictable. For example, a transaction defined as conversational on one IMS and non-conversational on another IMS causes unpredictable results when messages for that transaction are put on the shared queues.

If the TRANSACT macro statement is preceded by an APPLCTN macro defining it as remote (specifies SYSID), this transaction is generated as a remote transaction.

An IMS Fast Path-exclusive transaction is identified by a TRANSACT macro statement following an APPLCTN statement that specifies both PGMTYPE=TP and FPATH=YES.

TRANSACT macro statements that identify IMS Fast Path-exclusive transactions generate routing code entries. These entries are automatically generated and have the same name as the corresponding transaction codes. They are used by the IMS Fast Path Expedited Message Handler routines to locate the Fast Path application program named in the preceding APPLCTN macro statement that processes the input message.

An IMS Fast Path-potential transaction is identified by a TRANSACT macro statement that specifies FPATH=YES following an APPLCTN statement that specifies PGMTYPE=TP and FPATH=NO.

An application defined as FPATH=NO cannot be run in a Fast Path region (IFP), even though the application contains Fast Path-potential transactions. To run these Fast Path-potential transactions in an IFP region (a region that has FPATH=YES specified on the APPLCTN macro), both of the following conditions must be true:

- A routing code must be specified for the IFP on the RTCODE macro.
- The Fast Path-potential transaction must be routed to the IFP by the appropriate routing code.

Related reading: See the description of the Fast Path Routine for the Input Edit/Routing exit routine in *IMS Version 11 Exit Routines*.

All Fast Path transactions are implicitly defined as recoverable.

MSGTYPE=(SNGLSEG,RESPONSE) must be specified on the TRANSACT statement for Fast Path potential transactions.

Additional routing codes can be associated with a Fast Path application program by using the RTCODE macro statement following the corresponding APPLCTN macro statement.

TRANSACT macro statements with FPATH=YES and RTCODE macro statements are invalid after APPLCTN macro statements that define an IMS batch message processing (BMP) application.

Dynamic definition

If you do not include this macro during stage 1 system definition, no warning is issued, and it is assumed that you define your transaction codes dynamically.

To dynamically define the transaction codes that cause previously defined application programs to be scheduled for execution, you can use:

- The CREATE TRAN and UPDATE TRAN type-2 commands described in *IMS Version 11 Commands, Volume 1: IMS Commands A-M*.
- The Destination Creation exit routine described in *IMS Version 11 Exit Routines*.

Changes to existing transactions cannot be introduced through a MODBLKS system definition if those attributes are changeable by online commands.

When DRD is disabled, CREATE, DELETE, IMPORT, and most UPDATE commands that change the definitional attributes of a resource are rejected.

The following table compares the TRANSACT macro keywords and the equivalent CREATE and UPDATE command keywords used for dynamic definition. The table also includes equivalent values for the INSXTRNQ DSECT, which is mapped to by the DFSINSX0 parameter list. The information in INSXTRNQ is used to create a transaction control block if the destination is a transaction. Default values are shown in **bold**.

Table 61. TRANSACT macro keywords and the equivalent CREATE and UPDATE command keywords used in dynamic definition

TRANSACT macro keyword	CREATE | UPDATE TRAN keyword equivalent	INSXTRNQ DSECT information
Previous APPLCTN macro statement	PGM(*name*)	TRNQ_PGM
AOI=(**N** | TRAN | Y)	AOCMD(**N** | TRAN | Y)	TRNQ_F5_AOIY TRNQ_F5_AOIN TRNQ_F5_AOITRAN
CODE=transaction code	NAME(*name*)	Not applicable.
DCLWA=**YES** | NO	DCLWA(Y | N)	TRNQ_F3_DCLWAY TRNQ_F3_DCLWAN
EDIT=(**UC** | ULC, *editname*)	EDITUC(**Y** | N) EDITRTN(*name*)	TRNQ_F3_EDITUCY TRNQ_F3_EDITUCN TRNQ_EDITRTN
EXPRTIME= (*seconds*)	EXPRTIME(*number*)	TRNQ_EXPRTM
FPATH=(**NO** | YES | *size*)	FP(**N** | E | P) EMHBSZ(*size*)	TRNQFP TRNQFPP TRNQ_F2_FPN TRNQ_EMHBSZ
INQUIRY=(**NO** | YES, **RECOVER** | NORECOV)	INQ(**N** | Y) RECOVER(**Y** | N)	TRNQ_F4_INQY TRNQ_F4_INQN TRNQ_F4_RECOVERY TRNQ_F4_RECOVERN
MAXRGN=**0** | number	MAXRGN(**0** | number)	TRNQ_MAXRGN

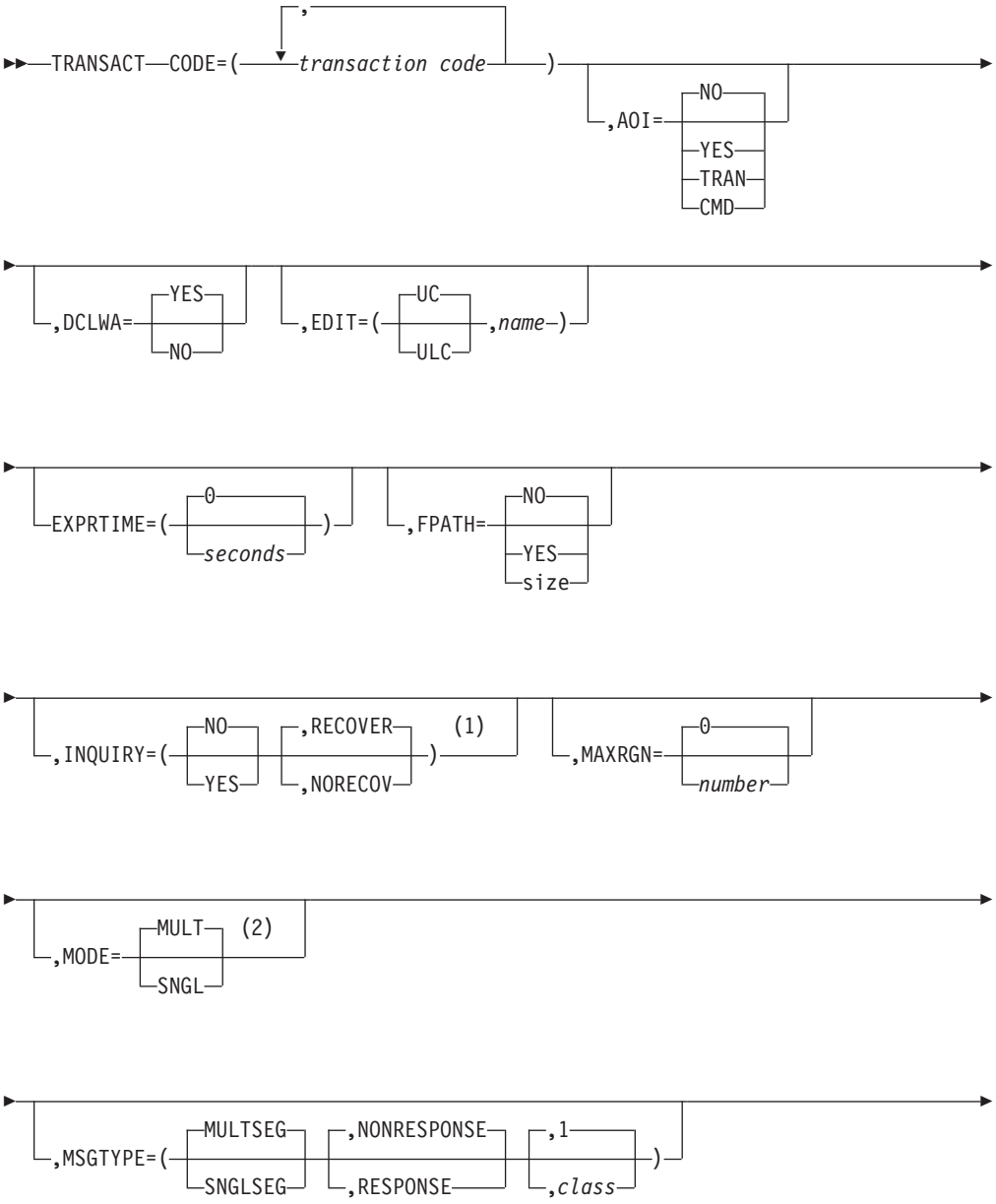
Table 61. TRANSACT macro keywords and the equivalent CREATE and UPDATE command keywords used in dynamic definition (continued)

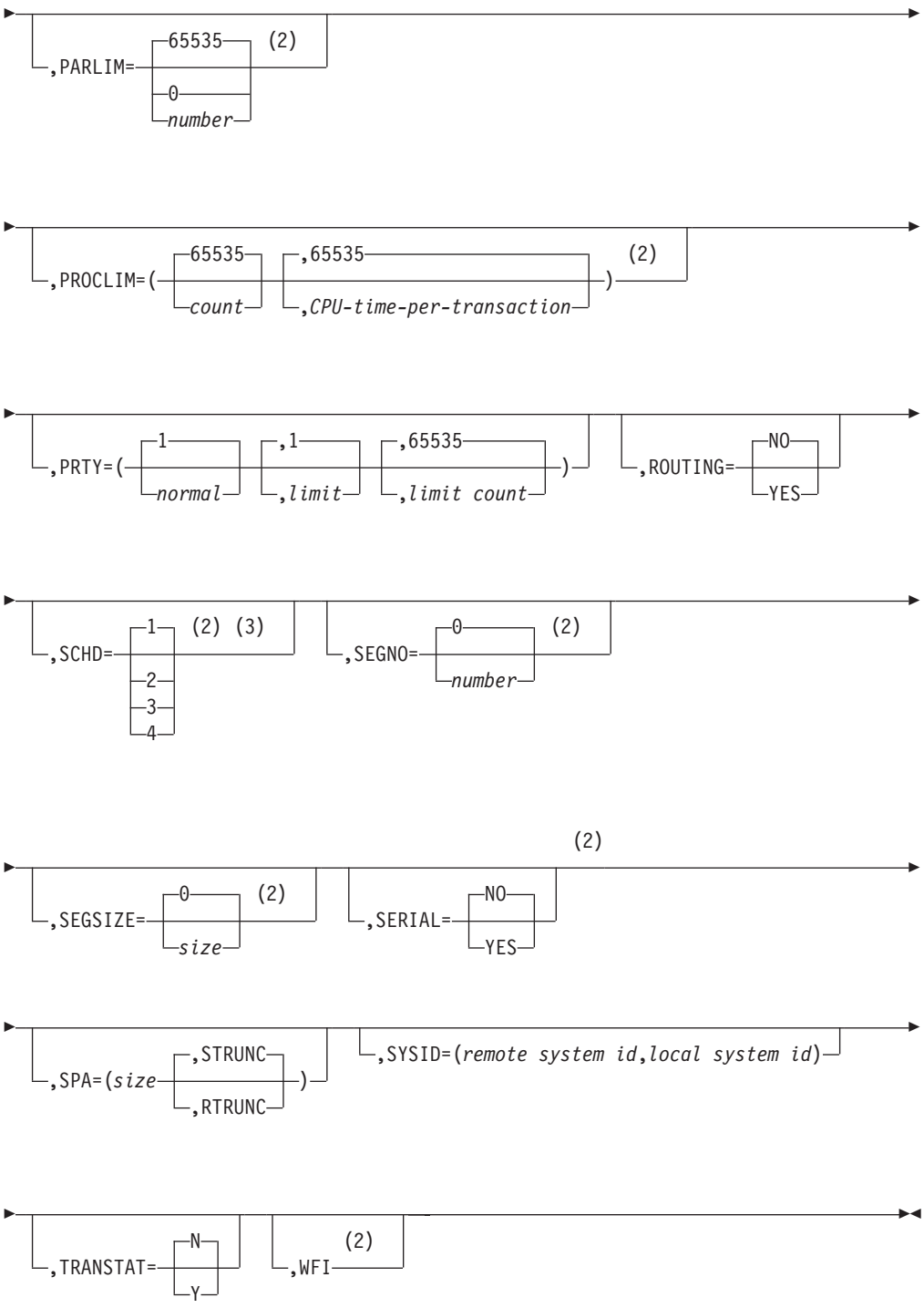
TRANSACT macro keyword	CREATE | UPDATE TRAN keyword equivalent	INSXTRNQ DSECT information
MODE=**MULT** | SNGL	CMTMODE(MULT | **SNGL**)	TRNQ_F3_CMTMODEM TRNQ_F3_CMTMODES
MSGTYPE= (**MULTSEG** | SNGLSEG, **NONRESPONSE** | RESPONSE, **1** | class)	MSGTYPE(**MULTSEG** | SNGLSEG) RESP(**N** | Y) CLASS(**1** | class)	TRNQMSEG TRNQ_F2_SSEG TRNQRESP TRNQ_F2_RESPN TRNQ_CLASS
PARLIM=**0** | number	PARLIM(**0** | number)	TRNQ_PARLIM
PROCLIM= (**65535** | count, **65535** | CPU-time-per-transaction) **Note:** For Fast Path potential transactions, IMS interprets the *CPU-time-per-transaction* value as hundredths of seconds.	PLCT(**65535** | value) PLCTTIME(**6553500** | CPU-time-per-transaction)	TRNQ_PLCT TRNQ_PLCTTIME
PRTY= (**1** | normal, **1** | limit, **65535** | limit count)	NPRI(**1** | value) LPRI(**1** | value) LCT(**65535** | value)	TRNQ_NPRI TRNQ_LPRI TRNQ_LCT
ROUTING=**NO** | YES	DIRROUTE(**N** | Y)	TRNQ_F3_DIRROUTEY TRNQ_F3_DIRROUTEN
SCHD=**1** | 2 | 3 | 4	Not supported.	Not supported.
SEGNO=**0** | number	SEGNO(**0** | number)	TRNQ_SEGNO
SEGSIZE=**0** | size	SEGSIZE(**0** | size)	TRNQ_SEGSZ
SERIAL=**NO** | YES	SERIAL(**N** | Y)	TRNQ_F4_SERIALY TRNQ_F4_SERIALN
SPA= (*size*, **STRUNC** | RTRUNC)	CONV(Y) SPASZ(*size*) SPATRUNC(S | R)	TRNQCONV TRNQ_F2_CONVN TRNQSPAL TRNQSTRU TRNQRTRU
SYSID=(*remote system id, local system id*) (specified either in the TRANSACT macro or in the previous APPLCTN macro)	SIDL(*localsysid*) SIDR(*remotesysid*) or MSNAME(*msname*) REMOTE(Y)	TRNQSIDL TRNQSIDR TRNQ_F4_REMOTEY TRNQ_F4_REMOTEN
TRANSTAT=**N** | Y	TRANSTAT(**N** | Y)	The TRANSTAT keyword is optional. If a value is not specified for the TRANSTAT keyword, the system default is used.
WFI	WFI(**N** | Y)	TRNQ_F5_WFIY TRNQ_F5_WFIN

Supported environments

The TRANSACT macro can be optionally used in IMS DB/DC and DCCTL environments.

Syntax





Notes:

1 NORECOV is not valid when NO is specified.

2 These operands are not appropriate if a transaction destined for processing in another (remote) system is being specified.

3 This parameter is not supported. If you include it in stage 1 system definition, it is ignored.

Positional parameters

WFI is the only positional parameter for the TRANSACT macro.

Keyword Parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

AOI=

For type-1 AOI: For Type 1 Automated Operator Interface (AOI), specifies whether (YES | TRAN | CMD) or not (NO) a particular transaction is allowed to issue the AOI command (CMD) call.

NO When AOI=NO, a particular transaction cannot issue the AOI command (CMD) call. If the EXEC parameter AOI1=N is defined, all transactions are allowed to issue the CMD call, including this transaction. AOI1=N overrides AOI=N.

YES When AOI=YES is specified, the authorization of the commands is like authorization for ICMD calls; the user ID or program name is used for authorization. For some environments, if a Get Unique call has not yet happened, then the program name is used for the authorization. Specifying AOI=YES requests that the user ID of the user who entered the transaction be authorized against the command (in CIMS class).

TRAN The TRAN specification is the same as YES, but it also requests that IMS use the transaction code, instead of the user ID of the user who entered the transaction, to check the authorization of the commands for the CMD calls issued by the transaction.

When a transaction is defined with AOI=TRAN, the first authorization check done for AOI for the transaction results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. In this case, the type-1 AOI transactions need to be defined to RACF (or equivalent product) as a user. The transactions are then specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this occurs, then either the transaction name or the existing user name needs to be changed.

CMD The CMD specification is the same as YES, but also indicates that authorization checking is based on which transactions can issue a particular command. In this case, the commands (or the first three characters of the commands) need to be defined to RACF (or equivalent product) as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified. Using the CMD specification requires you to create fewer user IDs than you need to create for the AOI=TRAN specification. However, using the CMD specification requires you to create or modify a larger number of resource profiles.

Note: If you specify AOI=YES, AOI=TRAN, or AOI=CMD for a transaction and the message being returned on the GU call was received by IMS before the start of IMS execution, a CF status code results. If your application has not specified CF as an acceptable status code, an abend results.

Related reading: See the description of CF and the related status codes in the DL/I status codes section in the *IMS Version 11 Messages and Codes, Volume 4: IMS Component Codes*.

The following table describes how the AOI= specification on the TRANSACT macro impacts the specifications on the EXEC parameter AOI1=.

Table 62. Relationship between AOI= and AOI1= specifications

EXEC parameter AOI1= specification	Effect on TRANSACT macro AOI= specifications
R | C | A	All specifications are retained.
N	NO specification is set to YES. YES specification is retained. TRAN and CMD specifications are reset.

For type-2 AOI: The AOI parameter is also available for type-2 AOI security. However, AOI=NO applies only to type-1 AOI. If AOI=NO is specified or set by default, RACF is used for security checking.

If the TRAN specification is used for type-2 AOI, IMS uses the transaction code, instead of the user ID of the user who entered the transaction, to check the authorization of the commands for the ICMD calls issued by the transaction.

If the CMD specification is used for type-2 AOI, authorization checking is based on which transactions can issue a particular command.

For IMS environments that do not support the use of a transaction, such as a non-message-driven BMP, the AOI=TRAN | CMD specification is not available, and the security checking that is performed is like the checking performed when only YES is specified.

CODE=

Specifies one or more one- to eight-character alphanumeric transaction codes or remote transaction codes. If more than one code is specified, each transaction code name is assumed to have the same characteristics as all other keyword or positional parameter specifications or defaults. Enclose multiple names in parentheses and separate them with commas. Each character of transaction codes and logical terminal names must be an alphanumeric character (A through Z, #, $, @, or 0 through 9). Transaction codes, logical terminals on the NAME macro, and linknames on MSNAME macros must comprise a set of values, each of which is unique in the system. That is, transaction codes, logical terminal names, and MSC linknames, collectively, cannot contain duplicates. The CODE operand is required.

Example:

```
CODE=(TRAN1,TRAN2,TRAN3)
```

DCLWA=

Specifies whether (YES) or not (NO) IMS should perform log write-ahead for recoverable, nonresponse mode input messages and transaction output messages. If not specified in the TRANSACT macro, the default is the DCLWA parameter in the IMSCTRL macro. The DCLWA parameter in the TRANSACT macro overrides the IMSCTRL macro parameter for this transaction.

Specify or accept a default of YES to ensure that both of the following occur:

- A nonresponse input transaction is made recoverable across IMS failures, before IMS acknowledging receipt of the input.
- Database changes are made recoverable before IMS sending associated output reply messages.

YES ensures that information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.

Specify or accept a default of YES for all VTAM terminal types.

Related reading: For information about the performance implications of selecting log write-ahead for transactions, see *IMS Version 11 System Administration*.

Specify NO if input message integrity and the consistency of output messages with associated database updates is not required. DCLWA does not apply to response mode or Fast Path input processing, and is ignored during IMS execution.

EDIT=

Specifies whether (UC) or not (ULC) the input data is to be translated to uppercase. The first parameter of this operand defines whether the transaction is uppercase/lowercase (ULC) as entered from the terminal, or if it is to be translated to uppercase (UC) before being presented to the processing program. The default is UC.

Specifying UC for VTAM terminals prevents the transmission of embedded device control characters.

You can also use EDIT to specify the 1- to 8-character name of your own transaction input edit routine that edits messages before the program receiving the message. This name must begin with an alphabetic character. The specified edit routine (load module) must reside on the USERLIB data set before IMS system definition stage 2 execution. This routine cannot be the same as the one that is used on a LINEGRP or TYPE EDIT= parameter.

If FPATH=YES is specified, the EDIT= keyword parameter specifies whether (UC) or not (ULC) the transaction is to be translated to uppercase before being presented to the edit/routing exit routine. Specification of a user edit routine is valid on a Fast Path potential transaction; this specification is used when the transaction is routed to IMS. It is invalid on a Fast Path exclusive transaction.

For input from LU 6.2 devices, the LU 6.2 Edit exit routine (DFSLUEE0) is called instead of the transaction input edit routine specified in EDIT.

For input from OTMA devices, the user edit exit routine (DFSYIOE0) is called instead of the transaction input edit routine specified in EDIT.

Related reading: For more information about DFSLUEE0, see *IMS Version 11 Exit Routines*.

EXPRTIME=

The length of elapsed time, in seconds, that IMS can use to cancel the input transaction. For transactions that exceed the specified elapsed time, IMS identifies the input transaction as expired and discards the transaction. The value can range from 0 to 65535. The default is 0, meaning that no expiration is set for this transaction. If an invalid value is specified, message G316 is issued.

Restriction: Transaction expiration checking is not done at the time of the GU call for Fast Path transactions, IMS conversational transactions, and program-to-program switch transactions.

FPATH=

Specifies whether (YES, size) or not (NO) the transaction code is a potential candidate for Fast Path processing. FPATH=YES is effective only when specified on a TRANSACT statement that follows an APPLCTN statement that

does not specify FPATH=YES; otherwise, the operand is ignored. FPATH=*size*, which determines the EMH buffer size required to run the transaction, overrides the EMHL execution parameter and implies FPATH=YES. The minimum specification for FPATH=*size* is 12; the maximum is 30720. The default is FPATH=NO.

Fast Path-potential transactions must be processed by a user edit/routing exit to determine whether the transaction is to be processed by IMS Fast Path. If it is to be processed by IMS Fast Path, the edit/routing exit routine associates the transaction with a routing code. This routing code identifies which Fast Path application program is to process the transaction.

If FPATH=YES is specified during a MODBLKS system definition, Fast Path must have been previously defined for the online system.

INQ= or
INQUIRY=

Specifies whether (YES) or not (NO) this is an inquiry transaction. The default is NO. You can specify this keyword either as INQ= or INQUIRY= If INQ= (or INQUIRY=) YES is specified, you can also specify whether (RECOVER) or not (NORECOV) this transaction should be recovered during an IMS emergency or normal restart. The specification of INQ=(NO,NORECOV) is invalid. The default is RECOVER.

For IMS Fast Path transactions, RECOVER must be specified.

INQ=YES should be specified only for those transactions that, when entered, do not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ=YES. An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ=YES. The INQ operands are not position dependent.

If the SPA parameter is specified (indicating that the transaction is conversational), INQ=(YES,NORECOV) cannot be specified.

If an attempt is made to enter a transaction that is not specified as INQ=YES from one of these terminals, the transaction is rejected. A message indicating that the update transaction cannot be processed is sent to the terminal that entered the transaction.

MAXRGN=

Limits the number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions.

If you specify zero, or if zero is defaulted to, no limit is imposed. The maximum number you can specify is 255.

If you specify SERIAL=YES in the TRANSACT macro or SCHDTYP=SERIAL in the APPLCTN macro, omit the MAXRGN parameter or set it equal to 0. Also, omit the PARLIM parameter in the TRANSACT macro and accept the default (65 535).

For non-zero values, you cannot specify MAXRGN= unless you also specify PARLIM=.

The value associated with MAXRGN= can be changed using a /CHANGE TRAN command; therefore, it is not affected by an online change sequence of operator commands for an existing transaction, whether it is altered during a MODBLKS system definition or not.

MODE=

Specifies that database buffers are to be written to direct access (flushed) upon each request for a new message (SNGL) by the processing program, or upon program termination (MULT). The default is MULT. Conversational and WFI transactions must be defined as SNGL. SNGL is forced for WFI applications.

This operand affects emergency restart. When MODE=SNGL, emergency restart only reprocesses the last completed message, regardless of whether one or more messages are scheduled and processed by a single load of an application program. Otherwise, emergency restart reprocesses those messages that were scheduled and processed by the single load of an application program, since the time of the previous checkpoint. The number of messages processed depends upon when the last checkpoint is issued.

The MODE= keyword parameters are checked on all IMS Fast Path-potential transactions for MODE=SNGL. If not specified, a warning diagnostic is issued.

If the transaction results in the application calling an external subsystem, such as DB2, the Commit Verify exit provided by the external subsystem can determine whether MODE=MULT is supported. See documentation under the Commit Verify exit routine in *IMS Version 11 Exit Routines*.

MSGTYPE=

Specifies the type of transaction code (single or multiple segment), and whether the communication line from which the transaction is entered is to be held until a response is received. The MSGTYPE operands are not position dependent.

The transaction code can be single segment (SNGLSEG), or multiple segment (MULTSEG). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing. The defaults are (MULTSEG,NONRESPONSE,1).

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

The MSGTYPE= keyword parameters are checked on all IMS Fast Path-potential transactions for MSGTYPE=(SNGLSEG,RESPONSE). If not specified, a warning diagnostic is issued.

The first parameter of the MSGTYPE keyword specifies one of the following choices for number of segments:

MULTSEG

Specifies that the incoming message can be more than one segment in length. It is not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS.

SNGLSEG

Specifies that the incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.

The second parameter of the MSGTYPE keyword specifies one of the following response choices:

NONRESPONSE
Specifies that, for terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered.

RESPONSE
Specifies that, for terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated by individual terminal definition.

The third parameter of the MSGTYPE= keyword specifies the class to which this transaction code is to be assigned. This parameter must be a decimal number from 1 to 999. The default is 1.

Because the values associated with the class keyword can be changed using an /ASSIGN command, they are not affected by an online change sequence of /MODIFY operator commands for an existing transaction, regardless of whether these values are altered during a MODBLKS system definition. The value specified must not exceed the MAXCLAS= value specified or accepted by default on the IMSCTRL macro statement. Regardless of specification made or default taken, remote transaction codes are assigned a class of zero. If the transaction code class is specified in the APPLCTN macro, this parameter need not be specified. If the transaction code class is specified in both the APPLCTN and TRANSACT macros, the APPLCTN macro specification is ignored for this transaction. Define CPI transactions with a different message class from that used for non-CPI transactions. IMS handles all CPI transactions as priority zero within the transaction class.

MSGTYPE=RESPONSE is ignored during online processing for all terminals that do not operate in response mode.

PARLIM=
Specifies the threshold value to be used when SCHDTYP=PARALLEL is specified in the preceding APPLCTN macro instruction. If the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction, an additional region is scheduled when a subsequent schedule event for this transaction's class occurs. In a shared-queues environment, the successful consecutive GU count is used instead of the enqueue count. Events can include the following:

- Message enqueue from terminal.
- Message insert from application. One region is allowed per sync point, EXPRESS=NO PCB. For example, when an application inserts two messages for the same transaction code, only one region is scheduled at sync point.
- /ASSIGN command.
- Application term thread.

If you do not specify a value for PARLIM, IMS:

- Assigns a value of 65 535 to PARLIM. You cannot specify 65 535 as a value for PARLIM.
- Allows the transaction to be scheduled in only one region at a time (that is, IMS disables transaction load balancing).

Valid values for PARLIM can be either any number from 0 to 32 767, or 65 535.

PARLIM=0 indicates that any input message can cause a new region to be scheduled because the scheduling condition is always met (the number of messages is greater than zero).

The value specified for PARLIM applies to message processing programs (MPPs) only. PARLIM is not supported for batch message processing programs (BMPs).

PARLIM is not applicable to FPE transactions and should be allowed to default to 65 535. If a PARLIM value is specified for FPE transactions, it is ignored by scheduling. Specifying a PARLIM value other than the default value results in a branch and link (BAL) status shown for an FPE transaction on commands such as /DISPLAY TRAN or QUERY TRAN. The PARLIM and BAL status can be ignored for FPE transactions.

The value for PARLIM is not affected by an online change sequence of operator commands for an existing transaction, regardless of whether it is altered during a MODBLKS system definition because the value associated with the PARLIM= keyword can be changed using an /ASSIGN command. For more information about the /ASSIGN command, see *IMS Version 11 Commands, Volume 1: IMS Commands A-M*.

Note: In a shared-queues environment, the PARLIM value behaves differently than it does in a non-shared-queues environment. In a non-shared-queues environment, the queue depth (the number of messages that are currently enqueued) for the transaction is used as the value that is compared with the PARLIM value to determine when to schedule another region. IMS responds to a growing queue of input transactions by scheduling more regions as the queue grows.

In a shared-queues environment, each individual IMS does not know the depth of the queue, because the queue is in the shared-queues coupling facility structure that is managed by Common Queue Server (CQS). The transaction queue might be added to by many different IMS systems. IMS is notified only when the first message is put in a queue (that is, when the queue becomes *not empty*). IMS is not notified for every subsequent message that is placed on the queue after that first message. In a shared-queues environment, the PARLIM comparison is done against a counter that each IMS keeps of the number of successful consecutive GU calls for the transaction by that IMS, rather than queue depth. IMS schedules more regions when it consistently gets messages from CQS when it asks for them. Thus, in a shared-queues environment, IMS infers the depth of the queue of messages based on processing activity, but it does not know the actual depth of the queue.

A PARLIM value of 0 in a shared-queues environment is the most responsive setting. PARLIM(0) ensures that message regions are scheduled until all messages are processed from the transaction queue, or until the maximum region value (MAXRGN) limit is reached. PARLIM(0) might, however, result in many unnecessary schedules (or *false schedules*). A false schedule occurs when a message region is scheduled and finds no more messages on the queue. This is a consequence of IMS not knowing how many messages are queued on the transaction queue, and having to try to read the queue to see if there are more messages.

Setting the PARLIM to a value greater than 0 can reduce the number of false schedules, because IMS then schedules a new message region only after a number of messages have been obtained consecutively without the queue

becoming empty. Setting the PARLIM to a value of 2 or greater is useful for reducing false schedules for transactions that are low-volume and that run relatively quickly (such that the queue depth is typically 1), because it avoids scheduling a second region until the first region has obtained at least two messages in a row. However, be aware that while a PARLIM value greater than 0 can reduce unnecessary schedules, it is also less responsive. If a transaction is long running, or if its processing is delayed (for example, because of locking contention), the consecutive GU count does not change while the transaction is executing, and no additional message regions are scheduled. This can result in delayed processing of other messages for this same transaction until a currently-scheduled message completes. This delay can occur even if message regions are available to process the transaction.

Recommendations:

- If you specify PARLIM=0, specify a MAXRGN value to limit the number of regions that can be scheduled to process a particular transaction. If you specify PARLIM=0 and you do not specify a value for MAXRGN, one transaction might monopolize all the available regions.
- If you specify SERIAL=YES in the TRANSACT macro or SCHDTYP=SERIAL in the APPLCTN macro, omit the PARLIM parameter and accept the default (65 535). Also, omit the MAXRGN parameter in the TRANSACT macro or set it equal to 0.

PROCLIM=

Specifies the number of messages (count) of this transaction code a program can process in a single scheduling, and the amount of time (for non-Fast-Path transactions, in seconds; for Fast Path transactions, in hundredths of seconds) allowable to process a single transaction (or message). Batch Message Programs (BMPs) are not affected by these settings.

The first parameter, *count*, can be changed using an /ASSIGN command. Therefore, its value is not affected by an online change sequence of operator commands for an existing transaction, regardless of whether it was altered during a MODBLKS system definition.

The *count* specifies the maximum number of messages sent to the application program by the IMS control program for processing without reloading the application program. The *count* value can range from 0 through 65 535. If 0 is coded, the maximum number of messages sent to the application is one and the application program is reloaded before receiving a subsequent message. Code the *count* value at 65 535 if no limit is to be placed upon the number of messages processed at a single program load. Values 1 through 65 535 are eligible for quick reschedule processing.

The *CPU-time-per-transaction* parameter specifies the processing limit count time. This is the amount of time (for non-Fast-Path transactions, in seconds; for Fast Path transactions, in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region. The value is a number that can range from 1 to 65 535. Specifying the maximum value means that no time limit is placed on the application program.

Note: For IMS Version 10 and later, changes have been made in the way IMS evaluates the value that you specify for *CPU-time-per-transaction* when a Fast Path potential transaction runs in an MPP region. The value that you specify is now evaluated as hundredths of seconds. See the sections describing changes to the PROCLIM parameter in *IMS Version 11 Release Planning*.

For Fast Path transactions (both potential and exclusive), *CPU-time-per-transaction* represents real time that elapses during transaction processing (not accumulated task time). Real time is used because the input terminal is in response mode and cannot enter another transaction until the response is sent. The *count* subparameter is ignored.

The defaults for PROCLIM are 65 535 and 65 535.

The *count* value assigned is used to determine how many messages an application program is allowed to process in a single scheduling cycle. When the application program requests, and receives, the number of messages indicated in the *count* value, any subsequent requests result in one of two things.

1. IMS indicates "no more messages exist" if any of the following conditions are true:
   - The region is not an MPP.
   - The currently scheduled mode is not MODE=SNGL.
   - Equal or higher— priority transactions are enqueued for the region.

   IMS might, in fact, have other messages enqueued for the application program. It is the responsibility of the application program to terminate when it receives an indicator that no more messages are available. Termination of the application program makes the region it occupied available for rescheduling. This feature makes it possible for IMS to allow scheduling of higher-priority transactions that entered the system while the previous transactions were in process. In addition, if any equal-priority transactions are enqueued, they become eligible for scheduling on a first-in, first-out (FIFO) basis.
2. The region goes through quick reschedule and returns the next message to the application if all the following conditions are true:
   - The region is an MPP.
   - The transaction is MODE=SNGL.
   - No higher priority transactions are enqueued.
   - Messages are still enqueued for the application.

If equal priority transactions are enqueued, IMS allows the transaction to quick reschedule if the PROCLIM value has not been reached. If the PROCLIM value has been reached, IMS disallows the quick reschedule and processes the other transactions.

Quick rescheduling is also affected by the IMS scheduling algorithm, which takes into account the following factors and more:

- MAXRGN value
- PARLIM value
- Current number of regions in which the transaction is scheduled

The *CPU-time-per-transaction* value controls application program looping. You are not required to optimize the *CPU-time-per-transaction* value for program-transaction execution time. However, the *CPU-time-per-transaction* time value assigned should not be less than the expected per-transaction execution time. If the scheduled application program exceeds the product of *CPU-time-per-transaction* and *count*, the application program abends (if the product of *CPU-time-per-transaction* and *count* exceeds 24 hours, 24 hours is used). If an IMS STIMER value of 2 is specified on the DFSMPR macro, the region does not abend until completion of the DL/I call.

The application must not use z/OS timer services, such as STIMER TASK, that override the IMS STIMER. IMS uses the IMS STIMER to time the execution of transactions. If a z/OS TIMER is issued, it disables the IMS STIMER.

The IMS STIMER task tracks processor time statistics, including time durations for:

- Application program execution time
- DL/I processing time
- CPU time for ESAF users such as DB2 for z/OS that continue to process under the TCB of the dependent region. If DB2 for z/OS switches to another TCB, which is the case with parallelism, logging, and prefetch, this time is not included.

PRTY=

Specifies the values that determine the scheduling priority of this transaction. This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system.

normal

The priority assigned to this transaction when the number of input transactions enqueued and waiting to be processed is less than the *limit count* value. The valid specification range is from 0 through 14. The default is 1.

limit

The priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the *limit count* value. The valid specification range is from 0 through 14. The default is 1.

limit count

The number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the *normal* or *limit* priority value is assigned to this transaction. The *limit count* value can range from 1 through 65 535. The default is 65 535.

When the limit priority is used, and the priority is raised to the specified limit priority value, the priority is not reduced to the normal priority until all messages enqueued for this transaction code are processed.

If you do not want the limit priority for this transaction, code equal values for the normal and limit priorities, and a limit count of 65 535.

When a transaction is processed exclusively by a batch message program (BMP), code the normal and limit priorities as 0. The limit count value is ignored for a transaction processed by a BMP.

The APPLCTN macro statement forces the scheduling priority of all transaction codes associated with it to 0 if the program type is batch (PGMTYPE=BATCH on APPLCTN macro statement). However, a batch message processing region (BMP) can process transactions with scheduling priorities other than 0.

For remote transactions, the PRTY parameter determines the priority used to send the transaction to the processing system, which is termed the *MSC link message priority*. The three MSC link message priority groups are:

- Low
- Medium
- High

The low priority group consists of primary requests in the input terminal system. This group is assigned remote transaction priorities from 0 to 6. The medium group consists of secondary requests, responses, primary requests in an intermediate system, and primary requests in the input terminal system. This group is assigned a remote transaction priority of 7. The high group consists of primary requests in the input terminal system. Messages in this group are assigned remote transaction priorities from 8 to 14. Within each group, messages have a priority based on the current priority value of the transaction or remote transaction in the input terminal system for primary requests, and on the latest processing system for secondary requests and responses.

In an MSC configuration, the transaction priority determines the priority used to send messages inserted by this transaction across an MSC link. If the transaction inserts multiple messages to the same destination (for example, pages to a printer) and these messages must be sent in the order inserted, the normal and limit priority values should be the same. If the normal and limit priority values are not identical, messages inserted at a higher priority than previously inserted messages could arrive at their destination first. (This restriction does not apply to multiple segments of the same message.)

A transaction must have the same characteristics in all systems where it is defined. These characteristics include:

- Non-conversational/conversational
- SPA size if conversational
- Single-/multi-segment messages
- Non-inquiry/inquiry
- Recoverable/nonrecoverable

ROUTING=

If MSC directed routing is used in a multiple IMS system configuration, specifies whether (YES) or not (NO) the application program processing a transaction is informed of the system which originated the transaction.

If ROUTING=YES, an MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB. If ROUTING=NO, the name of the originating LTERM is placed in the I/O PCB. The default is NO.

SCHD=

This parameter is not supported, but it is allowed for compatibility. If you include this parameter on the TRANSACT macro included in stage 1 system definition, it is ignored. IMS processes all transactions as if SCHD=3 was specified.

Specifies the scheduling option used for other transactions when this transaction cannot be scheduled for internal reasons (database intent or no more space in PSB pool or DMB pool to bring in needed blocks). Values include:

1 Schedule only transactions of equal or higher priority in the selected class. This is the default.

Advantages

- Lower priority transactions are not scheduled in the event of intent conflicts, reserving resources for equal or higher priority transactions.

- Excessive intent conflicts within a class are not encountered because only five consecutive intent conflicts are allowed within a class before IMS starts scheduling the next eligible class.

Disadvantages

- After an intent failure is encountered, processing of lower priority transactions within the class are delayed until a transaction of an equal or higher priority is successfully scheduled.
- Transactions of equal or higher priority within the current class might also be delayed because when IMS encounters five successive intent conflicts for different transactions, IMS starts scheduling the next class, thus bypassing schedule attempts for the remaining transactions within the current class.

2 Schedule higher-priority transactions in the selected class.

Advantages

- Equal and lower priority transactions are not scheduled in the event of intent conflicts, reserving resources for higher priority transactions.
- Excessive intent conflicts within a class are not encountered because only five consecutive intent conflicts are allowed within a class before IMS starts scheduling the next eligible class.

Disadvantages

- After an intent failure is encountered, IMS does not process transactions of equal or lower priority within the class until a higher priority transaction is successfully scheduled.
- Transactions of higher priority within the current class might also be delayed because when IMS encounters five successive intent conflicts for different transactions, IMS starts scheduling the next class, thus bypassing schedule attempts for the remaining transactions within the current class.

3 Schedule any transaction in the selected class.

Advantage

All transactions within a class remain eligible for scheduling should an intent conflict occur, regardless of their priority, thus avoiding temporary delays in processing lower priority transactions within a class.

Disadvantage

Higher priority transactions are not given priority in cases of intent conflicts.

Note: There is no intent conflict count for this option, so IMS starts scheduling the next eligible class after attempting to schedule all the transactions in the current class. If intent conflict delays are a concern with regards to moving to other eligible scheduling classes, then consider having message regions (MPRs and JMPs) with those other classes specified first in the CL1- CL4 startup parameters.

4 Skip to the next class and attempt to schedule the highest-priority transaction in that class.

Advantage
Scheduling moves immediately to the next eligible class when any intent conflict occurs, thus allowing the transactions in the next eligible class to be scheduled.

Disadvantage
All remaining transactions within the class that receives the intent conflict are excluded from current schedule attempts until scheduling is attempted in the next eligible class, thus resulting in scheduling delays for the other transactions in the class where the intent failure occurred.

A loop counter in MPP scheduling breaks potential scheduling loops between SCHD=1 or SCHD=2 and SCHD=3 transactions. If scheduling fails due to intent conflicts for an SCHD=1 or SCHD=2 transaction, the counter increments and the next logical transaction is selected based on the SCHD=parameter of the failed transaction.

Each time the scheduler fails to schedule a transaction for intent, the counter is checked. If the count is greater than five, the next class of transaction is selected and the counter reset. This processing method can cause a delay in the current class.

For example, a long-running BMP has update intent on a database. Several transactions that have update intent on the same database with SCHD=1 are entered into the IMS system. Transactions in the same class that do not reference the database are also added to the system. However, the first group of transactions fails for intent, and none of the transactions is scheduled until the BMP terminates.

After each failed transaction, the loop counter increments. After the counter exceeds 5, the next class is scheduled, bypassing the transactions that do not reference the database and causing delays in their processing.

To avoid these delays, place the second group of transactions into a separate class, or run the BMP job at a different time.

SEGNO=
Specifies the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program. It must be specified as a decimal number from 0 through 65 535. The default is 0. If the default specification of 0 is used, the number of segments is not checked by the online system at execution time.

Because the value associated with the SEGNO= keyword can be changed using an /ASSIGN command, it is not affected by an online change sequence of operator commands for an existing transaction, regardless of whether it is altered during a MODBLKS system definition.

SEGSIZE=
Specifies the maximum number of bytes allowed in any one output segment. It must be specified as a decimal number from 0 through 65 535. The default is 0. If the default specification of 0 is used, the segment size is not checked by the online system at execution time.

Because the value associated with the SEGSIZE= keyword can be changed using an /ASSIGN command, it is not affected by an online change sequence

of operator commands for an existing transaction, regardless of whether it is altered during a MODBLKS system definition.

The maximum output message segment to a LU 6.2 device is 32,767. If a transaction is expected to send output to a LU 6.2 device, the SEGSIZE parameter should be no greater than 32,767. However, this is not enforced during processing of the TRANSACT macro, because IMS cannot determine the device type for the message destination until output time.

SERIAL=

Forces serial processing of messages for a given transaction. When SERIAL=YES, U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP.

The USTOP of the transaction is removed when the transaction or the class is started with a /START command.

The default for this keyword is NO, which means message processing is done as before with the messages placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP.

If you specify SERIAL=YES, omit the MAXRGN parameter in the TRANSACT macro or set it equal to 0. Also, omit the PARLIM parameter in the TRANSACT macro and accept the default (65 535).

SPA=

Defines, by inclusion, that this transaction is a conversational transaction.

size

Specifies the size of the conversational scratchpad area (SPA). The size specified must be between 16 bytes and 32767 bytes inclusive.

STRUNC|RTRUNC

You can turn on the truncated data option (STRUNC) or off (RTRUNC).

If you specify SPA=STRUNC, IMS preserves all the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used.

If you specify SPA=RTRUNC, the truncated data is not preserved.

STRUNC is the default.

When a conversation initially starts, and when a program switches, the STRUNC|RTRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and if specified (either STRUNC or RTRUNC), that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

Restriction: The CORE, DASD, and FIXED operands can no longer be used. If you specify them, assembly errors occur.

Related reading: For information about structuring your conversational program, see "The Contents of SPA" in *IMS Version 11 Application Programming*.

SYSID=

In the multiple-IMS system configuration, specifies the system identification (SYSID) of the remote system (the system on which the application executes) and the SYSID of the local system (the originating system to which the responses are returned). The values specified must be from 1 through 2036. The remote SYSID specified must also be defined in an MSNAME macro statement, but the local SYSID can be defined in any or all the MSNAME, TRANSACT, and APPLCTN macro statements.

If the SYSID parameter is specified in the APPLCTN macro statement, you need not specify the SYSID in the TRANSACT macro statement. If the SYSID is specified in both the APPLCTN and the TRANSACT macro statements, the TRANSACT specification is ignored.

The SYSID parameter is independent of the link type (CTC, MTM, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement. Because the values associated with the SYSID= keyword can be changed using an /MSASSIGN command, they are not affected by an online change sequence of operator commands, regardless of whether the values are altered during a MODBLKS system definition. The values only take effect during an IMS cold start.

A Fast Path-exclusive transaction cannot have SYSID= specified. To assign a remote transaction as local, the associated APPLCTN macro must be defined as local. This means that no SYSID= parameter can be specified for the associated APPLCTN macro.

Restriction: A MODBLKS online change is rejected if you attempt to add the SYSID parameter when a change to any other parameter specified on the TRANSACT macro is also requested.

TRANSTAT=

Specifies whether transaction level statistics are logged. If you specify Y, transaction-level statistics are written to the log in an X'56FA' log record.

N Transaction-level statistics are not logged.

Y Transaction-level statistics are logged.

The TRANSTAT keyword is optional. If a value is not specified for the TRANSTAT keyword, and the default descriptor is the IMS-defined descriptor DFSDSTR1, the value in the DFSDFxxx PROCLIB member TRANSTAT parameter that was set at IMS cold start is used. Changing the DFSDFxxx TRANSTAT value across a restart does not affect the default descriptor TRANSTAT value. Use the TRANSTAT keyword on the CREATE TRAN or CREATE TRANDESC command to override the system default when creating a transaction or transaction descriptor.

WFI=

The positional parameter WFI specifies that this is a wait-for-input transaction. A message processing or batch processing application program that processes WFI transactions is scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if any of the following are true:

- The PROCLIM *count* is reached.

- A command is entered to change the status of the scheduled transaction, database, program, or class.
- The /DBR, /DBD, or /STA commands that relate to the databases used by the transaction are entered.
- IMS is terminated with a checkpoint shutdown.

MODE=SNGL is forced when WFI is specified.

Related concepts:

Conversational structure (Application Programming)

Related reference:

Commit Verify exit routine (Exit Routines)

TYPE macro

The TYPE macro statement defines the beginning of a set of macro statements for communication terminals and logical terminals, including the macros TERMINAL and NAME.

TYPE defines terminals attached to IMS through VTAM. It is equivalent to the LINEGRP/LINE macro set used to define terminals attached to IMS by means other than VTAM.

In addition to the keyword parameters described in this topic, any keyword parameter of the TERMINAL macro statement except EDIT, LTERM, NAME, COMPT, and COMPT1 through COMPT4 can be specified as a keyword operand of the TYPE macro statement. Any specified keyword value provides default values for subsequent TERMINAL macro statements within this terminal description set.

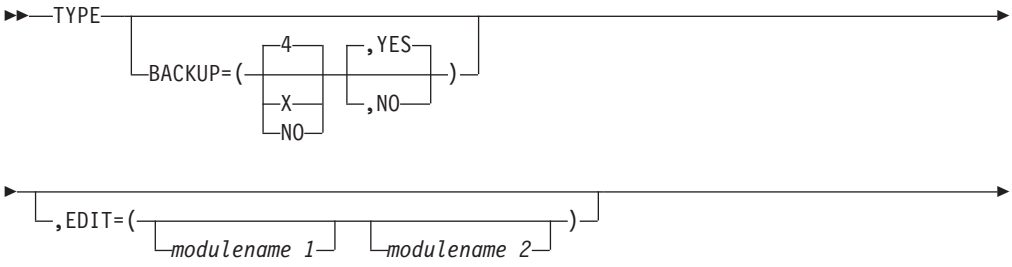
Dynamic definition

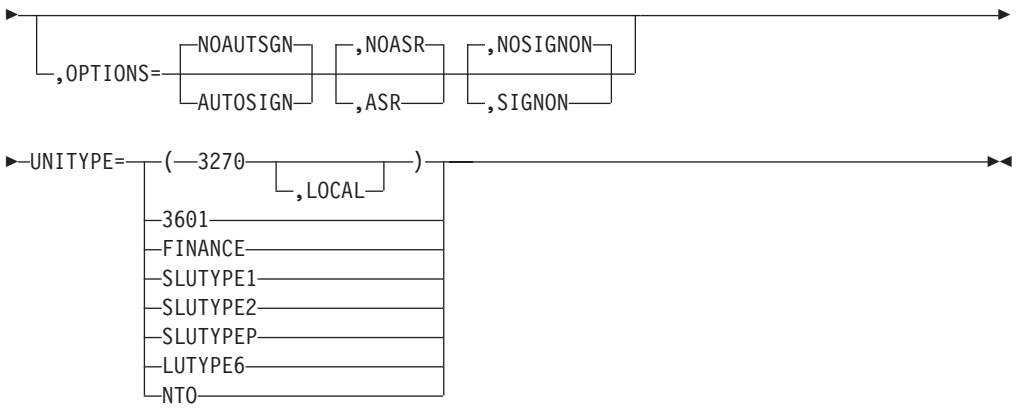
You cannot dynamically define the beginning of a set of communication terminals and logical terminal description macro statements.

Supported environments

The TYPE macro can be used in the DB/DC and DCCTL environments.

Syntax





Positional parameters

The TYPE macro does not include positional parameters.

Keyword parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

BACKUP=

Specifies (for XRF only) the control of session switching (VTAM) after takeover. Use only when HSB=YES is specified on the IMSCTRL macro.

X is a numeric integer from 1 to 7, inclusive, that specifies priority for reestablishing the session. The default is 4 when either keyword or parameter is omitted. NO suppresses session recovery of the terminal at takeover.

Although IMS sets priorities for VTAM requests, the active requests can be completed in any order because of internal VTAM conflicts and pacing.

The second parameter determines whether the backup system is to attempt to establish a backup session to the terminal when the active request establishes a session. When the entire keyword or this parameter is omitted, the default is YES. Specifying NO suppresses the establishment of a backup session.

EDIT=

modulename 1 specifies the 1- to 8-byte name of a user-supplied physical terminal output edit routine for the terminals in this communication description set.

modulename 2 specifies the 1- to 8-byte name of a user-supplied physical terminal input edit routine for the terminals in this communication description set.

This routine cannot be the same as the one that is used on a TRANSACT EDIT= parameter.

OPTIONS=

- Specifies (for the Session Outage Notification facility only) automatic session restart (ASR) processing on all nodes of the same type. The default is NOASR.

  The ASR and NOASR definitions on the TERMINAL macro override those definitions on the TYPE macro. However, the ASR option is not supported for nodes that can have XRF backup sessions.

Use the /DIS command to display the current ASR option, and the /CHANGE command to change it.

- Specifies signon verification security processing (SIGNON). The default is NOSIGNON.

  The SIGNON and NOSIGNON definitions on the TERMINAL macro override those definitions on the TYPE macro.

  Instead of specifying OPTIONS=SIGNON in more than one system definition macro, you can require all static terminals to sign on by specifying a single parameter, SIGNON=ALL, in the DFSDCxxx PROCLIB member.

  Related Reading: For more information about the SIGNON keyword, see "DFSDCxxx member of the IMS PROCLIB data set" on page 656.

- AUTOSIGN indicates that IMS automatically signs on to static terminals with the first LTERM name as the user ID, bypassing a password check. The user ID needs to be defined in RACF.

  For ISC static terminals, the SUBPOOL name are used as the user ID instead of the LTERM name.

  NOAUTSGN indicates that IMS does not automatically sign on to static terminals with the first LTERM name as the user ID. This is the default.

  The AUTOSIGN and NOAUTSGN definitions on the TERMINAL macro override those definitions on the TYPE macro.

  If you specify AUTOSIGN on the TYPE macro, every subsequent TERMINAL macro after that TYPE macro also specifies AUTOSIGN. You can specify exceptions to the AUTOSIGN specification by specifying NOAUTSGN for particular TERMINAL macros.

UNITYPE=

Specifies the terminal device type contained in this communication description set.

A terminal defined as either a 3770 or NTO is to be treated as a secondary logical unit type 1 terminal by IMS.

Terminals belonging to a Finance Communication System such as the 3600 Finance Communication System or the 4700 Finance Communication System can be specified as UNITYPE=FINANCE on the TYPE macro statement. Terminals belonging to the 3600 Finance Communication System can also be specified as UNITYPE=3601. The code generated to support the terminal is identical regardless of whether the terminal is specified as UNITYPE=3601 or UNITYPE=FINANCE.

VTAMPOOL macro

The VTAMPOOL macro, which is required for parallel session support, begins the definition of the LU 6.1 LTERM subpools.

You must define VTAMPOOL if any LU 6.1 node uses dynamic LTERM allocation. LTERMs, defined as LU 6.1 nodes, can be a fixed set or may be dynamically allocated depending on the composition of the VTAM macro set. This macro initiates a hierarchy of macros, consisting of a VTAMPOOL statement and one or more subsequent SUBPOOL macros, each of which can optionally be followed by one or more NAME macros (if at least one SUBPOOL statement has an associated NAME statement). This macro set can occur anywhere in the VTAM macro set if it does not break into the TYPE–TERMINAL–NAME statement hierarchy. This macro has no operands.

Dynamic definition

You cannot dynamically define the LU 6.1 LTERM subpools.

Supported environments

The VTAMPOOL macro can be used in the DB/DC and DCCTL environments.

Syntax

Positional parameters

The VTAMPOOL macro does not include positional parameters.

Keyword Parameters

To find which parameters apply to your IMS configuration, refer to "Selecting the appropriate macros to define your system" on page 5.

VTAMPOOL
All LU 6.1 SUBPOOLs can be defined under a single VTAMPOOL statement or under multiple VTAMPOOL statements. The only purpose of defining LU 6 SUBPOOLs using multiple VTAMPOOL statements is to aid in documenting the use of each LU 6 SUBPOOL.

Chapter 16. Procedures used in IMS environments

These topics include general information for using IMS procedures.

The specification of IMS procedures

When you specify IMS system definition-supplied procedures, you must consider where they are stored, whether they need to load the IMS SVC modules, and which IMS environments support the procedures.

For each of the IMS system definition-supplied procedures:

- The parameters are listed alphabetically in "Parameter descriptions for IMS procedures" on page 448.
- The DD statements are listed alphabetically in "DD statements for IMS procedures" on page 502.

Storing IMS system definition-supplied procedures

Store the procedures that are created by the system definition process in an appropriate z/OS PROCLIB. The JCL supplied in the IMSMSG and IMSWT*nnn* members of the IMS PROCLIB data set must be modified for your installation requirements and stored in IMS.JOBS.

When PROCLIB=YES is the default or if PROCLIB=YES is specified in the IMSGEN system definition macro statement, certain procedures and the IMSMSG and IMSWT*nnn* jobs are dynamically created and placed in the IMS PROCLIB data set. The created jobs and procedures should be examined carefully to determine whether the JCL is generated as you require. These procedures might not apply to all applications, but they can be used as guidelines for user-generated account oriented procedures. Depending on the type of system being defined, your IMS PROCLIB data set members can be a subset of the complete IMS PROCLIB data set that is presented here.

IMS conforms to z/OS rules for data set authorization. If an IMS job step is to run authorized, all libraries to be used in that job step must be authorized. To run an IMS batch region as unauthorized, a nonauthorized library must be concatenated to IMS.SDFSRESL.

Specifying the DFSRESLB DD statement in IMS batch procedures

Certain batch procedures must load the IMS SVC modules, all of which must be in an authorized library. The DFSRESLB DD statement specifies the library that contains these modules. All libraries that are specified by this statement must be authorized through the Authorized Program Facility (APF).

When you include the DFSRESLB DD statement in a batch procedure, all IMS SVC modules are loaded from the specified library. If this library does not contain the required modules, IMS does not search any other library for the modules and then terminates.

© Copyright IBM Corp. 1974, 2015

In the batch procedure for your DFSRESLB DD statement, if you identify a specific library from which the IMS SVC modules are to be loaded, this specification does not become the SDFSRESL library search order for the remaining IMS (non-SVC) modules. You must still specify your SDFSRESL library in a JOBLIB or STEPLIB DD statement.

When you do not include the DFSRESLB DD statement in a batch procedure, IMS searches the standard default libraries for the modules (JOBLIB/STEPLIB, followed by LINKLIB). If the IMS SVC modules are not in an authorized library, IMS terminates.

In addition to the DFSRESLB DD statement, the EXEC parameter SRCH, which is allowed on selected batch procedures, also affects the library search order. When SRCH=1, IMS searches the link pack area and directories before searching any other library. When SRCH=0 and the DFSRESLB DD statement is specified, IMS searches only the specified library and does not search the link pack area.

The following modules must be loaded from an authorized library during the IMS batch initialization:

DFSABND0
DFSAFMX0
DFSAOSF0
DFSAOS70
DFSBC000
DFSCBT10
DFSFSWA0
DFSKPXT0
DFSMODU0
DFSMODX0
DFSMRC00
DFSMRC20
DFSRTM00
DFSRTMI0
DFSSPF00
DFSSTM00
DFSTOPR0
DFSVCI00
DFSVCI10

Environments that support IMS system definition-supplied procedures

Table 63 shows all the IMS system definition-supplied cataloged procedures, control statements, and jobs. Each procedure is listed, followed by the environments in which it applies. Unless noted otherwise, all items in the **Name** column of the table are procedures.

Table 63. IMS system definition-supplied cataloged procedures and the environments in which they apply

Name	DB Batch	TM Batch	DBCTL	DB/DC	DCCTL	Description
DBBBATCH	X	X				A one-step procedure for offline DL/I batch processing regions that use IMS.ACBLIB:“DBBBATCH procedure” on page 520

Table 63. IMS system definition-supplied cataloged procedures and the environments in which they apply (continued)

Name	DB Batch	TM Batch	DBCTL	DB/DC	DCCTL	Description
DBC			X			An online execution procedure to initialize the DBCTL environment: "DBC procedure" on page 523
DBRC			X	X	X	A procedure to execute the DBRC address space: "DBRC procedure" on page 529
DCC					X	An online execution procedure to initialize the DCCTL environment: "DCC procedure" on page 532
DFSJBP	X	X				Starts a JVM non-message-driven dependent region, called a JBP region: "DFSJBP procedure" on page 543
DFSJMP				X		Starts a JVM message-driven dependent region, called a JMP region: "DFSJMP procedure" on page 545
DFSMPR				X	X	A procedure to execute an IMS message processing address space: "DFSMPR procedure" on page 547
DLIBATCH	X	X				A one-step procedure for an offline DL/I batch processing program that uses PSB and DBD libraries: "DLIBATCH procedure" on page 548
DLISAS			X	X		A procedure to execute the DL/I address space: "DLISAS procedure" on page 551
DXRJPROC	X		X	X		A procedure that defines the Internal Resource Lock Manager and its use to the system: "DXRJPROC procedure" on page 554
FDR			X	X		A procedure to execute an FDBR region: "FDR procedure" on page 555
FPUTIL			X	X		A procedure to execute the Fast Path utility program with DEDBs online: "FPUTIL procedure" on page 559
IMS				X		An online execution procedure to initialize the DB/DC environment: "IMS procedure" on page 560
IMSBATCH			X	X	X	A procedure to execute an IMS online batch message processing address space: "IMSBATCH procedure" on page 571
IMSCOBGO	X	X				A three-step compile, bind, and go procedure combining the IMSCOBOL procedure with an execution step for a stand-alone DL/I batch address space: "IMSCOBGO procedure" on page 572
CBLTDLI[2]	X	X	X	X	X	Control statements necessary to establish a COBOL-to-DL/I interface.
IMSCOBOL	X	X	X	X	X	A two-step compile and bind procedure for IMS applications written in COBOL: "IMSCOBOL procedure" on page 575

Table 63. IMS system definition-supplied cataloged procedures and the environments in which they apply (continued)

Name	DB Batch	TM Batch	DBCTL	DB/DC	DCCTL	Description
IMSFP				X	X	A procedure for executing a Fast Path application program: "IMSFP procedure" on page 578
IMSMSG[1]				X	X	A job to execute an IMS message processing program: "IMSMSG procedure" on page 579
IMSPLI	X	X	X	X	X	A two-step compile and bind procedure for IMS applications written in PL/I: "IMSPLI procedure" on page 580
IMSPLIGO	X	X				A three-step compile, bind, and go procedure combining the IMSPLI procedure with an execution step for a stand-alone DL/I batch processing address space: "IMSPLIGO procedure" on page 581
IMSRDR			X	X	X	A DASD read procedure to read an IMSMSG job into the operating system job stream from direct access storage devices: "IMSRDR procedure" on page 584
PLITDLI[2]	X	X	X	X	X	Control statements necessary to establish a PL/I-to-DL/I interface.
RDIBATCH	X		X	X		A procedure to help maintain database availability in failure situations: "RDIBATCH procedure" on page 584

Notes:

[1] Job

[2] Control Statement. For information about these control statements, refer to "Establishing a DL/I interface from COBOL or PL/I" in *IMS Version 11 Application Programming*.

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

Parameter descriptions for IMS procedures

All of the parameters that can be specified in startup procedures in the IMS environment are documented in the following subtopics.

ABND= through AUTO= procedure parameters

The procedure parameters ABND= through AUTO=, which are described in the following list, can be specified in procedures in the IMS environment.

ABND=
Specifies the abend code.

AGN=
The AGN parameter is ignored. However, it is a positional parameter, so it must remain in the corresponding procedures for compatibility reasons.

ALOT=
Specifies the auto logoff time in minutes. Valid values are 0 and 10 - 1440. If the ALOT value is not specified, the value from the JCL member is used except for FINANCE, SLU P, and ISC. If ALOT is not specified on the logon

descriptor or overridden by the logon exit (DFSLGNX0) for FINANCE, SLU P, and ISC, a value of 1440 is used (the value from the JCL member is ignored).

- ALOT=0

  The terminal is logged off immediately when no sign-on is in effect. ALOT=0 is normally specified when sign-on occurs by one of the following methods:

  – The user is signed on automatically during the logon process (autologon)
  – Signon data is supplied by logon (BIND) user data
  – Signon data is supplied by the logon exit (DFSLGNX0)

  Restriction: Do not specify ALOT=0 for interactive terminal sessions if the session expects a response to the DFS3649 (sign on required) salutation message.

- ALOT=(10-1439)

  The session is terminated after the specified number of minutes elapses without a signed-on user.

- ALOT=1440

  The session is never automatically terminated.

ALOT applies only to IMS ETO Support terminals.

ALTID=

Specifies a 1- to 4-character name for an alternate IMS system. When your IMS message processing region tries to link to an IMS system, it tries to link to the system specified by the IMSID= parameter. If no IMS system matches the name that is specified with IMSID, the message processing region tries the system that is specified with the ALTID= parameter. If no IMS system matches the name that is specified with ALTID=, what happens next depends upon the option you choose on the OPT= parameter.

AOI1=A | C | N | R

Specifies the security product and level of security to use for authorization of type-1 AOI commands. The default value for this parameter is set during system definition by using either the DCC, IMS, or TRANSACT procedure.

A Includes options C and R. RACF is called first, and then the system authorization facility (SAF) return code, RACF return code, and RACF reason code are passed to the Command Authorization exit routine (DFSCCMD0). These return codes are decoded into a security code, which is also passed to the user exit DFSCCMD0 for processing.

C Specifies that DFSCCMD0 is to be called for command authorization.

N Specifies that no authorization security checking is to be used for this function.

R Specifies that RACF is to be called for command authorization. R is the default.

Because the AOI1 specification is not included in a checkpoint record, you can change the AOI1 value each time IMS is initialized

AOIP=

Specifies a value for the upper expansion limit of the AOI buffer pool. The value can be specified as 1- to 6-numeric characters, or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or

G is not specified, K is the default. The maximum value is 2G-1. If the specified value exceeds 2G-1, the default is 2G-1. If a value is not specified, the upper limit default is 2G-1.

AOIS=A | C | N | R | S
Specifies whether RACF, the Command Authorization exit routine (DFSCCMD0), or both are used to secure the DL/I ICMD call in AO applications. The possible values are:

A Includes options C and R. RACF is called first. Then, the security authorization facility (SAF) return code, RACF return code, and RACF reason code are passed to DFSCCMD0. These return codes are decoded into a security code, which is also passed to DFSCCMD0 for processing.

C Specifies that DFSCCMD0 is to be called for command authorization.

N Specifies that the DL/I call ICMD cannot be issued by an application program. N is the default.

R Specifies that RACF is to be called for command authorization.

S Specifies that no authorization checking is to be done.

Because AOIS is not included in the checkpoint record, you can change the AOIS value each time IMS is initialized.

APAR=
Specifies the maintenance level of the failing module for searching the APAR.

APARM=
Specifies the 1- to 32-character parameter that is passed to the application program as part of the information that is returned in the INQY call with the ENVIRON subfunction. The parameter must be enclosed in single quotation marks (') if special characters are used. Embedded commas (,) are not allowed.

APPC=Y | N
Specifies to activate (Y) or not to activate (N) APPC/IMS LU 6.2 support. The default is (N). The /STA APPC command overrides APPC=N. An IMS APPLID override can also be specified by the existing APPLID1= execution parameter during initialization.

APPCSE=C | F | N | P
Specifies the type of APPC RACF security. The value that you specify for this keyword has an equivalent value you can specify in the /SECURE APPC command. The first character that you specify applies to the APPCSE keyword. The second character that you specify becomes the equivalent value for the /SECURE APPC command.

The /SECURE APPC command overrides the value that you specify in the APPCSE= keyword.

C (CHECK)
Specifies APPC RACF command and transaction security check.

F (FULL)
Specifies APPC RACF command, transaction security check, and additional security for dependent regions. The default is F.

N (NONE)
Specifies no APPC RACF security within IMS. (Specifying NONE does not affect APPC/MVS RACF security.)

The IMS security exits (DFSCTRN0 and DFSCCMD0) are still called if they are defined to the system.

P (PROFILE)
Specifies the use of APPC transaction security in the TP profile or the default profile.

APPLFE=
This optional parameter specifies a 1 to 8-character name of an application front-end routine that is called whenever a message processing program (MPP) is scheduled in a message processing region (MPR).

If APPLFE= is specified, the IMS MPP controller performs the following functions:

- Loads the application front-end routine.
- Calls the application front-end routine for initialization processing.
- Calls the application front-end routine when an MPP is scheduled in an MPR.
- Calls the application front-end routine for shutdown processing when the MPP terminates.

The following describes the interface between the IMS MPP controller and the application front-end routine:

1. The IMS MPP controller calls the application front-end routine for initialization processing with the input registers set as follows:
   - R0=0
   - R1=0
   - R13=address of caller's save area
   - R14=return address of caller
   - R15=entry point address of the application front-end routine

   The application front-end routine must restore all registers, except register 15, upon return to IMS. Register 15 contains the return code.
2. The IMS MPP controller calls the application front-end routine when an MPP is scheduled in an MPR with the input registers set as follows:
   - R0=entry point address of the application MPP
   - R1=PCB list to be passed to the application MPP
   - R13=address of caller's save area
   - R14=return address of caller
   - R15=entry point address of the application front-end routine

   The application front-end routine must restore all registers, except register 15, upon return to IMS. Register 15 contains the return code.
3. The IMS MPP controller calls the application front-end routine to perform shutdown processing when the MPP is terminating:
   - R0=non-zero
   - R1=0
   - R13=address of caller's save area
   - R14=return address of caller
   - R15=entry point address of the application front-end routine

   The application front-end routine must restore all registers, except register 15, upon return to IMS. Register 15 contains the return code.

Note: Because the IMS MPP controller establishes an ESTAE, a program interrupt is intercepted and treated as an application failure when it occurs after the application front-end routine is called and before it returns to the IMS MPP controller. As a result, the IMS MPP controller reinitializes itself, and once again loads and calls the application front-end routine for initialization processing. IMS stops the transaction that is responsible for the interrupt.

APPLID1=*name1*
Specifies the application identification for the active IMS subsystem.

The VTAM APPLID is overridden by the APPLIDn= startup parameter, and APPLID3 is always used for the tracking subsystem. You must take care not to cause the same IMS VTAM APPLID to be opened for more than one IMS subsystem. If necessary, the APPLID=*(name1,name2,name3)* on the COMM macro should all be overridden by using the APPLID1, APPLID2, and APPLID3 parameters to ensure unique APPLIDS.

In a generic resource group environment, the APPLID1 parameter overrides the value that is specified in the first position of the APPLID= field of the IMS COMM macro. Using this execution parameter allows IMS system definitions to be identical by providing a unique APPLID override when starting the IMS job.

If MNPS is used for XRF systems, the APPLID1 and APPLID2 parameters specify the application identification for the APPLID ACB. The MNPS parameter specifies the application identification for the MNPS ACB.

APPLID2=*name2*
Specifies the application identification for the XRF alternate IMS subsystem.

APPLID3=*name3*
Specifies the application identification for the RSR tracking IMS subsystem.

name3 can match either *name1* or *name2,* but no two application identifications can be active in the network at the same time.

ARC=
Nonzero entry (01-99) specifies that automatic archiving is to be performed. ARC= also indicates the number of online data sets (OLDSs) to be filled before issuing the DBRC GENJCL ARCHIVE command.

A /DBRECOVERY command can cause automatic archiving, even if the number specified in ARC= is not reached.

IF ARC=00 is specified, no automatic archiving is performed.

The default is 01.

ARMRST=Y | N
Specifies whether (Y) or not (N) IMS registers with z/OS ARM and allows ARM to restart IMS in case of a failure. The default is Y.

ASOT=
Specifies the autosignoff time, in minutes. Valid values are 0 and 10 - 1440. You can override this value during sign-on with the user descriptor. If ASOT is not specified on either the user descriptor, the logon descriptor, or overridden by either the logon (DFSLGNX0) or sign-on (DFSSGNX0) exits, the value from the JCL member is used except for FINANCE, SLU P, and ISC. If ASOT is not specified on either the user or logon descriptor for FINANCE, SLU P, and ISC, a value of 1440 is used (the value from the JCL member is ignored).

- ASOT=0

The user is signed off immediately when no output is available to be sent. ASOT=0 is normally specified when no IMS input or output is available, or after the last available output message completes.

Restriction: Do not specify ASOT=0 for interactive terminals, for example 3270s or SLU2.

- ASOT=(10-1439)

  The user is signed off after the specified number of minutes elapses without terminal activity, irrespective of message status.
- ASOT=1440

  The user is never automatically signed off.

ASOT applies only to IMS ETO Support terminals.

AUTO=Y | N
Specifies whether (Y) or not (N) automatic restart is to be invoked. The default is N.

When AUTO=Y is specified, you cannot cold start IMS or modify any restart options before IMS automatically executes either an emergency restart or warm restart, as determined by the circumstances of the prior termination.

BKO= through CSLG= procedure parameters

The procedure parameters BKO= through CSLG=, which are described in the following list, can be specified in procedures in the IMS environment.

BKO=Y | N
Specifies whether (Y) or not (N) dynamic backout is to be performed; BKO= applies only to pseudoabends. Dynamic backout is to the last sync point. Y applies only if the log data set is assigned to DASD. The default is N.

BPECFG=
Specifies an 8-character name for the BPE confirmation parameters PROCLIB member. This parameter is optional. If it is not specified, the BPE defaults used are no user exits, a trace level of error, and US English as the language. This parameter can be specified only as an execution parameter. If no PROCLIB member is specified, BPE uses default values for all parameters.

BPEINIT=
Specifies the name of the module that contains initialization values that are required by BEPINI00 to start an address space. This required parameter can be specified only as an execution parameter.

CSLDINI0
The module that contains initialization values for an Open Database Manager address space.

CSLOINI0
The module that contains initialization values for a Common Service Layer Operations Manager address space.

CSLRINI0
The module that contains initialization values for a Common Service Layer Resource Manager address space.

CSLSINI0
The module that contains initialization values for a Common Service Layer Structured Call Interface address space.

CQSINI00
The module that contains initialization values for a Common Queue Server address space.

DSPBINIO
The module that contains initialization values for a DBRC address space.

HWSINI00
The module that contains the initialization values for an IMS Connect address space.

BSIZ=*buffer_size*
Specifies the database buffer size; the value can be 512, 1024, 2048, 4096, or a multiple of 4 KB (4096). You must specify a value large enough to contain the largest defined DEDB control interval size. The maximum value is 28672.

BUF=
Specifies a 1- to 3-digit number defining the number of 1 KB blocks to be used in calculating the size of the OSAM subpools. This parameter applies only to a DL/I batch environment; it is used only if the DFSVSAMP data set is not supplied or does not include any IOBF control statements.

For more information, see "OSAM buffer pool compatibility definition" on page 181.

CCTCVCAN=N | Y
Specifies whether (Y) or not (N) IMS converts abend code 222 TO 08E when canceling a coordinator controller (CCTL) address space that is connected to IMS.

Attention: If the CCTL is registered with the automatic restart manager (ARM), converting the abend code to a 08E might result in ARM restarting the CCTL.

N IMS does not convert abend code 222 to 08E. When you cancel the CCTL, IMS might issue abend code 0113, but only if a DLI call contains an active CCTL thread. N is the default.

Y IMS converts abend code 222 to 08E. This enables the database resource adapter (DRA) to recover and also prevents IMS from issuing abend code 0113.

CIOP=
Specifies a value for the upper expansion limit of the communication I/O buffer pool. The value can be specified as 1- to 6-numeric characters, or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. If the value specified exceeds 2G-1, the default is 2G-1. If a value is not specified, the upper limit default is 2G-1.

CKPTID=
Specifies the checkpoint at which the program is to be restarted.

Enclose the checkpoint ID in single quotation marks if it contains any non-ANSI characters, such as the slash.

Related reading: For more information about specifying values for CKPTID, see XRST call (Application Programming APIs).

You can suppress message DFS681I or DFS0540I each time a checkpoint call is made:

- To suppress message DFS681I, code `NOMSG681`.

- To suppress message DFS0540I, code NOMSG540.
- To suppress both messages, code NOMSGS.

The values NOMSG681, NOMSG540, and NOMSGS are not valid checkpoint restart IDs.

CL1=,CL2=,CL3=,CL4=
Is a required positional parameter that specifies four 3-digit decimal numbers indicating which classes of messages are to be processed by this message region. For example, if classes 1, 2, and 3 are to be processed by this region, the **PARM** field must be specified as PARM='MSG,001002003000'. The maximum value for a class is 999.

The sequence of specifying the classes determines relative class priority within the message region. In the example, all class 1 messages are selected for scheduling before any class 2 messages are considered. Class numbers cannot be greater than the maximum number of classes that are specified during system definition.

CLASS=
Specifies the default message class for the job being submitted to the internal reader.

CMDLANG
Specifies the language to be used for IMS command text that is distributed to OM automation clients upon request. This affects only the command descriptions that are displayed on a workstation SPOC that requests command text from OM. This value defaults to ENU for US English.

The value is not validated at OM initialization time. It is validated only when a CSLOMQRY QUERY TYPE(CMDSYNTAX) request is issued. OM attempts to read a partitioned data set (PDS) member in the data set specified by the CMDTEXTDSN= with a member name of “CSLOT” concatenated with the 3 character CMDLANG= value. This is the member that contains the command syntax translatable text. The CMDLANG= value can be overridden on the CSLOMQRY request.

This is an optional execution parameter. If specified, it overrides the value specified in the CSLOIxxx PROCLIB member.

CMDMCS=
The following table shows the procedures that describe the valid keywords for each online system type.

Table 64. Procedures for online systems

Online system type	Procedure name
DB/DC	IMS
DCCTL	DCC
DBCTL	DBC

The following subparameters apply to the IMS and DCC procedures.

N Commands cannot be entered from an MCS console. N is the default.

Y Commands can be entered from an MCS or E-MSC console by entering the command recognition character (CRC) followed by the command text.

R Commands can be entered from an MCS console in the form CRC followed by the command text. The system calls RACF (or equivalent) to verify that the user ID of the console is authorized to issue the command.

C Commands can be entered from an MCS console in the form CRC followed by the command text. DFSCCMDO is called to verify that the user ID of the console is authorized to issue the command.

B Include options R and C. RACF (or equivalent) is called first. After RACF is called, the SAF return code, RACF return code, and RACF reason code are passed to DFSCCMD0. The return codes are decoded into a security code that is also passed to DFSCCMD0. DFSCCMD0 determines whether the command is authorized for the user ID of the console.

This is an optional subparameter. If you do not select B, C, or R, commands can be entered from an MCS console. Neither RACF nor DFSCCMD0 is called to verify that the user ID of the console is authorized to issue the command.

The following subparameters apply to the DBC procedure.

R RACF (or equivalent) is called to verify that the user ID of the console is authorized to issue the command.

C DFSCCMD0 is called to verify that the user ID of the console is authorized to issue the command.

B Include options R and C. RACF (or equivalent) is called first. After RACF is called, the SAF return code, RACF return code, and RACF reason code are passed to DFSCCMD0. These return codes are decoded into a security code that is also passed to DFSCCMD0. DFSCCMD0 determines whether the command is authorized for the user ID of the console.

Note: CRC for DB/DC does not work for restart commands. The ability to issue IMS commands using the CRC or IMSID in a DB/DC environment is available if the proper specification of the CMDMCS parameter is supplied. However, this does not work for restart commands /ERESTART and /NRESTART. You must enter these commands the same as before. After restart is complete, you can enter commands by using the CRC or the IMSID.

CMDP=

Specifies a value for the upper expansion limit of the command pool. Some of the DRD commands, for example, use this pool. Specify the value as either 1- to 6-numeric characters, or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If you do not specify K, M, or G, K is the default. The maximum value that can be specified is 2G-1 (2 gigabytes minus 1 byte). If the value specified exceeds 2G-1, the default is 2G-1. If a value is not specified, the upper limit default is 2G-1.

CMDSEC=

Specifies the security method to be used for OM command security.

A Specifies that both RACF (or an equivalent security product) and the OM Security exit are to be called (options E and R). RACF is called first. Then the security authorization facility (SAF) return code, RACF return code, and RACF reason code are passed to the exit. These return codes are decoded into a security code, which is also passed to the exit for processing.

E Specifies that the OM Security user exit routine is to be called for command authorization.

N Specifies that no authorization checking is to be done. This is the default.

R Specifies that RACF (or an equivalent security product) is to be called for command authorization.

This is an optional execution parameter. If specified, it overrides the value specified in the CSLOIxxx PROCLIB member. If not specified, the value in the CSLOIxxx PROCLIB member is used.

CORE=

Note: This parameter is no longer used. CORE= was replaced by the PIINCR= and PIMAX= execution parameters.

CPLOG=

Specifies the number of system log records between system-generated checkpoints. The values that are valid are between 1K and 16M. The default is 500K. The value that is specified must be one or more numeric characters, followed either by K (for thousand) or M (for million). For example, specify 1K, not 1000.

CPUTIME=

Specifies the task timing option for either BMP or Fast Path.

0 No task timing is performed for this BMP or Fast Path region. A default value is placed in the 07 accounting log record to indicate that no timing was done. This is the default.

N A 1- to 4-digit number that specifies the maximum task time in minutes to be allowed for the execution of this BMP or Fast Path region. The values can range from 1 - 1440 minutes. If a value of 1440 or greater is specified, a time of 24 hours is used. When *N* is nonzero, the STIMER=1 option is in effect.

For BMP task-timing, the measured time includes the DL/I processing that occurs in the dependent region. This includes CPU time for External Subsystem Attach Facility (ESAF) users such as DB2 for z/OS that continue to process under the TCB of the dependent region. When the dependent region exceeds the specified time, a U0240 abend occurs after the current DL/I call completes. To control the execution time of a BMP, use CPUTIME=N instead of the TIME= of the operating system. If the operating system time limit is included with a job that specifies CPUTIME=N, ensure that the former is a sufficiently higher value to allow for initialization and one completed DL/I call beyond the CPUTIME value. When the CPUTIME=N option is used, the application program must not use the STIMER or TTIMER CANCEL.

For Fast Path programs, the timer is set at the beginning of program invocation. However, message-driven programs are not allowed to exceed the amount of time that is defined for the program to process a single transaction. In this case, CPUTIME=*n* is ignored.

CRC=*x*

CRC specifies a 1-character command recognition character for DBCTL, DB/DC or DCCTL. Commands that are entered at a z/OS system console with the CRC as a prefix, are processed by the control region that has the CRC defined.

CRC overrides the value that is specified on the CMDCHAR keyword of the system definition IMSCTRL macro.

If CMDCHAR is not specified, and CRC is not specified, the default value is /.

If CMDCHAR=NONE is specified, and CRC= is not specified, commands to the DBCTL control region are prefixed by the IMSID.

The CRC must be different from any other CRC in DBCTL regions already running. Choose an unreserved character that is appropriate for your particular system.

Do not use a comma (,) as a command recognition character.

Do not use less than (<) or greater than (>) as a command recognition character. The results would be unpredictable, because the Operations Manager (OM) uses these characters in its XML output.

Do not use a character for the CRC that is the same as the beginning character of a z/OS command. If the CRC is the same as the beginning character of a z/OS command, that command does not work after you start IMS. For example, if you start IMS with CRC=D, z/OS does not respond to any of the z/OS display commands such as D A,L.

The CRC value that you select might affect multisegment command processing. The last character of the command is checked during multisegment command processing. If that character matches the CRC, the command is assumed to be a multisegment command. Processing of the command waits for the rest of the command. For example, if you use B for the command recognition character, and issue the command 'BMODIFY PREPARE ACBLIB', the command does not run. The ending 'B' in 'ACBLIB' is interpreted to indicate a multisegment command. To enter this command correctly, enter:

```
'BMODIFY PREPARE ACBLIB . '
```

CSAPSB=

When the DL/I address space option (LSO=S) is specified, two program specification block pools are used. CSAPSB specifies the size of pool in the z/OS common storage area (CSA), and DLIPSB specifies the size in the DL/I address space. In an LSO=S system, the PSB= parameter is ignored.

The output of the ACBGEN utility indicates maximum CSA space, average CSA space, and the CSA space required by each PSB in the pool. This output is useful in determining what value to specify for the CSAPSB= parameter. Similarly, before specifying the DLIPSB= parameter, you should consider the secondary address space (SAS) size that is required for each PSB, maximum SAS space, and average SAS space. Neither parameter should be zero. Normally, DLIPSB should be larger than CSAPSB.

The values for the pool sizes can be specified as either 1- to 6-numeric characters, or 1- to 5-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements are true:

- The specified value is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The specified value exceeds 2G-1.
- No value is specified.

The sizes that are specified are rounded up to the nearest page boundary. The sizes of the two pools can also be specified during IMS system definition by using the SASPSB parameter in the BUFPOOLS macro.

For the FDR procedure, the sum of the values for the CSAPSB and DLIPSB defines the PSB pool size. If PSB is also specified, the larger value (PSB or the sum of CSAPSB and DLIPSB) is used.

For CSAPSB and PSBW, DFSIINS0 obtains contiguous space in the z/OS common area. If this storage is unavailable in the z/OS common area, ABENDU0717 occurs.

CSLG=*xxx*
Specifies a 3-character suffix for the Common Service Layer IMS.PROCLIB member, DFSCGxxx. When you specify this parameter, IMS uses the Common Service Layer (CSL) to manage and operate the IMSplex. The minimum configuration for the CSL is to have one SCI and one OM on each z/OS image where an IMS control region resides.

This parameter must be explicitly specified. There are no default values.

DBBF= through DIRCA= procedure parameters

The procedure parameters DBBF= through DIRCA=, which are described in the following list, can be specified in procedures in the IMS environment.

DBBF=
Specifies the maximum number of Fast Path buffers. Storage is obtained as needed from either:

- The extended common storage area (ECSA).
- FDBR private storage, if the default value FPBUFF=LOCAL is specified in the DFSFDRxx IMS.PROCLIB member.

You can adjust the ECSA or region to meet your requirements. If you do not specify a value for DBBF, IMS uses a default of 10 buffers. If you specify a value for DBBF, IMS overrides the default with your value, and you must adjust for any change in DBFX specification.

The maximum number of Fast Path buffers can vary based on the buffer size (BSIZ):

Buffer size (BSIZ)	Maximum number of Fast Path buffers
512	4 000 000
1024	2 000 000
2048	1 000 000
4096	500 000
8192	250 000
12 288	175 000
16 384	130 000
20 480	100 000
24 576	87 000
28 672	74 000

DBD=
Specifies the Fast Path Data Entry Database (DEDB) DBD name. DBD= can be 1 - 8 characters.

DBFP=
Specifies a 1- to 4-digit value that determines how often the IMS Fast Path buffer pool manager can page free unneeded buffers. The default value of 0 indicates that any unneeded buffers are page freed at each dependent region termination.

This parameter applies only to the Fast Path buffer pool as defined by the DBBF, DBFX, and BSIZ parameters. It is not valid for the Fast Path 64-bit buffer manager.

Specifying a value for this parameter can improve z/OS performance when many IMS dependent regions with NBA and OBA specified are started and terminated in a short time period (for example, within a period of 1 second). When many page-fixed buffers exist within the z/OS system, specifying a value for this parameter can decrease the z/OS overhead that is required to page fix and page free buffers.

This parameter alters only the method of managing the page fixing and page freeing of the IMS Fast Path buffers in the IMS Fast Path buffer pool. It does not affect the number of buffers in the buffer pool.

0 Allows the number of page-fixed buffers to fluctuate at each termination.

1 Allows the number of IMS Fast Path page-fixed I/O buffers to increase, without a corresponding free. If this option is selected, the number of page-fixed buffers only increases.

2-3600 Determines how often (in seconds) unneeded buffers can be page freed. During this time interval, the number of page-fixed buffers only increases; no page freeing is performed. When the specified time interval expires, the number of page-fixed buffers is reset to the minimum number of buffers needed to satisfy the current requirements.

DBFX=

Specifies a 1- to 5-digit number of additional buffers (not separate pools) from DBBF to be set aside and page-fixed at control region initialization. This specification allows for asynchronous processing, which ensures that the DEDB updates are held until the associated log buffer is written. The default is 10.

DBLDL=

An optional 4-digit number that specifies the maximum number of BLDL entries to be kept for application modules that are not preloaded. The default maximum is 20. The maximum value is 9999.

For a region that is used to test new or changed programs, set the DBLDL parameter to 0, ensuring that the most current version of the program is loaded for each execution. Specifying the DOPT parameter on the APPLCTN macro disables quick rescheduling.

Related reading: For more information, see references to BLDL in *IMS Version 11 System Administration*.

DBRC=

Specifies whether database recovery control is to be used during this execution of IMS. The possible values for DBRC= are:

C This value has no meaning except during a batch backout run of IMS. If specified for other than a batch backout run, it is treated as a null.

With DBRC=C, you can back out batch jobs that terminate normally during an execution that includes DBRC, but not IRLM. During batch backout of jobs that terminated normally, IMS performs authorization for databases and then backs out database changes, as is done for jobs that terminate abnormally.

If DBRC=C is specified when the previous execution of IMS included both DBRC and IRLM, batch backout fails if the previous execution completed normally.

N Specifies that DBRC is not used during this execution unless DBRC=FORCE is defined in DFSIDEF0. If DBRC=FORCE is defined in DFSIDEF0, and if this is not a batch backout execution of IMS, a message is issued, and a nonzero return code is returned.

If this is a batch backout execution of IMS, the DBRC=FORCE specification defined in DFSIDEF0 can be overridden by specifying DBRC=N on the EXEC procedure. Thus, if the previous execution of IMS used DBRC but not IRLM, batch backout executes without DBRC.

Null Specifies whether DBRC is included, based on the specification in the installation defaults module, DFSIDEF0. If DFSIDEF0 is not used, DBRC=Y is the default.

Y Specifies that DBRC is used during this execution of IMS.

If the IRLM is not being used, DBRC provides additional database security. If a common RECON data set is shared among subsystems, DBRC allows proper authorization to registered databases so that the subsystems can share data at the database level.

Attention: If DBRC is used to maintain the integrity of databases, DBRC=N should be specified only when DBRC is unavailable or when you are sure that the integrity of databases is not destroyed. When DBRC=N, before reactivating DBRC, you must register information about the log volumes created while DBRC was inactive.

DBRCGRP=*id*
Specifies the 1- to 3-character identifier (ID) assigned to a group of DBRC instances that access the same RECON data set in an IMSplex. The DBRCGRP ID must be unique. If no DBRCGRP ID is specified, and the DBRC SCI Registration exit routine (DSPSCIX0) is not being used, a value of '001' is used.

DBRCINIT=*nnn*
Specifies a 3-character suffix for the DBRC initialization parameters PROCLIB member, DSPBIxxx. The default suffix is 000. This optional parameter can be specified only as an execution parameter.

DBRCNM=
Specifies the member in SYS1.PROCLIB containing the procedure for the DBRC address space. The IMS control region automatically starts the DBRC procedure, using this specification. The name can be up to 8 characters, the first of which must be alphabetic. The default is DBRC.

DBRSE=
Specifies the 8-character RSENAME. It is used by a primary and an alternate DBCTL region in a “DBCTL standby” configuration, and it must be the same in both DBCTL regions.

A cold start is required for any change to this parameter to take effect.

Related reading: For more information about RSENAME, refer to *IMS Version 11 System Administration*.

DBWP=
Specifies the amount of subpool 231 storage to be allocated to the database work area pool. The value can be specified as either 1- to 6-numeric characters, or 1- to 5-numeric characters followed by either K (kilobyte), M (megabyte), or

G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. If the value specified exceeds 2G-1, the default is 2G-1. The pool size that is specified is rounded up to the nearest page boundary.

If the logical record size is greater than 8K, the DBWP size must be specified as the largest logical record size times the number of active regions.

If the logical record size is less than 8K, the DBWP size must be specified as 8K times the number of active regions.

Note: During DL/I database OPEN/CLOSE processing in a non-batch environment, DL/I separate address space (DLISAS) E-private storage is used instead of the database work area pool.

DC=*nnn*

Specifies the 3-character suffix for the DFSDCxxx member in IMS.PROCLIB. The default is 000.

DFSDCxxx parameters allow the override of the VTAM node name and the LTERM names for the master and secondary terminals that are defined in the system definition.

DEADLOK='*iiii***,***kkkk***'**

Specifies the local deadlock-detection interval in seconds or milliseconds (*iiii*) and the number of local cycles (*kkkk*) that are to occur before a global detection is initiated.

The LOCKTIME parameter in the DFSVSMxx member of the IMS PROCLIB data set is also associated with IRLM deadlock management. See "Enabling the IRLM lock timeout function" on page 768 for more information.

iiii A 1- to 4-digit number from 1 to 9999 that specifies the length of time in seconds (1 - 99) or milliseconds (100 - 9999) that is used for the IRLM local deadlock detection interval. Values less than 100 are recognized in seconds with 5 seconds being the maximum used. Values from 100 to 9999 are treated as milliseconds and IRLM truncates this to even 100 millisecond increments, with a maximum value of 5000 (5 seconds).

Note: Once IMS TIMEOUT candidates time out, they remain timeout candidates and are presented to the timeout exit each Global deadlock cycle. IMS creates SMF 79.15 records when candidates are presented which are written to the SMF data sets (if enabled). If timeout candidates are found and the value for *iiii* is subsecond, there are many SMF 79.15 records written per second until the tasks are no longer waiting in IRLM.

kkkk Specifies the number of local deadlock cycles performed before global deadlock detection is performed. The default is 1.

The valid value for *iiii* is a number 1 - 5. The value that is used for *kkkk* is always 1.

In data sharing environments, IRLM synchronizes all of the DEADLOK values in the group to the values specified on the most recent IRLM to join the group. The DEADLOK value may be changed by starting a member with the values desired, or by issuing the MODIFY irlmproc, SET, DEADLOCK= command.

Recommendation: Make sure that the installation specifies the same value for DEADLOK on all its IRLM start-up procedures, and use the START irlmproc command or the MODIFY command to override this value only when the

interval must be changed from its original value.
If all members of a sysplex group do not support subsecond deadlock processing, the value range used by the group is 1 to 5 seconds, with values specified less than 1 second, changed to 1 second.

Recommendation: Specify the same values for the DEADLOK parameter on all your IRLM startup procedures if you do not want the value for the last IRLM to connect to the group to become the value for all.

Deadlock processing time is minimal if the number of locks that are HELD and WAITING is small. However, deadlock processing can affect system performance when IRLM encounters a real deadlock situation.

The amount of time to run deadlock processing is directly proportional to the number of resources HELD (this is a minor time component) plus the number of WAITERs (this is a major time component) on those resources.

If the WAITERs increase as a result of a real deadlock situation, the deadlock must be found as quickly as possible. The larger the number of WAITERs, the longer deadlock processing takes to find and break the deadlock. While deadlock processing is running, nothing else is happening with respect to that particular IRLM! The effect on performance is particularly significant in a Sysplex environment because any IRLM that is running deadlock processing can cause other members to slow down or stop processing if they need this IRLM to process another request.

The previous paragraph might imply that you should have the DEADLOK parameter on the local IRLM set as high as possible, which would cause deadlock processing to run the least. However, if a deadlock situation exists, the faster it is detected and broken, the better the system runs.

Recommendation: Carefully choose a value for the deadlock frequency that is high enough so that deadlock processing runs often enough to resolve deadlock situations and at the same time is low enough so that system performance is minimally affected. If you have many threads that are capable of becoming WAITERs, it is important to find and break that deadlock as quickly as possible before the WAIT chains require deadlock processing to run seconds (or even minutes) to perform a single local deadlock cycle. To find and break the deadlock is especially important in a Sysplex environment where multiple local cycles are required to break each deadlock, which in turn causes WAITERs to back up on all systems.

Recommendation: Monitor your system application programs for the actual number of deadlocks that IRLM had to break. If you find that the number of these deadlocks is low (or non-existent), you could raise the local deadlock value. If, however, you find that there are a lot of deadlocks that needed to be broken, you might consider having a low deadlock value.

Another aspect of deadlock processing and how it affects system performance is the number of locks that are held. IRLM must evaluate all held resources to check for WAITERs. Therefore, the amount of time that deadlock processing takes depends on the number of held resources (more held resources equals more deadlock processing time).

DESC=

Specifies the message descriptor code to be assigned to the IMS system console messages. If the parameter is not specified, then the value specified in the IMSCTF system definition macro is used.

The DESC parameter overrides the value specified on the IMSCTRL macro at system definition time.

DFSDF=*xxx*
Use this parameter to specify the 3-character suffix for the IMS.PROCLIB member, DFSDFxxx. You can also specify the suffix for the DFSDFxxx PROCLIB member in the DFSPBxxx PROCLIB member.

If the FP 64-bit Buffer Manager is being used on systems that are being tracked by a Fast Database Recovery (FDBR) address space, the DFSDF= keyword must be specified on the FDR procedure.

DIRCA=
Specifies the size of the dependent region interregion communication area (DIRCA); the size specified must be a 3-digit number (for example, 001) representing the number of 1 KB blocks of subpool 251 to be reserved to hold a copy of your PCBs.

If you do not specify a DIRCA size, or if you specify a size of 000, IMS uses the system default for this message region. The size of the system default, determined during system initialization, is the maximum size required for any PSB. If necessary, IMS updates this default when an online change is made or when a dynamic PSB is scheduled.

Normally, you should not specify this parameter. You can use it to reduce the DIRCA size for a region where only PSBs with small DIRCA requirements are scheduled by class scheduling.

The output from message DFS6891 occurring during ACBGEN facilitates DIRCA size calculation. Each PSB has a PCB=value that is the sum of the PCBs within it. The PCB value plus 64 is the required DIRCA size, in bytes.

The Fast Path utility program uses a PSB that contains an I/O PCB and a DB PCB for the DBD specified in the JCL parameters. A DIRCA size of 002 is sufficient to contain the control blocks built for the PSB.

DLINM= through EXVR= procedure parameters

The procedure parameters DLINM= through EXVR=, which are described in the following list, can be specified in procedures in the IMS environment.

DLINM=
Specifies the member in SYS1.PROCLIB containing the procedure for the optional DL/I address space. If LSO=S is specified, DLINM= automatically starts the DL/I address space procedure. The name can be up to 8 characters, the first of which must be alphabetic. The default is DLISAS.

DLIPSB=
When the DL/I address space option (LSO=S) is specified, two program specification block pools are used. CSAPSB specifies the size of pool in the z/OS common storage area (CSA), and DLIPSB specifies the size in the DL/I address space. In an LSO=S system, the PSB= parameter is ignored.

The output of the ACBGEN utility indicates maximum CSA space, average CSA space, and the CSA space required by each PSB in the pool. This output is useful in determining what value to specify for the CSAPSB= parameter. Similarly, before specifying the DLIPSB= parameter, you should consider the secondary address space (SAS) size required for each PSB, maximum SAS space, and average SAS space. Neither parameter should be zero. Normally, DLIPSB should be larger than CSAPSB.

The values for the pool sizes can be specified as either 1- to 6-numeric characters, or 1- to 5-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements is true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

The sizes specified are rounded up to the nearest page boundary. The sizes of the two pools can also be specified during IMS system definition, using the SASPSB parameter in the BUFPOOLS macro.

For the FDR procedure, the sum of the values for the CSAPSB and DLIPSB defines the PSB pool size. If PSB is also specified, the larger value (PSB or the sum of CSAPSB and DLIPSB) is used.

DLQT=
Specifies a 1- to 3-digit number for the size of the dead letter queue, in days. Valid values are from 0 to 365. The default is 60. If you specify a null or invalid value, 60 is assumed.

DMB=
Specifies the amount of subpool 231 storage to be allocated to the DMB pool. The value can be specified as either 1- to 6-numeric characters, or 1- to 5-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. If the value specified exceeds 2G-1, the default is 2G-1. The pool size specified is rounded up to the nearest page boundary.

The output of the ACBGEN utility indicates the size of each DMB processed. This output should be examined before specifying the DMB= parameter.

DMHVF=
Specifies (in megabytes) how much storage in the Fast Path DEDB VSO emergency restart data space is to be page fixed. Valid values are from 0 to 99.

This startup parameter is valid only on an XRF tracking system. The emergency restart data space is deleted at the end of the XRF takeover. At that time, the storage is also page freed.

DPRTY=
This parameter is ignored by z/OS, but retained in IMS for compatibility.

Related reading: For details on DPRTY, see *MVS/ESA Job Control Language Reference*.

DSCT=
Specifies a 1-character suffix identifying the user-supplied descriptor table, DFSDSCTy, in IMS.PROCLIB. The default is 0.

DYNP=
Specifies a value for the upper expansion limit of the TM dynamic storage private buffer pool. The value can be specified as 1- to 6-numeric characters, or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements is true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

The IMS TM code uses this pool for module dynamic storage.

EMHB=

Specifies a value for the upper expansion limit of the Fast Path expedited message handler buffer pool. The value can be specified as 1- to 6-numeric characters, or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G, is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements is true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

EMHL=

Specifies a 1- to 5-digit number for the length of the Fast Path expedited message handler buffer. If no buffer length is available from either the TRANSACT or APPL macro or, in the case of a static terminal, an FPBUF specification, this EMHL buffer length is used. Valid values are from 12 to 30720. The default is 2K. If you specify a null or invalid value, 2K is assumed.

ENVIRON=

Specifies the name of the PROCLIB member that contains the environment settings. A sample member DFSJVMEV is supplied in the IMS sample library. ENVIRON is a required parameter.

Environment variables in the form of X=Y, where X is the environment variable and Y is the value of the environment variable, can be specified. For example, you can specify:`JAVA_DUMP_OPTS=ONINTERRUPT(ALL),ONANYSIGNAL(ALL)` in the member, and IMS sets the environment variable `JAVA_DUMP_OPTS` to the value `ONINTERRUPT(ALL),ONANYSIGNAL(ALL)`.

The member named by the ENVIRON parameter can have comments but must begin with an asterisk (*) in the first column. Each line in the options file must be no longer than 72 bytes including the continuation mark. To mark the continuation of a line, place the greater-than sign (>) at the end of the line. Path name strings are restricted to 255 bytes in length. A path name string can be one path name or several path names separated by a colon (:). The portion of the path name string greater than 255 bytes is ignored.

This parameter can be specified for non-Java dependent regions (MPPs, BMPs, and IFPs) also, in order to create and manage the JVM in MPP, BMP, and IFP regions.

JLEOPT= Y | N

Specifies whether (Y) or not (N) you want IMS to terminate the IBM Language Environment® for z/OS (LE) enclave and reinitialize the LE enclave after each application program scheduling. This would enable (or not) the LE dynamic runtime parameter overrides for each application program scheduled in the dependent region.

LIBPATH=

Required variable that must specify the environment variable. There must not be a space between the equal sign (=) in the LIBPATH= and the beginning of the first path name.

The LIBPATH= parameter must contain a list of directories, separated by colons, that includes the locations of:

- The libjvm.so and libwrappers.so libraries (DLLs).
- The Java class libraries for IMS (libJavTDLI.so).

For example, if you code ENVIRON=DFSENV on your JMP or JBP procedure, the DFSENV member must reside in IMS.PROCLIB or equivalent. The DFSENV member must contain the following (comments are optional):

```
************************************************************
*LIBPATH environment variable*
*--------------------------*
*/usr/lpp/java150/J5.0/bin/j9vm is path to libjvm.so
*/usr/lpp/java150/J5.0/bin is path to libwrappers.so
************************************************************
LIBPATH=/usr/lpp/java150/J5.0/bin/j9vm: >
/usr/lpp/java150/J5.0/bin:/usr/lpp/ims/imsjava10
```

EPCB=

Specifies the amount of subpool 231 storage to be allocated to the EPCB pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2G-1. If the value specified exceeds 2G-1 the default is 2G-1. The pool size specified is rounded up to the nearest page boundary. The size specified on the EPCB= parameter overrides the size specified in the BUFPOOLS macro.

If Fast Path is not generated, the EPCB pool is not allocated. If Fast Path is generated, an EPCB pool is required when the PSB, used by the IMS dependent region, has PCBs for Fast Path databases. This is true regardless of whether the region ever makes a call to these Fast Path databases. If nothing is specified on the EPCB= parameter or on the BUFPOOLS macro and if Fast Path is generated, the default pool size is 8K.

See Table 49 on page 314 for more information about calculating EPCB storage for PSBs.

ETO=N | M | Y

Specifies whether IMS Extended Terminal Option Support (IMS ETO Support) is enabled. The default is N.

N Specifies that terminals that are not defined to IMS have session initiation attempts rejected. Logon user data is not supported for static terminals; any user data provided at logon time is ignored.

M Specifies that terminals that are not defined to IMS have session establishment attempts rejected. Logon user data is supported for static terminals; any user data provided at logon time allows automatic sign on to occur during session initiation.

Y Specifies that terminals that are not defined through the system definition process can establish sessions with IMS if the following conditions are true:

- The terminal type is supported by IMS for dynamic allocation.
- A default logon descriptor (provided either by IMS or by you) has sufficient information to create the control blocks necessary to accept the session request.

- The IMS ETO Support feature is installed.

EXCPVR=
Specifies whether (1) or not (0) the OSAM Database Buffer Pool is to be page fixed. A value of 0 or 1 must appear in the generated JCL statement for this parameter.

EXVR=
Specifies whether (1 or Y) or not (0 or N) the Queue Manager buffer pools are to be long-term page fixed.

FBP= through HSBMBR= procedure parameters

The procedure parameters FBP= through HSBMBR=, which are described in the following list, can be specified in procedures in the IMS environment.

FBP=
Specifies the amount of subpool 0 storage to be allocated to the message format buffer pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G, is not specified, K is the default. The maximum value that can be specified is 2G-1. If the value specified exceeds 2G-1 the default is 2G-1. The pool size specified is rounded up to the nearest page boundary.

FDRMBR=*xx*
Specifies the 2-character suffix for the FDR member DFSFDRxx in the IMS.PROCLIB.

Restriction: When you specify the FDR parameter on the IMS and DBCTL procedures you must also specify IRLM=Y.

FESTIM=
Specifies a timeout value in seconds to be used for front-end switching. The minimum value is 1, the maximum value is 300. If this parameter is not specified, the default is the value specified at system definition.

FIX=*xx*
Specifies the 2-character suffix for DFSFIXxx and DFSDRFxx. This specifies the IMS.PROCLIB member to control both of the following:

- Page fixing portions of the control program
- Loading portions of the control program into DREF storage

For information about defining the content of these members, see "DFSFIXnn member of the IMS PROCLIB data set" on page 704.

FMID=
Specifies the component's FMID to search on the preventive service planning (PSP) database.

FMTO=
Specifies the type of dump output to be produced.

FP=N | Y
Indicates whether Fast Path is enabled. You can only change the FP value at IMS cold start. If IMS is either warm started or emergency restarted with a different FP value, restart terminates. The FP value must be the same on the active IMS system and the tracking system (XRF, FDBR, and RSR). If the tracking system has a different FP value from the active system, it terminates.

N Fast Path is not enabled. All other Fast Path execute parameters are

ignored. This is the default. However, if FP=N, and you attempt to use a Fast Path resource or command, results are unpredictable.

Y Fast Path is enabled.

FPOPN=N | P | R | A | D
Specifies whether to reopen DEDB areas automatically during IMS restart.

N Specifies that areas should not be reopened automatically during IMS restart. Areas that are registered with DBRC for preopening are preopened before the completion of an IMS start or restart as usual; other areas are not preopened.

This is the default.

P Specifies that after an IMS normal or emergency restart, IMS should preopen those areas that are registered with DBRC to be preopened asynchronously and concurrently with normal IMS online processing.

R Specifies that during IMS emergency restart, IMS should automatically reopen all areas that were open at the time of the previous IMS failure.

IMS reopens the areas after restart processing is complete. Reopening the areas occurs asynchronously and concurrently with normal IMS online processing.

During IMS normal restart, this option behaves the same as option P.

A Specifies that during IMS emergency restart, IMS should automatically reopen all areas that were open at the time of the previous IMS failure. This option also specifies that IMS should preopen those areas that are registered with DBRC to be preopened.

IMS reopens the areas after completing restart processing. Reopening and preopening the areas occurs asynchronously and concurrently with normal IMS online processing.

During IMS normal restart, this option behaves the same as option P.

D Specifies that during IMS control region initialization, the DEDB area preopen process is disabled. Areas that are defined to DBRC with the PREOPEN option are not opened. The open process occurs when the dependent region first requests data or when a /STA AREA command is issued after restart is complete. This option is valid for an IMS cold start, normal restart, and emergency restart. This option has no effect on an XRF tracking system or an FDBR system.

The FPOPN option does not change any area options in DBRC. This includes PREOPEN, VSO, and PRELOAD. These options remain set in the RECON data set.

This option does not affect DEDB forward recovery (REDO) processing. If an area requires forward recovery during an emergency restart, the area is opened, updated, and then closed.

FPRLM=N | P | S | R | A
Specifies whether to restart DEDB areas automatically during IRLM reconnect.

N Specifies that areas should not be restarted automatically during IRLM reconnect. All DEDB areas remain stopped and unopened until you issue a /START DATABASE or /START AREA command.

This is the default.

P Specifies that after IRLM reconnect, IMS preopens those areas that are registered with DBRC to be preopened asynchronously and concurrently with normal IMS online processing.

S Specifies that after IRLM reconnect, IMS automatically restarts (but does not reopen) all areas that were started at the time of the IRLM failure.

IMS restarts the areas after IRLM completes reconnect processing. Restarting the areas occurs asynchronously and concurrently with normal IMS online processing.

R Specifies that during IRLM reconnect, IMS automatically restarts and reopens all areas that were open and started at the time of the IRLM failure. After IRLM reconnects, IMS restores all DEDB areas to their state at the time of the IRLM failure, restarting and reopening DEDB areas regardless of whether the DEDB areas have preopen status.

IMS restarts the areas after IRLM completes reconnect processing. After IMS restarts all the areas that were started at the time of the IRLM failure, IMS reopens them. If an area is not yet reopened when an IMS application program requests access, it is reopened immediately.

Restarting and reopening the areas occurs asynchronously and concurrently with normal IMS online processing.

A After IRLM reconnects, IMS restarts and reopens all DEDB areas that were open at the time of the IRLM failure and all DEDB areas that have a preopen status, even if they were closed at the time of the IRLM failure. IMS restores the DEDB areas asynchronously to the resumption of application processing.

FPWP=
Specifies a value representing the upper expansion limit of the Fast Path work pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2G-1. If the value specified exceeds 2G-1 the default is 2G-1. If a value is not specified, the upper limit default is 2G-1.

FRE=
Specifies a 1- to 5-digit number of fetch request elements used to load MFS blocks into the message format buffer pool.

GEN=
Specifies a general argument string.

GRNAME=
Specifies a 1- to 8-character name of the z/OS cross-system coupling facility group for IMS Open Transaction Manager Access (OTMA). The group name should be alphanumeric (uppercase only) or specified with the following special characters: `#, $, @`.

The OTMA XCF group name must be unique within an IMSplex. The first three characters must not conflict with any other IMS XCF group names. Some known IMS XCF group names are:

- DFSxxxxx
- FDRxxxxx
- CQSxxxxx
- CSLxxxxx
- TOIxxxxx

For IMS systems that do not use RSR or XRF, OTMA uses the OTMANM= parameter as the IMS XCF member name. If OTMANM= is not specified, OTMA uses the non-APPC VTAM LU name (APPLID1) as the member name.

For IMS systems that use RSR or XRF, OTMA uses the USERVAR value as the IMS XCF member name.

This parameter is only valid in IMS DB/DC or IMS TM-DB2 environments.

GRSNAME=*generic_resources_group_name*
Specifies a 1- to 8-byte character name of the generic resources group. The group name should be specified in uppercase alphanumeric characters, or with the following special characters: `#, $, @`. GRSNAME must begin with an alphabetic character. You must specify the same GRSNAME for all IMS systems participating within the same generic resources group.

GSGNAME=*global_service_group_name*
Specifies the global service group name to be used. If GSGNAME= is not specified in the DBBBATCH procedure, the default is the GSGNAME specified in the IMSCTRL macro.

If GSGNAME=NONE is specified, this procedure does not reference any global service group and any activity performed is not tracked by RSR.

If a GSGNAME is supplied either by the GSGNAME= specification or from the system definition IMSCTRL macro, then *DBRC=Y* must be used.

HIOP=
Specifies a value for the upper expansion limit of the high communication I/O pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2G-1. If the value specified exceeds 2G-1, the default is 2G-1. If a value is not specified, the upper limit default is 2G-1.

HSBID=
Specifies (for XRF systems only) the XRF system identification. One system is associated with digit 1 and one is associated with digit 2. Either can be brought up as an active or an alternate system.

Through this parameter, IMS identifies the master terminal and the message queues that are associated with it. For example, an IMS system with HSBID=1 uses the first master terminal in the stage 1 definition.

Recommendation: Code the HSBID parameter if you are planning to run XRF anytime in the future. Having HSBID coded could save performing an IMS definition at a later date.

XRF requires that two APPLIDs be specified on the system definition COMM macro. XRF also requires that two *nodenames* be specified by the NAME parameter on the TERMINAL macro. These are for the master and secondary master terminal.

If a null value is specified for HSBID (for example, HSBID=' '), XRF is disabled. Changing the HSBID from usable ('1' or '2') to null or from null to usable requires a cold start.

HSBMBR=*xx*
Specifies (for XRF systems only) the 2-character suffix for XRF member DFSHSBxx in the IMS.PROCLIB. The default is 00.

IMS= through LHTS= procedure parameters

The procedure parameters IMS= through LHTS=, which are described in the following list, can be specified in procedures in the IMS environment.

IMS=*ims_id*
Specifies the subsystem IMS ID.

IMSGROUP=
Specifies a 1- to 4-character IMS group name. This parameter is user-generated and is typically the IMSID value most often found in the BMP or other dependent region PROCLIB members. There is no default value. If no IMS group name is specified, the z/OS name token is *not* built.

The IMSGROUP= parameter must equal the same value as the IMSID of the BMP regions.

The IMSID of the CTL region should be different because it must be unique within the sysplex. Using the IMSID of the BMP regions as the IMSGROUP= value in the CTL region is to eliminate the need to add any new parameters to the BMP (and other dependent region's) JCL.

Example:

```
CTL Region 1 IMSID=IMS1, IMSGROUP=IMSB
CTL Region 2 IMSID=IMS2, IMSGROUP=IMSB
BMP Region 1 IMSID=IMSB
MPP Region 1 IMSID=IMSB
```

In the example, BMP region 1 connects to either IMS1 or 2 using z/OS names services to find the correct IMSID to use. The IMSID of the BMP JCL is used to construct the z/OS name. This example also holds true for the MPP region and any IFP regions.

The one EXEC parameter, IMSGROUP= is added to the control region procedure.

IMSID=
For IMS control region: Specifies a 1- to 4-character identifier that is a valid subsystem identifier to the operating system being used. It overrides the identifier specified at the time of system definition of the running IMS system.

For IMS dependent regions: Specifies a 1- to 4-character identifier that is a valid subsystem identifier to the operating system to which this dependent region connects. It overrides the identifier specified at the time of system definition of the running IMS system.

For IMS batch region: Specifies a 1- to 4-character IMS identifier that is used in IMS messages that are written to the system log. It overrides the identifier specified at the time of system definition of the running IMS system.

Recommendation: Specify a unique IMSID for each batch region. This helps to avoid confusion as to which region has issued a console message.

Do not use characters for the IMSID that match the beginning characters of a z/OS command. If the IMSID is the same as the beginning characters of a z/OS command, that command does not work after you start IMS. For example, if you start IMS with IMSID=D, z/OS does not respond to any of the z/OS display commands such as `D A,L`.

For online control regions, the IMSID must be different from any other IMS subsystem identifier or non-IMS subsystem identifier defined to the operating system under which IMS is running. This identifier is also used to relate messages that are routed to the z/OS system console with the corresponding

IMS system. To avoid confusion as to which region has issued a console message, specify a unique IMSID for each batch region. This is however, not a requirement.

The IMSID name must not be the same as the procedure name that starts IMS unless one of the following is true:

- All DD statements in the startup procedure are cataloged in the master catalog.
- The unit and volume are specified on each DD statement.

This parameter cannot be changed at emergency restart.

An IMSID name specified in the FDR procedure identifies the subsystem name for the FDR region.

IMSPLEX=*ccccc*
Specifies the 5-character IMSplex group name, left-justified and padded with blanks, if necessary. The IMSplex group name is passed to the SCI registration exit routine DSPSCIX0. The sample version of DSPSCIX0 shipped with IMS returns the value that you supply to DBRC as the IMSplex name.

Restriction: If you want to use Automatic RECON Loss Notification but you are not going to use DSPSCIX0, you must add the IMSPLEX= keyword and its value to the DBRC procedure. You must also manually ensure that all IMS instances that use the procedure are in the same IMSplex. This is because the z/OS START command that IMS issues internally does not include an IMSPLEX= value.

Recommendation: Use the SCI registration exit, DSPSCIX0.

IN=*input_transaction_code*
Specifies an input transaction code. This parameter is necessary only when the application program intends to access the message queues. You cannot schedule a BMP with an IN= parameter specified against a PDIR with an associated SMB that is already scheduled and is SCHDTYP=SERIAL. Doing so results in ABENDU0457 when the BMP is scheduled.

If the IN= parameter is omitted, the BMP is treated as a batch-oriented BMP.

IOB=
This parameter is no longer used. Requests for I/O are now dynamically allocated.

IOVFI=
Specifies how often the 'count of unused IOVF control intervals' is updated. This parameter sets a timer, in seconds, which triggers an IMS internal task to begin a count of unused IOVF control intervals.

The default value is 7200 seconds (2 hours). To disable the timer, specify a time of 1 second. A value of 0 sets the timer to the default value (7200 seconds). The maximum allowed value is 86400 (24 hours).

If the counter task is currently executing and the time interval expires to initiate another counter task, the second count request is ignored. The start time, stop time, and accumulate number of runs and corresponding total time for these runs are tracked in the 59FF logrec.

IRLM=Y | N
Specifies whether you want to use the IRLM during this execution. The default for IRLM= depends on the values you specify in the IRLM= and the IRLMNM= parameters of the IMSCTRL macro. If IRLMNM= is not specified

on the IMSCTRL macro or on the execution JCL and IRLM= is specified on the execution JCL, then the IRLM name used is *IRLM*.

This parameter cannot be changed at emergency restart.

Y Indicates that you are using IRLM. This is the default under either of the following conditions:

- The IMSCTRL macro specifies IRLM=Y.
- The IMSCTRL macro specifies an IRLMNM and IRLM=N is not specified.

N Indicates that you are not using IRLM. This is the default under either of the following conditions:

- The IMSCTRL macro specifies IRLM=N.
- The IRLM= and IRLMNM= keywords of the IMSCTRL macro do not specify a value.

IRLMGRP=*xcf_group_name*
Specifies the name of the XCF group to which this IRLM belongs. All IRLMs in the same group must specify the same LOCKTABL parameter and an IRLMID value that is unique within the group. The default is IRLMDS.

The group name (IRLMDS by default) is used as the XCF group name. This name cannot start with "SYS" and cannot be the same as the LOCKTABL parameter.

IRLMID=
Specifies a decimal number that is used to distinguish between IRLMs in a data sharing group. The IRLM with the lowest ID number in the group becomes the global deadlock manager for the group when you are in a data sharing module. There is no default value. A unique value must be specified for each IRLM in the data sharing group.

You can specify this parameter as either a 1- to 3-digit number from 1 to 255, or as a printable character. You must enclose a printable character with seven single quotation marks on either side of the character. Thus, you must specify the character **D** as **IRLMID="""""D"""""**. When you specify a printable character, IRLM uses the EBCDIC value of the character as the IRLMID.

IRLMNM=*name*
When used with the DXRJPROC procedure, specifies the 4-byte z/OS subsystem name assigned to this IRLM.

IRLM requires a 4-byte name for its internal processing, even though z/OS can accept names with fewer than 4 bytes for subsystem names. When used with the DBBBATCH, DBC, DLIBATCH, IMS, IMSCOBGO, IMSPLIGO, or RDI procedure, specifies the name of the IRLM started by the IMS online or batch system.

ISIS=A | C | N | R
Specifies whether resource access security checking is to be performed.

If this parameter is not specified, the default specification on the TYPE parameter of the SECURITY macro is used.

A Specifies that resource access security checking using both RACF and a user exit routine is to be performed.

C Specifies that resource access security checking using a user exit routine is to be performed.

N Specifies that no application group name security or resource access

security is to be performed. N replaces values of 0, 1, and 2. If you specify 0, 1, or 2, the system interprets them as ISIS=N.

R Specifies that resource access security checking using RACF is to be performed.

The DFSRAS00 user exit, if it exists, is loaded in order to provide the extended functionality of the exit.

This parameter cannot be changed at emergency restart.

JVMOPMAS=*name*

Sets the name of the member in the IMS.PROCLIB data set that contains the JVM options for the standalone JVM for Java dependent regions (JBPs and JMPs).

This parameter can be specified for non-Java dependent regions (MPPs, BMPs, and IFPs) also, in order to create and manage the JVM in MPP, BMP, and IFP regions.

The member name is a maximum of 8 uppercase characters.

The JVM options member must contain either of the following:

- Specify -Xoptionsfile=HFS_JVM_properties_file and then specify the -Djava.class.path=application_class_path option in the options file.

  -Xoptionsfile allows you to specify path names greater than 255 characters in length on the -Djava.class.path option.
- Specify -Djava.class.path=application_class_path directly in this member.

Specify the path name (or path names) of your IMS Java application class files. If your .class files are contained in a .jar file, the path name to the .jar file must be fully qualified, including the name of the .jar file.

Comments are supported for this options member. The comments begin with an asterisk (*) in the first column.

Each line in the options file must not be longer than 72 bytes, including the continuation mark. Use a greater-than symbol (>) at the end of the line as a continuation character.

If you do not use the -Xoptionsfile JVM option, path strings can be a maximum of 255 bytes in length (any characters over 255 bytes are ignored). A path string can be one path name or several path names. If you are specifying multiple path names, separate each by a colon (:).

The sample member, DFSJVMMS, demonstrates how to specify the JVM options for Java dependent regions (JBPs and JMPs).

JVMOPWKR=

This parameter is obsolete. If it is specified, it is ignored.

LGMSGSZ=

Specifies a 1- to 5-digit number that represents the size in bytes of a long message record. The length specified is rounded up to the nearest multiple of 4.

Restriction: The length must be greater than or equal to the length specified for the short message record or one and one-half times the maximum message prefix size plus 4. If you specify a value less than the short message record length or one and one-half times the maximum prefix size plus 4, the length of the large message size is set to the greater value between the large message DCB LRECL length, the short message size, or one and one-half times the

maximum message prefix size plus 4. The length of the long message record cannot exceed 30632 bytes. If you specify a value greater than 30632, the size is set to 30632.

This parameter is valid only in the shared-queues environment. If you do not specify a value for the SHAREDQ= parameter, the value specified for LGMSGSZ= is ignored. If you specify a value for the SHAREDQ= parameter, but not for the LGMSGSZ= parameter, the size of the long message record is obtained from whichever is greater: the DCB generated for the long message queue data set or the short message record length.

LGNR=
Specifies the maximum number of Fast Path DEDB buffer alterations that is to be held before the entire control interval is logged. The parameter is used by Fast Path to build the Fast Path buffer header (DMHR) and to control the DEDB update logging mechanism. The number of DEDB buffer alterations has no direct relationship to the number of DEDB calls. The parameter can be specified as a 1- to 2-digit numeric value. If a value less than 7 is specified, a minimum value of 7 is set.

LHTS=
Specifies a 1- to 5-digit number of LTERM hash table slots. Valid values are from 0 to 32767. The default is 256. If you specify a null or invalid value, 256 is assumed.

LOCKMAX= through MNPSPW= procedure parameters

The procedure parameters LOCKMAX= through MNPSPW=, which are described in the following list, can be specified in procedures in the IMS environment.

LOCKMAX=
Specifies a value between 1 and 32767 (in units of 1000). This parameter overrides the PSBGEN LOCKMAX value if one was specified. An override parameter of LOCKMAX=0 turns off all locking limitation.

LOCKMAX=10
Allows for 10000 locks.

LOCKMAX=0
Turns off locking limitation.

LOCKTAB=
This parameter is optional and specifies the lock table to be used by this group. The lock table must have previously been defined through XCF Resource Manager panels and must be an element of the currently active locking policy. The default is IRLMLT1.

This parameter must be the same for all IRLMs that specify the same value for the GROUP parameter.

LOGA=
This parameter is no longer used and is ignored. In earlier releases, it specified whether IMS was to use the BSAM (0) or OSAM (1) logging access method.

LOGT=
Specifies the tape device type where the log data set is to be mounted. The tape device type specified by the LOGT= parameter substitutes for the device parameter specified in the IEFRDER DD statement. The default is device type 2400.

LSO=Y | S

Specifies whether the Local Storage Option is to be used. When it is used, some IMS modules and buffers are moved from the CSA to the private storage area of the control region.

Y Specifies that LSO is to be used. This is the default.

S Specifies the DL/I subordinate address space option. This is required for a CCTL connected to an IMS control region or DBCTL region.

If LSO=Y is specified for an RSR tracking subsystem, DL/I database tracking is not initialized; only Fast Path database tracking can be performed.

LTE=

Specifies the number of lock table entries available in the coupling facility lock structure (in units of 1,048,576). LTE can have a value of blank, zero, or any exact power of 2 up to a maximum of 1024.

For example, a value of LTE=32 would result in a lock table size of 64 MB, assuming a width of 2 bytes for each lock table entry. If the value specified on LTE is incorrect, START terminates with `DXR116E CODE=24` and `ABENDU2018`. This value is only used if SCOPE=GLOBAL or NODISCON and has a default of blank (IRLM calculates the value). The number of lock table entries in the group is used in the order outlined in the list, and the width is controlled by the value specified on MAXUSRS. Both of these are dictated by the first IRLM to connect to the group during initial structure allocation or during a REBUILD:

1 The value specified on the `MODIFY irlmproc,SET,LTE=` command if greater than zero.

2 The value from the LTE= in the IRLMPROC if greater than zero.

3 The existing logic, which determines the nearest power of 2 after dividing the XES structure size returned on the IXCQUERY call by 2 times the LTE width (based on MAXUSRS).

If an attempt is made to use a nonzero value from either number 1 or number 2 and that value would require more storage than is available in the structure size returned by the XES IXCQUERY issued by IRLM, then the value from the next priority order is used (number 2 or number 3 from the previous list). IRLM does not try to determine how many record table entries the user wants. If the LTE= value consumes most or all the coupling facility lock structure size available, IRLM allows the connect to succeed.

Note: IRLM supports a maximum of 1024 MB lock entries. If a user specifies a high value for lock structure size in a CFRM policy, IRLM calculates the lock entries for the hash table depending upon the number of users (MAXUSRS) and the structure size, and limits the lock entries to a maximum of 1024 MB.

The following table illustrates some of the commonly desired values for lock table entries and the STORAGE that would be needed for the LOCK TABLE portion as a result. The storage available for record table entries is the INITSIZE minus the storage size listed in this table.

Table 65. Common lock table entry values

For LTE=	Storage if 2-byte entries	Storage if 4-byte entries
8	16 MB	32 MB
16	32 MB	64 MB
32	64 MB	128 MB

Table 65. Common lock table entry values (continued)

For LTE=	Storage if 2-byte entries	Storage if 4-byte entries
64	128 MB	256 MB
128	256 MB	512 MB
256	512 MB	1024 MB

LTERM=Y | N
Specifies whether the LTERM name of the static input terminal is used in the DFSAPPC.

Y Specifies that the LTERM is used. Y is the default

N Specifies that the LTERM is not used. If LTERM=N is specified, and a user ID is not provided or a user is not signed on, the DFSAPPC process is rejected with an error message DFS1957E.

LUMC=
Specifies a value for the upper expansion limit of the LU 6.2 device manager common buffer pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements is true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

The LUM code uses the pool for work area processing.

LUMP=
Specifies a value for the upper expansion limit of the LU 6.2 device manager private buffer pool and OTMA. The value can be specified as 1- to 6-numeric characters, or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements is true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

The LUM code uses the pool for work area processing when not processing in the dependent region.

Some of the OTMA and APPC message processing modules have been updated to use DYNP for module dynamic storage. For more information, see the description of DYNP= in DLINM= through EXVR= procedure parameters (System Definition).

MAXCSA=
Specifies the maximum amount of common service area (CSA) and extended CSA (ECSA) that the IRLM for this IMS uses for its lock control block structures. The acceptable value range is 1M - 999M. IRLM is not prevented from using additional CSA and ECSA for other purposes. You can enter the value in bytes (such as 5242880), or use the abbreviations K for kilobytes (such as 5000K) and M for megabytes (such as 5M).

When PC=YES, the MAXCSA parameter is ignored. A value of zero is allowed in IRLM 2.2.

MAXPST=
Specifies the maximum number of blocks that the user permits the online control region to allocate.

The default is 255; the maximum is 999. When system activity increases, IMS allocates PST blocks up to the maximum specified. When the workload diminishes, these blocks are unallocated and storage is released to as low as the value specified by PST=.

An example of how these parameters can be specified follows:

Example:

```
...,PARM='PST=48,MAXPST=300,...
```

Restriction: Reducing the value of the MAXPST parameter from a previous IMS start requires a cold start. Otherwise, results are unpredictable.

MAXUSRS=
Specifies the maximum number of IRLMs that are to connect to the data sharing group. A value from 2 to 248 can be specified.

The initial allocation of locking resources in XES is dependent on the number of entries and the maximum number of users, so some assessment of available resources is made when IRLM joins the group. However, nothing is done to keep the number of users below the specified value.

MBR=*name*
Specifies an application program name.

MCS=(x,y)
Specifies the z/OS routing code to be assigned to the IMS system console if multiple console support (MCS) is included in the operating system. If the parameter is not specified, the value specified in the IMSCTF macro is used.

The MCS parameter overrides the value specified on the IMSCTRL macro at system definition time.

Although z/OS supports more than 16 route codes, IMS uses only route codes 1 through 16.

MNPS=*name*
Specifies the name of the MNPS ACB to use for terminal recovery. This parameter is optional and valid only for XRF systems. If specified:

- This parameter overrides the MNPS name specified on the DFSHSBxx PROCLIB member.
- The USERVAR specification is ignored.

MNPSPW=*name*
Specifies the password to use for the MNPS ACB. If specified, this parameter overrides the MNPSPW name specified on the DFSHSBxx IMS.PROCLIB member. VTAM checks this password. If VTAM requires a password during VTAM system definition and no password is specified, then the MNPS ACB cannot be opened.

MOD= through ORSMBR= procedure parameters

The procedure parameters MOD= through ORSMBR=, which are described in the following list, can be specified in procedures in the IMS environment.

MOD=*module_name*
Specifies the module name, when available.

MON=Y | N
Specifies whether (Y) or not (N) the IMS monitor is to be active for this execution.

MSDB=*x*
Specifies the 1-character suffix for DBFMSDBx. (See "DBFMSDBx member of the IMS PROCLIB data set" on page 646.)

MSG=
Specifies the message number for which information is to be searched.

NBA=*nnnn*
A 4-digit number specifying the number of Fast Path database buffers to be made available when the Fast Path region is activated. If you are using the Fast Path 64-bit buffer manager, the buffer pools are loaded there. Otherwise, they are loaded into the 31-bit common storage area. DEDBs use these buffers to access VSAM control intervals; MSDBs and DEDBs use them to hold information for updating between the DL/I call and synchronization point.

These buffers become part of the system database resources, but the number specified is reserved for exclusive use by this region when the application program accesses the MSDBs or DEDBs. Although you are guaranteed access to a number of buffers, a maximum of the combined values of NBA and OBA, you might have to wait for them. The combined values of NBA and OBA also define the maximum number of buffers that the Fast Path 64-bit buffer manger BM can allocate to a dependent region. The default is 0.

NHTS=
Specifies a 1- to 5-digit number of terminal hash table slots. Valid values are 0 - 32767. The default is 256. If you specify a null or invalid value, 256 is assumed.

NLXB=
Specifies the number of parallel sessions to be added during the startup of an IMS system. The NLXB value is added to the value specified in the SESSION parameter during system definition to increase the number of link extension blocks (LXB) generated for each link control block (LCB). The default for NLXB is 0. If NLXB is specified when the maximum value is already specified on the SESSION parameter of the MSPLINK macro, the NLXB value is ignored. This parameter is valid for MSC with VTAM only and does not apply to Intersystem Communication (ISC).

OBA=*nnnn*
A 4-digit number that specifies the number of additional page-fixed buffers to be made available to a Fast Path application region if the normal allotment is used. If you are using the Fast Path 64-bit buffer manager, the buffer pools are loaded there. Otherwise, they are loaded in the 31-bit common storage area. In the common storage area, the system page fixes only enough additional buffers to handle the largest overflow specification from all active regions. The default is 0.

To prevent a situation in which every program is waiting for buffers when no buffers are available, the system allows only one program at a time to use its overflow allocation. If an application needs more buffers than the combined values of NBA and OBA, it abnormally ends (for message-driven programs) or an FR status code is issued (for non-message-driven programs).

When the 64-bit buffer manager is used, the use of OBA buffers is not serialized. Multiple dependent regions or threads might be using their OBA

allocations at the same time, which eliminates a potential bottleneck in buffer use. In practice, it means that each region or thread might be using the number of buffers equal to the combined values of the NBA and OBA specifications.

ODBASE=Y | N
Specifies whether (Y) or not (N) SAF security checking is to be done on an APSB request from an ODBA thread by using the AIMS resource class. The default is N.

Y Specifies that a SAF RACROUTE AUTH call is to be made. The AIMS class is used to verify that the user can access the PSB.

The DFSRAS00 user exit, if it exists, is loaded in order to provide the extended functionality of the exit.

N Specifies that no SAF calls are to be made against the AIMS resource class.

ODBMCFG=*xxx*
Specifies a 3-character suffix for the ODBM configuration parameters PROCLIB member, CSLDCxxx, which includes definitions for the ODBA connection initialization parameters and any ODBM configuration statements. The default suffix is 000.

ODBMINIT=*xxx*
Specifies a 3-character suffix for the ODBM initialization parameters PROCLIB member, CSLDIxxx. This parameter can only be specified as an execution parameter. The default suffix is 000.

ODBMNAME=*odbm_address_space_name*
Specifies the name for the ODBM address space. This is an optional 1-6 character name. If specified, it overrides the value specified in the CSLDIxxx PROCLIB member. You must specify this parameter either as an execution parameter or in the CSLDIxxx PROCLIB member.

OMINIT=
Specifies a 3-character suffix for the OM initialization parameters PROCLIB member, CSLOIxxx. This parameter can only be specified as an execution parameter. The default suffix is 000.

OMNAME=
Specifies the name for the OM address space. This is an optional 1-6 character name. If specified, it overrides the value specified in the CSLOIxxx PROCLIB member. You must specify this parameter either as an execution parameter or in the CSLOIxxx PROCLIB member. This name is used to create the OMID, which is used in OM processing. The 8-character OMID is the OMNAME followed by the characters "OM". Trailing blanks in the OMNAME are deleted, and the OMID is padded with blanks. For example, if OMNAME=ABC then OMID="ABCOM".

OPT=N | W | C
Specifies what action to take when the dependent region starts, but when no system identifier matches the names given on IMSID or ALTID. The actions are:

N Ask operator for a decision. This is the default.

W Wait for the control program to start.

C Cancel message region automatically.

ORS=
Identifies this address space as part of IMS Database Recovery Facility to DFSMVRC0 so that the Recovery Data Manager is started.

ORSMBR=*xx*
Specifies the 2-character suffix for the IMS Database Recovery Facility member (DFSORSxx) in IMS.PROCLIB that is currently in effect. There is no default.

OTHR= through PIMAX= procedure parameters

The procedure parameters OTHR= through PIMAX=, which are described in the following list, can be specified in procedures in the IMS environment.

OTHR=
Specifies the number of concurrent output threads that Fast Path is to support for the entire Fast Path system. OTHR= *n* causes a total of *n* service request blocks (SRBs) and extended service request blocks (ESRBs) to be created at system initialization. These blocks are used in scheduling asynchronous DEDB output. If an insufficient number of SRBs is specified, write buffers are queued until one becomes available.

The OTHR= parameter can have any value from 1 to 32,767. If a value is not specified, the default is 255. If `0` or a value greater than `32767` is specified, the default is 2. If a value from 1 to `32767` is specified, the specified value is used.

OTMA=Y | N | M
Specifies that the IMS Open Transaction Manager Access (OTMA) function is to be enabled during IMS initialization. Valid values are Y, N, and M. The default value is N.

Y IMS uses the security level that is currently in effect (CHECK, FULL, NONE, or PROFILE) to control the RACF security level for input from IMS OTMA clients. You can issue the /DISPLAY OTMA command to show the security level that is currently in effect.

If Y is specified, IMS attempts to create an XCF group, specified by GRNAME, during initialization and then attempts to join that group.

N IMS recovers the /START OTMA command across a warm or emergency restart. If a warm or emergency restart of IMS occurs after OTMA has been started by the /START OTMA command, OTMA is restarted when IMS is restarted, regardless of the OTMA=N specification.

M IMS does not recover the /START OTMA command across a warm or emergency restart. If a warm or emergency restart of IMS occurs after OTMA has been started by the /START OTMA command, OTMA is not restarted when IMS restarts.

If N or M is specified, OTMA is not started during IMS initialization, but can be started later by issuing the /START OTMA command.

The /START OTMA command is recovered during emergency restart; however, if a CHKPT is issued after the /START OTMA command is issued, the command is not recovered, and the /START OTMA command must be reissued after restart. The /START OTMA command is not recovered, in either case, if OTMA=M is specified.

After an IMS cold start, the security default is FULL.

The OTMA parameter is valid only in IMS DB/DC or DCCTL environments.

OTMAASY= Y | N | S
Specifies whether (Y) or not (N) a non-response transaction originating from a program-to-program switch is to be scheduled asynchronously.

The default value is N.

Y Specifies that a non-response transaction originating from a program-to-program switch is scheduled asynchronously.

By specifying Y, this parameter can be used in the multiple program-to-program switches environment to ensure that only response transactions can be scheduled synchronously.

N Specifies that a non-response transaction originating from a program-to-program switch is not scheduled asynchronously.

S Specifies that a non-response transaction originating from a program-to-program switch is scheduled synchronously.

If you specify OTMAASY=S, the first program-to-program switch that is able to continue the synchronous conversation is synchronous. Every further program switch is asynchronous.

If you specify OTMAASY=S, IMS overwrites the APPCASY value to S. If you specify APPCASY=S, IMS overwrites the OTMAASY value.

In a shared-queues environment, if you specify OTMAASY=S, the program-to-program switch transactions do not have any affinity.

Tip: No DFS2082 is issued for a transaction that is scheduled asynchronously.

OTMAMD=Y | N
Specifies whether the member override field in the parameter list of the OTMA Prerouting Exit routine (DFSYPRX0) can be used for a transaction initiated from an OTMA client. Valid values are Y (yes) or N (no). The default value is N.

OTMANM=*member_name*
Specifies the member name that IMS uses when joining the z/OS cross-system coupling facility (XCF) group for non-RSR (Remote Site Recovery) or non-XRF (Extended Recovery Facility) systems.

For IMS systems without RSR or XRF, OTMA uses the OTMANM= parameter as the IMS XCF member name. If OTMANM= is not specified, OTMA uses the non-APPC VTAM LU name (APPLID1) as the member name.

For IMS systems with RSR or XRF, OTMA uses USERVAR as the IMS XCF member name.

This parameter is only valid in IMS DB/DC or IMS TM-DB2 environments.

OTMASE=C | F | J | N | P
Specifies the type of OTMA RACF security that you want to use. Valid parameter values include:

C OTMA RACF security is CHECK. IMS commands are checked against the CIMS class. IMS transactions are checked against the TIMS class.

F OTMA RACF security is FULL. This is the same type of security as CHECK, but additional security checking is performed against dependent regions. The default value is F.

N OTMA RACF security is NONE. No calls to RACF are made.

P OTMA RACF security is PROFILE. Each OTMA message defines the level of security checking to be done.

The /SECURE OTMA command overrides the value you specify in the OTMASE= keyword.

If no value is specified, a value of X appears in the initial DFS1929I message that is issued when IMS is started. During IMS restart processing, this value is changed to F. A value of F appears in the active DFS1929I message that is issued when restart processing is complete.

If you do not specify the OTMASE keyword, IMS retains the OTMA security settings (which are established by the /SECURE OTMA command) after a warm start or an emergency restart.

OTMASP=Y | N
Specifies whether a non-synchronous Tpipe or a synchronous Tpipe is created to deliver the OTMA output. Valid values are Y (create a synchronous Tpipe) or N (create a non-synchronous Tpipe). The default value is N. If the value of this parameter is Y or if the OTMA Prerouting Exit routine (DFSYDRU0) indicates the need for a synchronous Tpipe, the synchronous Tpipe is created.

OUT=
Specifies the transaction code or logical terminal name to which an output message is to be sent. This parameter is necessary when the application program wants to send output without accessing the input queues. A remote LTERM (CNT) cannot be specified.

OVLA=0 | 1
Specifies the overlay supervisor option:

0 Allow z/OS to load and delete the overlay supervisor for every overlay application program. This is the default.

1 Load and retain a copy of the overlay supervisor when the message region is initialized.

PAGES=
Specifies the amount of primary and secondary space allocated for the SYSPRINT DD statement data set.

PARDLI=0 | 1
Specifies the parallel DL/I option.

0 DL/I processing is to be performed within the region. This is the default.

1 All DL/I processing for this region is to be performed in the IMS control region. If data capture (EXIT= on the DBD statement) is enabled and this is a delete, replace, or insert call, PARDLI=1 is ignored for the DL/I call.

PARDLI=1 prevents control region system 113 abends resulting from system X22 abends in the region. If PARDLI=1, parallel DL/I is disabled, which can degrade performance.

Important: Using PARDLI=1 for MPP, JMP, or IFP regions can degrade performance. Use PARDLI=1 for MPP, JMP, or IFP regions only for application debugging purposes, if needed.

PASSWD=
This parameter is obsolete. Use the PASSWD1 parameter.

PASSWD1=*password*
Specifies a password to override the value specified on the PASSWD= parameter in the COMM macro.

PC=YES | NO

IRLM places the lock storage in the 64-bit private address space.

Restriction: IRLM does not run with PC=NO. PC=NO can still be specified in the startup procedure, but IRLM sets PC=YES during initialization.

PCB=
Specifies the size of the dependent region interregion communication area (DIRCA); the size specified must be a 3-digit number (for example, *001*) representing the number of 1 KB blocks of subpool 251 to be reserved to hold a copy of your PCBs.

If you do not specify a DIRCA size or if you specify a size of *000*, IMS uses the system default for this message region. The size of the system default, determined during system initialization, is the maximum size required for any PSB. If necessary, this system default is updated when an online change is made or when a dynamic PSB is scheduled.

If you specify this parameter, the specified size is always used. If the size required is larger than the specified size, the region abends with a user 0242 abend code.

Normally, you should not specify this parameter. You can use it to reduce the DIRCA size for a region when PSBs with small DIRCA requirements are scheduled into this region.

The output from message DFS589I occurring during ACBGEN facilitates DIRCA size calculation. Each PSB has a PCB= value that is the sum of the PCBs within it. The PCB value + 64 becomes the required DIRCA in bytes.

PGPROT=YES | NO
Specifies whether the IRLM is to place load modules that are resident in common storage into z/OS Page Protected Storage during initialization. If the load modules are in Page Protected Storage, any application that attempts to overlay the modules is terminated. The default value is YES.

PIINCR=
Specifies the increment to be used in a conditional GETMAIN to obtain the dynamic storage. The default value is 64K.

No storage is allocated for ENQUEUE/DEQUEUE blocks at initialization. The first block of storage, obtained when the ENQUEUE/DEQUEUE routine is first called, is in the increment specified in PIINCR. Storage continues to be obtained in the increment specified until the maximum is reached (see PIMAX=). IMS does not release any storage that is dynamically obtained for the ENQUEUE/DEQUEUE routine. After storage is obtained, it is kept for the duration of the control region execution.

PIINCR can be specified as either 1- to 6-numeric characters, or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements is true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

PIMAX=
Specifies the maximum amount of dynamic storage available to the exclusive control of the ENQUEUE/DEQUEUE routine. The default value is 1024K (1M).

No storage is allocated for ENQUEUE/DEQUEUE blocks at initialization. The first block of storage, obtained when the ENQUEUE/DEQUEUE routine is first called, is in the increment specified by PIINCR=. Storage continues to be obtained in the increment specified until the maximum is reached. IMS does not release any storage that is dynamically obtained for the ENQUEUE/DEQUEUE routine. After storage is obtained, it is kept for the duration of the control region execution.

PIMAX can be specified as either 1- to 6-numeric characters, or 1- to 5-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2G-1. The upper limit defaults to 2G-1 if any of the following statements is true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.

PRDR= through QTU= procedure parameters

The procedure parameters PRDR= through QTU=, which are described in the following list, can be specified in procedures in the IMS environment.

PRDR=*name*
Specifies the name of the IMSRDR procedure in IMS.PROCLIB, or SYS1.PROCLIB, used in /START REGION commands. The name can be up to 8 characters in length. The default is IMSRDR.

PREINIT=*xx*
Specifies a 2-character suffix for DFSINTxx, the IMS.PROCLIB member that lists the preinitialization modules that are to receive control. For details on how to define the member, see "DFSINTxx member of the IMS PROCLIB data set" on page 714.

PREMSG=Y | N
Specifies whether (Y) or (N) you want to receive or suppress the DFS000I prefix message that precedes all DBCTL system messages and command responses. Y is the default.

If the DFS000I prefix message is suppressed (PREMSG=N), all system messages are issued with the IMSID appended to the last line. The message can be a single line with a maximum of 121 characters of message text followed by *XXXX*, where *XXXX* is the IMSID. Or the message can be multiple lines with each line containing a maximum of 71 characters of message text. The IMSID is appended to the last line of a multi-line message or is the only text of a final line.

Display command output is prefaced by a prefix message:

```
DFS4444I DISPLAY FROM ID=XXXX
```

where XXXX is the IMSID. The message text is in columns 2-71 of all subsequent lines. Column 1 of display output contains + if the message text continues to the next line; otherwise, column 1 is blank.

If PREMSG=Y (the default value) is selected, all system messages and command responses are issued as multi-line messages.

The first line is:

```
DFS000I MESSAGE(S) FROM ID=XXXX
```

where XXXX is the IMSID. The message starts on the second line.

PRLD=*xx*

Specifies a 2-character suffix for DFSMPLxx, the IMS.PROCLIB member identifying the modules to be preloaded in the region or partition. For more information about making program modules resident in a region, see "DFSMPLxx member of the IMS PROCLIB data set" on page 716.

PSB=

For batch procedures: An optional parameter specifying a PSB name when the PSB name and application program name are different.

The use of generated PSBs (GPSB) is supported in a TM batch environment. If the application program requires a GPSB, it must use the PSB parameter to indicate that request. In this case the PSB parameter does not specify the name of the PSB. Instead, it specifies a code that requests the use of a generated PSB and its language type. The specification of the code as the PSB parameter causes the specified GPSB to be used for the batch application program. The application program name (MBR parameter) is used as the GPSB name. GPSBs are not available in DB batch.

The following coded character strings are used to identify the use of a GPSB and its language:

DFS$$ASM
GPSB for the assembler language format.

DFS$$COB
GPSB for the COBOL language format.

DFS$$PLI
GPSB for the PL/I language format.

DFS$$PAS
GPSB for the Pascal language format.

Note: The coded character strings (DFS$$xxx) are valid only in an IMS DCCTL environment.

For online procedures: Specifies the amount of subpool 231 storage to be allocated to the PSB pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2G-1. If the value specified exceeds 2G-1 the default is 2G-1. The value specified is rounded up to the nearest page boundary. The default is 0.

The output of the ACBGEN utility indicates the maximum PSB size and the size of each PSB processed. This output should be examined before specifying the PSB= parameter.

For the FDR procedure, when LSO=S is specified, the sum of the values for the CSAPSB and DLIPSB defines the PSB pool size. If PSB is also specified, the larger value (PSB or the sum of CSAPSB and DLIPSB) is used.

PSBW=

Specifies the amount of subpool 231 storage to be allocated to the PSB work area pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2G-1. If the value specified exceeds 2G-1 the default is 2G-1. The pool size specified is rounded up to the nearest page boundary.

The output of the ACBGEN utility indicates the maximum work area size among the work area sizes required by each PSB. Depending on the execution environment, the work area size for a given PSB may be increased to the size of the long message queue buffer or to the size specified plus the maximum segment size of the segments processed by this PSB. The ACBGEN output should be examined before specifying the PSBW= parameter

This parameter is not needed for an RSR tracking subsystem.

For CSAPSB and PSBW, DFSIINS0 obtains contiguous space in the z/OS common area. If this storage is unavailable in the z/OS common area, ABENDU0717 occurs.

PST=

Specifies a 1- to 3-digit number of partition specification tables (PSTs) to be allocated during system initialization.

Each active dependent region, DBCTL thread, or ODBA thread uses a PST. If additional dependent regions or threads are required during peak periods, IMS dynamically allocates additional PSTs as the dependent regions or threads are started. PSTs can be allocated up to the MAXPST value. During idle periods, the PSTs are released. The number of PSTs specified on PST= is maintained. For example, if PST=15, it is assumed that approximately 15 dependent regions or threads are always active. During peak periods, if additional dependent regions or threads are started, IMS dynamically allocates a PST for each dependent region or thread, up to the MAXPST value. As PSTs are terminated, the PSTs are released until a value of 15 PSTs is reached. This support is within the constraints of the operating system or subsystem being used. The default is 0.

PSWDC=M | U | R

Specifies whether mixed-case passwords are supported.

M IMS supports the use of mixed-case passwords. If you intend to support mixed-case passwords, be aware of this support wherever you manipulate passwords, such as in exit routines.

U IMS forces all password to uppercase.

R IMS uses whatever is defined for mixed-case passwords in RACF. If mixed-case passwords are active in RACF (which is done through the SETROPTS command) then IMS uses it. If mixed-case passwords are not active in RACF, then IMS uses uppercase passwords. Whenever there are changes to the mixed-case password definition in RACF, IMS adjusts without requiring a restart. R is the default.

PWFI=Y | N

Specifies the pseudo-wait-for-input (PWFI) parameter. N is the default.

Y Activates pseudo wait-for-input (PWFI). If the application program issues a Get Unique (GU) call to get a message from the IMS message queue, MODE=SINGLE is specified on the TRANSACT macro and no message is available, IMS checks for other work for this region. If no other work is available, instead of returning status QC to the application program, IMS enqueues this region on scheduler subqueue 6 indicating that it is a PWFI region. If the next message is for the transaction scheduled in this MPP region, this region is dequeued from subqueue 6 and posted. IMS then returns the new message to the application program. This eliminates the rescheduling of IMS resources. If the processing limit (PROCLIM) for the transaction is reached, the

application receives a QC status and is expected to terminate. After the application program terminates, it is reloaded on the subsequent schedule.

Related reading: For more information about PWFI, see *IMS Version 11 System Administration*.

N Disables PWFI. If the program issues a Get Unique (GU) call to get a message from the IMS message queue but no message is available, the application program receives status QC.

QBUF=
Specifies the 1- to 4-digit number of message queue buffers in subpool 0 to be allocated to the queue pool. In the shared-queues environment, the number you specify for QBUF= is used as the initial number of message queue buffers allocated to the queue pool. The number of message queue buffers is dynamically expandable in the shared-queues environment. The minimum number you can specify is 3, and the maximum is 9999. If you specify a value less than 3, the number of buffers in the queue pool is set to 3. If you do not specify a value, the number defaults to the value generated during system definition for the BUFFERS= parameter of the MSGQUEUE macro.

QBUFHITH=
Specifies a 1- to 3-digit number from 1 to 100 that establishes the high threshold percentage for the message queue buffer. When the buffer reaches the high threshold, it is dynamically expanded. If you do not specify a value for this parameter, the default is 80%.

QBUFLWTH=
Specifies a 1- to 3-digit number from 1 to 100 that establishes the low threshold percentage for the message queue buffer. When the message queue buffer reaches the low threshold, it is compressed. Compression only occurs when the number of message queue buffers exceeds the number allocated by system definition. If you do not specify a value for this parameter, the default is 50%. This parameter is valid only in the shared-queues environment.

QBUFMAX=
Specifies the 1- to 4-digit number that indicates the maximum number of message queue buffers for the queue pool. The minimum number you can specify is 200, and the maximum number is 9999. If you specify a value less than 200, the number of buffers in the queue pool is set to 200. If you specify the parameter without a value (QBUFMAX=,) the default of 9999 is taken. If this parameter is not specified, the number of buffers in the queue pool is unlimited.

This parameter is only valid in a shared-queues environment. If you do not specify a value for the SHAREDQ= parameter, this parameter is ignored.

QBUFPCTX
Specifies a 1- to 3-digit number in the range of 1 to 100 that establishes the percentage of the originally allocated message queue buffers that are dynamically expanded when the limit specified by the QBUFHITH= parameter is reached. The default for this parameter is 20%.

QBUFSZ=
Specifies a 1- to 5-digit number that represents the size in bytes of the message queue buffers (in core buffers) used by the queue manager. You must specify a size greater than or equal to the length specified by LGMSGSZ= The size must be equal to or less than 30632 bytes which is the maximum length of the long message record. The size specified is rounded up to the nearest multiple of 4.

If you specify a value greater than 30632, the size is set to 30632. If you do not specify a value, or if you specify a value less than the length of the long message record, the size is obtained from the record length of the long message record. The DCBs for the message queue data sets are built during IMS system definition processing, even though the data sets are not used.

This parameter is only valid in the shared-queues environment. If you do not specify a value for the SHAREDQ= parameter, this parameter is ignored. If you do specify a value for the SHAREDQ= parameter, the value specified in the QBUFSZ= parameter is used to determine the size of the message queue buffers.

QTL=
For a non-shared-queues environment, specifies the lower threshold percentage (1 to 99) for each message queue data set. The default is 60%. If this threshold is crossed and if the associated exit has not been modified, a message is issued.

For a shared-queues environment, specifies the lower threshold percentage (1 to 99) for the device relative record number (DRRN) in-use count. The default value is 60%. If this threshold is crossed, and if the high threshold count was reached, a message is issued and all messages in an IWAIT state are posted.

QTU must be higher than QTL, and 0 is not valid. Either error results in the defaults being used.

QTU=
For a non-shared-queues environment, specifies the upper threshold percentage (2 through 100) for each message queue data set. The default is 75%. If this threshold is crossed, and the associated exit routine has not been modified, a message is issued.

For a shared-queues environment, specifies the upper threshold percentage (2 to 100) for the DRRN in-use count. The default value is 75%. If this threshold is crossed, and if the high threshold count was reached, a message is issued and all messages being retrieved from shared queues are put in an IWAIT state until the lower threshold is reached.

RC= through RVFY= procedure parameters

The procedure parameters RC= through RVFY=, which are described in the following list, can be specified in procedures in the IMS environment.

RC=
Specifies the message number for which information is to be searched.

RCF=A | C | N | S | T | Y
Specifies whether RACF is to be used for transaction authorization or signon authorization checking.

A Includes options T, C, and S.

C Specifies that RACF is to be used for ETO terminal signon and command authorization.

N Specifies that no signon, transaction, or command authorization is to be performed by RACF.

S Specifies that RACF is to be used for static and IMS ETO Support terminal command authorization. If signon verification is not explicitly specified during IMS system definition (or with the SGN= parameter), signon verification security is set on by this specification of command authority.

T Specifies that RACF is to be used for signon and transaction authorization.

Y Includes options T and C.

Changes to the value of this parameter require an IMS cold start.

If the RCF parameter is not specified, the default is the value specified at system definition.

RCFTCB=
Specifies a number between 1 and 20 indicating how many RCF TCBs are defined in your system. If you specify a null value or an invalid value, IMS uses the default of 1 RCF TCB. Increasing the number of RCF TCBs allows you to customize your system to achieve maximum parallelism and improved performance during RACF sign on and sign off calls.

RCLASS=*identifier*

Specifies an identifier of 1 - 7 alphanumeric characters that is to be used to identify the IMS system as a resource class to RACF for transaction authorization and user ID verification. The default identifier is "IMS".
IMS class names do not have to be unique. If you do make them unique, you can, for example, define the same transaction name on a production subsystem and a test subsystem, with different access lists for each subsystem, with no ambiguity.
The specification is valid only if one of the following statements is true:

- TYPE=RACFTERM is specified on the stage-1 system definition SECURITY macro; or RCF=A, RCF=T, or RCF=Y is specified on the IMS procedure.
- TYPE=RASRACF or TYPE=RAS is specified on the SECURITY macro; or ISIS=R or ISIS=A is specified on the IMS procedure.

The RCLASS= specification on the DFSPBxxx PROCLIB member overrides the RCLASS= specification on both the SECURITY macro and the DFSDCxxx member. The RCLASS= specification on the DFSDCxxx PROCLIB member overrides the RCLASS= specification on the SECURITY macro.

RDMNM=*name*
Specifies the Recovery Data Manager PROCLIB member name. If this optional parameter is not specified, the default RDM is used. If the Recovery Data Manager PROCLIB member is not found, IMS Database Recovery Facility uses default values.

READNUM=
Specifies the number of input devices that are to be used for IMS Database Recovery Facility. If input data resides on tape, READNUM specifies the maximum number of tape drives that can be concurrently allocated for IMS Database Recovery Facility. The allowable range is 1 through 99. The default value is 3.

RECA=
Specifies the number of receive-any buffers (1 to 500). This parameter overrides what is specified in the RECANY= parameter on the COMM macro.

If MNPS is used for XRF, the RECA= specification indicates the number of buffers for the MNPS ACB. IMS automatically allocates one additional buffer for the APPLID ACB.

RECASZ=
Specifies a 1- to 5-digit number indicating the size of the receive-any buffers. Valid values are from 112 to 30720. If you specify a null or invalid value, IMS

uses the value specified in the RECANY parameter of the COMM macro used in the IMS definition. If that value is 0, IMS uses a default value of 2100.

RES=Y | N
Specifies whether (Y) or not (N) the PSBs or DMBs defined in system definition macros APPLCTN or DATABASE as RESIDENT should be made resident during system initialization. The default is Y.

REST=*nn*
Specifies a 2-digit number. A number greater than 00 indicates that the utility is to be restarted. A specification of 00 indicates no restart.

REST=*nn* specifies from which set of utility control statements starts processing. Values 00 and 01 start with the first set. Sets of utility control statements with lower values than that specified by *nn* are not processed. A set of control statements is defined by the GO statement.

RGN=
Specifies the region size for this execution. For Fast Path, the default is 100K.

If TLIM is specified as a number greater than 1, the RGN symbolic parameter must be increased approximately 50K bytes to allow virtual storage overhead for handling RE-IN-STATE situations. Insufficient virtual storage specification results in an S106 abend of the IFP region.

RGSUF=*xxx*
Specifies a 3-character suffix for DFSPBxxx. This member contains default values for JCL EXEC statement parameters. It minimizes the number of EXEC parameter overrides needed so the EXEC parameter string does not exceed the 100 byte limit.

RMINIT=
Specifies a 3-character suffix for the RM initialization parameters PROCLIB member, CSLRIxxx. This parameter can be specified only as an execution parameter. The default suffix is 000.

RMNAME=
Specifies the name for the RM address space. This is an optional 1-6 character name. If specified, it overrides the value specified in the CSLRIxxx PROCLIB member. You must specify this parameter either as an execution parameter or in the CSLRIxxx PROCLIB member. This name is used to create the RMID, which is used in RM processing. The 8-character RMID is the RMNAME followed by the characters RM. Trailing blanks in the RMNAME are deleted, and the RMID is padded with blanks. For example, if RMNAME=ABC then RMID="ABCRM".

RRS=Y | N
Specifies whether (Y) or not (N) the registration and connection to z/OS Resource Recovery Services (RRS) should be made. The default is N.

When using the RRS feature, be aware that the optional RRS archive log stream typically generates large amounts of additional logging to the z/OS logger. If you use RRS Archive Logging, monitor it closely for system log stream performance impact.

RSRMBR=*xx*
Specifies (for RSR systems only) the 2-character suffix for an RSR member DFSRSRxx in IMS.PROCLIB. The default is 0.

RST=0 | 1
Specifies UCF restart. A value of 0 (no) or 1 (yes) must appear in the generated JCL statement for this parameter.

Related reading: For more information about UCF restart, see *IMS Version 11 Database Utilities*.

RVFY=Y | N
Specifies whether password reverification is activated (Y) or not (N). The default is N.

SAV= through SUF= procedure parameters

The procedure parameters SAV= through SUF=, which are described in the following list, can be specified in procedures in the IMS environment.

SAV=
Specifies a 1- to 3-digit number of dynamic save area sets for communication terminal I/O requests. IMS expands the number of SAPs up to 10 times this number as it needs to process the work being given to the system.

You can use the following formula for calculating the number of SAPs that your IMS system requires:

Formula for calculating the number of SAPs:

```
((# of VTAM terminals + 39) ÷ 40)
+ ((# of MSC links + 1) ÷ 2)
+ 1 (if VTAM required) + 60 (if ETO defined)
+ 4 (base ITASKs)
```

Valid values are from 1 to 999.

Note: Message DFS0769I could be issued while IMS is expanding the number of SAPs. If this message is issued, see *IMS Version 11 Messages and Codes, Volume 1: DFS Messages* for additional information.

SCIINIT=*xxx*
Specifies a 3-character suffix for the SCI initialization parameters PROCLIB member, CSLSIxxx. This parameter can be specified only as an execution parameter. The default suffix is 000.

SCINAME=*scimbrname*
Specifies the name for the SCI address space. This is an optional 1- to 6-character name. If specified, it overrides the value specified in the CSLSIxxx PROCLIB member. You must specify this parameter either as an execution parameter or in the CSLSIxxx PROCLIB member. This name is used to create the SCIID which is used in SCI processing. The 8-character SCIID is the SCINAME followed by the characters "SC". Trailing blanks in the SCINAME are deleted and the SCIID is padded with blanks. For example, if SCINAME=ABC then SCIID="ABCSC ".

SCOPE=LOCAL | GLOBAL | NODISCON
Specifies whether intersystem sharing is to be performed. If SCOPE=LOCAL is specified, sharing is limited to intrasystem, and XCF and SLM are neither required or used. If SCOPE=GLOBAL or SCOPE=NODISCON is specified, intersystem sharing is performed, and both XCF and SLM are required. This parameter must always be specified because no default value exists.

With NODISCON, there is less impact on other systems when an IMS fails because z/OS is not required to perform certain recovery actions when IRLM DISCONNECTS from the group. NODISCON can also mean that IMS restarts more quickly after an IMS terminates normally or abnormally because it does not have to wait for IRLM to rejoin the IRLM data sharing group. Under

normal situations, IRLM Disconnects from the group ONLY when there are no IMS systems identified to it. NODISCON has no affect as long as there is at least one IMS identified to the IRLM.

Sharing can be performed between up to 248 IRLMs on a single z/OS image. This can be done for pre-installation testing. In this case, SCOPE=GLOBAL must be specified for all IRLMs.

SGN=F | G | M | N | Y | Z
Specifies whether the signon verification function is active.

The possible values are:

F Specifies that the MTO cannot negate the activation of the signon verification function.

G Includes options F and M.

M Specifies that a single user ID can sign on to multiple terminals. If the user structure name is different than the user ID for an IMS ETO Support user (DFSSGNX0 exit returned a user structure name that is not the same as the user ID), SGN=M must be specified for this user to be able to have multiple sign ons. SGN=M does not activate the signon verification function.

The SGN= parameter has ramifications when specified on the first IMS to become active in an IMSplex. The SGN= value specified on that first active IMS is used (and enforced) for the entire IMSplex unless overridden by a /NRESTART or /ERESTART command. For example, if the first IMS to connect to the IMSplex has single sign on specified, this particular IMSplex requires all IMS systems that later connect to this IMSplex to also have single sign on specified. Similarly, if the first IMS to connect to the IMSplex has multiple sign on specified, this particular IMSplex requires all IMS systems that later connect to this IMSplex to also have multiple sign on specified.

If IMS systems try to join the IMSplex and they have a different value specified on the SGN= parameter, IMS issues an error message. For example, if IMSA connects to IMSplex1 first (with SGN=G) and then IMSB connects to IMSplex1 (with SGN=Z), an error message is issued.

If all the IMS systems in the IMSplex leave that IMSplex, then the first IMS to rejoin the IMSplex sets the SGN= value.

Important: Implementing multiple sign on (SGN=M) requires a cold start of IMS.

N Specifies that the signon verification function is not to be activated unless overridden by the MTO. For a cold start, this implies that signon verification security is not going to be in effect.

Y Specifies that the signon verification function is to be activated unless overridden by the MTO.

Z Includes options Y and M.

SHAREDQ=*xxx*
Specifies a 3-character suffix for the shared queues IMS.PROCLIB member, DFSSQxxx. When you specify this parameter, IMS uses the Common Queue Server (CQS) to place messages on the shared queues that reside in a coupling facility structure. If you do not specify this parameter, messages are placed on the message queue data sets.

There is no default for this parameter, it must be explicitly specified.

SHMSGSZ=
Specifies a 1- to 5-digit number that represents the length in bytes of a short message record. The length of the short message record cannot exceed 30632 bytes. The length must also be greater than or equal to the number of bytes in one and one-half times the maximum message prefix length plus 4. If you specify a smaller value, the message size is set to the greater value between the size in the DCB LRECL and one and one-half times the maximum message prefix size plus 4. If you specify a value greater than 30632, the size is set to 30632. If the specified value is usable, it is rounded up to the nearest multiple of 4.

This parameter is only valid in the shared-queues environment. If you do not specify a value for the SHAREDQ= parameter, the SHMSGSZ= parameter is ignored. If you specify a value for SHAREDQ=, but not for SHMSGSZ=, the size of the short message record is obtained from the DCB generated for the short message queue data set. The DCBs for the message queue data sets are built during the IMS system generation process even thought the data sets are not used.

Choose values for the SHMSGSZ= and the LGMSGSZ= based on the sizes of the messages processed by the queue manager. When determining which size message queue record to use, the queue manager calculates the size of the message prefix, subtracts that value from the size of the short message queue buffer and then doubles the remaining value. It then compares that value to the average user data length for the destination. If the remainder buffer size is greater than or equal to the average user data size, then the short message queue buffer is used. If the remainder buffer size is less than the average user data size, then the large message queue buffer is used. The length of the message prefix varies based on the IMS system options specified.

Related reading: For a list of message prefix lengths see Table 56 on page 370.

SOD=
Specifies a 1-character class for SYSOUT to be used for the spin-off main storage dump. If omitted or specified as 0, no spin-off dump is taken.

SOUT=
Specifies the class assigned to SYSOUT DD statements. The default is A.

SPAP=
This parameter is no longer used.

SPIE=0 | 1
Specifies the SPIE option to:

0 Allow your SPIE (the SPIE established by the application program), if any, to remain in effect while the application program call is being processed.

1 Negate your SPIE while the application program call is being processed. Negated SPIEs are reinstated before returning to the application program.

Under z/OS, if SPIE=1, a SPIE must be established; otherwise, system abend ABEND46D is issued.

A value of 0 or 1 must appear in the generated JCL statement for this parameter.

Related reading: For more information about IMS and SPIEs, see the topic, "How PL/I-IMS Error Handling Operates" in *OS PL/I Version 2 Programming Guide, SC26-4307.*

SPM=*xx*
Specifies a 2-character suffix for the storage pool manager PROCLIB member, DFSSPMxx.

SRCH=0 | 1
Is the module search indicator for directed load.

0 Standard search

1 Search JPA and LPA before searching PDS

SSM=

Specifies a 1- to 4-character identifier for a member that identifies the external subsystems that IMS connects to. When building IEBUPDTE JCL, you must generate the member name by concatenating this SSM identifier to the IMSID. You can specify the values for SSM= with alphabetic characters (A - Z), $, #, or @, and numeric digits (0 - 9).

An SSM PROCLIB member can contain either positional format statements in a fixed order or keyword format statements in any order:

Positional format
Fixed parameter format. Keywords are not allowed. Values must be entered in the following order: *ssn, lit, esmt, rtt, reo, crc*.

For example,

```
V10A,SYS1,DSNMIN10,RTTDB2A,R,-
```

Keyword format
Keyword-only. Values can be specified in any order. If COORD is specified, only SST= and SSN= may also be specified; all other keywords are ignored.

```
SST=,SSN=,LIT=,ESMT=,REO=,CRC=
SST=,SSN=,COORD=
```

For example,

```
SST=DB2,SSN=DB21,LIT=SYS1,ESMT=DSNMIN10,REO=R,CRC=-
CRC=-,REO=R,ESMT=DSNMIN10,LIT=SYS1,SSN=DB21,SST=DB2
SST=DB2,SSN=SYS3,LIT=MQM3,ESMT=CSQQESMT
SST=DB2,SSN=V71A,COORD=RRS
```

Restriction: A single subsystem definition cannot use both positional and keyword parameters simultaneously.

All external subsystems to be accessed by any region must be defined in this member. Essentially, the member specified by the IMS procedure is a master list. Dependent region procedures (DFSMPR, IMSBATCH, and IMSFP) can define subset lists. Each subset list can define all, some, or none of the external subsystems defined in the IMS master list. The SSM entry in each dependent region controls that region's access to subsystems.

Specifying SSM in a dependent region procedure but not in the IMS procedure is invalid, and no connection is made.

- To allow a region to access all subsystems defined in the IMS procedure, do not code SSM. However, if preferred, the SSM entry can specify the same member as the IMS procedure.

- To allow access to selected subsystems, specify a member that defines only those subsystems a region is allowed to access. For example, if the IMS procedure member defines subsystems A, B, and C, but a region is only to access A and C, the region's SSM should specify a member defining only A and C. Thus, the SSM member is a subset of the IMS procedure member.
- To prevent this region from accessing any subsystem, specify a member that contains no entries.

STIMER=0 | 1 | 2

Important: The following description applies to message-driven programs only.

Specifies the processor time statistics to be gathered.

0 No processor time statistics are to be gathered. No STIMER/TTIMER sequence is issued.

1 No DL/I processor time is to be included in the processor time statistics. An STIMER/TTIMER sequence is issued once for each program invocation and once for each DL/I call.

Specify this value for a BMP when you want to gather statistics. Specifying STIMER=1 for a BMP or IFP is like specifying STIMER=2 for an MPP. The overhead associated with it is a STIMER macro, a TTIMER macro, a load instruction, and a subtract register instruction per BMP execution.

2 If the LSO=S region startup parameter has been selected, processor time statistics include time durations for:

- Application program time
- DL/I processing time
- CPU time for ESAF users such as DB2 for z/OS that continue to process under the TCB of the dependent region. If DB2 for z/OS switches to another TCB, which is the case with parallelism, logging, and prefetch, this time is not included.

One STIMER/TTIMER sequence is issued for each program invocation. This is the default.

If you use the IMS timing services STIMER=1 or STIMER=2, you should ensure that your application programs do not use or invoke SVC2E or SVC2F (STIMER or TTIMER). Specifying STIMER=2 is invalid for an IFP region. If these services are invoked, they negate the function of controlling application programming loops, and invalid time statistics might be gathered. Additionally, ABENDU240 might occur. You should also be aware that the LSO option you select affects the meaning of the processor time statistics. If LSO= is specified, most DL/I processing time is not included in the statistics.

SUF=*n*

Specifies the 1-character suffix for the control program name. This suffix allows multiple copies of the IMS nucleus to reside on IMS.SDFSRESL.

This parameter cannot be changed at emergency restart.

SVC2= through TMINAME= procedure parameters

The procedure parameters SVC2= through TMINAME=, which are described in the following list, can be specified in procedures in the IMS environment.

SVC2=

Specifies the Type 2 SVC number that IMS uses. If the parameter is not specified, then the value specified in the IMSCTF system definition macro is used.

The SVC2 parameter overrides the value specified on the IMSCTF macro at system definition time.

SVSODR=NONE | AUTO | DRRS | WTOR

This parameter is optional. It specifies options for processing shared VSO areas at a remote site. The options take effect during an emergency restart when IMS connects to and creates a structure for a shared VSO area.

NONE

Specifies that processing is unchanged from existing shared VSO emergency restart processing. This is the default.

AUTO

Specifies that during emergency restart, if IMS connects to a new structure for a shared VSO area, that area is automatically marked as recovery needed in the RECON data set. If the connection is made to a failed persistent structure, processing continues as normal.

DRRS This option is specified for a system that is to be restarted at the disaster recovery remote site. Typically, when an emergency restart occurs at a remote site, the connection creates a structure and the areas need to be marked as recovery needed in the RECON data set. However, when the DRRS option is specified, IMS recognizes that the emergency restart is at a remote site and processes accordingly. During area-open processing for a shared VSO area, IMS does not attempt to connect to shared VSO structures. The areas are marked as recovery needed in DBRC. Specifying this option saves the overhead of doing a connect and then a disconnect for each shared VSO area.

WTOR

Gives the user the option of marking shared VSO areas as recovery needed. During an emergency restart, IMS connects to a shared VSO structure. If the connection creates a structure, a WTOR is generated. The prompt then gives the user the option of continuing to process with the new structure or marking the area as recovery needed. Based on the reply to the WTOR, IMS processes accordingly. A WTOR is issued for each shared VSO area that is opened during an emergency restart when the connection creates a structure.

The WTOR message is:

```
DFS2853A A NEW STRUCTURE WAS CREATED FOR AREA AAAAAAAA.
REPLY 'C' TO CONTINUE OR 'R' TO RECOVER
```

Note: These options can also be specified at a local site.

SWAP=Y | N

Makes address space swappable (Y) or nonswappable (N). The default is Y.

If you are using DBRC, you might want to choose option N because of the possibility of being swapped out while holding the reserve on RECON.

If you are using IRLM for block-level data sharing, the SWAP parameter has no effect. If you specify IRLM=Y, a batch job is always made nonswappable to prevent the job from obtaining locks for resources and then being swapped out.

SYS=
Specifies an optional second-level dsname qualifier for those data sets designated as “mandatory shared” in an XRF complex. When specified, the operand must be enclosed in single quotation marks and must include a trailing period.

Example: SYS='IMSA.'

SYSID=
Specifies the SYSID of the subsystem that failed with the given abend code.

SYS1=
Specifies an optional second-level dsname qualifier for those data sets designated as “mandatory replicate” in an XRF complex. When specified, the operand must be enclosed in single quotation marks and must include a trailing period.

Example: SYS1='IMSA.'

SYS2=
Specifies an optional second-level dsname qualifier for those data sets designated as “optional replicate” in an XRF complex. When specified, the operand must be enclosed in single quotation marks and must include a trailing period.

Example: SYS2='IMSA.'

T=S | U
Specifies the abend type. Valid values are S for system abends and U for user abends.

TCORACF=Y | N
Specifies whether (Y) or not (N) RACF should be called to perform an authorization check of commands from a TCO script. The default is N.

Because the TCORACF specification is not included in a checkpoint record, you can change the TCORACF value each time IMS is initialized.

TEST=0 | 1
Specifies whether (1) or not (0) the addresses in your call list should be checked for validity. A value of 0 or 1 must appear in the generated JCL statement for this parameter. An address is invalid if it is either lower than the lowest address not in the z/OS nucleus or higher than the highest address in virtual storage of the machine.

TLIM=*nn*
Specifies a 2-digit termination limit option with a decimal number between 01 and 99. When the number of application program abends reaches this limit, the message region is automatically terminated. The accumulated number of abends for a region is incremented only if the transaction is not requeued. TLIM is not incremented in the case of a pseudoabend. This allows z/OS to print the accumulated SYSOUT data sets.

The TLIM default for an IFP region is 01. For all other regions, the default is 00. If 00 is specified, no limit on program abends is set, and the message region is not automatically terminated.

For an application defined as PGMTYPE=BATCH in the APPLCTN macro statement, the optional parameter is ignored if specified.

TMINAME=*name*
Specifies the Transport Manager Subsystem (TMS) instance name the batch job

is to use. If TMINAME= is not specified in the DBBBATCH procedure, the default TMI name is the TMI name specified in the IMSCTRL macro, or blanks if no TMI name is specified.

TRACE= through YEAR4= procedure parameters

The procedure parameters TRACE= through YEAR4=, which are described in the following list, can be specified in procedures in the IMS environment.

TRACE=NO | YES
Specifies whether the IRLM is to turn on traces. Tracing is initialized at IRLM startup. The default is TRACE=NO.

NO Turns off the high activity traces in IRLM. However, IRLM always keeps the low-activity subtraces on by default.

YES Turns on all internal traces in IRLM.

z/OS TRACE CT commands can be used to turn on or off individual IRLM traces, except the RQH, EXP, RLE, INT traces, which are always on.

TRACK=NO | RLT | DLT
Specifies the level of recovery tracking for the RSR tracking subsystem.

NO Used by the IMS active subsystem and XRF alternates subsystems. NO specifies that no recovery tracking be done. This is the default, so existing procedures do not need to be changed.

RLT
Used by the RSR tracking subsystem. RLT specifies that recovery-level tracking be done.

DLT
Used by the RSR tracking subsystem. DLT specifies that database-level tracking be done.

TRN=F | N | Y
Specifies whether transaction authorization checking is to be performed. If signon verification security is not explicitly specified during IMS system definition or with the SGN= parameter, it is set on by the specification of transaction authorization.

F Specifies that the MTO cannot negate transaction authorization checking when issuing the /NRESTART command.

N Specifies that the MTO can optionally enable transaction authorization checking by specifically requesting transaction authorization checking on the /NRESTART command.

Y Specifies that the MTO can optionally override transaction authorization checking by specifically requesting no transaction authorization checking on the /NRESTART command.

TSR=U | L
Specifies the chosen time stamp representation.

U The time stamp extension to the I/O PCB contains UTC.

L The time stamp extension to the I/O PCB contains local time. The default is L extension to the I/O PCB. If you specify L, local time becomes the date extension to the I/O PCB.

The default is L.

UHASH=

This parameter is ignored. DBFLHSH0 is always used as the Fast Path user-hashing module.

UHTS=

Specifies a 1- to 5-digit number of user (SPQB) hash table slots. Valid values are 0 - 32767. The default is 256. If you specify a null or invalid value, 256 is assumed.

USERVAR=*username*

Specifies the user name for the IMS active subsystem. It is optional.

The *username* specified for the USERVAR= parameter for a Remote Site Recovery (RSR) active site must remain the same in the event of a takeover and restart from a remote site. Likewise, the logical unit type 6.2 (LU 6.2) *uservars* specified at an RSR active site must remain the same in the event of a takeover and restart from a remote site.

For an Extended Recovery Facility (XRF)-capable system, OTMA uses USERVAR as the IMS XCF (z/OS cross-system coupling facility) member name. The USERVAR= parameter overrides the USERVAR specified in the DFSHSBxx member of the IMS PROCLIB data set.

(For IMS systems without RSR or XRF, OTMA uses the OTMANM= parameter as the IMS XCF member name. If OTMANM= is not specified, OTMA uses the non-APPC VTAM LU name (APPLID1) as the member name.)

When MNPS=*name* is specified, the USERVAR specification is ignored, and IMS uses MNPS for XRF terminal switching.

VALCK=0 | 1

Specifies the validity-check option:

0 Do not check the validity of addresses in the user's call list. This is the default.

1 Check the validity of the addresses in the user's call list.

An address is invalid if it is either lower than the lowest address not in the z/OS nucleus or higher than the highest address in virtual storage.

VAUT=0 | 1

Specifies whether (1) or not (0) IMS is to use the VTAM authorized path facility.

VFREE=

Virtual Fetch is no longer supported by the operating system. This keyword, if specified, is ignored by IMS. However, do not remove the comma from the procedure, because this parameter is positional.

VSFX=

Virtual Fetch is no longer supported by the operating system. This keyword, if specified, is ignored by IMS. However, do not remove the comma from the procedure, because this parameter is positional.

VSPEC=*xx*

Specifies a 2-character suffix for DFSVSMxx, the IMS.PROCLIB member that contains control statements. The default is 00. The default member in IMS.PROCLIB is DFSVSM00.

Control statements in DFSVSMxx define the following:

- Size and number of buffers in a subpool
- Number and use of local shared resource pools

- Whether sequential buffering is to be used
- Various performance and trace options
- DASD logging data set requirements
- Whether dynamic allocation for IMS batch is to be used
- Coupling facility structure names for sysplex data sharing

The VSPEC= parameter performs the same function in an IMS online system that the DFSVSAMP DD statement performs in a batch system. For more information, see "DFSVSMxx member of the IMS PROCLIB data set" on page 738.

WADS=S | D
Specifies whether single (S) or dual (D) logging is to be done on the write ahead data set. The default is S.

This parameter cannot be changed at emergency restart.

WKAP=
Specifies the amount of subpool 231 storage to be allocated to the work area pool. The value can be specified as 1- to 6-numeric characters or 1- to 5-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2G-1. If the value specified exceeds 2G-1 the default is 2G-1. The value entered is rounded up to the nearest 4K page boundary. The default is 5K. The WKAP pool size cannot be specified at system definition.

XPLINK=Y | N
This parameter is obsolete. If it is specified, it is ignored.

For Java dependent regions, this parameter is internally set to XPLINK=Y, which means that the JVM LE enclave is initialized with XPLINK(ON).

YEAR4=N | Y
Specifies whether the year is displayed as 4 digits in the time stamp as a result of the display command. The default is YEAR4=N.

N Displays 2 digits.

Y Displays 4 digits.

DD statements for IMS procedures

The IMS-supplied procedures can contain various DD statements, which are described and defined in this topic.

DD statement descriptions

The following list contains a description of all the DD statements that can be used by the procedures supplied for IMS.

DFSCTL DD
Use sequential buffering control statements to:

- Specify I/O operations that require sequential buffering (SB).
- Capture internal calls to the SB buffer handler.
- Take a snapshot of SB control blocks.
- Perform an SB buffer handler self-check.

For more information, see Specifying sequential buffering control statements (System Definition).

Use HSSP control statements to:

- Set up the environment in which you process a selected PCB with HSSP.
- Create an image copy of a designated DEDB area.
- Suppress index maintenance for any DEDB databases with a secondary index for a specific PSB.
- Restrict access of a specific HSSP or non-HSSP application program to only designated DEDB areas.

For more information, see Using high-speed sequential processing control statements (System Definition).

DFSDB2AF DD

Points to the DB2 for z/OS libraries that contain modules used by DB2 Resource Recovery Services attachment facility (RRSAF). Use this DD statement in the DFSJMP or DFSJBP procedure of a JMP or JBP that accesses DB2 for z/OS databases.

The following is an example of a DFSDB2AF DD statement:

```
//DFSDB2AF DD  DISP=SHR,DSN=IMS.SDFSRESL
//         DD  DISP=SHR,DSN=DSNxxx.DSNLOAD
//         DD  DISP=SHR,DSN=DSNyyy.DSNLOAD
```

DFSHALDB DD

The data set can be either a member of a PDS, a PS, or instream. The data set attributes must be LRECL=80 and RECFM=FB. The contents of this data set should be the HALDB control statements that restrict processing to a single partition.

DFSESL DD

Points to the external subsystem libraries that contain modules used by External Subsystem Attach Facility (ESAF).

Use this DD statement in the dependent region procedure of an IFP, MPP, or BMP region that accesses DB2 for z/OS databases using ESAF. You can also concatenate these libraries with the JOBLIB/STEPLIB libraries in the control region instead of using this DD statement. The external subsystem libraries must follow IMS.SDFSRESL.

The following is an example of a DFSESL DD statement:

```
//DFSESL      DD  DISP=SHR,DSN=IMS.SDFSRESL
//            DD  DISP=SHR,DSN=DSNxxx.DSNLOAD
//            DD  DISP=SHR,DSN=DSNyyy.DSNLOAD
```

DFSOLPnn DD

Defines the primary online DASD log data set, where *nn* can be any numeric value. You can specify between 3 and 100 primary DASD data sets. The block size for the online log data sets must be a multiple of 2K bytes.

DFSOLSnn DD

Defines the optional secondary online DASD log data sets, where *nn* can be any numeric value. You can specify between 3 and 100 secondary DASD data sets. The block size for the online log data sets must be a multiple of 2 K. These statements are required only if dual logging is requested. Suffixes must be consistent with the primary log data sets.

DFSRESLB DD

Points to an authorized library that contains the IMS SVC modules. For IMS batch, IMS.SDFSRESL and any data set that is concatenated to it on the DFSRESLB DD statement must be authorized through the Authorized Program Facility (APF).

DFSSTAT DD

Defines a data set describing DB call and buffering activity during an application's execution. The reports are written when the application terminates. If you are interested in receiving //DFSSTAT reports, include a //DFSSTAT DD statement in this procedure. For example: //DFSSTAT DD SYSOUT=A

For more information about interpreting //DFSSTAT reports, see *IMS Version 11 System Utilities*.

DFSTCF DD

Defines the data set specified if the time-controlled operation function is used.

DFSVSAMP DD

Defines the following:

- Size and number of buffers in a subpool
- Number and use of local shared resource pools
- Whether sequential buffering is to be used
- Various performance and trace options
- DASD logging data set requirements
- Whether dynamic allocation for IMS batch is to be used
- Coupling facility structure names for sysplex data sharing
- Caching option for the OSAM subpools.

The DFSVSAMP DD statement performs the same function in an IMS batch system that the DFSVSMxx DD statement performs in an online system. In a batch system, the control statements defining the preceding options are in a data set with the ddname of DFSVSAMP; in an online system, the control statements are in a member of the IMS.PROCLIB data set with the member name of DFSVSMxx.

DFSWADSn DD

Defines the required write-ahead data sets, where *n* can be a number from zero to nine. If dual logging to the WADS is requested, at least two WADS data sets must be provided. If an I/O error occurs on the current WADS, the next WADS is used.

Dual WADS logging provides backup on a read error while terminating the OLDS from the WADS. The primary and secondary WADS contain the same data. Single or dual WADS logging is determined from an execution-time parameter you specify. Regardless of whether dual logging is selected, as many as 10 WADS DD statements can be included. The extra data sets serve as spares on write errors. When a write error occurs, write-ahead logging continues whether single mode or dual mode is selected. One of the spare data sets is substituted for the one with the error. When required WADSs are unavailable because of I/O errors, IMS issues a warning message and continues processing in a degraded mode, truncating OLDS buffers for LWA. When switching to a new WADS after a write error, the current online log buffer is truncated and written to the OLDS, making obsolete all data currently in the WADS.

FORMATA DD and FORMATB DD

Points to IMS.FORMATA and IMS.FORMATB, which contain online MFS definitions to be used as the format library by the online system. They are required by MFS-supported terminals.

IMS limits the number of data sets that can be concatenated for MFS format libraries, IMS.FORMATA and IMS.FORMATB, to 16. All the concatenations must have like attributes.

FPTRACE DD
Defines the destination for the Fast Path trace output. Activate the trace with the DD statement in the procedure and the following command:

```
/TRACE SET ON TABLE FAST
```

IEFRDER DD
Defines the primary system log data sets. If DASD instead of tape is used for logging, substitute the appropriate DD statements. IEFRDER and IEFRDER2 can be a combination of tape and DASD. This statement is not required in DB/DC or DBCTL environments if the job does not declare database-update intent.

Striped extended-format data sets cannot be used for the system logs.

IEFRDER2 DD
Defines the secondary system log data sets. This statement is included only when dual system log data sets are used. If DASD is used for logging instead of tape, substitute the appropriate DD statements. IEFRDER and IEFRDER2 can be a combination of tape and DASD.

Striped extended-format data sets cannot be used for the system logs.

IMS DD
Add an IMS DD statement for IMS.PSBLIB, concatenated with IMS.DBDLIB, if GSAM or GLOBAL databases are accessed by the batch application. The statements are:

```
//IMS   DD   DSN=IMS.PSBLIB,DISP=SHR
//      DD   DSN=IMS.DBDLIB,DISP=SHR
```

The PSB for the batch application program must be contained in the IMS.PSBLIB, and the DBDs for the GSAM or GLOBAL databases referenced by the PSB must be contained in the IMS.DBDLIB.

IMSACB DD
Points to a partitioned data set that contains the output from ACBGEN. The IMSACB DD applies to the DBBBATCH procedure. The IMSACB DD should only be used if online change is not used. To avoid user error, do not use the IMSACB DD if local online change or global online change is enabled. If local online change is enabled, use the MODSTAT DD instead. If global online change is enabled, use the OLCSTAT DD instead.

IMSACBA DD and IMSACBB DD
Point to partitioned data sets containing the output from ACBGEN. Two statements are needed for online change. If any DOPT PSBs exist for an online system, they must reside in any concatenation other than the first. If OLCSTAT is omitted or cannot be accessed, the MODSTAT DD is used, if defined. If OLCSTAT and MODSTAT are omitted or cannot be used, IMSACB DD is used.

If you are using the DFSMDA member with TYPE=IMSACBA and TYPE=IMSACBB to dynamically allocate the IMS IMSACBA and IMSACBB library data sets, remove the IMSACBA and IMSACBB DD statements from your FDR, DBC, DCC, DLISAS, and IMS procedures.

IMSDALIB DD
Defines the non-authorized partitioned data set (PDS) in which members to be dynamically allocated can be stored. Examples include RECON definitions, database definitions, and anything that is in a DFSMDA member.

As implied by the absence of IMSDALIB from the list of valid DD statements in "DLIBATCH procedure" on page 548 (for the DLIBATCH procedure) and "DBBBATCH procedure" on page 520 (for the DBBBATCH procedure), IMSDALIB is not supported in batch mode.

IMSIRD DD
Defines the data set that submits the modified start-up JCL for a dependent region to the JES internal reader.

IMSLOGR
Defines the input log data set for extended restart. This statement is required if an application program is performing an XRST call. Any program that issues symbolic checkpoint calls must also issue the XRST call and therefore requires the IMSLOGR DD statement.

Striped extended-format data sets cannot be used for the system logs.

IMSMON DD
Describes the recording device that the IMS monitor uses. This statement is produced only if LOG=MONITOR is specified in the IMSCTF macro.

IMSRDS DD
Defines the first restart data set; this data set contains information required for recovery, including the checkpoint ID table needed for restarting IMS. This data set does not contain any log records.

IMSRDS2 DD
Defines the second restart data set. It is used only in an XRF system. This data set is identical in function to IMSRDS DD.

IMSTFMTA DD and IMSTFMTB DD
Point to IMS.TFORMAT, which contains the online MFS descriptors for test mode online execution. These DD statements consist of IMS.TFORMAT, concatenated in front of IMS.FORMATA and IMS.TFORMAT, concatenated in front of IMS.FORMATB. If you change MFS formats online, the two DD statements might point to this single TFORMAT data set, or the DD statements might point to two separate TFORMAT data sets.

INPARMS DD
INPARMS is a DD statement used by the IMS abend search and notification program, DFSIASNP. It points to a data set member created by the setup panel that contains keywords and their corresponding values. The keywords are used in email and text messages, where a parser substitutes corresponding values in place of the keywords inside the email and text messages.

JAVAIN DD, JAVAOUT DD, and JAVAERR DD
Point to z/OS UNIX System Services file system files that Java opens and assigns to stdin, stdout, and stder. System.in.read() reads from the UNIX System Services file system file that is specified in the JAVAIN DD statement. All System.out.print() calls from the Java application are directed to the UNIX System Services file system file that is specified in the JAVAOUT DD statement. All System.err.print() output from the Java application is directed to the UNIX System Services file system file that is specified in the JAVAERR DD statement. For example:

```
//JAVAIN DD PATH=/tmp/javain,DISP=SHR
```

Recommendation: Specify these DD statements if you want to take advantage of the function that they offer. If you do not specify the JAVAOUT DD or the JAVAERR DD statements, System.out.print() and System.err.print() calls in the Java application do not generate any output.

JCLOUT DD
Defines a data set where output produced by a GENJCL command is to be written.

JCLPDS DD
Defines a partitioned data set that contains the skeletal JCL execution and default members used by the GENJCL commands.

LGMSG DD
Points to the normal long message data set. If a batch only execution is planned, you do not need the message queue data sets. The Queue Manager Concurrent I/O provides a facility for the IMS customer to provide multiple Normal short and long message queue data sets. This facility is optional and is invoked by the IMS customer providing from 1 to 10 DD statements for the Normal short and long message queues data sets.

The current implementation of the Normal short and long message queues allows only one DD statement for each.

LGMSGL DD
Points to the local long message data set. This statement is used only in an XRF complex.

MODBLKSA DD
Points to IMS.MODBLKSA, which contains the system definition output to be brought online using a `/MODIFY` command when IMS.MODBLKSB is actively being used by the control region. MODBLKSA must be authorized by APF.

MODBLKSB DD
Points to IMS.MODBLKSB, which contains the system definition output to be brought online using a `/MODIFY` command when IMS.MODBLKSA is actively being used by the control region. MODSBLKB must be authorized by APF.

MODSTAT DD
Points to IMS.MODSTAT, the modify status data set. The MODSTAT data set contains online change status for an IMS that has local online change enabled. The MODSTAT data set keeps track of the DD names of libraries that are active:

- ACBLIBA or ACBLIBB
- FORMATA or FORMATB
- MODBLKSA or MODBLKSB

The MODSTAT DD applies to the IMS procedure and the DBBBATCH procedure. The MODSTAT DD is not needed in the IMS procedure when global online change is enabled.

If the OLCSTAT DD is omitted or the OLCSTAT data set cannot be accessed, the MODSTAT DD (if defined) determines which ddname to use. If both the OLCSTAT DD and the MODSTAT DD are omitted or the OLCSTAT and MODSTAT data sets cannot be accessed, the IMSACB DD is used.

For the IMS procedure, the ddnames you can use are ACBLIBA (or ACBLIBB), FORMATA (or FORMATB), and MODBLKSA (or MODBLKSB). For the DBBBATCH procedure, the ddname to use is IMSACBA or IMSACBB.

MODSTAT2 DD
Points to IMS.MODSTAT2, which indicates the active or inactive data sets that the IMS online system should use during initialization. This statement is used only in an XRF system.

The MODSTAT2 data set, when used for online changes, is used only for local online change. Global online change (in an IMSplex) uses the OLCSTAT data set.

MSDBCP1 DD and MSDBCP2 DD
Defines the first pair of MSDB checkpoint data sets. IMS alternates between the two when taking an MSDB checkpoint. During an /NRE, /ERE, or IMS procedure, the most current MSDB is determined, and the MSDBs are loaded from that data set (unless the MSDBLOAD keyword is specified during warm start).

MSDBCP3 DD and MSDBCP4 DD
Defines the second pair of MSDB checkpoint data sets. Used only in an XRF environment, these data sets are identical in function to MSDBCP1 and MSDBCP2. In an XRF environment, you must specify all four data sets, because both the active and the alternate subsystem each require two. Any two of the four can contain the latest MSDB checkpoint. Although an active subsystem can select the data set containing the latest MSDB checkpoint, the alternate subsystem must select the two data sets not used by the active.

MSDBDUMP DD
Defines a data set that contains a dump of all MSDBs. This data set specifies where to dump the MSDBs when a /DBD DB MSDB command is entered. Successive executions of the command cause the previous contents to be overlaid.

MSDBINIT DD
Defines a data set that contains unloaded or reconstructed MSDBs. This data set specifies where to start the initial load of the MSDBs during all cold starts and during a normal restart if the MSDBLOAD parameter is specified for the /NRESTART command. It is produced by executing the MSDB Dump Recovery or MSDB Maintenance utility. MSDBINIT can contain one, several, or all MSDBs defined.

OLCSTAT DD
Points to IMS.OLCSTAT, the online change status data set. The OLCSTAT data set contains online change status for an IMSplex that has global online change enabled. The OLCSTAT data set keeps track of the ddnames of libraries that are active:

- ACBLIBA or ACBLIBB
- FORMATA or FORMATB
- MODBLKSA or MODBLKSB

The OLCSTAT DD applies to the DBBBATCH procedure.

If the OLCSTAT data set is defined, it determines whether to use the IMSACBA or IMSACBB DD name. The OLCSTAT data set name is selected from either:

- The OLCSTAT DD statement that is coded in the JCL.
- The dynamic allocation MDA member, if either the OLCSTAT DD statement is omitted, or the data set from the OLCSTAT DD cannot be accessed.

If the MODSTAT DD statement is defined, it determines whether to use the IMSACBA or IMSACBB DD name only when both of the following are true:

- The OLCSTAT DD statement is omitted.
- No dynamic allocation MDA member is defined for the OLCSTAT data set.

If both the MODSTAT and OLCSTAT DD statements are omitted, the IMSACB DD is used. If either the OLCSTAT DD statement or the MDA member exists, and the OLCSTAT data set cannot be accessed, the DBB batch job fails with an abend U0821.

PRINTDD DD
Defines the output data set for the test program, including displays of control blocks using the SNAP call. It must conform to the z/OS SNAP data set requirement.

PROCLIB DD
Points to IMS.PROCLIB, which contains all IMS-generated cataloged procedures, jobs, and control statements.

QBLKS DD
Points to the normal queue blocks message data set. If a batch-only execution is planned, you do not need the message queue data sets.

QBLKSL DD
Points to the local queue blocks message data set. This statement is used only in an XRF complex.

RECONn DD
Defines the DBRC RECON data sets, where *n* is 1, 2, or 3. Add RECON DD statements if DBRC is being used. If dynamic allocation is being used, the DD statements are not necessary. The names of these data sets must be consistent with all subsystems that share the RECON data set.

Related reading: For more information about creating a RECON data set, see *IMS Version 11 System Administration*.

SHMSG DD
Points to the normal short message data set. If a batch-only execution is planned, you do not need the message queue data sets.

SHMSGL DD
Points to the local short message data set. This statement is used only in an XRF complex.

STEPLIB DD
Points to IMS.SDFSRESL, which contains the IMS nucleus and required action modules. If STEPLIB is unauthorized because of having unauthorized libraries concatenated to IMS.SDFSRESL, a DFSRESLB DD statement must be included. The STEPLIB statement need not be authorized for IMS batch.

SYSABEND DD
Defines a dump data set. If both a SYSABEND DD statement and a SYSUDUMP DD statement are used, the last occurrence is used to define the dump.

SYSHALDB DD
Contains a report of all card images of the HALDB control statements with corresponding reasons for either accepting or rejecting the control statements.

SYSLMOD DD
Defines the destination of the output for the binder.

SYSPRINT DD
Defines the output data set for control messages, statistics and reports produced by a utility.

SYSTSPRT DD
Defines the output data set for TSO Parse. The data set contains all TSO Parse

error messages, such as errors in parsing the DFSRSRxx PROCLIB member. This DD statement is optional and is not generated by IMS.

SYSUDUMP DD
Defines a dump data set. If both a SYSABEND DD statement and a SYSUDUMP DD statement are used, the last occurrence is used to define the dump.

SYSIN DD
Defines the control statement input data set.

SYSLIB DD
Points to the libraries that contain the other modules needed for the final executable module.

SYSLIN DD
Defines the destination of the output from the compiler.

SYSUTn DD
Is a temporary work data set used by the compiler, where *n* is 1, 2, 3, or 4. This data set must reside on DASD.

CQS startup procedure

Use the CQS startup procedure to dynamically override the settings in the CQS initialization parameters member of the IMS PROCLIB data set (CQSIPxxx).

You can start CQS as a started procedure or with JCL. The started procedure JCL for CQS can execute either the CQSINIT0 or the BPEINI00 program.

A sample startup procedure called CQS can be found in the IMS PROCLIB data set. When using CQSINIT0, there is no change to the sample procedure shown in "Sample Common Queue Server startup procedure using CQSINIT0" on page 514.

JCL

The JCL for the CQS procedure using CQSINIT0 is shown in "Sample Common Queue Server startup procedure using CQSINIT0" on page 514.

Note: Make sure to use a sufficient value for RGN=. Using a value that is too small can impact IMS availability and response times. It is suggested that you use an initial value of 300M, which you can adjust higher or lower after analyzing information provided in the JES2 step termination message IEF374I. That message includes the fields SMF3OURB and SMF30EUR, which are prefixed by, respectively, the labels VIRT and EXT.

When using BPEINI00, use the same procedure shown in "Sample Common Queue Server startup procedure using CQSINIT0" on page 514, but change the EXEC statement so that PGM=BPEINI00 is specified. You must also add the BPEINIT=CSQINI00 parameter, which identifies the CQS BPE initialization parameters module. A sample startup procedure using BPEINI00 is shown in "Sample Common Queue Server startup procedure using BPEINI00" on page 515.

Note: Make sure to use a sufficient value for RGN=. Using a value that is too small can impact IMS availability and response times. It is suggested that you use an initial value of 300M, which you can adjust higher or lower after analyzing

information provided in the JES2 step termination message IEF374I. That message includes the fields SMF3OURB and SMF30EUR, which are prefixed by, respectively, the labels VIRT and EXT.

Usage

Figure 42 shows the general relationship between execution parameters and members of the IMS PROCLIB data set. In the figure, CQSSL001 and CQSSL002 contain structure definition parameters that are unique to each CQS. CQSSG00A contains the structure definition parameters that are shared by all CQS address spaces connected to the shared queues.

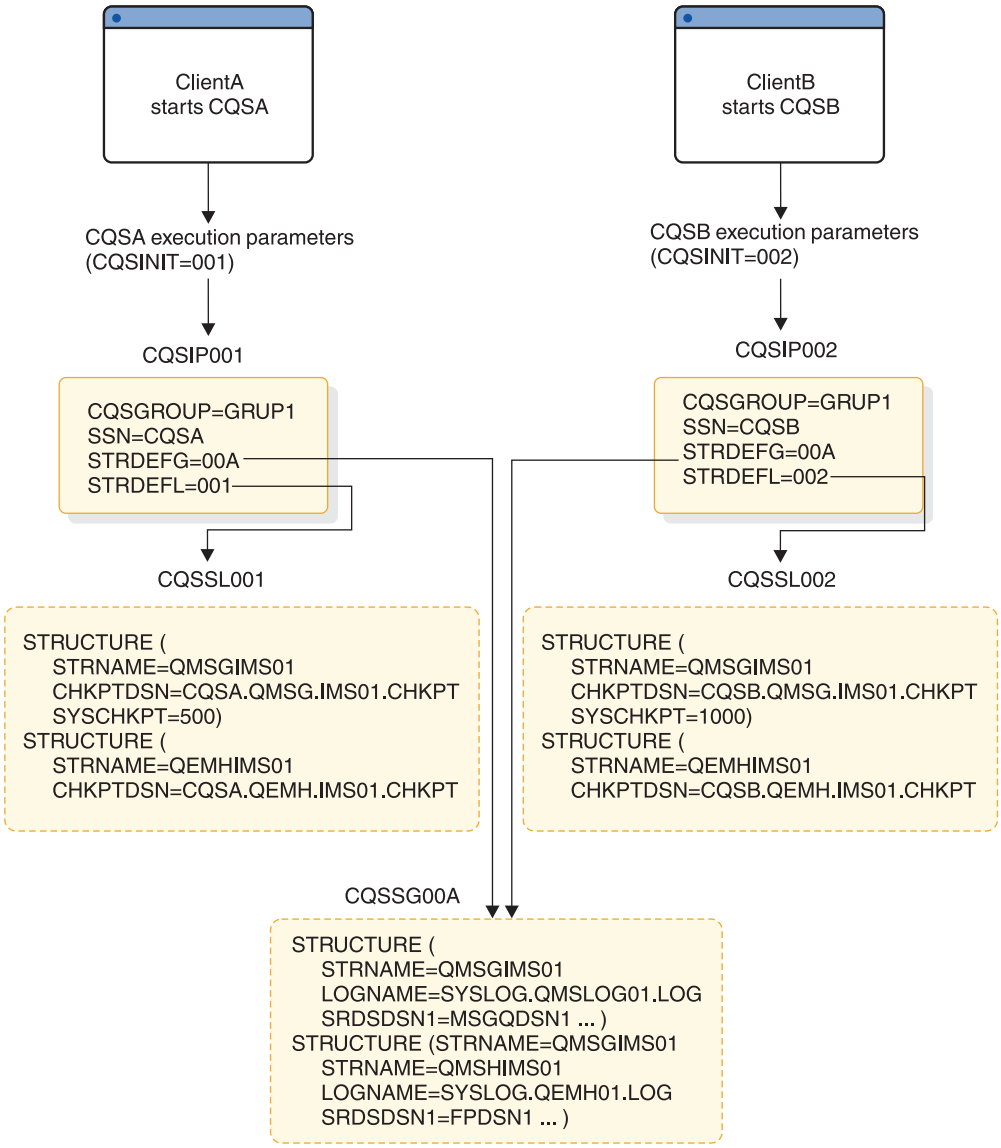


Figure 42. Specifying IMS and CQS parameters

Parameters

You can specify the following execution parameters on the CQS startup procedure. Read the descriptions of the parameters to determine whether you want to accept the system defaults or tailor the system to fit the requirements of your environment.

```
ARMRST=
 BPECFG=
BPEINIT=CQSINI00
CQSGROUP=
CQSINIT=
 SSN=
STRDEFG=
STRDEFL=
```

Execution data sets

CQS uses two types of data sets, the system checkpoint data set and the structure recovery data set. Both of these data sets must be VSAM entry-sequenced data sets (ESDSs). These data sets are user data sets (and are not known to SMP/E).

CQS system checkpoint data set

Each CQS address space that is connected to a queue structure maintains a system checkpoint data set for each structure pair. Neither CQS address spaces or queue structures share the system checkpoint data set. The CQS initialization process dynamically allocates the system checkpoint data set.

Use the MVS/DFP `DEFINE CLUSTER` functional command to define the data set to the installation. The `DEFINE CLUSTER` function command and its parameters are described in *z/OS DFSMS Access Method Services for Catalogs*.

When you use the `DEFINE CLUSTER` functional command to define the system checkpoint data set, you must specify the following parameters:

NAME
Specifies the same name that you specify using the CHKPTDSN= parameter of the CQS local structure definition member of the IMS PROCLIB data set.

NONINDEXED
Specifies that the data set is to be an ESDS.

NONSPANNED
Specifies that the records must be contained in a single control interval.

SHAREOPTIONS
Specifies how a cluster can be shared among users. You must specify SHAREOPTIONS (2,3).

REUSE
Specifies that the cluster can be reused.

The following parameters are recommended when you specify the DEFINE CLUSTER functional command:

RECORDSIZE
Specifies the average and maximum length in bytes for the records in the cluster. Both the maximum and minimum length of the RECORDSIZE should be 7 bytes less than the CONTROLINTERVALSIZE.

CONTROLINTERVALSIZE
Specifies the size of the control interval for the cluster. The recommended CONTROLINTERVALSIZE is 512; whatever value you specify must be a multiple of 512.

An example of the system checkpoint data set is shown here:

```
DEFINE CLUSTER                                          -
     (NAME (MSGQ.CHKPT)                                 -
     TRK(2,2) VOL (IMSQAV) NONINDEXED SHAREOPTIONS (2,3)  -
     RECSZ(505,505) REUSE CISZ (512))
```

CQS structure recovery data sets

CQS uses two structure recovery data sets per structure pair for its structure checkpoint processing.

Structure recovery data sets are not used for the resource structure.

When a structure checkpoint is requested, CQS dynamically allocates the structure recovery data sets. Structure checkpoint requests alternate between the two structure recovery data sets. During structure checkpoint processing all recoverable data objects on a structure are written to the structure recovery data sets. Thus, the size of each data set should be approximately the size of the primary structure plus the overflow structure to ensure that the entire structure fits in the data set.

Use the MVS/DFP `DEFINE CLUSTER` functional command to define the data set to the installation. The `DEFINE CLUSTER` function command and its parameters are described in *z/OS DFSMS Access Method Services for Catalogs*.

When you use the `DEFINE CLUSTER` functional command to define the structure recovery data set, you must specify the following parameters.

NAME
Specify the same name you specify in the CQS global structure definition member of the IMS PROCLIB data set using the SRSDSDSN1= and the SRSDSDSN2= parameters.

NONINDEXED
Specifies that the data set is to be an ESDS.

NONSPANNED
Specifies that the records must be contained in a single control interval.

SHAREOPTIONS
Specifies how a cluster can be shared among users. You must specify SHAREOPTIONS (2,3).

REUSE
Specifies that the cluster can be reused.

The following parameters are recommended when you specify the `DEFINE CLUSTER` functional command:

RECORDSIZE
Specifies the average and maximum length in bytes for the records in the cluster. Both the maximum and minimum length of the RECORDSIZE should be 7 bytes less than the CONTROLINTERVALSIZE.

CONTROLINTERVALSIZE
Specifies the size of the control interval for the cluster. The recommended control interval size is 32 768; whatever value you specify must be a multiple of 512.

An example of the structure recovery data set is shown here:

```
 DEFINE CLUSTER                                          -
     (NAME (MSGQ.SRDS1)                                  -
     TRK(45,5) VOL (DSHR03) NONINDEXED SHAREOPTIONS (2,3)  -
     RECSZ(32761,32761) REUSE  CISZ (32768)
```

Sample Common Queue Server startup procedure using CQSINIT0

```
//*******************************************************************
//*     CQS Procedure
//*
//*
//*     Parameters:
//*     BPECFG  - Name of BPE member
//*     CQSINIT - Suffix for your CQSIPxxx member
//*     PARM1   - Other override parameters
//*               ARMRST   - ARM restart on CQS failure (Y/N)
//*               CQSGROUP - XCF Group for sharing CQSs
//*               SSN      - CQS subsystem name
//*               STRDEFG  - Suffix for CQSSGxxx member
//*               STRDEFL  - Suffix for CQSSLxxx member
//*
//*               example:
//*               PARM1='ARMRST=Y,SSN=CQSA,STRDEFG=001,STRDEFL=001'
//*
//*******************************************************************
//*
//*    Licensed Materials - Property of IBM
//*
//*    Restricted Materials of IBM
//*
//*    5655-C56
//*
//*    (C) Copyright IBM Corp. 1996,1998
//*
//*******************************************************************
//*
//CQS      PROC RGN=300M,SOUT=A,
//             RESLIB='IMS.SDFSRESL',
//             BPECFG=BPECONFG,
//             CQSINIT=,
//             PARM1=
//*
//CQSPROC  EXEC PGM=CQSINIT0,
//             REGION=&RGN,
// PARM='BPECFG=&BPECFG,CQSINIT=&CQSINIT,&PARM1'
//*
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=&BPERES,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//*
```

Sample Common Queue Server startup procedure using BPEINI00

```
//*****************************************************************
//*     CQS Procedure
//*
//*
//*     Parameters:
//*     BPECFG  - Name of BPE member
//*     CQSINIT - Suffix for your CQSIPxxx member
//*     PARM1   - Other override parameters
//*               ARMRST   - ARM restart on CQS failure (Y/N)
//*               CQSGROUP - XCF Group for sharing CQSs
//*               SSN      - CQS subsystem name
//*               STRDEFG  - Suffix for CQSSGxxx member
//*               STRDEFL  - Suffix for CQSSLxxx member
//*
//*               example:
//*               PARM1='ARMRST=Y,SSN=CQSA,STRDEFG=001,STRDEFL=001'
//*
//*****************************************************************
//*
//*    Licensed Materials - Property of IBM
//*
//*    Restricted Materials of IBM
//*
//*    5655-C56
//*
//*    (C) Copyright IBM Corp. 1996,1998
//*
//*****************************************************************
//*
//CQS      PROC RGN=300M,SOUT=A,
//             RESLIB='IMS.SDFSRESL',
//             BPECFG=BPECONFG,
//             CQSINIT=,
//             PARM1=
//*
//CQSPROC  EXEC PGM=BPEINI00,
//             REGION=&RGN,
// PARM='BPECFG=&BPECFG,BPEINIT=CQSINI00,CQSINIT=&CQSINIT,&PARM1'
//*
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=&BPERES,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//*
```

CSLODBM procedure

Use the CSLODBM procedure to dynamically override the settings in the ODBM initialization parameters member of the IMS PROCLIB data set (CSLDIxxx).

You can start ODBM as a started procedure or with JCL. A sample startup procedure called CSLODBM can be found in IMS.SDFSISRC.

Parameters

The following list identifies the parameters that you can specify as execution parameters on the startup procedure for ODBM. The ARMRST, ODBMCFG, and ODBMNAME parameters that are required for ODBM initialization and the optional RRS parameter can also be specified in the CSLDIxxx member of the IMS

PROCLIB data set, documented in "CSLDIxxx member of the IMS PROCLIB data set" on page 633. Parameters specified on the ODBM procedure override those specified in the CSLDIxxx member of the IMS PROCLIB data set.

ARMRST=
BPECFG=
BPEINIT=
ODBMCFG=
ODBMINIT=
ODBMNAME=
RRS=

Parameters are described in "Parameter descriptions for IMS procedures" on page 448.

Sample ODBM startup procedure

The JCL for CSLODBM is shown in the following example.

```
//*****************************************************************
//*     ODBM Procedure
//*
//*
//*     Parameters:
//*     BPECFG  - Name of BPE member
//*     BPEINIT  - CSLDINI0, the module that contains the ODBM start up values
//*     ODBMINIT  - Suffix for your CSLDIxxx member
//*     PARM1   - other override parameters:
//*               ARMRST  - Indicates if ARM should be used
//*               ODBMNAME  - Name of ODBM being started
//*               ODBMCFG  - Suffix for your CSLDCxxx member
//*               RRS     - Indicates RRS is (Y) or is not (N) used
//*
//*               example:
//*               PARM1='ARMRST=Y,ODBMNAME=ODBM1,ODBMCFG=000,RRS=N'
//*
//*************************************************************@SCPYRT**
//*
//*  Licensed Materials - Property of IBM
//*
//*  5635-A02
//*
//*  Copyright IBM Corp. 2009      All Rights Reserved
//*
//*  US Government Users Restricted Rights - Use, duplication or
//*  disclosure restricted by GSA ADP Schedule contract with
//*  IBM Corp.
//*
//*************************************************************@ECPYRT**
//*
//CSLODBM    PROC RGN=3000K,SOUT=A,
//             RESLIB='IMS.SDFSRESL',
//             BPECFG=BPECONFG,
//             ODBMINIT=000,
//             PARM1=
//*
//ODBMPROC   EXEC PGM=BPEINI00,REGION=&RGN,
//  PARM='BPECFG=&BPECFG,BPEINIT=CSLDINI0,ODBMINIT=&ODBMINIT,&PARM1'
//*
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//*
```

CSLOM procedure

Use the CSLOM procedure to dynamically override the settings in the OM initialization parameters member of the IMS PROCLIB data set.

You can start OM as a started procedure or with JCL. A sample startup procedure called CSLOM can be found in IMS.SDFSISRC.

Parameters

You can specify the following execution parameters on the startup procedure for OM. Some parameters that are required for OM initialization can also be specified in "CSLOIxxx member of the IMS PROCLIB data set" on page 636.

ARMRST=
BPECFG=
BPEINIT=
CMDLANG=
CMDSEC=
OMINIT=
OMNAME=

Parameters are described in "Parameter descriptions for IMS procedures" on page 448.

Sample OM startup procedure

The JCL for CSLOM is shown here.

```
//*****************************************************************
//*     OM Procedure
//*
//*
//*     Parameters:
//*     BPECFG  - Name of BPE member
//*     OMINIT  - Suffix for your CSLOIxxx member
//*     PARM1   - other override parameters:
//*               ARMRST  - Indicates if ARM should be used
//*               CMDLANG - Language for command description text
//*               CMDSEC  - Command security method
//*               OMNAME  - Name of OM being started
//*
//*               example:
//*               PARM1='ARMRST=Y,CMDSEC=R,OMNAME=OM1,CMDLANG=ENU'
//*
//*************************************************************@SCPYRT**
//*                                                                   *
//*       Licensed Materials - Property of IBM                        *
//*                                                                   *
//*       "Restricted Materials of IBM"                               *
//*                                                                   *
//*       5635-A01 (C) Copyright IBM Corp. 2009                       *
//*                                                                   *
//*************************************************************@ECPYRT**
//*
//CSLOM    PROC RGN=3000K,SOUT=A,
//             RESLIB='IMS.SDFSRESL',
//             BPECFG=BPECONFG,
//             OMINIT=000,
//             PARM1=
//*
//OMPROC   EXEC PGM=BPEINI00,REGION=&RGN,
//  PARM='BPECFG=&BPECFG,BPEINIT=CSLOINI0,OMINIT=&OMINIT,&PARM1'
```

```
//*
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//*
```

CSLRM procedure

Use the CSLRM procedure to dynamically override the settings in the RM initialization parameters member of the IMS PROCLIB data set (CSLRIxxx).

You can start RM as a started procedure or with JCL. A sample startup procedure named CSLRM can be found in IMS.SDFSISRC.

Parameters

The following parameters can be specified as execution parameters on the RM startup procedure. Some parameters that are required for RM initialization can also be specified in the RM initialization parameters member of the PROCLIB data set (CSLRIxxx).

ARMRST=
BPECFG=
BPEINIT=
RMINIT=
RMNAME=

Sample Resource Manager startup procedure

```
//*******************************************************************
//*     RM Procedure
//*
//*
//*     Parameters:
//*     BPECFG  - Name of BPE member
//*     RMINIT  - Suffix for your CSLRIxxx member
//*     PARM1   - other override parameters:
//*               ARMRST - Indicates if ARM should be used
//*               RMNAME - Name of RM being started
//*
//*               example:
//*               PARM1='ARMRST=Y,RMNAME=RM1'
//*
//*************************************************************@SCPYRT**
//*                                                                   *
//*       Licensed Materials - Property of IBM                        *
//*                                                                   *
//*       "Restricted Materials of IBM"                               *
//*                                                                   *
//*       5635-A01 (C) Copyright IBM Corp. 2009                       *
//*                                                                   *
//*************************************************************@ECPYRT**
//*
//CSLRM    PROC RGN=3000K,SOUT=A,
//             RESLIB='IMS.SDFSRESL',
//             BPECFG=BPECONFG,
//             RMINIT=000,
//             PARM1=
//*
//RMPROC   EXEC PGM=BPEINI00,REGION=&RGN,
//  PARM='BPECFG=&BPECFG,BPEINIT=CSLRINI0,RMINIT=&RMINIT,&PARM1'
//*
```

```
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//*
```

CSLSCI procedure

Use the CSLSCI procedure to dynamically override the settings in the SCI initialization parameters member of the IMS PROCLIB data set (CSLSIxxx).

The CSLSCI startup procedure is required, but setting values for the execution parameters is optional. A sample startup procedure called CSLSCI can be found in IMS.SDFSISRC.

Parameters

You can specify the following parameters as execution parameters on the EXEC statement in the SCI startup procedure. Certain parameters that are required for SCI address space initialization can also be specified in the SCI initialization parameters member of the IMS PROCLIB data set (CSLSIxxx).

ARMRST=
BPECFG=
BPEINIT=
FORCE=
SCIINIT=
SCINAME=

Sample SCI startup procedure

```
//*****************************************************************
//*     SCI Procedure
//*
//*
//*     Parameters:
//*     BPECFG  - Name of BPE member
//*     SCIINIT - Suffix for your CSLSIxxx member
//*     PARM1   - other override parameters:
//*               ARMRST  - Indicates if ARM should be used
//*               SCINAME - Name of SCI being started
//*
//*               example:
//*               PARM1='ARMRST=Y,SCINAME=SCI1'
//*
//*************************************************************@SCPYRT**
//*                                                                    *
//*       Licensed Materials - Property of IBM                         *
//*                                                                    *
//*       "Restricted Materials of IBM"                                *
//*                                                                    *
//*       5655-C56 (C) Copyright IBM Corp. 2000                        *
//*                                                                    *
//*************************************************************@ECPYRT**
//*
//CSLSCI   PROC RGN=3000K,SOUT=A,
//             RESLIB='IMS.SDFSRESL',
//             BPECFG=BPECONFG,
//             SCIINIT=000,
//             PARM1=
//*
//SCIPROC  EXEC PGM=BPEINI00,REGION=&RGN,
//  PARM='BPECFG=&BPECFG,BPEINIT=CSLSINI0,SCIINIT=&SCIINIT,&PARM1'
```

```
//*
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//*
```

DBBBATCH procedure

The DBBBATCH procedure is a one-step procedure for running an offline DL/I batch processing region using IMS.ACBLIB.

The JCL shown in "Sample procedure to run offline DL/I batch processing region using IMS.ACBLIB" on page 522 can be used to run an offline DL/I batch processing region using IMS.ACBLIB.

If VSAM databases are used, see "IMS buffer pools" on page 179.

Usage

In the example shown in "Sample procedure to run offline DL/I batch processing region using IMS.ACBLIB" on page 522:

- The parameters in parentheses are positional.
- The JCL is for an XRF-capable system only.
- The IEFRDER statement is not required in DB/DC or DBCTL environments if the job does not declare database update intent.

For a job step declaring database update intent, DD DUMMY can be specified, if the job step is not using DBRC. This specification is valid where an image copy of the database is taken before the update job step.

Log initialization calculates the smallest value necessary for logical record length. If the JCL logical record length value is larger than the calculated value, the JCL value is used. Otherwise, log initialization uses the calculated value for logical record length and adds 4 for the block size.

If multiple volumes are required for the system log, specify a volume count value in the VOL parameter of the DD statement.

For information about the number of volumes required per DD statement, see *MVS/ESA Job Control Language User's Guide*.

If the IBM 3480 tape drive is used for the IMS log data set, IMS forces tape write mode (DCB=OPTCD=W). The default on the 3480 is to buffer the write so that IMS cannot detect when the write is performed. If a power failure occurs after a log record is written to the 3480, and the database is updated but the log record is not yet written to tape, database integrity is lost. Tape write mode is forced for the log in batch and GSAM data sets.

Parameters

The following parameters are valid for the DBBBATCH procedure:

APARM=
BKO=
BUF=

CKPTID=
DBRC=
DBRCGRP=
FMTO=
GSGNAME=
IMSID=
IRLM=
IRLMNM=
IMSPLEX=
LOCKMAX=
MON=
PRLD=
PSB= (optional)
RGN=
RGSUF= (no default when specified in the batch procedure)
RRS=
SOUT=
SRCH=
SSM=
SWAP=
SYS=
SYS2=
TMINAME=

The following parameters cannot be specified in the PARM1= and PARM2= parameters:
EXCPVR=
MBR=
RST=
SPIE=
TEST=

The IOB parameter is no longer used and is ignored if it is specified.

See "Parameter descriptions for IMS procedures" on page 448 for parameter definitions.

DD statements

The following DD statements are valid for the DBBBATCH procedure.

In addition to these DD statements, add statements for data sets representing IMS databases that are not to be dynamically allocated.

The following DD statements are required:
DFSRESLB DD
DFSVSAMP DD
IEFRDER DD
IMSACB DD
IMSACBA DD
IMSACBB DD
IMS LOGR DD
MODSTAT DD
MODSTAT2 DD
OLCSTAT DD
PROCLIB DD

STEPLIB DD
SYSUDUMP DD

The following DD statements are optional:
DFSHALDB DD
DFSSTAT DD
IEFRDER2 DD
IMS DD
RECONn DD
SYSHALDB DD

IMSDALIB is not included in the list of valid DD statements for the DBBBATCH procedure because it is not supported in batch mode.

See "DD statements for IMS procedures" on page 502 for explanations of the DD statements.

Restrictions

The following restrictions apply when you run DL/I batch jobs in an RSR environment:

- Batch-only databases that are identified to be tracked must be included in the system definition (DATABASE macro) for the tracking subsystem.
- The default GSG and TMI names used by batch jobs are specified in the IMSCTRL macro, but they can be overridden in the DLIBATCH procedure.
- Batch jobs do not use the DFSRSRxx PROCLIB member, and are therefore limited to the default VTAM modename (TMDEFLT).

Sample procedure to run offline DL/I batch processing region using IMS.ACBLIB

```
//       PROC MBR=TEMPNAME,PSB=,BUF=7,
//            SPIE=0,TEST=0,EXCPVR=0,RST=0,PRLD=,
//            SRCH=0,CKPTID=,MON=N,LOGA=0,FMTO=T,
//            IMSID=,SWAP=,DBRC=,IRLM=,IRLMNM=,
//             BKO=N,IOB=,SSM=,APARM=,
//            RGN=4M,RGSUF=,PARM1=,PARM2=,
//            SOUT=A,LOGT=2400,
//            SYS=,
//            SYS2=,LOCKMAX=,
//            GSGNAME=,TMINAME=,RRS=N,
//            IMSPLEX=
//G      EXEC PGM=DFSRRC00,REGION=&RGN,
//            PARM=(DBB,&MBR,&PSB,&BUF,
//            &SPIE&TEST&EXCPVR&RST,&PRLD,
//            &SRCH,&CKPTID,&MON,&LOGA,&FMTO,
//            &IMSID,&SWAP,&DBRC,&IRLM,&IRLMNM,
//             &BKO,&IOB,&SSM,
//            '&APARM',&LOCKMAX,
//            &GSGNAME,&TMINAME,&RRS,
//            &IMSPLEX,&RGSUF,
//            '&PARM1','&PARM2')
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//DFSRESLB DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//IMSACBA  DD DSN=IMS.&SYS2.ACBLIBA,DISP=SHR
//IMSACBB  DD DSN=IMS.&SYS2.ACBLIBB,DISP=SHR
//MODSTAT  DD DSN=IMS.&SYS.MODSTAT,DISP=SHR
//MODSTAT2 DD DSN=IMS.&SYS.MODSTAT2,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//IEFRDER  DD DSN=IMSLOG,DISP=(,KEEP),VOL=(,,,99),
```

```
//         UNIT=(&LOGT,,DEFER),
//         DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//IEFRDER2 DD DSN=IMSLOG2,DISP=(,KEEP),VOL=(,,,99),
//         UNIT=(&LOGT,,DEFER,SEP=IEFRDER),
//         DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605),
//         SPACE=(605,(500,500),RLSE,,ROUND)
//IMSMON   DD DUMMY
```

DBC procedure

The DBC procedure is an online execution procedure to initialize the DBCTL environment.

If dynamic resource definition (DRD) for resources in the IMS.MODBLKS data set is enabled, and you are no longer using the MODBLKS data sets, this procedure no longer requires the DD statements for the MODBLKS data sets, MODBLKSA and MODBLKSB.

JCL

The procedure shown in the following example executes an IMS DBCTL control region.

```
//       PROC RGN=64M,SOUT=A,DPTY='(14,15)',
//            SYS=,SYS2=,
//            LOGT=2400,SYS1=,
//            RGSUF=DBC,PARM1=,PARM2=
//IEFPROC EXEC PGM=DFSMVRC0,DPRTY=&DPTY,REGION=&RGN,
//            PARM='DBC,&RGSUF,&PARM1,&PARM2'
//*
//*
//* THE MEANING AND MAXIMUM SIZE OF EACH PARAMETER
//* IS AS FOLLOWS:
//*****************************************************
//*   RGSUF  XXX   EXEC PARM DEFAULT BLOCK SUFFIX FOR
//*                MEMBER DFSPBXXX.
//*****************************************************
//*
//*  PARM1 , PARM2 PARAMETERS BOTH ARE USED TO SPECIFY
//*  CHARACTER STRINGS THAT CONTAIN IMS KEYWORD
//*  PARAMETERS. I.E. PARM1='AUTO=Y,PST=222,RES=Y'
//*
//*  ALL OF THE VALID DBCTL KEYWORD PARAMETERS
//*  ARE DESCRIBED BELOW
//*****************************************************
//*
//******** CONTROL REGION SPECIFICATIONS ********
//*   RES    X     BLOCK RESIDENT (N = NO, Y = YES)
//*   PST    XXX   NUMBER OF PST'S PERMANENTLY ALLOC
//*   MAXPST XXX   MAXIMUM NUMBER OF PST'S
//*   SRCH   X     MODULE SEARCH INDICATOR FOR DIRECTED LOAD
//*                0 = STANDARD SEARCH
//*                1 = SEARCH JPA AND LPA BEFORE PDS
//*   FMTO         T = ONLINE FORMATTED DUMP WITH
//*                    STORAGE IMAGE DELETIONS.
//*                    TERM=NO SDUMPS ALLOWED.
//*                P = FULL ONLINE FORMATTED DUMP.
//*                    TERM=NO SDUMPS ALLOWED.
//*                F = FULL ONLINE FORMATTED DUMP.
//*                    TERM=NO SDUMPS NOT ALLOWED.
//*                N = NO FORMATTED DUMP, NO OFFLINE
```

```
//*                    DUMP. TERM=NO SDUMPS ALLOWED.
//*                Z = NO FORMATTED DUMP, NO OFFLINE
//*                    DUMP. TERM=NO SDUMPS NOT
//*                    ALLOWED.
//*  (DEFAULT)     D = OFFLINE DUMP, OR ONLINE FORMAT-
//*                    TED DUMP WITH STORAGE IMAGE
//*                    DELETIONS IF OFFLINE DUMPING
//*                    FAILS. TERM=NO SDUMPS ALLOWED.
//*                X = OFFLINE DUMP, OR ONLINE FORMAT-
//*                    TED DUMP WITH STORAGE IMAGE
//*                    DELETIONS IF OFFLINE DUMPING
//*                    FAILS. TERM=NO SDUMPS NOT
//*                    ALLOWED.
//*                M = OFFLINE DUMP, ONLINE IMS DUMP
//*                    FORMATTING NOT PERMITTED
//*                    TERM=NO SDUMPS ALLOWED.
//*                R = OFFLINE DUMP, ONLINE IMS DUMP
//*                    FORMATTING NOT PERMITTED
//*                    TERM=NO SDUMPS NOT ALLOWED.
//*   AUTO   X     Y = AUTOMATIC RESTART DESIRED
//*                N = NO AUTOMATIC RESTART
//*   IMSID  XXXX  IMS SUBSYSTEM IDENTIFIER
//*   ISIS   X     N = NO RESOURCE ACCESS SECURITY
//*                R = RACF RESOURCE ACCESS SECURITY
//*                C = RACF RESOURCE ACCESS SECURITY
//*                A = RACF RESOURCE ACCESS SECURITY
//*   ARMRST X     Y = ALLOW MVS ARM TO RESTART
//*                N = ARM NOT RESTART IMS
//*   RRS     X    Y = ENABLE PROT CONV SUPPORT
//*                N = DISABLE PROT CONV SUPPORT
//*   IRLM   X     Y = YES, N = NO
//*   IRLMNM XXXX  IRLM SUBSYSTEM NAME
//*   SSM    XXXX  EXT SUBSYSTEM PROCLIB MEMBER ID
//*   WADS   X     SINGLE OR DUAL WADS,S=SINGLE,D=DUAL
//*   ARC    XX    AUTOMATIC ARCHIVE.
//*                0 = NOT AUTOMATIC
//*                1-99 = AUTOMATIC
//*   UHASH  XXXXXXXX USER HASH MODULE NAME
//*   DBRCNM XXXXXXXX DBRC PROCLIB MEMBER NAME
//*   DLINM  XXXXXXXX DL/I PROCLIB MEMBER NAME
//*   PRDR   XXXXXXXX IMSRDR PROCLIB MEMBER NAME
//*   DBRSE  XXXXXXXX 8 CHAR DBCTL RSENAME
//*   CRC    X     COMMAND RECOGNITION CHARACTER
//*   CMDMCS X     COMMAND SECURITY OPTION
//*                R=RACF COMMAND SECURITY
//*                C=DFSCCMD0 COMMAND SECURITY
//*                B=RACF AND DFSCCMD0 CMD SEC
//*   PREMSG X     PREFIX MESSAGE OPTION
//*                N=NO DFS000I PREFIX
//*                Y=DEFAULT, DFS000I PREFIX
//*   PIMAX  XXXXXX ENQ/DEQ POOL MAXIMUM BYTES
//*   PIINCR XXXXXX ENQ/DEQ POOL INCREMENT
//*   AOIS    X     ICMD SECURITY OPTION
//*   PSWDC   X     PASSWORD CASE
//*                 U=UPPER CASE
//*                 M=MIXED CASE
//*                 R=USES RACF SETTING (DEFAULT)
//*   YEAR4   X     N = 2-DIGIT DATE (DEFAULT)
//*                 Y = 4-DIGIT DATE
//*   CPLOG   XXXXXK CHECKPOINT LOG INTERVAL
//*           OR
//*   CPLOG   XXM    CHECKPOINT LOG INTERVAL
//*   IMSGROUP XXXX  4 CHAR USER SPEC NAME
//*   DESC    XX    MSG DESC CODE
//*   MCS     (XX,XX) MSG ROUTE CODES
//*   SVC2    XXX   TYPE 2 SVC NUMBER
//*   CCTCVCAN X    Y = CCTL CANCEL WILL BE CONVERTED
```

```
//*                    TO ABEND SYSTEM 08E
//*                N = CCTL CANCEL IS NOT CONVERTED
//*
//******** FDR PARAMETER ****************************
//*
//*   FDRMBR  XX    SUFFIX FOR FDR MEMBER IN
//*                 IMS.PROCLIB
//*
//******** FAST PATH PARAMETERS ********
//*
//*   BSIZ   XXX   DATA BASE BUFFER SIZE
//*   OTHR   XXX   NUMBER OF OUTPUT THREADS
//*   DBFX   XXX   SYSTEM ALLOCATION OF DATA BASE BUFFERS TO BE
//*                FIXED AT START OF 1ST FAST PATH DEP REGION
//*   DBBF   XXX   NUMBER OF DATABASE BUFFERS
//*   DBFP   XXXX  PAGE FIX/FREE ADJUST TIMER
//*                0: FIX/FREE AT SCHED/TERM
//*                1: ALLOW PAGEFIX ONLY
//*                2-3600: SEC PAGEFREE FREQ
//*   LGNR   XX    NUMBER OF LOG ENTRIES IN DEDB BUFFERHEADER
//*   SVSODR XXXX  SVSO DISASTER RECOVERY OPTIONS
//*                NONE: DEFAULT. NO CHANGE TO ERE.
//*                AUTO: AREA MARKED RECOV NEEDED IF
//*
//*                DRRS: AREA MARKED RECOV NEEDED AT
//*
//*                WTOR: USER OPTION TO MARK AREAS
//*
//*   DMHVF  XX    MEGS TO FIX PAGEFIX FOR VSO ERE DATASPACE
//*   FPOPN   X    PREOPEN/REOPEN OPTIONS FOR DEDBS
//*                N: DEFAULT. PREOPEN OF DEDB AREAS
//*                   DONE BEFORE IMS RESTART
//*                   COMPLETES
//*                R: REOPEN AREAS OPENED WHEN IMS
//*                   ABNORMALLY TERMINATED.  BEHAVE
//*                   LIKE OPTION P FOR NON /ERE
//*                P: PREOPEN OF DEDB AREA
//*                   INITIATED AT THE END OF
//*                   RESTART
//*                A: OPTION R AND P COMBINED
//*   FPRLM   X    DEDB OPTIONS FOR IRLM RECONNECT
//*                N: DEFAULT. NO ACTION TAKEN
//*                S: RESTART ALL DEDB AREAS WHICH
//*                   WERE STARTED WHEN IRLM
//*                   DISCONNECTED
//*                R: RESTART AND REOPEN ALL DEDB
//*                   AREAS WHICH WERE STARTED WHEN
//*                   IRLM DISCONNECTED
//*                P: PREOPEN OF DEDB AREA
//*                   INITIATED AT THE END OF
//*                   IRLM RECONNECT
//*                A: COMBINE OPTIONS R AND P
//*   FP      X    INCLUDE FASTPATH IN THIS IMS
//*                N: THE DEFAULT.  THIS IMS DOES
//*                   NOT INCLUDE FASTPATH
//*                Y: THIS IMS INCLUDES FASTPATH
//******** RSR PARAMETERS *************************
//*
//*   RSRMBR XX    SUFFIX FOR RSR MEMBER
//*   TRACK  XXX   NO = NO RECOVERY TRACKING DONE
//*                RLT = RECOVERY TRACKING DONE
//*                DLT = DATABASE TRACKING DONE
//******** STORAGE POOL VALUES IN K, M OR G ******
//*   DMB    XXXXXX DMB POOL SIZE
//*   CIOP   XXXXXX CIOP POOL UPPER LIMIT
//*   WKAP   XXXXXX WORKING STORAGE BUFFER POOL SIZE
//*   PSBW   XXXXXX PSB WORK POOL SIZE
```

```
//*   DBWP   XXXXXX DATABASE WORK POOL SIZE
//*   CSAPSB XXXXXX DLISAS: CSA PSB POOL SIZE
//*   DLIPSB XXXXXX DLISAS: DLI PSB POOL SIZE
//*   EPCB   XXXXXX EPCB POOL SIZE
//*   FPWP   XXXXXX FP WORK POOL UPPER LIMIT
//*   AOIP   XXXXXX AOI POOL UPPER LIMIT
//*   CMDP   XXXXXX CMDP POOL UPPER LIMIT
//*
//******** MEMBER SUFFIXES **********************
//*
//*   SUF    X     LAST CHARACTER OF CTL PROGRAM LOAD
//*                MODULE MEMBER NAME
//*   FIX    XX    2 CHARACTER FIX PROCEDURE MODULE SUFFIX
//*   PRLD   XX    2 CHARACTER PROCLIB MEMBER SUFFIX FOR PRELOAD
//*   VSPEC  XX    2 CHARACTER BUFFER POOL SPEC MODULE SUFFIX
//*   SPM    XX    STORAGE POOL OPTIONS (DFSSPMXX)
//*   CSLG   XXX   CSL GLOBAL MEMBER (DFSCGXXX)
//*   DFSDF  XXX   DRD, CSL, AND SQ MEMBER (DFSDFXXX)
//*********************************************
//*
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//*
//*************************************************
//* IN ORDER TO START A DEPENDENT REGION, MODIFIED
//* START-UP JCL IS WRITTEN FROM INTERNAL STORAGE TO
//* THE INTERNAL READER.
//*
//IMSIRD   DD SYSOUT=(A,INTRDR)
//*********** DASD  LOGGING DD CARDS *************
//* THE FOLLOWING DD CARDS DESCRIBE THE DASD LOGGING
//* OLDS AND WADS. THESE CARDS ARE FOR EXAMPLE ONLY.
//* ALL OLDS AND WADS DATA SETS MAY BE DYNAMICALLY
//* ALLOCATED.  DD CARDS ARE NOT REQUIRED.
//* THE OLDS AND WADS TO BE USED DURING STARTUP MUST
//* BE SPECIFIED VIA OLDSDEF AND WADSDEF CONTROL
//* STATEMENTS IN THE DFSVSMXX MEMBER OF IMS PROCLIB.
//* THE ACTUAL SELECTION OF OLDS AND WADS MUST BE
//* TAILORED TO INSTALLATION REQUIREMENTS.  THE OLDS
//* AND WADS MUST BE PREDEFINED BY A SET UP JOB.
//* THE BLOCK SIZE OF ALL OLDS MUST BE THE SAME.
//* THE BLOCK SIZE AND DEVICE TYPE OF ALL WADS MUST
//* BE THE SAME.  AT LEAST 3 PRIMARY OLDS AND 1 WADS
//* MUST BE AVAILABLE FOR STARTUP.  THE BLOCK SIZE
//* SHOULD NOT BE SPECIFIED IN THIS JCL.  THE LOGGER
//* WILL GET THE BLOCK SIZE FROM THE VTOC.
//*
//DFSOLP00 DD DSN=IMS.&SYS.OLP00,DISP=SHR
//DFSOLP01 DD DSN=IMS.&SYS.OLP01,DISP=SHR
//DFSOLP02 DD DSN=IMS.&SYS.OLP02,DISP=SHR
//DFSOLP03 DD DSN=IMS.&SYS.OLP03,DISP=SHR
//DFSOLP04 DD DSN=IMS.&SYS.OLP04,DISP=SHR
//DFSOLP05 DD DSN=IMS.&SYS.OLP05,DISP=SHR
//*
//DFSOLS00 DD DSN=IMS.&SYS.OLS00,DISP=SHR
//DFSOLS01 DD DSN=IMS.&SYS.OLS01,DISP=SHR
//DFSOLS02 DD DSN=IMS.&SYS.OLS02,DISP=SHR
//DFSOLS03 DD DSN=IMS.&SYS.OLS03,DISP=SHR
//DFSOLS04 DD DSN=IMS.&SYS.OLS04,DISP=SHR
//DFSOLS05 DD DSN=IMS.&SYS.OLS05,DISP=SHR
//*
//DFSWADS0 DD DSN=IMS.&SYS.WADS0,DISP=SHR
//DFSWADS1 DD DSN=IMS.&SYS.WADS1,DISP=SHR
//*//*
//*********** IMSMON  DD  CARDS  *****************
//*
//* THE IMSMON DD STATEMENT MUST BE REMOVED IF
```

```
//* THIS DATA SET IS TO BE DYNAMICALLY ALLOCATED.
//* THE IMSACBA AND IMSACBB DD STATEMENTS MUST BE
//* REMOVED IF YOU WISH TO DYNAMICALLY ALLOCATE THE
//* ACBLIB DATA SETS THROUGH THE DFSMDA MEMBER.
//*
//IMSMON   DD DSN=IMS.&SYS1.IMSMON,DISP=(,KEEP),
//         VOL=(,,,99),UNIT=(&LOGT,,DEFER)
//*
//IMSACBA  DD DSN=IMS.&SYS2.ACBLIBA,DISP=SHR
//IMSACBB  DD DSN=IMS.&SYS2.ACBLIBB,DISP=SHR
//MODBLKSA DD DSN=IMS.&SYS2.MODBLKSA,DISP=SHR
//MODBLKSB DD DSN=IMS.&SYS2.MODBLKSB,DISP=SHR
//MODSTAT  DD DSN=IMS.&SYS.MODSTAT,DISP=SHR
//******** SYSTEM REQUIRED DD CARDS **************
//*
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=125,RECFM=FBA,BLKSIZE=3129),
//         SPACE=(6050,300,,,ROUND)
//IMSRDS   DD DSN=IMS.&SYS.RDS,DISP=SHR
//PRINTDD  DD SYSOUT=&SOUT
//*
//****** EXTERNAL SUBSYSTEM DD CARDS  ***********
//*
//* USER MAY OPTIONALLY ADD THE DFSESL DD CARD
//* FOR EXTERNAL SUBSYSTEM CONNECTION.
//*
//************ DATA BASE DD CARDS ***************
//*
//* USER MAY OPTIONALLY SUPPLY THE DD STATEMENTS
//* FOR THE ON-LINE DATA BASES TO BE
//* INSERTED HERE PRIOR TO ATTEMPTING
//* AN ON-LINE SYSTEM EXECUTION USING
//* THIS PROCEDURE.
//* IF NO DD STATEMENTS ARE SUPPLIED FOR
//* A DATA BASE, IMS ASSUMES THAT THIS
//* DATA BASE HAS BEEN DESCRIBED THROUGH
//* THE DFSMDA MACRO.
//* IF THE USER WILL BE EXECUTING WITH THE DL/I
//* SAS OPTION, THESE DD STATEMENTS SHOULD BE ADDED
//* TO THE DLISAS PROCLIB MEMBER OR DESCRIBED
//* THROUGH THE DFSMDA MACRO.
```

Usage

If DL/I databases are used, a member DFSVSMxx is built in the IMS PROCLIB data set. See “IMS buffer pools” on page 179.

DEDB areas must be allocated in the DBCTL control region. One of the following must occur:

- The DD statements must be in the control region JCL.
- The dynamic allocation members must be in a library, as addressed by JOBLIB and STEPLIB.

If you are defining your resources dynamically, the DD statements for the MODBLKS data sets are no longer required.

DBCTL execution parameters can be specified on the DBC EXECUTE statement, in the DBC procedure, or in the defaults member DFSPBDBC, or the parameter default values that are specified during system definition can be used.

Parameters

The following parameters are valid for the DBC procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions of the parameters.

AOIP= to EPCB=	FDRMBR= to PREMSG=	PRLD= to YEAR4=
AOIP=	FDRMBR=	PRLD=
AOIS=	FIX=	PSBW=
ARC=	FMTO=	PST=
ARMRST=	FP=	PSWDC=
AUTO=	FPDSSIZE=	RCLASS=
BSIZ=	FPOPN=	RES=
CCTCVCAN=	FPRLM=	RGSUF=
CIOP=	FPWP=	RRS=
CMDMCS=	IMSGROUP=	RSRMBR=
CRC=	IMSID=	SPM=
CSAPSB=	IOVF=	SRCH=
CSLG=	IRLM=	SSM=
DBBF=	IRLMNM=	SUF=
DBFP=	ISIS=	SVC2=
DBFX=	LGNR=	SVSODR=
DBRCNM=	MAXPST=	TRACK=
DBRSE=	MCS=	TSR=
DBWP=	ODBASE=	UHASH=
DESC=	ORSMBR=	VSPEC=
DFSDF=	OTHR=	WADS=
DLINM=	PIINCR=	WKAP=
DLIPSB=	PIMAX=	YEAR4=
DMB=	PRDR=	
DMHVF=	PREMSG=	
EPCB=		

DD statements

The following DD statements are valid for the DBC procedure.

In addition to the following DD statements, add statements for data sets representing IMS databases that are not to be dynamically allocated.

See "DD statements for IMS procedures" on page 502 for descriptions of the DD statements.

DFSESL DD (optional)
DFSOLPnn DD
DFSOLSNnn DD
DFSWADSn DD
IMSACBA DD
IMSACBB DD

IMSIRD DD
IMSMON DD
IMSRDS DD
MODBLKSA DD
MODBLKSB DD
MODSTAT DD
PRINTDD DD
PROCLIB DD
RECON1 DD (optional)
RECON2 DD (optional)
RECON3 DD (optional)
STEPLIB DD
SYSUDUMP DD

Important:

- If you are defining your resources dynamically, the DD statements for the MODBLKS data sets (MODBLKSA DD and MODBLKSB DD) are no longer required.
- IMSACBA DD and IMSACBB DD must be removed if you want to dynamically allocate the ACBLIB data sets through the DFSMDA member.

Restrictions

The DBC procedure is used "as is" for both regions in a DBCTL stand-by configuration.

DBRC procedure

The DBRC procedure is an online execution procedure that initiates DBRC. To take advantage of additional features provided with Base Primitive Environment (BPE) (such as BPE tracing or improved DBRC user exit services), start DBRC in a BPE.

JCL

Both the standard DBRC procedure and the BPE-based DBRC procedure run in a separate address space.

A sample procedure to run DBRC in BPE, DSPBPROC, is supplied in IMS.SDFSISRC.

The following procedure initializes DBRC and is executed by IMS using a z/OS `START` command during control region initialization.

```
//       PROC RGN=64M,DPTY='(14,15)',SOUT=A,
//            IMSID=SYS3,SYS2=
//IEFPROC EXEC PGM=DFSMVRC0,REGION=&RGN,
//            DPRTY=&DPTY,PARM='DRC,&IMSID'
//************************************************
//*
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//JCLOUT   DD SYSOUT=(A,INTRDR)
//JCLPDS   DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT
//SYSABEND DD SYSOUT=&SOUT
//*
//************ DBRC RECON DD CARDS ******************
//*
```

```
//* USER MAY OPTIONALLY SUPPLY THE DD CARDS
//* REQUIRED FOR THE DBRC RECON DATA SET.
//* IF NO DD STATEMENTS ARE SUPPLIED FOR RECON
//* DATASETS, IMS ASSUMES THAT THE DATASETS
//* HAVE BEEN DESCRIBED THROUGH THE DFSMDA MACRO.
```

The following procedure initializes DBRC in a BPE and is executed by IMS using a z/OS START command during control region initialization.

```
//*******************************************************************
//*     DBRC Procedure -  BPE Environment
//*
//*     Parameters:
//*     BPECFG   - Name of BPE member
//*     DBRCINIT - Suffix for your DSPBIxxx member
//*     IMSID    - IMS Control region ID (required)
//*     PARM1    - other override parameters:
//*                BPEINIT=DSPBINI0 (required)
//*                IMSPLEX - IMSplex name
//*                DBRCGRP - DBRC sharing group ID
//*
//*                example:
//*                PARM1='BPEINIT=DSPBINI0,IMSPLEX=PLEX1'
//*
//*************************************************************@SCPYRT**
//*
//*  Licensed Materials - Property of IBM
//*
//*  5635-A02
//*
//*  (C) Copyright IBM Corp. 2008      All Rights Reserved
//*
//*  US Government Users Restricted Rights - Use, duplication or
//*  disclosure restricted by GSA ADP Schedule contract with
//*  IBM Corp.
//*
//*************************************************************@ECPYRT**
//*
//DBRC    PROC RGN=0M,SOUT=A,
//             RESLIB='IMS.SDFSRESL',
//             BPECFG=BPECONFG,
//             DBRCINIT=000,
//             IMSID=IMS1,
//             PARM1='BPEINIT=DSPBINI0'
//*
//DBRCPROC   EXEC PGM=BPEINI00,REGION=&RGN,
// PARM='BPECFG=&BPECFG,DBRCINIT=&DBRCINIT,IMSID=&IMSID,&PARM1'
//*
//STEPLIB  DD  DSN=&RESLIB,DISP=SHR
//         DD  DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD  DSN=IMS.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT
//JCLOUT   DD SYSOUT=(A,INTRDR)
//JCLPDS   DD DSN=IMS.PROCLIB,DISP=SHR
//SYSABEND DD SYSOUT=&SOUT
//*
//*
//* User may optionally supply the DD cards
//* required for the DBRC RECON data sets.
//* If no DD statements are supplied for RECON
//* data sets, IMS assumes that the data sets
//* have been described through the DFSMDA macro.
```

Usage

IMS system definition stores the cataloged procedure for DBRC in the member of the IMS PROCLIB data set specified in the DBRCNM=*member* parameter on the IMSCTRL macro. This *member* must be copied to SYS1.PROCLIB.

The IMS control region automatically initiates the DBRC procedure. The PDS member name that IMS uses for the DBRC procedure is defined and can be overridden as follows:

1. DBRC is the default name.
2. The value specified on the DBRCNM= keyword in the IMSCTRL macro overrides the default name.
3. The value specified on the DBRCNM= keyword in the DFSPBxxx member of the IMS PROCLIB data set overrides the value specified in the IMSCTRL macro.
4. The value specified on the DBRCNM= keyword in the JCL EXEC parameter overrides the value specified in the DFSPBxxx member of the IMS PROCLIB data set.

Example `START` commands:

- `START IMSRDR,MBR=DBRC`: the PDS member specified is read as a job through the z/OS internal reader. It starts as a job rather than as a started task.
- `START DBRC,IMSID=IMSA`: DBRC executes as a started task. DBRC connects to IMSA.

When DBRC connects to the IMS control region, the DBRC procedure name is compared with the DBRCNM= specification. If they are not the same, DBRC abends.

Automatic initiation

A z/OS `START` command is issued for DBRCNM. The z/OS `START` command is coded to override the PARM='DRC,&IMSID' procedure statement. The override defines the IMSID of the subsystem issuing the `START` command, and allows a generic procedure to be used by multiple IMS control regions.

IMS does not issue the `IMS READY` message until DBRC successfully initiates. If the DBRC procedure started by the control region is in error, correct the procedure and restart it.

Manual initiation

You can supply a `START` command instead of the internally defined `START`. The `START` command is supplied in member =DBRCNM of a PDS defined in //PROCLIB in the control region JCL. If columns 1-5 are not "START", message DFS1930I is issued and the internal `START` command is issued as if the DBRCNM member was not found in the IMS PROCLIB data set.

Parameters

The following parameters are valid for the DBRC procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions of the parameters.

BPECFG=
BPEINIT=

DBRCGRP=
DBRCINIT=
DPRTY=
IMSID=
IMSPLEX=
RGN=
SOUT=
SYS2=

DD statements

The following DD statements are valid for the DBRC procedure. See "DD statements for IMS procedures" on page 502 for descriptions of the statements.

JCLOUT DD
JCLPDS DD
PROCLIB DD
RECONn DD (optional)
STEPLIB DD
SYSABEND DD
SYSUDUMP DD

DCC procedure

The DCC procedure is an online execution procedure to initialize the DCCTL environment.

If dynamic resource definition (DRD) is enabled for your runtime resource definitions, and you are no longer using the IMS.MODBLKS data sets, this procedure no longer requires the DD statements for the IMS.MODBLKS data sets, MODBLKSA and MODBLKSB.

Parameters

The following parameters are valid for the DCC procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions of the parameters.

ALOT= to EPCB=	ETO= to OMAASY=	OTMAMD= to RRS=	RSRMBR= to YEAR4=
ALOT=	ETO=	OTMAMD=	RSRMBR=
AOIP=	EXVR=	OTMANM=	RVFY=
AOIS=	FBP=	OTMASP=	SAV=
AOI1=	FESTIM=	PASSWD1=	SGN=
APPC=	FIX=	PRDR=	SHAREQ=
APPCSE=	FMTO=	PRLD=	SHMSGSZ=
APPLID1=	FRE=	PSB=	SOD=
APPLID2=	GRNAME=	PSBW=	SOUT=
APPLID3=	GRSNAME=	PST=	SPM=
ARC=	HIOP=	PSWDC=	SRCH=
ARMRST=	HSBID=	QBUF=	SSM=
ASOT=	HSBMBR=	QBUFHITH=	SUF=
AUTO=	IMSGROUP=	QBUFLWTH=	SVC2=
CIOP=	IMSID=	QBUFMAX=	SYS=

ALOT= to EPCB=	ETO= to OMAASY=	OTMAMD= to RRS=	RSRMBR= to YEAR4=
CMDMCS=	ISIS=	QBUFPCTX=	SYS1=
CPLOG=	LGMSGSZ=	QBUFSZ=	SYS2=
CRC=	LHTS=	QTL=	TCORACF=
CSLG=	LTERM=	QTU=	TRACK=
DBRCNM=	LUMC=	RCF=	TRN=
DC=	LUMP=	RCFTCB=	TSR=
DESC=	MAXPST=	RCLASS=	UHTS=
DSCT=	MCS=	RECA=	USERVAR=
DYNP=	NHTS=	RECASZ=	VAUT=
EMHB=	NLXB=	RES=	VSPEC=
EMHL=	OTMA=	RGN=	WADS=
EPCB=	OTMAASY=	RGSUF=	WKAP=
		RRS=	YEAR4=

DD statements

The following table lists the valid DD statements for the DCC procedure. See also "DD statements for IMS procedures" on page 502.

DFSDUPPM to IMSMON	IMSRDS to PRINT	PROCLIB to SYSUDUMP
DFSDUPPM DD	IMSRDS DD	PROCLIB DD
DFSESL DD (optional)	IMSRDS2 DD	QBLKS DD
DFSOLPnn DD	IMSTFMTA DD	QBLKSL DD
DFSOLSnn DD	IMSTFMTB DD	RECON1 DD (optional)
DFSTCF DD (optional)	LGMSG DD	RECON2 DD1 (optional)
FORMATA DD	LGMSGL DD	RECON3 DD1 (optional)
FORMATB DD	MODBLKSA DD	SHMSG DD
IMSACBA DD	MODBLKSB DD	SHMSGL DD
IMSACBB DD	MODSTAT DD	STEPLIB DD
IMSIRD DD	MODSTAT2 DD	SYSABEND DD
IMSMON DD	PRINT DD	SYSUDUMP DD

Important:

- If you are defining your resources dynamically, the DD statements for the MODBLKS data sets (MODBLKSA DD and MODBLKSB DD) are no longer required.
- IMSACBA DD and IMSACBB DD must be removed if you want to dynamically allocate the ACBLIB data sets through the DFSMDA member.

JCL

The following procedure executes an IMS DCCTL address space.

```
//       PROC  RGN=64M,SOUT=A,DPTY='(14,15)',
//             SYS=,SYS1=,SYS2=,
//             LOGT=2400,
```

```
//             RGSUF=DCC,PARM1=,PARM2=
//IEFPROC EXEC PGM=DFSMVRC0,DPRTY=&DPTY,REGION=&RGN,
//             PARM='DCC,&RGSUF,&PARM1,&PARM2'
//*
//*************************************************
//*                                               *
//*  THE DESCRIPTION AND CHARACTERISTICS OF EACH   *
//*  PARAMETER IS DEFINED AS FOLLOWS:             *
//*                                               *
//*************************************************
//*
//********** CONTROL REGION SPECIFICATIONS ***********
//*
//*   RGSUF   XXX   EXEC PARM DEFAULT BLOCK SUFFIX FOR
//*                 MEMBER DFSPBXXX.
//*************************************************
//*
//*  PARM1 , PARM2 PARAMETERS BOTH ARE USED TO SPECIFY
//*  CHARACTER STRINGS THAT CONTAIN IMS KEYWORD
//*  PARAMETERS. I.E. PARM1='AUTO=Y,PST=222,RES=Y'
//*
//*  ALL OF THE VALID DCCTL KEYWORD PARAMETERS
//*  ARE DESCRIBED BELOW
//*************************************************
//*   APPLID1 XXXXXXXX VTAM APPLID OF ACTIVE IMS SYSTEM
//*   APPLID2 XXXXXXXX VTAM APPLID OF XRF ALTERNATE SYSTEM
//*   APPLID3 XXXXXXXX VTAM APPLID OF RSR TRACKING SYSTEM
//*   RES     X     BLOCK RESIDENT (N = NO, Y = YES)
//*   FRE     XXXXX NUMBER OF FETCH REQUEST ELEMENTS
//*   PST     XXX   NUMBER OF PST'S PERMANENTLY ALLOC
//*   MAXPST  XXX   MAXIMUM NUMBER OF PST'S
//*   SAV     XXX   NUMBER OF DYNAMIC SAVE AREA SETS
//*   SRCH    X     IMODULE LOAD LIBRARY SEARCH INDICATOR
//*                 0 = SEARCH JPA AND LIBS BEFORE LPA
//*                 1 = SEARCH JPA AND LPA BEFORE LIBS
//*   SOD     X     SPIN OFF DUMP SYSOUT CLASS
//*   VAUT    X     VTAM AUTH PATH OPTION (1=YES,0=NO)
//*   FMTO          T = ONLINE FORMATTED DUMP WITH
//*                     STORAGE IMAGE DELETIONS.
//*                     OFFLINE SDUMPS PERMITTED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 P = FULL ONLINE FORMATTED DUMP.
//*                     OFFLINE SDUMPS PERMITTED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 F = FULL ONLINE FORMATTED DUMP.
//*                     OFFLINE SDUMPS SUPPRESSED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 N = NO FORMATTED DUMP, NO OFFLINE
//*                     DUMP. OFFLINE SDUMPS PERMITTED
//*                     FOR NON-IMS TERMINATING ERRORS
//*                 Z = NO FORMATTED DUMP, NO OFFLINE
//*                     DUMP. OFFLINE SDUMPS
//*                     SUPPRESSED FOR NON-IMS
//*                     TERMINATING ERRORS.
//*   (DEFAULT)     D = OFFLINE DUMP, OR ONLINE FOR-
//*                     MATTED DUMP WITH STORAGE IMAGE
//*                     DELETIONS IF OFFLINE DUMPING
//*                     FAILS.  OFFLINE SDUMPS
//*                     PERMITTED FOR NON-IMS
//*                     TERMINATING ERRORS.
//*                 X = OFFLINE DUMP, OR ONLINE FOR-
//*                     MATTED DUMP WITH STORAGE IMAGE
//*                     DELETIONS IF OFFLINE DUMPING
//*                     FAILS. OFFLINE SDUMPS
//*                     SUPPRESSED FOR NON-IMS
//*                     TERMINATING ERRORS.
//*                 M = OFFLINE DUMP, ONLINE IMS DUMP
```

```
//*                     FORMATTING NOT PERMITTED.
//*                     OFFLINE SDUMPS PERMITTED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 R = OFFLINE DUMP, ONLINE IMS DUMP
//*                     FORMATTING NOT PERMITTED.
//*                     OFFLINE SDUMPS SUPPRESSED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*   AUTO    X     AUTOMATIC RESTART OPTION
//*                 Y = AUTOMATIC RESTART DESIRED
//*                 N = NO AUTOMATIC RESTART
//*   IMSID   XXXX  IMS DCCTL SUBSYSTEM ID
//*   NLXB    XXX   NUMBER OF PARALLEL VTAM SESSIONS
//*   APPC    X     Y = ACTIVATE APPC/IMS
//*                 N = DO NOT ACTIVATE APPC/IMS
//*   LTERM   X     Y = LTERM USED IN DFSAPPC PROCESS
//*                 N = LTERM NOT USED IN DFSAPPC
//*                     PROCESS
//*   ARMRST  X     Y = ALLOW MVS ARM TO RESTART
//*                 N = ARM NOT RESTART IMS
//*   RRS     X     Y = ENABLE PROT CONV SUPPORT
//*                 N = DISABLE PROT CONV SUPPORT
//*   DBRCNM  XXXXXXXX DBRC PROCLIB MEMBER NAME
//*   PRDR    XXXXXXXX IMSRDR PROCLIB MEMBER NAME
//*   SSM     XXXX  EXT SUBSYSTEM PROCLIB MEMBER ID
//*   WADS    X     SINGLE OR DUAL WRITE AHEAD OPTION
//*                 S=SINGLE, D=DUAL
//*   ARC     XX    AUTOMATIC ARCHIVING OPTION
//*                 0 = NOT AUTOMATIC
//*                 1-99 = AUTOMATIC
//*   RECA    XXX   RECEIVE ANY BUFFERS
//*   FESTIM  XXXX  FRONTENDSWITCH TIMEOUT
//*   RECASZ  XXXXX RECEIVE ANY BUFFER SIZE
//*   CRC     X     COMMAND RECOGNITION CHARACTER
//*   TSR     X     U = UTC TIME
//*                 L = LOCAL TIME (DEFAULT)
//*   YEAR4   X     N = 2-DIGIT DATE (DEFAULT)
//*                 Y = 4-DIGIT DATE
//*   DC      XXX   DC PROC MEMBER SUFFIX IN
//*                 IMS.PROCLIB
//*                 DEFAULT VALUE IS 000
//*   CPLOG   XXXXXK CHECKPOINT LOG INTERVAL
//*           OR
//*   CPLOG   XXM    CHECKPOINT LOG INTERVAL
//*   PASSWD1 XXXXXXXX VTAM ACB PASSWORD
//*   ORSMBR  XX    SUFFIX FOR ORS MEMBER
//*   IMSGROUP XXXX  4 CHAR USER SPEC NAME
//*   DESC    XX    MSG DESC CODE
//*   MCS     (XX,XX) MSG ROUTE CODES
//*   SVC2    XXX   TYPE 2 SVC NUMBER
//*
//******** OTMA PARAMETERS ***************************
//*
//*   OTMA    X     Y = OTMA ENABLED
//*                 N = OTMA NOT ENABLED
//*                 DEFAULT VALUE IS N
//*   OTMANM  XXXXXXXX IMS OTMA XCF MEMBER NAME
//*   GRNAME  XXXXXXXX OTMA XCF GROUP NAME
//*                    NO DEFAULT VALUE
//*   GRSNAME XXXXXXXX GENERIC RESOURCE GROUP
//*                    NAME
//*                    NO DEFAULT VALUE
//******** SECURITY PARAMETERS ***********************
//*
//*   AOIS    X     ICMD SECURITY OPTION
//*   AOI1    X     CMD SECURITY OPTION
//*                 A = ALL
//*                 N = NONE
```

```
//*                C = DFSCCMD0 EXIT
//*                R = RACF
//*   TCORACF X    TCO RACF SECURITY OPTION
//*                Y = YES
//*                N = NO
//*   APPCSE  X    C = APPC RACF SECURITY IS CHECK
//*                F = APPC RACF SECURITY IS FULL
//*                N = APPC RACF SECURITY IS NONE
//*                P = APPC RACF SECURITY IS PROFILE
//*   CMDMCS  X    MCS/EMCS COMMAND OPTION
//*                N=COMMANDS NOT ALLOWED WITH CRC
//*                Y=ALL COMMANDS ALLOWED WITH CRC
//*                R=RACF COMMAND SECURITY
//*                C=DFSCCMD0 COMMAND SECURITY
//*                B=RACF AND DFSCCMD0 CMD SEC
//*   ISIS    X    RAS SECURITY OPTION
//*                N = NO RESOURCE ACCESS SECURITY
//*                R = RACF RESOURCE ACCESS SECURITY
//*                C = RACF RESOURCE ACCESS SECURITY
//*                A = RACF RESOURCE ACCESS SECURITY
//*   RCF     X    RACF TRAN/CMD AUTHORIZATION
//*                N = NO TRAN/CMD AUTHORIZATION
//*                C = ETO CMD AUTHORIZATION
//*                T = TRAN AUTHORIZATION
//*                S = STATIC LTERM CMD AUTH
//*                Y = C AND T
//*                A = C AND T AND S
//*   RVFY    X    RACF REVERIFY OPTION
//*                Y = YES, N = NO
//*   SGN     X    SIGNON AUTHORIZATION CHECKING
//*                F = MTO CANNOT NEGATE ACTIVATION
//*                    OF SIGNON VERIFICATION
//*                    SECURITY
//*                Y = SIGNON VERIFICATION SECURITY
//*                    WILL BE ACTIVATED
//*                N = SIGNON VERIFICATION SECURITY
//*                    WILL NOT BE ACTIVATED
//*                M = SINGLE USERID CAN SIGNON
//*                    TO MULTIPLE STATIC TERMINALS
//*                G = 'F' + 'M'
//*                Z = 'Y' + 'M'
//*   TRN     X    TRANSACTION AUTHORIZATION CHECKING
//*                F = FORCED, Y = YES, N = NO
//*   RCFTCB  XX   NUMBER OF RCF TCB'S
//*   PSWDC   X    PASSWORD CASE
//*                U=UPPER CASE
//*                M=MIXED CASE
//*                R=USES RACF SETTING (DEFAULT)
//******** MESSAGE QUEUE PARAMETERS ******************
//*
//*   EXVR    X    PAGEFIX QMGR BUFFER POOLS
//*                (1=YES, 0=NO)
//*   QBUF    XXXX NUMBER OF MESSAGE QUEUE BUFFERS
//*   QTU     XXX  MSG QUEUE UPPER THRESHOLD (%)
//*   QTL     XXX  MSG QUEUE LOWER THRESHOLD (%)
//*
//******** SHARED QUEUES PARAMETERS ******************
//*
//*   LGMSGSZ XXXXX LONG MESSAGE SIZE
//*   QBUFHITH XXX  MSG QBUF HIGH THESHOLD %
//*   QBUFLWTH XXX  MSG QBUF LOW THESHOLD %
//*   QBUFMAX  XXXX MAX NUMBER OF MSG QUEUE BUFFERS
//*   QBUFPCTX XXX  % MSG QBUF DYNAMIC EXPAND
//*                 WHEN QBUFHITH EXCEEDED
//*                 DEFAULT IS 20%
//*   QBUFSZ  XXXXX SIZE OF MESSAGE QUEUE BUFFERS
//*   SHMSGSZ XXXXX SHORT MESSAGE SIZE
```

```
//*   SHAREDQ XXX   SQ PROC MEMBER SUFFIX IN
//*                 IMS.PROCLIB
//*                 NO DEFAULT VALUE
//*

//******** ETO PARAMETERS ***************************
//*
//*   ETO     X     Y = EXTENDED TERMINAL OPTION
//*                 N = NO EXTENDED TERMINAL OPTION
//*                 M = NO EXTENDED TERMINAL OPTION
//*                     BUT LOGON USERDATA SUPPORTED
//*                     FOR STATIC TERMINALS
//*   ASOT    XXXX  ETO AUTO SIGNOFF TIME
//*   ALOT    XXXX  ETO AUTO LOGNOFF TIME
//*   DLQT    XXX   ETO DEAD LETTER QUEUE SIZE
//*
//******** RSR PARAMETERS ***************************
//*
//*   RSRMBR  XX    SUFFIX FOR RSR MEMBER
//*   TRACK   XXX   NO = NO RECOVERY TRACKING DONE
//*                 RLT = RECOVERY TRACKING DONE
//*   USERVAR XXXXXXXX USER NAME OF ACTIVE IMS SYSTEM FOR RSR
//*
//******* HASH TABLE PARAMETERS **********************
//*
//*   LHTS    XXXXX # OF CNT HASH TABLE SLOTS
//*   NHTS    XXXXX # OF VTCB HASH TABLE SLOTS
//*   UHTS    XXXXX # OF SPQB HASH TABLE SLOTS
//************ FAST PATH PARAMETERS ******************
//*
//*   EPCB    XXXX  EPCB POOL SIZE (1K BLOCKS)
//*   EMHL    XXXXX SIZE OF EMH BUFFER IN BYTES
//*   FP      X     INCLUDE FASTPATH IN THIS IMS
//*                 N: THE DEFAULT.  THIS IMS DOES
//*                    NOT INCLUDE FASTPATH
//*                 Y: THIS IMS INCLUDES FASTPATH
//*
//********** STORAGE POOL VALUES IN K, M OR G ********
//*
//*   FBP     XXXXXX MESSAGE FORMAT BLOCK POOL SIZE
//*   PSB     XXXXXX DCCTL PSB POOL SIZE
//*   CIOP    XXXXXX TP DEVICE I/O POOL UPPER LIMIT
//*   WKAP    XXXXXX WORKING STORAGE POOL SIZE
//*   PSBW    XXXXXX PSB WORK POOL SIZE
//*   HIOP    XXXXXX HIGH I/O POOL UPPER LIMIT
//*   EMHB    XXXXXX EMHB POOL UPPER LIMIT
//*   LUMP    XXXXXX LUMP POOL UPPER LIMIT
//*   LUMC    XXXXXX LUMC POOL UPPER LIMIT
//*   DYNP    XXXXXX DYNP POOL UPPER LIMIT
//*   AOIP    XXXXXX AOI POOL UPPER LIMIT
//*   CMDP    XXXXXX CMDP POOL UPPER LIMIT
//*
//******* MEMBER SUFFIXES ****************************
//*
//*   SUF     X     CONTROL PGM NAME (DCCTL NUCLEUS)
//*   FIX     XX    PAGE FIX NAME (DFSFIX..)
//*   PRLD    XX    MODULE PRELOAD NAME (DFSMPL..)
//*   VSPEC   XX    TRACE/LOG DEF (DFSVSM..)
//*   SPM     XX    STORAGE POOL OPTIONS (DFSSPM..)
//*   CSLG    XXX   CSL GLOBAL MEMBER (DFSCGXXX)
//*   DSCT    X     ETO USER DESCRIPTOR TABLE(DFSDSCTX)
//*   DFSDF   XXX   DRD, CSL, AND SQ MEMBER (DFSDFXXX)
//*
//******* XRF PARAMETERS *****************************
//*
//*   HSBID   X     XRF SYSTEM IDENTIFICATION
//*                 1 FOR FIRST SYSTEM
```

```
//*                2 FOR SECOND SYSTEM
//*   HSBMBR  XX    XRF OPTIONS (DFSHSB..)
//*
//*   MNPS    XXXXXXXX NAME OF MNPS ACB
//*                    USERVAR WILL BE IGNORED
//*   MNPSPW  XXXXXXXX MNPS ACB PASSWORD
//*************************************************
//*                                               *
//*        XRF SYSTEM DATA SET INFORMATION        *
//*                                               *
//*************************************************
//*
//* THE FOLLOWING DATA SETS MUST RESIDE ON SHARED
//* VOLUMES:
//*
//*     DDNAMES:   DFSOLPXX, DFSOLSXX
//*                DFSWADSX
//*                IMSRDS   - FIRST RDS
//*                IMSRDS2  - SECOND RDS (MANDATORY)
//*                MODSTAT  - FIRST MODSTAT
//*                MODSTAT2 - 2ND MODSTAT (MANDATORY)
//*                RECON1, RECON2, RECON3
//*
//* THE FOLLOWING DATA SETS MUST BE REPLICATED AND
//* SHOULD BE ON NON-SHARED VOLUMES.  IF YOUR XRF
//* CONFIGURATION REQUIRES THAT BOTH IMS SUBSYSTEMS
//* BE EXECUTABLE ON EITHER CEC, THEN THESE DATA
//* SETS MUST BE ON SHARED VOLUMES.
//*
//*     DDNAMES:   IMSSPA
//*                LGMSG, LGMSGL
//*                QBLKS, QBLKSL
//*                SHMSG, SHMSGL
//*
//*                SYSABEND, SYSUDUMP
//*                IMSMON
//*
//* THE FOLLOWING DATA SETS MAY BE REPLICATED.  IF
//* REPLICATED, THESE DATA SETS SHOULD BE ON
//* NON-SHARED VOLUMES.  IF THESE DATA SETS ARE NOT
//* REPLICATED, THEY MUST BE ON SHARED VOLUMES.  IF
//* YOUR XRF CONFIGURATION REQUIRES THAT BOTH IMS
//* SUBSYSTEMS BE EXECUTABLE ON EITHER CEC,
//* THESE DATA SETS MUST BE ON SHARED VOLUMES.
//*
//*     DDNAMES:   FORMATA, FORMATB
//*                IMSACBA, IMSACBB
//*                IMSTFMTA, IMSTFMTB
//*                MODBLKSA, MODBLKSB
//*                PROCLIB, RESLIB
//*                OTHER STEPLIB DATA SETS
//* THE FOLLOWING DATA SETS SHOULD ALSO BE CONSIDERED
//* FOR REPLICATION ON EITHER SHARED OR NON-SHARED
//* VOLUMES (AS APPROPRIATE).
//*
//*     DATA SET:  JOBS   USED IN THE IMSRDR PROCEDURE
//*                PGMLIB
//*                PSBLIB USED BY GSAM
//*                DBDLIB USED BY GSAM
//*
//* THE FOLLOWING IMS DATA SETS MAY ALSO BE IMPACTED
//* BY THE CONFIGURATION OF YOUR XRF COMPLEX.
//*
//* INSTALIB -  USED DURING INSTALLATION/IVP
//* RESLIB   -  CREATED BY SYSDEF - MANAGED BY SMP
//* PROCLIB  -  CREATED BY SYSDEF
//* MACLIB   -  CREATED BY SYSDEF - MANAGED BY SMP
```

```
//* OPTIONS  -  CREATED BY SYSDEF - USED BY SMP
//* OBJDSET  -  CREATED BY SYSDEF
//* MODBLKS  -  CREATED BY SYSDEF - MANAGED BY SMP
//* ACBLIB   -  ONLINE CHANGE STAGING LIBRARY
//* FORMAT   -  ONLINE CHANGE STAGING LIBRARY
//* TFORMAT  -  ONLINE CHANGE STAGING LIBRARY
//* REFERAL  -  USED IN CONJUNCTION WITH FORMAT
//* DBSOURCE -  USED BY SYSDEF - MANAGED BY SMP
//* DCSOURCE -  USED BY SYSDEF - MANAGED BY SMP
//* SRSOURCE -  USED BY SYSDEF - MANAGED BY SMP
//* ADFSMAC  -  USED BY SYSDEF - MANAGED BY SMP
//* LOAD     -  USED BY SYSDEF - MANAGED BY SMP
//* JCLLIB   -  MANAGED BY SMP
//* SMP DATA SETS
//*
//*************************************************
//*                                               *
//*       DATA DEFINITION STATEMENTS FOLLOW       *
//*                                               *
//*************************************************
//*
//******** LIBRARY STATEMENTS ***********************
//*
//STEPLIB  DD  DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD  DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//*
//*************************************************
//*
//* IN ORDER TO START A DEPENDENT REGION, MODIFIED
//* START-UP JCL IS WRITTEN FROM INTERNAL STORAGE TO
//* THE INTERNAL READER.
//*
//IMSIRD   DD SYSOUT=(A,INTRDR)
//*
//*
//*
//*
//******** DASD LOGGING STATEMENTS ******************
//*
//* THE FOLLOWING DD CARDS DESCRIBE THE DASD LOGGING
//* OLDS AND WADS. THESE CARDS ARE FOR EXAMPLE ONLY.
//* ALL OLDS AND WADS DATA SETS MAY BE DYNAMICALLY
//* ALLOCATED.  DD CARDS ARE NOT REQUIRED.
//* THE OLDS AND WADS TO BE USED DURING STARTUP MUST
//* BE SPECIFIED VIA OLDSDEF AND WADSDEF CONTROL
//* STATEMENTS IN THE DFSVSMXX MEMBER OF IMS PROCLIB.
//* THE ACTUAL SELECTION OF OLDS AND WADS MUST BE
//* TAILORED TO INSTALLATION REQUIREMENTS.  THE OLDS
//* AND WADS MUST BE PREDEFINED BY A SET UP JOB.
//* THE BLOCK SIZE OF ALL OLDS MUST BE THE SAME.
//* THE BLOCK SIZE AND DEVICE TYPE OF ALL WADS MUST
//* BE THE SAME.  AT LEAST 3 PRIMARY OLDS AND 1 WADS
//* MUST BE AVAILABLE FOR STARTUP.  THE BLOCK SIZE
//* SHOULD NOT BE SPECIFIED IN THIS JCL.  THE LOGGER
//* WILL GET THE BLOCK SIZE FROM THE VTOC.
//*
//DFSOLP00 DD  DSN=IMS.&SYS.OLP00,DISP=SHR
//DFSOLP01 DD  DSN=IMS.&SYS.OLP01,DISP=SHR
//DFSOLP02 DD  DSN=IMS.&SYS.OLP02,DISP=SHR
//DFSOLP03 DD  DSN=IMS.&SYS.OLP03,DISP=SHR
//DFSOLP04 DD  DSN=IMS.&SYS.OLP04,DISP=SHR
//DFSOLP05 DD  DSN=IMS.&SYS.OLP05,DISP=SHR
//*
//DFSOLS00 DD  DSN=IMS.&SYS.OLS00,DISP=SHR
//DFSOLS01 DD  DSN=IMS.&SYS.OLS01,DISP=SHR
//DFSOLS02 DD  DSN=IMS.&SYS.OLS02,DISP=SHR
//DFSOLS03 DD  DSN=IMS.&SYS.OLS03,DISP=SHR
```

```
//DFSOLS04 DD  DSN=IMS.&SYS.OLS04,DISP=SHR
//DFSOLS05 DD  DSN=IMS.&SYS.OLS05,DISP=SHR
//*
//DFSWADS0 DD  DSN=IMS.&SYS.WADS0,DISP=SHR
//DFSWADS1 DD  DSN=IMS.&SYS.WADS1,DISP=SHR
//*
//******** MONITOR LOGGING STATEMENTS ****************
//*
//* THE IMSMON DD STATEMENT MUST BE REMOVED IF
//* THIS DATA SET IS TO BE DYNAMICALLY ALLOCATED.
//*
//IMSMON   DD  DSN=IMS.&SYS1.IMSMON,DISP=(,KEEP),
// VOL=(,,,99),UNIT=(&LOGT,,DEFER)
//*
//******** MESSAGE QUEUE STATEMENTS ******************
//*
//QBLKS    DD  DSN=IMS.&SYS1.QBLKS,DISP=OLD
//SHMSG    DD  DSN=IMS.&SYS1.SHMSG,DISP=OLD
//LGMSG    DD  DSN=IMS.&SYS1.LGMSG,DISP=OLD
//QBLKSL   DD  DSN=IMS.&SYS1.QBLKSL,DISP=OLD
//SHMSGL   DD  DSN=IMS.&SYS1.SHMSGL,DISP=OLD
//LGMSGL   DD  DSN=IMS.&SYS1.LGMSGL,DISP=OLD
//*
//******** ONLINE CHANGE STATEMENTS ******************
//* THE IMSACBA AND IMSACBB DD STATEMENTS MUST BE
//* REMOVED IF YOU WISH TO DYNAMICALLY ALLOCATE THE
//* ACBLIB DATA SETS THROUGH THE DFSMDA MEMBER.
//*
//IMSACBA  DD  DSN=IMS.&SYS2.ACBLIBA,DISP=SHR
//IMSACBB  DD  DSN=IMS.&SYS2.ACBLIBB,DISP=SHR
//MODBLKSA DD  DSN=IMS.&SYS2.MODBLKSA,DISP=SHR
//MODBLKSB DD  DSN=IMS.&SYS2.MODBLKSB,DISP=SHR
//MODSTAT  DD  DSN=IMS.&SYS.MODSTAT,DISP=SHR
//MODSTAT2 DD  DSN=IMS.&SYS.MODSTAT2,DISP=SHR
//*
//******** MFS STATEMENTS ****************************
//*
//FORMATA  DD  DSN=IMS.&SYS2.FORMATA,DISP=SHR
//FORMATB  DD  DSN=IMS.&SYS2.FORMATB,DISP=SHR
//IMSTFMTA DD DSN=IMS.&SYS2.TFORMAT,DISP=SHR
//         DD DSN=IMS.&SYS2.FORMATA,DISP=SHR
//IMSTFMTB DD DSN=IMS.&SYS2.TFORMAT,DISP=SHR
//         DD DSN=IMS.&SYS2.FORMATB,DISP=SHR
//*
//******** DCCTL SYSTEM STATEMENTS *******************
//*
//IMSRDS   DD  DSN=IMS.&SYS.RDS,DISP=SHR
//IMSRDS2  DD  DSN=IMS.&SYS.RDS2,DISP=SHR
//*DFSTCF  DD  DSN=IMS.TCFSLIB,DISP=SHR
//PRINTDD  DD  SYSOUT=&SOUT
//SYSUDUMP DD  SYSOUT=&SOUT,
// DCB=(LRECL=125,RECFM=FBA,BLKSIZE=3129),
// SPACE=(6050,300,,,ROUND)
//*
//******** TELEPROCESSING LINE STATEMENTS ************
//******** GENERATED FROM DCCTL DEFINITION ***********
//*
//* *** THE GENERATED TELEPROCESSING LINE STATEMENTS WOULD BE HERE ***
//*
//******** EXTERNAL SUBSYSTEM STATEMENTS *************
//*
//*  INCLUDE THE DFSESL DD STATEMENT IF AN EXTERNAL
//*  SUBSYSTEM CONNECTION TO DB2 IS DESIRED.
```

DFSIASN0 procedure

DFSIASN0 is a sample procedure that invokes the IMS abend search and notification function, a diagnostic tool that you can customize so that you receive an informational email if an abend occurs.

Before you can use the DFSIASN0 procedure, you must tailor it, create and populate input data sets for it to use, and perform the IMS abend search and notification setup tasks described in "Setting up the IMS abend search and notification function" on page 245.

JCL

The JCL and SMTP control statements to invoke IMS abend and search and notification follow.

```
//*****************************************
//* DFSIASN0
//* NOTE: THIS MEMBER MUST BE COPIED TO
//* A CONCATENATED Z/OS PROCEDURE LIBRARY
//* FOR THE EVENT-DRIVEN ACTIVATION
//*****************************************
//DFSIASN0   PROC
//           PARM1=,PARM2=
//*
//*************************************
//* REMOVE THE EMAIL DATA SET IF IT EXISTS
//*************************************
//IASN1 EXEC PGM=IEFBR14
//LSTEMAIL DD DSN=USERID.IASN.SDFSSWAN.LSTEMAIL,
//         DCB=(LRECL=255,BLKSIZE=2550,RECFM=FB),
//         DISP=(MOD,DELETE), UNIT=SYSDA,
//         SPACE=(TRK,(1,1))
//*
//*************************************
//* REMOVE THE SMS DATA SET IF IT EXISTS
//*************************************
//IASN1B EXEC PGM=IEFBR14
//LSTSMS   DD DSN=USERID.IASN.SDFSSWAN.LSTSMS,
//         DCB=(LRECL=255,BLKSIZE=2550,RECFM=FB),
//         DISP=(MOD,DELETE),
//         UNIT=SYSDA,
//         SPACE=(TRK,(1,1))
//*
//************************************************
//* WRITE THE FORMATTED EMAIL OUT TO EMAIL DATA SET
//************************************************
//IASN2 EXEC PGM=DFSIASNP,PARM='&amp;PARM1;,&amp;PARM2;'
//      STEPLIB DD DISP=SHR,DSN=STLSERV.QPPTEST.IMS.SDFSRESL
//      SYSUT1  DD DISP=SHR,DSN=RUNTIME.DS(DFSIAEML)
//      URLS    DD DISP=SHR,DSN=RUNTIME.DS(DFSIAURL)
//      CONTROL DD DISP=SHR,DSN=RUNTIME.DS(DFSIACTL)
//      INPARMS DD DISP=SHR,DSN=SKELETON.DS(DFSIAPRM)
//     SYSPRINT DD SYSOUT=*
//      SYSUT2  DD DSN=*.IASN1.LSTEMAIL,DISP=(NEW,CATLG),
//                 DCB=(LRECL=255,BLKSIZE=2550,RECFM=FB),
//                 UNIT=SYSDA,SPACE=(TRK,(1,1))
//*
//*******************************************
//* WRITE THE FORMATTED SMS OUT TO SMS DATA SET
//*******************************************
//IASN2B EXEC PGM=DFSIASNP,PARM='&amp;PARM1;,&amp;PARM2;'
//       STEPLIB DD DISP=SHR,DSN=STLSERV.QPPTEST.IMS.SDFSRESL
//       SYSUT1  DD DISP=SHR,DSN=RUNTIME.DS(DFSIASMS)
//       URLS    DD DISP=SHR,DSN=RUNTIME.DS(DFSIAURL)
```

```
//       CONTROL DD DISP=SHR,DSN=RUNTIME.DS(DFSIACST)
//       INPARMS DD DISP=SHR,DSN=SKELETON.DS(DFSIAPRM)
//      SYSPRINT DD SYSOUT=*
//       SYSUT2  DD DSN=*.IASN1B.LSTSMS,DISP=(NEW,CATLG),
//                  DCB=(LRECL=255,BLKSIZE=2550,RECFM=FB),
//                  UNIT=SYSDA,SPACE=(TRK,(1,1))
//*
//*******************************************
//* SENDS FORMATTED EMAIL OUT THROUGH THE Z/OS SMTP
//*******************************************
//IASN3 EXEC PGM=IEBGENER
//      SYSUT1   DD DISP=SHR,DSN=*.IASN1.LSTEMAIL
//      SYSUT2   DD SYSOUT=(B,SMTP),DCB=(LRECL=255,BLKSIZE=2550,RECFM=FB)
//      SYSPRINT DD SYSOUT=*
//      SYSIN    DD DUMMY
//*
//*******************************************
//* SENDS FORMATTED SMS OUT THROUGH THE Z/OS SMTP
//*******************************************
//IASN3B EXEC PGM=IEBGENER
//       SYSUT1   DD DISP=SHR,DSN=*.IASN1B.LSTSMS
//       SYSUT2   DD SYSOUT=(B,SMTP),DCB=(LRECL=255,BLKSIZE=2550,RECFM=FB)
//       SYSPRINT DD SYSOUT=*
//       SYSIN    DD DUMMY
//*
//*************************************************
//*  Steps IASN3 and IASN3B if external SMTP server
//*  is specified on "System Setup" panel:
//*************************************************
//* SENDS FORMATTED EMAIL OUT THROUGH EXTERNAL SMTP
//*************************************************
//IASN3 EXEC PGM=IRXJCL,PARM='DFSRASNT SMTPHOST.COMPANY.COM 25'
//      SYSEXEC  DD DISP=SHR,DSN=STLSERV.QPPTEST.IMS.SDFSEXEC
//      SYSTSIN  DD  DUMMY
//      SYSTSPRT DD  SYSOUT=*,DCB=LRECL=256
//      SYSTIN   DD  DUMMY
//      TELOUT   DD  SYSOUT=*,DCB=LRECL=256
//      TELIN    DD  DISP=SHR,DSN=*.IASN1.LSTEMAIL
//*
//*********************************************
//* SENDS FORMATTED SMS OUT THROUGH EXTERNAL SMTP
//*********************************************
//IASN3B EXEC PGM=IRXJCL,PARM='DFSRASNT SMTPHOST.COMPANY.COM 25'
//       SYSEXEC  DD DISP=SHR,DSN=STLSERV.QPPTEST.IMS.SDFSEXEC
//       SYSTSIN  DD  DUMMY
//       SYSTSPRT DD  SYSOUT=*,DCB=LRECL=256
//       SYSTIN   DD  DUMMY
//       TELOUT   DD  SYSOUT=*,DCB=LRECL=256
//       TELIN    DD  DISP=SHR,DSN=*.IASN1B.LSTSMS
//*
```

Usage

There are several job steps in DFSIASN0:

- Steps IASN1 and IASN1B clean up the data sets that are used for each invocation of the IMS abend and search and notification function.
- Steps IASN2 and IASN2B parse the parameter string and create the email or text message to be sent.
- Steps IASN3 and IASN3B send the email message and the text message using either the z/OS SMTP server or an external SMTP server, depending on which server is specified during the system setup.

Parameters

PARM1= and PARM2= represent a character string that is passed to the DFSIASN0 procedure either from:

- System setup during an abend invocation
- JCL run deck during an on-demand invocation

This character string contains the input parameters that are passed to the IMS abend search and notification parsing module. The following input parameters can be used with the DFSIASN0 procedure:

ABND=
APAR=
FMID=
GEN=
IMS=
MOD=
MSG=
RC=
SYSID=
T=

For descriptions of these parameters, see "Parameter descriptions for IMS procedures" on page 448.

DD statements

The following DD statements can be used with the DFSIASN0 procedure.

CONTROL DD
INPARMS DD
SYSUT1 DD
SYSUT2 DD
SYSPRINT DD
URLS DD

For descriptions of these DD statements, see "DD statements for IMS procedures" on page 502.

DFSJBP procedure

The DFSJBP procedure starts a Java non-message-driven dependent region that resembles a non-message-driven BMP region (for example, similar procedure parameters and z/OS TCB structure).

The DFSJBP procedure resides in the IMS PROCLIB data set as a standard IMS system definition-supplied procedure and can be invoked as the IMSBATCH procedure is invoked.

The procedure shown in "Procedure to start a JBP region" on page 545 starts a JBP region.

Usage

The DFSJVMAP member of the IMS PROCLIB data set can be used with a DFSJBP procedure. DFSJVMAP maps all the 8-byte or less uppercase Java application names (specified to IMS) to the true OMVS path name for the ".class" file

associated with that Java application. For more information, see "DFSJVMAP member of the IMS PROCLIB data set" on page 714.

Parameters

The following parameters are valid for the DFSJBP procedure.

AGN=
ALTID=
APARM=
CKPTID=
CPUTIME=
DIRCA=
ENVIRON=
IMSID=
JVMOPMAS=
LOCKMAX=
MBR=
NBA=
OBA=
OPT=
OUT=
PARDLI=
PREINIT=
PSB=
RGN=
SOUT=
SPIE=
SYS2=
TEST=

See "Parameter descriptions for IMS procedures" on page 448 for parameter descriptions.

DD statements

The following DD statements are valid for the DFSJBP procedure.

DFSDB2AF DD
JAVAERR DD
JAVAIN DD
JAVAOUT DD
PROCLIB DD
STEPLIB DD
SYSUDUMP DD

See "DD statements for IMS procedures" on page 502 for descriptions.

Restrictions

- Only a Java application can be scheduled in this region.
- The JVMOPMAS=<*member name*> procedure parameter, which sets the name of the member of the IMS PROCLIB data set that contains the JVM options for the standalone JVM, is specific to initializing a JVM and is required.
- The ENVIRON= parameter, which specifies the name of the PROCLIB member that contains the environment settings, is a required parameter.
- The existing IN=, PRLD=, and SSM= parameters on the IMSBATCH procedure are not supported on the DFSJBP procedure.

Procedure to start a JBP region

```
//JBPJOB   JOB 1,IMS,MSGLEVEL=1,PRTY=11,CLASS=K,MSGCLASS=A,REGION=56K
//       EXEC DFSJBP,
//            IMSID=



//       PROC MBR=TEMPNAME,PSB=,JVMOPMAS=,OUT=,
//            OPT=N,SPIE=0,TEST=0,DIRCA=000,
//            STIMER=,CKPTID=,PARDLI=,
//            CPUTIME=,NBA=,OBA=,IMSID=,AGN=,
//            PREINIT=,RGN=56K,SOUT=A,
//            SYS2=,ALTID=,APARM=,ENVIRON=,LOCKMAX=,
//            PRLD=,SSM=
//*
//JBPRGN EXEC PGM=DFSRRC00,REGION=&RGN,
//            PARM=(JBP,&MBR,&PSB,&JVMOPMAS,&OUT,
//            &OPT&SPIE&TEST&DIRCA,
//            &STIMER,&CKPTID,&PARDLI,&CPUTIME,
//            &NBA,&OBA,&IMSID,&AGN,
//            &PREINIT,&ALTID,
//            '&APARM',&ENVIRON,&LOCKMAX,
//            &PRLD,&SSM)
//STEPLIB  DD DSN=IMS.&SYS2.SDFSJLIB,DISP=SHR
//         DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//         DD DSN=CEE.SCEERUN,DISP=SHR
//         DD DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=VBA,BLKSIZE=3129),
//         SPACE=(125,(2500,100),RLSE,,ROUND)
```

DFSJMP procedure

The DFSJMP procedure starts a Java message-driven dependent region that resembles an MPP region (for example, similar procedure parameters and z/OS TCB structure).

The DFSJMP procedure resides in the IMS PROCLIB data set as a standard IMS System Definition-supplied procedure and can be invoked as DFSMPR would be invoked.

The procedure shown in “Sample procedure to start a JMP region” on page 546 starts a JMP region.

Usage

The DFSJVMAP member of the IMS PROCLIB data set can be used with a DFSJMP procedure. DFSJVMAP maps all the 8-byte or less uppercase Java application names (specified to IMS) to the true OMVS path name for the “.class” file associated with that Java application. For more information, see “DFSJVMAP member of the IMS PROCLIB data set” on page 714.

Parameters

The following parameters are valid for the DFSJMP procedure.

AGN=
ALTID=
APARM=
CL1=, CL2=, CL3=, CL4=

ENVIRON=
IMSID=
JVMOPMAS=
LOCKMAX=
NBA=
OBA=
OPT=
OVLA=
PARDLI=
PCB=
PREINIT=
PWFI=
RGN=
SOD=
SOUT=
SPIE=
STIMER=
SYS2=
TLIM=
VALCK=

See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

DD statements

The following DD statements are valid for the DFSJMP procedure.

DFSDB2AF DD
JAVAERR DD
JAVAIN DD
JAVAOUT DD
PROCLIB DD
STEPLIB DD
SYSUDUMP DD

See "DD statements for IMS procedures" on page 502 for descriptions.

Restrictions

- The JMP region can schedule only a Java application.
- The JVMOPMAS=<*member name*> sets the name of the member of the IMS PROCLIB data set that contains the JVM options for the master JVM, and is required.
- The ENVIRON= parameter specifies the name of the member of the IMS PROCLIB data set that contains the environment settings, and is required.
- The existing APPLFE=, DBLDL=, PRLD=, SSM=, VSFX=, and VFREE= parameters on the DFSMPR procedure are not supported on the DFSJMP procedure.

Sample procedure to start a JMP region

```
//       PROC SOUT=A,RGN=56K,SYS2=,
//            CL1=001,CL2=000,CL3=000,CL4=000,
//            OPT=N,OVLA=0,SPIE=0,VALCK=0,TLIM=00,
//            PCB=000,STIMER=,SOD=,
//            NBA=,OBA=,IMSID=,AGN=,
//            PREINIT=,ALTID=,PWFI=N,APARM=,
//            LOCKMAX=,ENVIRON=,
//            JVMOPMAS=,PARDLI=
```

```
//*
//JMPRGN EXEC PGM=DFSRRC00,REGION=&RGN,
//            TIME=1440,DPRTY=(12,0),
//            PARM=(JMP,&CL1&CL2&CL3&CL4,
//            &OPT&OVLA&SPIE&VALCK&TLIM&PCB,
//            &STIMER,&SOD,&NBA,
//            &OBA,&IMSID,&AGN,&PREINIT,
//            &ALTID,&PWFI,'&APARM',&LOCKMAX,
//            &ENVIRON,,&JVMOPMAS,,,&PARDLI)
//*
//STEPLIB  DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//         DD DSN=IMS.&SYS2.SDFSJLIB,DISP=SHR
//         DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=CEE.SCEERUN,DISP=SHR
//         DD DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,BLKSIZE=3129,RECFM=VBA),
//         SPACE=(125,(2500,100),RLSE,,ROUND)
```

DFSMPR procedure

The DFSMPR procedure is an online execution procedure that initiates an IMS message processing address space.

The procedure shown in "Sample procedure to execute an IMS message processing address space" on page 548 executes an IMS message processing address space.

Parameters

The following parameters are valid for the DFSMPR procedure.

AGN=
ALTID=
APARM=
APPLFE=
CL1=, CL2=, CL3=, CL4=
DBLDL=
ENVIRON=
IMSID=
JVMOPMAS=
LOCKMAX=
NBA=
OBA=
OPT=
OVLA=
PARDLI=
PCB=
PREINIT=
PRLD=
PWFI=
RGN=
SOD=
SOUT=
SPIE=
SSM=
STIMER=
SYS2=
TLIM=
VALCK=

VFREE=
VSFX=

See “Parameter descriptions for IMS procedures” on page 448 for descriptions.

DD statements

The following DD statements are valid for the DFSMPR procedure.
DFSESL DD (optional)
FPTRACE DD (optional)
PROCLIB DD
STEPLIB DD
SYSUDUMP DD

See “DD statements for IMS procedures” on page 502 for descriptions.

Sample procedure to execute an IMS message processing address space

```
//       PROC SOUT=A,RGN=56K,SYS2=,
//            CL1=001,CL2=000,CL3=000,CL4=000,
//            OPT=N,OVLA=0,SPIE=0,VALCK=0,TLIM=00,
//            PCB=000,PRLD=,STIMER=,SOD=,DBLDL=,
//            NBA=,OBA=,IMSID=,AGN=,VSFX=,VFREE=,
//             SSM=,PREINIT=,ALTID=,PWFI=N,
//             APARM=,LOCKMAX=,APPLFE=,ENVIRON=,
//            JVMOPMAS=,PARDLI=
//*
//REGION EXEC PGM=DFSRRC00,REGION=&RGN,
//            TIME=1440,DPRTY=(12,0),
//            PARM=(MSG,&CL1&CL2&CL3&CL4,
//            &OPT&OVLA&SPIE&VALCK&TLIM&PCB,
//            &PRLD,&STIMER,&SOD,&DBLDL,&NBA,
//            &OBA,&IMSID,&AGN,&VSFX,&VFREE,
//             &SSM,&PREINIT,&ALTID,&PWFI,
//             '&APARM',&LOCKMAX,&APPLFE,
//            &ENVIRON,&JVMOPMAS,&PARDLI)
//*
//STEPLIB  DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//         DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,BLKSIZE=3129,RECFM=VBA),
//         SPACE=(125,(2500,100),RLSE,,ROUND)
```

DLIBATCH procedure

Use the DLIBATCH procedure to enable DL/I batch jobs to run in a Remote Site Recovery (RSR) environment.

When using the DLIBATCH procedure, be aware that:

- The batch-only databases that are identified to be tracked must be included in the system definition (DATABASE macro) for the tracking subsystem.
- The default GSG name and TMI name used by batch jobs are specified in the IMSCTRL macro, but they can be overridden in the DLIBATCH procedure.
- Batch jobs do not use the DFSRSRxx PROCLIB member, and are therefore limited to the default VTAM modename (TMDEFLT).
- As implied by the absence of IMSDALIB from the list of valid DD statements for the DLIBATCH procedure, IMSDALIB is not supported in batch mode.

"Procedure for an offline DL/I batch processing program using PSB and DBD libraries" on page 550 shows a one-step procedure for an offline DL/I batch processing program using PSB and DBD libraries.

If VSAM databases are used, refer to "IMS buffer pools" on page 179.

Usage

In the example shown in "Procedure for an offline DL/I batch processing program using PSB and DBD libraries" on page 550:

- Parameters in parentheses are positional.
- The IEFRDER statement is not required in DB/DC or DBCTL environments if the job does not declare database update intent.

  For a job step declaring database-update intent, DD DUMMY can be specified if the job step is not using DBRC. This is valid where an image copy of the database is taken before the update job step.

  Log initialization calculates the smallest value necessary for logical record length. If the JCL logical record length value is larger than the calculated value, the JCL value is used; otherwise, log initialization uses the calculated value for logical record length and adds 4 for the block size.

  If multiple volumes are required for the system log, a volume count value should be specified in the VOL parameter of the DD statement.

  If the IBM 3480 tape drive is used for the IMS log data set, IMS forces tape write mode (DCB=OPCD=W). The default on the 3480 is to serve as a buffer for the write so that IMS cannot detect when the write is performed. If a power failure occurs after a log record is written to the 3480, and the database is updated but the log record is not yet written to tape, database integrity is lost. Tape write mode is forced for the log in batch and for GSAM data sets.

Parameters

The following parameters are valid for the DLIBATCH procedure.

APARM=
BKO=
BUF=
CKPTID=
DBRC=
DBRCGRP=
FMTO=
GSGNAME=
IMSID=
IMSPLEX=
OPB=
IRLM=
IRLMNM=
LOCKMAX=
LOGA=
LOGT=
MON=
PRLD=
PSB=
RGN=
RGSUF= (no default when specified in the batch procedure)
RRS=

SOUT=
SRCH=
SSM=
SWAP=
SYS2=
TMINAME=

These parameters cannot be specified in the PARM1= and PARM2= parameters:
EXCPVR=
MBR=
RST=
SPIE=
TEST=

The IOB parameter is no longer used and is ignored if it is specified.

See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

DD statements

In addition to whichever of the following DD statements you use, your procedure must include DD statements for database data sets that will not be dynamically allocated.

The following DD statements are required for the DLIBATCH procedure.
DFSRESLB DD
DFSVSAMP DD
IEFRDER DD
IMS DD
IMSLOGR DD
IMSMON DD
PROCLIB DD
STEPLIB DD
SYSABEND DD
SYSUDUMP DD

The following DD statements are optional for the DLIBATCH procedure.
DFSHALDB DD
DFSSTAT DD
IEFRDER2 DD
SYSHALDB DD

See "DD statement descriptions" on page 502 for descriptions.

Procedure for an offline DL/I batch processing program using PSB and DBD libraries

```
//       PROC MBR=TEMPNAME,PSB=,BUF=7,
//            SPIE=0,TEST=0,EXCPVR=0,RST=0,PRLD=,
//            SRCH=0,CKPTID=,MON=N,LOGA=0,FMTO=T,
//            IMSID=,SWAP=,DBRC=,IRLM=,IRLMNM=,
//             BKO=N,IOB=,SSM=,APARM=,
//            RGN=4M,RGSUF=,PARM1=,PARM2=,
//            SOUT=A,LOGT=2400,SYS2=,
//            LOCKMAX=,GSGNAME=,TMINAME=,
//            RRS=N,IMSPLEX=
//G      EXEC PGM=DFSRRC00,REGION=&RGN,
//            PARM=(DLI,&MBR,&PSB,&BUF,
//            &SPIE&TEST&EXCPVR&RST,&PRLD,
```

```
//            &SRCH,&CKPTID,&MON,&LOGA,&FMTO,
//            &IMSID,&SWAP,&DBRC,&IRLM,&IRLMNM,
//            &BKO,&IOB,&SSM,'&APARM',
//            &LOCKMAX,&GSGNAME,&TMINAME,
//            &RRS,&IMSPLEX,&RGSUF,
//            '&PARM1','&PARM2')
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//DFSRESLB DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//IMS      DD DSN=IMS.&SYS2.PSBLIB,DISP=SHR
//         DD DSN=IMS.&SYS2.DBDLIB,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//IEFRDER  DD DSN=IMSLOG,DISP=(,KEEP),VOL=(,,,99),
//         UNIT=(&LOGT,,DEFER),
//         DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//IEFRDER2 DD DSN=IMSLOG2,DISP=(,KEEP),VOL=(,,,99),
//         UNIT=(&LOGT,,DEFER,SEP=IEFRDER),
//         DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605),
//         SPACE=(605,(500,500),RLSE,,ROUND)
//IMSMON   DD DUMMY
```

DLISAS procedure

The DLISAS procedure initializes a DL/I separate address space (DLISAS).

If you specify LSO=S in the IMS procedure, automatically invoking the DLISAS procedure, the DD statements for the DL/I databases must be in the DLISAS procedure, not in the IMS procedure. Dynamic allocation members remain in the STEPLIB library.

For information about LSO=S, see “Using a DL/I separate address space” on page 71.

The DLISAS procedure is shown in “Sample procedure to initialize a DL/I separate address space” on page 553.

Usage

Other parameters in the IMS procedure that are applicable with the LSO=S option are DLINM, CSAPSB, and DLIPSB. DLINM specifies the partitioned data set (PDS) member name that IMS uses for the DLISAS procedure. The following list shows how the member name is assigned and how the name can be overridden.

1. DLISAS is the default name
2. The value specified on the DLINM= keyword in the IMSCTRL macro overrides the default name
3. The value specified on the DLINM= keyword in the DFSPBxxx member of the IMS PROCLIB data set overrides the value specified in the IMSCTRL macro
4. The value specified on the DLINM= keyword in JCL EXEC parameter overrides the value specified in the DFSPBxxx member of the IMS PROCLIB data set

As specified by the DLINM parameter on the IMSCTRL macro, the cataloged procedure (for the DLISAS) is stored in the IMS PROCLIB data set member during system definition. This member must be copied to SYS1.PROCLIB.

CSAPSB and DLIPSB specify the PSB pool sizes and override the values specified with the SASPSB parameter on the BUFPOOLS system definition macro.

Automatic initiation

An internally defined z/OS START command is issued for the DLINM value and is coded to override the PARM=(DLS,&IMSID) procedure statement. The override defines the IMSID of the subsystem issuing the START, allowing a generic procedure to be used by multiple IMS control regions.

IMS does not issue the `IMS READY` message until the DLISAS successfully initiates. If the started DLISAS procedure is in error, you should correct and restart it.

When the DL/I address space connects to the IMS control region, the DLISAS procedure name is compared with the DLINM value. If they differ, the DLISAS abends.

Manual initiation

You can supply a START command (in member=DLINM of a PDS defined in the control region JCL //PROCLIB) instead of using the internally defined z/OS START command. Columns 1-5 must be "START", or message DFS1930I is issued and the default process is performed.

You can specify two types of START commands:

- `START IMSRDR,MBR=DLISAS`: the specified PDS member is read in as a job (rather than as a started task) through the z/OS internal reader.
- `START DLIAA,PARM=(DLS,IMSA)`: DL/I executes as a started task and DLIAA connects to IMSA.

Recommendation: Run your DL/I region as a started task, and specify the NODETAIL parameter in the SMFPRMxx member in SYS1.PARMLIB. Do not run your DL/I region as a job. While IMS supports running DL/I as a job, doing so can lead to storage shortages for long running DL/I regions that are managing a large number of databases. This is because z/OS SMF captures performance information in control blocks, which are kept in storage until the job terminates. These control blocks accumulate over time. A DL/I region that is active for a long period of time, and which is managing a large number of databases, can eventually run out of storage due to the SMF measurement data that is collected. This storage buildup does not occur with started tasks when NODETAIL is specified in the SMFPRMxx member.

For active and inactive ACBLIBs, the data set names and their concatenation order must be identical (same DSN and VOLSER) in the DLISAS procedure and in the IMS procedure.

The IMS PROCLIB data set must be defined in the DLISAS procedure and in the IMS procedure. Sometimes the same DLINM member is read from the IMS PROCLIB data set by both the control region and DLISAS.

Important: If the resources (such as DL/I databases) are RACF-protected, the user ID associated with DLISAS procedure must be authorized to access them. For more information about how to authorize the user ID, see *IMS Version 11 System Administration*.

Parameters

The following parameters are valid for the DLISAS procedure.

DPRTY=
IMSID=
RGN=
SOUT=
SYS2=

See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

DD statements

The following DD statements are valid for the DLISAS procedure.

In addition to these DD statements, DD statements for DL/I databases must be in the DLISAS procedure, not in the IMS procedure; add statements for data sets representing databases that are not to be dynamically allocated. DD statements for Fast Path databases remain in the IMS procedure.

IMSACBA DD
IMSACBB DD
PROCLIB DD
STEPLIB DD
SYSUDUMP DD

The IMSACBA and IMSACBB DD statements must be removed if you want to dynamically allocate the ACBLIB data sets through the DFSMDA member.

See "DD statement descriptions" on page 502 for descriptions.

Sample procedure to initialize a DL/I separate address space

```
//       PROC RGN=64M,DPTY='(14,15)',SOUT=A,
//            IMSID=SYS3,SYS2=
//IEFPROC EXEC PGM=DFSMVRC0,REGION=&RGN,
//            DPRTY=&DPTY,PARM=(DLS,&IMSID)
//*************************************************
//*
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//****************** ACBLIB **************************
//*
//* THE SPECIFICATION OF THE ACBLIB DATASETS
//* IN THE DLI/SAS REGION PROCEDURE MUST
//* CORRESPOND EXACTLY WITH THE SPECIFICATION
//* IN THE CONTROL REGION JCL. THE IMSACBA AND
//* IMSACBB DD STATEMENTS MUST BE REMOVED IF YOU
//* WISH TO DYNAMICALLY ALLOCATE THE ACBLIB DATA
//* SETS THROUGH THE DFSMDA MEMBER.
//*
//IMSACBA  DD DSN=IMS.&SYS2.ACBLIBA,DISP=SHR
//IMSACBB  DD DSN=IMS.&SYS2.ACBLIBB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT
//SYSABEND DD SYSOUT=&SOUT
//************ DATA BASE DD CARDS ********************
//*
//* USER MAY OPTIONALLY SUPPLY THE DD STATEMENTS
//* FOR THE ON-LINE DATA BASES TO BE
//* INSERTED HERE PRIOR TO ATTEMPTING
//* AN ON-LINE SYSTEM EXECUTION USING
//* THIS PROCEDURE.
```

```
//* IF NO DD STATEMENTS ARE SUPPLIED FOR
//* A DATA BASE, IMS ASSUMES THAT THIS
//* DATA BASE HAS BEEN DESCRIBED THROUGH
//* THE DFSMDA MACRO.
```

DXRJPROC procedure

The DXRJPROC procedure starts the internal resource lock manager (IRLM). This procedure is supplied on the IRLM distribution tape.

For information about obtaining the IRLM from the distribution tape and preparing it for use, see *IMS Version 11 Installation*.

The procedure shown in "Sample procedure to start IRLM 2.2" on page 555 executes IRLM 2.2.

Usage

When you install the IRLM, a unique copy of DXRJPROC should be created for each IRLM that might run concurrently. The procedure name and parameters should be changed, as required, for each IRLM. Because no default value is provided for the IRLMID parameter, the DXRJPROC procedure must at least be altered to assign a value for IRLMID.

In a configuration in which two systems are sharing data, the two IRLMs must execute concurrently. A typical way to define this configuration is:

- Procedure name IRLM1 with IRLMID=1
- Procedure name IRLM2 with IRLMID=2

For testing purposes, two IRLMs can communicate with one another by using the z/OS cross-system coupling facility while executing concurrently on a single system. A typical way to define this environment is:

- Procedure name JRLM1 with IRLMID=1 and IRLMNM=JRLM
- Procedure name KRLM8 with IRLMID=8 and IRLMNM=KRLM

IRLM has a global deadlock manager, which is designated as the IRLM with the lowest IRLMID. IRLMs in a Parallel Sysplex group dynamically readjust the global manager identity as members join and leave the group. If you are concerned about the placement of the global deadlock manager, you can put the lowest IRLMID on a specific processor.

Parameters

The following parameters are valid for the DXRJPROC procedure.

DEADLOK=
IRLMGRP=
IRLMID=
IRLMNM=
LOCKTAB=
LTE=
MAXCSA= (ignored by IRLM)
MAXUSRS=
PC= (IRLM sets PC=YES regardless of what is specified in the DXRJPROC procedure.)
PGPROT=

SCOPE=
TRACE=

See “Parameter descriptions for IMS procedures” on page 448 for descriptions.

DD statements

The following DD statements are valid for the DXRJPROC procedure:
STEPLIB DD
SYSABEND DD

See “DD statement descriptions” on page 502 for descriptions.

Sample procedure to start IRLM 2.2

```
//DXRJPROC PROC  RGN=3072K,
//            IRLMNM=IRLM,
//            IRLMID=,
//            SCOPE=LOCAL,
//            DEADLOK='5,1',
//            MAXCSA=8,
//            PC=NO,
//            MAXUSRS=,
//            IRLMGRP=,
//            LOCKTAB=IRLMT1,
//            TRACE=YES,
//            PGPROT=YES,
//            LTE=
//         EXEC PGM=DXRRLM00,DPRTY=(15,15),
//         PARM=(&IRLMNM,&IRLMID,&SCOPE,&DEADLOK,&MAXCSA,
//         &PC,&MAXUSRS,&IRLMGRP,&LOCKTAB,&TRACE,&PGPROT,&LTE),
//         REGION=&RGN
//STEPLIB  DD DSN=SDXRLM21.SDXRRESL,DISP=SHR
//*
//* The following DUMP dd statement should not be specified unless
//* you are having IRLM STARTUP problems and are not getting the
//* dump needed to diagnosis the problem.
//*
//SYSABEND DD SYSOUT=A
```

FDR procedure

The FDR procedure executes a Fast Database Recovery (FDBR) address space.

Usage

For the FDR procedure, the sum of the values for the CSAPSB and DLIPSB defines the PSB pool size. If PSB is also specified, the larger value (PSB or the sum of CSAPSB and DLIPSB) is used.

If the Fast Path 64-bit buffer manager is being used, the DFSDF= parameter must be specified on the FDR procedure.

Parameters

The following parameters are valid for the FDR procedure.
ARC=
ARMRST=
BSIZ=
CSAPSB=
CSLG=

DBBF=
DBRCGRP=
DBWP=
DESC=
DFSDF=
DLIPSB=
DMB=
DPRTY=
FMTO=
FP=
IMSID=
IMSPLEX=
IRLMNM=
LGNR=
MCS=
PSB=
RGN=
RGSUF=
SOUT=
SPM=
SUF=
SVC2=
SYS=
SYS2=
UHASH=
VSPEC=
WADS=
WKAP=

See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

DD statements

The following DD statements are valid for the FDR procedure. In addition to the following DD statements, add statements for data sets that are not to be dynamically allocated.

DFSOLPn DD
DFSOLSn DD
DFSWADSn DD
IMSACBA DD
IMSACBB DD
IMSRDS DD
JCLOUT DD
JCLPDS DD
MODBLKSA DD (optional)
MODBLKSB DD (optional)
MODSTAT DD
PROCLIB DD
STEPLIB DD
SYSUDUMP DD

Note: The IMSACBA and IMSACBB DD statements must be removed if you want to dynamically allocate the ACBLIB data sets through the DFSMDA member.

See "DD statement descriptions" on page 502 for descriptions.

JCL

The FDR procedure shown in the example executes an FDBR address space:

```
//       PROC RGN=64M,SOUT=A,DPTY='(14,15)',
//            SYS=,SYS2=,
//            RGSUF=IMS,PARM1=,PARM2=
//IEFPROC EXEC PGM=DFSMVRC0,DPRTY=&DPTY,
//            REGION=&RGN,
//            PARM='FDR,&RGSUF,&PARM1,&PARM2'
//*
//*
//* THE MEANING AND MAXIMUM SIZE OF EACH PARAMETER
//* IS AS FOLLOWS:
//*
//******** CONTROL REGION SPECIFICATIONS ********
//*****************************************************
//*   RGSUF  XXX   EXEC PARM DEFAULT BLOCK SUFFIX FOR
//*                MEMBER DFSPBXXX.
//*****************************************************
//*
//*  PARM1 , PARM2 PARAMETERS BOTH ARE USED TO SPECIFY
//*  CHARACTER STRINGS THAT CONTAIN IMS KEYWORD
//*  PARAMETERS. I.E. PARM1='AUTO=Y,PST=222,RES=Y'
//*
//*  ALL OF THE VALID IMS KEYWORD PARAMETERS
//*  ARE DESCRIBED BELOW
//*****************************************************
//*   FMTO         T = ONLINE FORMATTED DUMP WITH
//*                    STORAGE IMAGE DELETIONS.
//*                    OFFLINE SDUMPS PERMITTED FOR
//*                    NON-IMS TERMINATING ERRORS.
//*                P = FULL ONLINE FORMATTED DUMP.
//*                    OFFLINE SDUMPS PERMITTED FOR
//*                    NON-IMS TERMINATING ERRORS.
//*                F = FULL ONLINE FORMATTED DUMP.
//*                    OFFLINE SDUMPS SUPPRESSED FOR
//*                    NON-IMS TERMINATING ERRORS.
//*                N = NO FORMATTED DUMP, NO OFFLINE
//*                    DUMP. OFFLINE SDUMPS PERMITTED
//*                    FOR NON-IMS TERMINATING ERRORS
//*                Z = NO FORMATTED DUMP, NO OFFLINE
//*                    DUMP. OFFLINE SDUMPS
//*                    SUPPRESSED FOR NON-IMS
//*                    TERMINATING ERRORS.
//*  (DEFAULT)     D = OFFLINE DUMP, OR ONLINE FORMAT-
//*                    TED DUMP WITH STORAGE IMAGE
//*                    DELETIONS IF OFFLINE DUMPING
//*                    FAILS.  OFFLINE SDUMPS
//*                    PERMITTED FOR NON-IMS
//*                    TERMINATING ERRORS.
//*                X = OFFLINE DUMP, OR ONLINE FORMAT-
//*                    TED DUMP WITH STORAGE IMAGE
//*                    DELETIONS IF OFFLINE DUMPING
//*                    FAILS. OFFLINE SDUMPS
//*                    SUPPRESSED FOR NON-IMS
//*                    TERMINATING ERRORS.
//*                M = OFFLINE DUMP, ONLINE IMS DUMP
//*                    FORMATTING NOT PERMITTED.
//*                    OFFLINE SDUMPS PERMITTED FOR
//*                    NON-IMS TERMINATING ERRORS.
//*                R = OFFLINE DUMP, ONLINE IMS DUMP
//*                    FORMATTING NOT PERMITTED.
//*                    OFFLINE SDUMPS SUPPRESSED FOR
//*                    NON-IMS TERMINATING ERRORS.
//*   IMSID  XXXX  IMS SUBSYSTEM IDENTIFIER
//*   ARMRST X     Y = ALLOW MVS ARM TO RESTART
```

```
//*               N = ARM NOT RESTART IMS
//*   IRLMNM XXXX  IRLM SUBSYSTEM NAME
//*   WADS   X     SINGLE OR DUAL WADS,S=SINGLE,D=DUAL
//*   ARC    XX    AUTOMATIC ARCHIVE.
//*               0 = NOT AUTOMATIC
//*               1-99 = AUTOMATIC
//*   UHASH  XXXXXXXX USER HASH MODULE NAME
//*   IMSPLEX XXXXX IMSPLEX NAME
//*   DESC    XX    MSG DESC CODE
//*   MCS     (XX,XX) MSG ROUTE CODES
//*   SVC2    XXX   TYPE 2 SVC NUMBER
//*   DBRCGRP XXX  DBRC SHARING GROUP ID
//******** FAST PATH PARAMETERS ********
//*
//*   BSIZ   XXXXX DATA BASE BUFFER SIZE
//*   DBBF   XXXXX NUMBER OF DATABASE BUFFERS
//*   LGNR   XX    NUMBER OF LOG ENTRIES IN DEDB BUFFERHEADER
//******** STORAGE POOL VALUES IN K, M OR G *******
//*
//*   PSB    XXXXXX PSB POOL SIZE - NON DLISAS
//*   DMB    XXXXXX DMB POOL SIZE
//*   WKAP   XXXXXX WORKING STORAGE BUFFER POOL SIZE
//*   DBWP   XXXXXX DATABASE WORK POOL SIZE
//*   CSAPSB XXXXXX DLISAS: CSA PSB POOL SIZE
//*   DLIPSB XXXXXX DLISAS: DLI PSB POOL SIZE
//*
//******** MEMBER SUFFIXES **********************
//*
//*   SUF    X     LAST CHARACTER OF CTL PROGRAM LOAD
//*               MODULE MEMBER NAME
//*   VSPEC  XX    2 CHARACTER BUFFER POOL SPEC MODULE SUFFIX
//*   SPM    XX     STG POOL MGR PROCLIB MEMBER SUFFIX
//*   CSLG   XXX CSL GLOBAL MEMBER (DFSCGXXX)
//*
//*********************************************
//*
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//JCLOUT   DD SYSOUT=(A,INTRDR)
//JCLPDS   DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//*
//*********** DASD  LOGGING DD CARDS *************
//* THE FOLLOWING DD CARDS DESCRIBE THE DASD LOGGING
//* OLDS AND WADS. THESE CARDS ARE FOR EXAMPLE ONLY.
//* ALL OLDS AND WADS DATA SETS MAY BE DYNAMICALLY
//* ALLOCATED.  DD CARDS ARE NOT REQUIRED.
//* THE OLDS AND WADS TO BE USED DURING STARTUP MUST
//* BE SPECIFIED VIA OLDSDEF AND WADSDEF CONTROL
//* STATEMENTS IN THE DFSVSMXX MEMBER OF IMS PROCLIB.
//* THE ACTUAL SELECTION OF OLDS AND WADS MUST BE
//* TAILORED TO INSTALLATION REQUIREMENTS.  THE OLDS
//* AND WADS MUST BE PREDEFINED BY A SET UP JOB.
//* THE BLOCK SIZE OF ALL OLDS MUST BE THE SAME.
//* THE BLOCK SIZE AND DEVICE TYPE OF ALL WADS MUST
//* BE THE SAME.  AT LEAST 3 PRIMARY OLDS AND 1 WADS
//* MUST BE AVAILABLE FOR STARTUP.  THE BLOCK SIZE
//* SHOULD NOT BE SPECIFIED IN THIS JCL.  THE LOGGER
//* WILL GET THE BLOCK SIZE FROM THE VTOC.
//* THE IMSACBA AND IMSACBB DD STATEMENTS MUST BE
//* REMOVED IF YOU WISH TO DYNAMICALLY ALLOCATE THE
//* ACBLIB DATA SETS THROUGH THE DFSMDA MEMBER.
//*
//DFSOLP00 DD DSN=IMS.&SYS.OLP00,DISP=SHR
//DFSOLP01 DD DSN=IMS.&SYS.OLP01,DISP=SHR
//DFSOLP02 DD DSN=IMS.&SYS.OLP02,DISP=SHR
//DFSOLP03 DD DSN=IMS.&SYS.OLP03,DISP=SHR
//DFSOLP04 DD DSN=IMS.&SYS.OLP04,DISP=SHR
```

```
//DFSOLP05 DD DSN=IMS.&SYS.OLP05,DISP=SHR
//*
//DFSOLS00 DD DSN=IMS.&SYS.OLS00,DISP=SHR
//DFSOLS01 DD DSN=IMS.&SYS.OLS01,DISP=SHR
//DFSOLS02 DD DSN=IMS.&SYS.OLS02,DISP=SHR
//DFSOLS03 DD DSN=IMS.&SYS.OLS03,DISP=SHR
//DFSOLS04 DD DSN=IMS.&SYS.OLS04,DISP=SHR
//DFSOLS05 DD DSN=IMS.&SYS.OLS05,DISP=SHR
//*
//DFSWADS0 DD DSN=IMS.&SYS.WADS0,DISP=SHR
//DFSWADS1 DD DSN=IMS.&SYS.WADS1,DISP=SHR
//*
//IMSACBA  DD DSN=IMS.&SYS2.ACBLIBA,DISP=SHR
//IMSACBB  DD DSN=IMS.&SYS2.ACBLIBB,DISP=SHR
//MODBLKSA DD DSN=IMS.&SYS2.MODBLKSA,DISP=SHR
//MODBLKSB DD DSN=IMS.&SYS2.MODBLKSB,DISP=SHR
//MODSTAT  DD DSN=IMS.&SYS.MODSTAT,DISP=SHR
//******** SYSTEM STATEMENTS ************************
//*
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=125,RECFM=FBA,BLKSIZE=3129),
//         SPACE=(6050,300,,,ROUND)
//IMSRDS   DD DSN=IMS.&SYS.RDS,DISP=SHR
//*
//************ DATA BASE DD CARDS ***************
//*
//* USER MAY OPTIONALLY SUPPLY THE DD STATEMENTS
//* FOR THE ON-LINE DATA BASES TO BE
//* INSERTED HERE PRIOR TO ATTEMPTING
//* AN ON-LINE SYSTEM EXECUTION USING
//* THIS PROCEDURE.
//* IF NO DD STATEMENTS ARE SUPPLIED FOR
//* A DATA BASE, IMS ASSUMES THAT THIS
//* DATA BASE HAS BEEN DESCRIBED THROUGH
//* THE DFSMDA MACRO.
//* IF THE USER WILL BE EXECUTING WITH THE DL/I
//* SAS OPTION, THESE DD STATEMENTS SHOULD BE ADDED
//* TO THE DLISAS PROCLIB MEMBER OR DESCRIBED
//* THROUGH THE DFSMDA MACRO.
//*
//************ DBRC RECON DD CARDS **************
//*
//* USER MAY OPTIONALLY SUPPLY THE DD CARDS
//* REQUIRED FOR THE DBRC RECON DATA SET.
//* IF NO DD STATEMENTS ARE SUPPLIED FOR RECON
//* DATASETS, IMS ASSUMES THAT THE DATASETS
//* HAVE BEEN DESCRIBED THROUGH THE DFSMDA MACRO.
```

FPUTIL procedure

The FPUTIL procedure executes the Fast Path utility programs with Data Entry Database (DEDB) online.

The JCL for the FPUTIL procedure is shown in "Sample procedure to execute Fast Path utility programs" on page 560. Parameters in parentheses are positional.

Parameters

The following parameters are valid for the FPUTIL procedure.

AGN=
ALTID=
DBD=
DIRCA=

IMSID=
PRLD=
REST=
RGN=
SOUT=
SSM=
SYS2=

See “Parameter descriptions for IMS procedures” on page 448 for descriptions.

DD statements

The following DD statements are valid for the FPUTIL procedure.

PROCLIB DD
STEPLIB DD
SYSPRINT DD
SYSUDUMP DD

See “DD statement descriptions” on page 502 for descriptions.

Restrictions

NBA= and OBA= are not valid parameters for this procedure. Instead, the default values NBA=7 and OBA=0 are used.

Sample procedure to execute Fast Path utility programs

```
//FPUTIL PROC SOUT=A,RGN=4M,SYS2=,
//            DBD=,REST=00,DIRCA=002,
//            PRLD=,IMSID=,AGN=,SSM=,ALTID=
//FPU   EXEC  PGM=DFSRRC00,REGION=&RGN,
//      PARM=(IFP,&DBD,DBF#FPU0,&REST,00,,1,
//      &DIRCA,&PRLD,0,,,,&IMSID,&AGN,&SSM,,
//      &ALTID)
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&SOUT
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=VBA,BLKSIZE=3129),
//         SPACE=(125,(2500,100),RLSE,,ROUND)
```

IMS procedure

The IMS procedure is an online execution procedure that initializes the IMS DB/DC environment.

Usage

In the sample shown in “JCL” on page 563:

- The PARM2 parameter is needed because z/OS:
  - Allows up to 100 characters in the EXEC parameter area
  - Does not allow symbolic parameters to be continued on next record
  - Does not allow more than 66 characters for each keyword option phrase on the START command
- The XRF parameters are used for an XRF-capable system only.
- The RSR parameters are used for an RSR-capable system only.

- On the Monitor statements, specify DCB=BLKSIZE=*nnnnn* to ensure good performance when running the IMS Monitor. Make a block size as large as possible for your environment. A value greater than 20000 is typically optimal. If DCB=BLKSIZE is not specified, the default block size is 1048 for a JCL-allocated IMSMON data set, and any data set block size value (in the DSCB) is ignored. The default block size value is likely to cause performance degradation for a busy IMS system.
- The IMSMON DD statement is produced only if requested in the IMSCTF macro.
- For the Fast Path statements, the DEDB areas must be allocated in the IMS control region address space, regardless of whether the DL/I SAS option is used. Allocation is attempted in the following order:
  - The DD statements are in the control region JCL
  - DEDB dynamic allocation from the RECON data set
  - Dynamic allocation members in the IMSDALIB concatenation or JOBLIB/STEPLIB concatenation

  Recommendation: Register the databases with DBRC and do not use DFSMDA.
- The security parameters AGN=, AOI1=S, and ISIS=< 0 | 1 |2 > continue to be accepted, but are ignored if specified. The security parameters RCF=, SGN=, and TRN= are no longer documented, but continue to perform the function that was provided in previous versions of IMS.

Dynamic resource definition

If you are defining your resources dynamically rather than using online change, the IMS procedure does not require the DD statements for the IMS.MODBLKS data sets, MODBLKSA and MODBLKSB. In addition, if the MODBLKSA and MODBLKSB DD statements are defined to an IMS system that was initially defined with dynamic resource definition and global online change enabled, they are ignored.

RACF and resource protection

If the resources (such as DL/I databases) are RACF protected, the user ID associated with the IMS procedure must be authorized to access them. For more information about how to authorize the user ID, see *IMS Version 11 System Administration*.

DLISAS procedure

If you specify LSO=S in the IMS procedure, automatically invoking the DLISAS procedure, the DD statements for the DL/I databases must be in the DLISAS procedure, not in the IMS procedure. Dynamic allocation members remain in the STEPLIB library.

Online change

IMSACB DD statements can optionally be added to the IMS procedure for the IMS staging ACBLIB that is used by ACBLIB member online change. If IMSACB DD statements are added, a dynamic allocation (DFSMDA) member for the IMS staging ACBLIB is not required.

Program Specification Block (PSB) pools

If you do not specify LSO=S in the IMS procedure, one PSB pool is created, as specified by the PSB parameter on the BUFPOOLS macro. If you specify LSO=S, two PSB pools are created and the PSB parameter is ignored. The pool sizes are specified with the SASPSB parameter on the BUFPOOLS system definition macro, and overridden with the CSAPSB and DLIPSB parameters on the IMS procedure. For an overview of information about the DL/I address space options, see *IMS Version 11 System Administration*.

Allocating terminal devices

If an online IMS system has been defined, carefully consider the terminal device allocation generated within the IMS procedure. A list of terminal addresses and logical and physical terminals is printed by stage 1 of IMS system definition. Examples of the procedure jobs in this information show the contents of the members as they are supplied. No card column image is intended. When coding your own procedures, follow JCL and assembler language coding practices.

Parameters

The following parameters are valid for the IMS procedure.

ALOT= to CSLG=	DBBF= to EXVR=	FBP= to ISIS=	LGMSGSZ= to PIMAX=	PRDR= to SHAREDQ=	RECASZ= to YEAR4
ALOT=	DBBF=	FBP=	LGMSGSZ=	PRDR=	SHMSGSZ=
AOI1=	DBFP=	FDRMBR=	LGNR=	PRLD=	SOD=
AOIP=	DBFX=	FESTIM=	LHTS=	PSB=	SOUT=
AOIS=	DBRCNM=	FIX=	LSO=	PSBW=	SPM=
APPC=	DBWP=	FMTO=	LTERM=	PST=	SRCH=
APPCSE=	DC=	FP=	LUMC=	PSWDC=	SSM=
APPLID1=	DESC=	FPDSSIZE=	LUMP=	QBUF=	SUF=
APPLID2=	DFSDF=	FPOPN=	MAXPST=	QBUFHITH=	SVC2=
APPLID3=	DLIDSIZE=	FPRLM=	MCS=	QBUFLWTH=	SVSODR=
ARC=	DLINM=	FPWP=	MNPS=	QBUFMAX=	SYS=
ARMRST=	DLIPSB=	FRE=	MNPSW=	QBUFSZ=	SYS1=
ASOT=	DLQT=	GRNAME=	MSDB=	QTL=	TCORACF=
AUTO=	DMB=	GRSNAME=	NHTS=	QTU=	TRACK=
BSIZ=	DMHVF=	HIOP=	NLXB=	RCF=	TRN=
CCTCVCAN=	DPRTY=	HSBID=	ODBASE=	RCFTCB=	TSR=
CIOP=	DSCT=	HSBMBR=	ORSMBR=	RCLASS=	UHASH=
CMDMCS=	DYNP=	IMSGROUP=	OTHR=	RECA=	UHTS=
CPLOG=	EMHB=	IMSID=	OTMA=	RECASZ=	USERVAR=
CRC=	EMHL=	IOVFI=	OTMAASY=	RES=	VAUT=
CSAPSB=	EPCB=	IRLM=	OTMAMD=	RGN=	VSPEC=
CSLG=	ETO=	IRLMNM=	OTMANM=	RGSUF=	WADS=
	EXVR=	ISIS=	OTMASE=	RRS=	WKAP=
			OTMASP=	RSRMBR=	YEAR4
			PASSWD1=	RVFY=	
			PIINCR=	SAV=	

ALOT= to CSLG=	DBBF= to EXVR=	FBP= to ISIS=	LGMSGSZ= to PIMAX=	PRDR= to SHAREDQ=	RECASZ= to YEAR4
			PIMAX=	SGN=	
				SHAREDQ=	

The following parameters are Fast Path parameters: BSIZ, DBBF, DBFP, DBFX, DMHVF, EPCB, FPOPN, FPRLM, IOVFI, LGNR, MSDB, OTHR, and UHASH.

See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

DD statements

The following parameters are valid for the IMS procedure.

In addition to the following DD statements, add statements for data sets representing IMS databases that are not to be dynamically allocated.

DFSDUPPM to IMSACBA	IMSACBB to MODBLKSB	MODSTAT to PROCLIB	QBLKS to SYSUDUMP
DFSDUPPM DD	IMSACBB DD (optional)	MODSTAT DD	QBLKS DD
DFSESL DD (optional)	IMSMON DD	MODSTAT2 DD	QBLKSL DD
DFSOLPnn DD	IMSRDS DD	MSDBCP1 DD	RECON1 DD (optional)
DFSOLSnn DD	IMSRDS2 DD	MSDBCP2 DD	RECON2 DD (optional)
DFSTCF DD (optional)	IMSTFMTA DD	MSDBCP3 DD	RECON3 DD (optional)
DFSWADSn DD	IMSTFMTB DD	MSDBCP4 DD	SHMSG DD
FORMATA DD	LGMSG DD	MSDBDUMP DD	SHMSGL DD
FORMATB DD	LGMSGL DD	MSDBINIT DD	STEPLIB DD
IMSACB DD	MODBLKSA	PRINTDD DD	SYSTPRT DD
IMSACBA DD (optional)	MODBLKSB	PROCLIB DD	SYSUDUMP DD

Note: The IMSACBA and IMSACBB DD statements must be removed if you want to dynamically allocate the ACBLIB data sets through the DFSMDA member.

Note: If you are defining your resources dynamically, the DD statements for the MODBLKS data sets (MODBLKSA and MODBLKSB) are not required.

See "DD statement descriptions" on page 502 for descriptions.

JCL

The following procedure executes an IMS DB/DC online control program.

```
//       PROC RGN=64M,SOUT=A,DPTY='(14,15)',
//            SYS=,SYS1=,SYS2=,
//            LOGT=2400,
//            RGSUF=IMS,PARM1=,PARM2=
//IEFPROC EXEC PGM=DFSMVRC0,DPRTY=&DPTY,
//            REGION=&RGN,
//            PARM='CTL,&RGSUF,&PARM1,&PARM2'
//*
//*
//* THE MEANING AND MAXIMUM SIZE OF EACH PARAMETER
//* IS AS FOLLOWS:
//*
```

```
//********** CONTROL REGION SPECIFICATIONS ***********
//*************************************************
//*   RGSUF   XXX   EXEC PARM DEFAULT BLOCK SUFFIX FOR
//*                 MEMBER DFSPBXXX.
//*************************************************
//*
//*  PARM1 , PARM2 PARAMETERS BOTH ARE USED TO SPECIFY
//*  CHARACTER STRINGS THAT CONTAIN IMS KEYWORD
//*  PARAMETERS. I.E. PARM1='AUTO=Y,PST=222,RES=Y'
//*
//*  ALL OF THE VALID IMS KEYWORD PARAMETERS
//*  ARE DESCRIBED BELOW
//*************************************************
//*   APPLID1 XXXXXXXX VTAM APPLID OF ACTIVE IMS SYSTEM
//*   APPLID2 XXXXXXXX VTAM APPLID OF XRF ALTERNATE SYSTEM
//*   APPLID3 XXXXXXXX VTAM APPLID OF RSR TRACKING SYSTEM
//*   RES     X     BLOCK RESIDENT (N = NO, Y = YES)
//*   FRE     XXXXX NUMBER OF FETCH REQUEST ELEMENTS
//*   PST     XXX   NUMBER OF PST'S PERMANENTLY ALLOC
//*   MAXPST  XXX   MAXIMUM NUMBER OF PST'S
//*   SAV     XXX   NUMBER OF DYNAMIC SAVE AREA SETS
//*   SRCH    X     MODULE SEARCH INDICATOR FOR DIRECTED LOAD
//*                 0 = STANDARD SEARCH
//*                 1 = SEARCH JPA AND LPA BEFORE PDS
//*   SOD     X     1 CHARACTER SYSOUT CLASS
//*   VAUT    X     VTAM AUTH PATH OPTION (1=YES,0=NO)
//*   FMTO          T = ONLINE FORMATTED DUMP WITH
//*                     STORAGE IMAGE DELETIONS.
//*                     OFFLINE SDUMPS PERMITTED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 P = FULL ONLINE FORMATTED DUMP.
//*                     OFFLINE SDUMPS PERMITTED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 F = FULL ONLINE FORMATTED DUMP.
//*                     OFFLINE SDUMPS SUPPRESSED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 N = NO FORMATTED DUMP, NO OFFLINE
//*                     DUMP. OFFLINE SDUMPS PERMITTED
//*                     FOR NON-IMS TERMINATING ERRORS
//*                 Z = NO FORMATTED DUMP, NO OFFLINE
//*                     DUMP. OFFLINE SDUMPS
//*                     SUPPRESSED FOR NON-IMS
//*                     TERMINATING ERRORS.
//*  (DEFAULT)      D = OFFLINE DUMP, OR ONLINE FOR-
//*                     MATTED DUMP WITH STORAGE IMAGE
//*                     DELETIONS IF OFFLINE DUMPING
//*                     FAILS.  OFFLINE SDUMPS
//*                     PERMITTED FOR NON-IMS
//*                     TERMINATING ERRORS.
//*                 X = OFFLINE DUMP, OR ONLINE FOR-
//*                     MATTED DUMP WITH STORAGE IMAGE
//*                     DELETIONS IF OFFLINE DUMPING
//*                     FAILS. OFFLINE SDUMPS
//*                     SUPPRESSED FOR NON-IMS
//*                     TERMINATING ERRORS.
//*                 M = OFFLINE DUMP, ONLINE IMS DUMP
//*                     FORMATTING NOT PERMITTED.
//*                     OFFLINE SDUMPS PERMITTED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*                 R = OFFLINE DUMP, ONLINE IMS DUMP
//*                     FORMATTING NOT PERMITTED.
//*                     OFFLINE SDUMPS SUPPRESSED FOR
//*                     NON-IMS TERMINATING ERRORS.
//*   AUTO    X     Y = AUTOMATIC RESTART DESIRED
//*                 N = NO AUTOMATIC RESTART
//*   IMSID   XXXX  IMS SUBSYSTEM IDENTIFIER
//*   NLXB    XXX   # ADD'L LXBS FOR MSC VTAM
```

```
//*   LSO     X     Y = DL/I LOCAL STORAGE OPTION ON
//*                 S = DLI/SAS OPTION
//*
//*
//*   APPC    X     Y = ACTIVATE APPC/IMS
//*                 N = DO NOT ACTIVATE APPC/IMS
//*   LTERM   X     Y = LTERM USED IN DFSAPPC PROCESS
//*                 N = LTERM NOT USED IN DFSAPPC
//*                     PROCESS
//*   ARMRST  X     Y = ALLOW MVS ARM TO RESTART
//*                 N = ARM NOT RESTART IMS
//*   RRS     X     Y = ENABLE PROT CONV SUPPORT
//*                 N = DISABLE PROT CONV SUPPORT
//*   IRLM    X     Y = YES, N = NO
//*   IRLMNM  XXXX  IRLM SUBSYSTEM NAME
//*   SSM     XXXX  EXT SUBSYSTEM PROCLIB MEMBER ID
//*   WADS    X     SINGLE/DUAL WADS,S=SINGLE,D=DUAL
//*   ARC     XX    AUTOMATIC ARCHIVE.
//*                 0 = NOT AUTOMATIC
//*                 1-99 = AUTOMATIC
//*   UHASH   XXXXXXXX USER HASH MODULE NAME
//*   DBRCNM  XXXXXXXX DBRC PROCLIB MEMBER NAME
//*   DLINM   XXXXXXXX DL/I PROCLIB MEMBER NAME
//*   PRDR    XXXXXXXX IMSRDR PROCLIB MEMBER NAME
//*   FESTIM  XXXX  FRONTENDSWITCH TIMEOUT (SECONDS)
//*   RECASZ  XXXXX RECEIVE ANY BUFFER SIZE
//*   PIMAX   XXXXXX ENQ/DEQ POOL MAXIMUM BYTES
//*   PIINCR  XXXXXX ENQ/DEQ POOL INCREMENT
//*   RECA    XXX   NUMBER OF RECEIVE ANY BUFFERS
//*   CRC     X     COMMAND RECOGNITION CHARACTER
//*   TSR     X     U = UTC TIME
//*                 L = LOCAL TIME (DEFAULT)
//*   YEAR4   X     N = 2-DIGIT DATE (DEFAULT)
//*                 Y = 4-DIGIT DATE
//*   DC      XXX   DC PROC MEMBER SUFFIX IN
//*                 IMS.PROCLIB
//*                 DEFAULT VALUE IS 000
//*   CPLOG   XXXXXK CHECKPOINT LOG INTERVAL
//*           OR
//*   CPLOG   XXM    CHECKPOINT LOG INTERVAL
//*   PASSWD1 XXXXXXXX VTAM ACB PASSWORD
//*   ORSMBR  XX    SUFFIX FOR ORS MEMBER
//*   IMSGROUP XXXX  4 CHAR USER SPEC NAME
//*   DESC    XX    MSG DESC CODE
//*   MCS     (XX,XX) MSG ROUTE CODES
//*   SVC2    XXX   TYPE 2 SVC NUMBER
//*   CCTCVCAN X    Y = CCTL CANCEL WILL BE CONVERTED
//*                     TO ABEND SYSTEM 08E
//*                 N = CCTL CANCEL IS NOT CONVERTED
//*
//******** OTMA PARAMETERS ***************************
//*
//*   OTMA    X     Y = OTMA ENABLED
//*                 N = OTMA NOT ENABLED
//*                 DEFAULT VALUE IS N
//*   OTMANM  XXXXXXXX IMS OTMA XCF MEMBER NAME
//*   GRNAME  XXXXXXXX OTMA XCF GROUP NAME
//*                    NO DEFAULT VALUE
//*   GRSNAME XXXXXXXX GENERIC RESOURCE GROUP
//*                    NAME
//*                    NO DEFAULT VALUE
//*
//******** SECURITY PARAMETERS ***********************
//*
//*   AOIS    X     ICMD SECURITY OPTION
//*   AOI1    X     CMD SECURITY OPTION
//*                 A = ALL
```

```
//*                N = NONE
//*                C = DFSCCMD0 EXIT
//*                R = RACF
//*   TCORACF X    TCO RACF SECURITY OPTION
//*                Y = YES
//*                N = NO
//*   APPCSE  X    C = APPC RACF SECURITY IS CHECK
//*                F = APPC RACF SECURITY IS FULL
//*                N = APPC RACF SECURITY IS NONE
//*                P = APPC RACF SECURITY IS PROFILE
//*   CMDMCS  X    MCS/EMCS COMMAND OPTION
//*                N=COMMANDS NOT ALLOWED WITH CRC
//*                Y=ALL COMMANDS ALLOWED WITH CRC
//*                R=RACF COMMAND SECURITY
//*                C=DFSCCMD0 COMMAND SECURITY
//*                B=RACF AND DFSCCMD0 CMD SEC
//*   ISIS    X    N = NO RESOURCE ACCESS SECURITY
//*                R = RACF RESOURCE ACCESS SECURITY
//*                C = RACF RESOURCE ACCESS SECURITY
//*                A = RACF RESOURCE ACCESS SECURITY
//*   RCF     X    RACF USED FOR TRANS. AND SIGNON
//*                A = Y + S, Y = T + C, S = S + C.
//*   RVFY    X    RACF REVERIFY OPTION
//*                Y = YES, N = NO
//*   SGN     X    SIGNON AUTHORIZATION CHECKING
//*                F = MTO CANNOT NEGATE ACTIVATION
//*                    OF SIGNON VERIFICATION
//*                    SECURITY
//*                Y = SIGNON VERIFICATION SECURITY
//*                    WILL BE ACTIVATED
//*                N = SIGNON VERIFICATION SECURITY
//*                    WILL NOT BE ACTIVATED
//*                M = SINGLE USERID CAN SIGNON
//*                    TO MULTIPLE STATIC TERMINALS
//*                G = 'F' + 'M'
//*                Z = 'Y' + 'M'
//*   TRN     X    TRANSACTION AUTHORIZATION CHECKING
//*                F = FORCED, Y = YES, N = NO
//*   RCFTCB  XX   NUMBER OF RCF TCB'S
//*   PSWDC   X    PASSWORD CASE
//*                U=UPPER CASE
//*                M=MIXED CASE
//*                R=USES RACF SETTING (DEFAULT)
//*
//******** MESSAGE QUEUE PARAMETERS ******************
//*
//*   EXVR    X    PAGEFIX QMGR BUFFER POOLS
//*                (1=YES, 0=NO)
//*   QBUF    XXXX NUMBER OF MESSAGE QUEUE BUFFERS
//*   QTL     XXX  QUEUE LOWER THRESHOLD (%)
//*   QTU     XXX  QUEUE UPPER THRESHOLD (%)
//*
//******** SHARED QUEUES PARAMETERS ******************
//*
//*   LGMSGSZ XXXXX LONG MESSAGE SIZE
//*   QBUFHITH XXX  MSG QBUF HIGH THESHOLD %
//*   QBUFLWTH XXX  MSG QBUF LOW THESHOLD %
//*   QBUFMAX  XXXX MAX NUMBER OF MSG QUEUE BUFFERS
//*   QBUFPCTX XXX  % MSG QBUF DYNAMIC EXPAND
//*                 WHEN QBUFHITH EXCEEDED
//*                 DEFAULT IS 20%
//*   QBUFSZ  XXXXX SIZE OF MESSAGE QUEUE BUFFERS
//*   SHMSGSZ XXXXX SHORT MESSAGE SIZE
//*   SHAREDQ XXX   SQ PROC MEMBER SUFFIX IN
//*                 IMS.PROCLIB
//*                 NO DEFAULT VALUE
//*
```

```
//******** XRF PARAMETERS ****************************
//*
//*   HSBID   X     XRF SYSTEM ID
//*                 1 FOR FIRST SYSTEM
//*                 2 FOR SECOND SYSTEM
//*
//*   HSBMBR  XX    SUFFIX FOR XRF MEMBER IN
//*                 IMS.PROCLIB
//*                 00 IS DEFAULT
//*
//*   MNPS    XXXXXXXX NAME OF MNPS ACB
//*                    USERVAR WILL BE IGNORED
//*   MNPSPW  XXXXXXXX MNPS ACB PASSWORD
//*
//******** FDR PARAMETER *****************************
//*
//*   FDRMBR  XX    SUFFIX FOR FDR MEMBER IN
//*                 IMS.PROCLIB
//*
//******** FAST PATH PARAMETERS **********************
//*
//*   BSIZ    XXXXX DATA BASE BUFFER SIZE
//*   OTHR    XXX   NUMBER OF OUTPUT THREADS
//*   DBFX    XXXXX SYSTEM ALLOCATION OF DATA BASE BUFFERS TO BE
//*                 FIXED AT START OF 1ST FAST PATH DEP REGION
//*   DBBF    XXXXX NUMBER OF DATABASE BUFFERS
//*   DBFP    XXXX  PAGE FIX/FREE ADJUST TIMER
//*                 0: FIX/FREE AT SCHED/TERM
//*                 1: ALLOW PAGEFIX ONLY
//*                 2-3600: SEC PAGEFREE FREQ
//*   MSDB    X     SUFFIX FOR MSDB MEMBER ON
//*                 IMS.PROCLIB
//*   LGNR    XX    NUMBER OF LOG ENTRIES IN DEDB BUFFERHEADER
//*   EPCB    XXXX  EPCB POOL SIZE (1K BLOCKS)
//*   EMHL    XXXXX SIZE OF EMH BUFFER IN BYTES
//*   SVSODR  XXXX SVSO DISASTER RECOVERY OPTIONS
//*               NONE: DEFAULT. NO CHANGE TO ERE.
//*               AUTO: AREA MARKED RECOV NEEDED IF
//*
//*               DRRS: AREA MARKED RECOV NEEDED AT
//*
//*               WTOR: USER OPTION TO MARK AREAS
//*
//*
//*   DMHVF  XX     MEGS TO FIX PAGEFIX FOR VSO ERE DATASPACE
//*   FPOPN   X     PREOPEN/REOPEN OPTIONS FOR DEDBS
//*               N: DEFAULT. PREOPEN OF DEDB AREAS
//*                  DONE BEFORE IMS RESTART
//*                  COMPLETES
//*               D: DEDB PREOPEN/PRELOAD IS DIS-
//*                  ABLED AT CTL REGION INIT
//*               R: REOPEN AREAS OPENED WHEN IMS
//*                  ABNORMALLY TERMINATED.  BEHAVE
//*                  LIKE OPTION P FOR NON /ERE
//*               P: PREOPEN OF DEDB AREA
//*                  INITIATED AT THE END OF
//*                  RESTART
//*               A: OPTION R AND P COMBINED
//*   FPRLM   X     DEDB OPTIONS FOR IRLM RECONNECT
//*               N: DEFAULT. NO ACTION TAKEN
//*               S: RESTART ALL DEDB AREAS WHICH
//*                  WERE STARTED WHEN IRLM
//*                  DISCONNECTED
//*               R: RESTART AND REOPEN ALL DEDB
//*                  AREAS WHICH WERE STARTED WHEN
//*                  IRLM DISCONNECTED
//*               P: PREOPEN OF DEDB AREA
```

```
//*                   INITIATED AT THE END OF
//*                   IRLM RECONNECT
//*                A: COMBINE OPTIONS R AND P
//*   FP      X    INCLUDE FASTPATH IN THIS IMS
//*                N: THE DEFAULT.  THIS IMS DOES
//*                   NOT INCLUDE FASTPATH
//*                Y: THIS IMS INCLUDES FASTPATH
//*

//******** ETO PARAMETERS ****************************
//*
//*   ETO     X     Y = EXTENDED TERMINAL OPTION
//*                 N = NO EXTENDED TERMINAL OPTION
//*                 M = NO EXTENDED TERMINAL OPTION
//*                     BUT LOGON USERDATA SUPPORTED
//*                     FOR STATIC TERMINALS
//*   ASOT    XXXX  ETO AUTO SIGNOFF TIME
//*   ALOT    XXXX  ETO AUTO LOGNOFF TIME
//*   DLQT    XXX   ETO DEAD LETTER QUEUE SIZE
//*
//******** RSR PARAMETERS ****************************
//*
//*   RSRMBR  XX    SUFFIX FOR RSR MEMBER
//*   TRACK   XXX   NO = NO RECOVERY TRACKING DONE
//*                 RLT = RECOVERY TRACKING DONE
//*                 DLT = DATABASE TRACKING DONE
//*   USERVAR  XXXXXXXX USER NAME OF ACTIVE IMS SYSTEM FOR RSR
//*
//******* HASH TABLE PARAMETERS **********************
//*
//*   LHTS    XXXXX # OF CNT HASH TABLE SLOTS
//*   NHTS    XXXXX # OF VTCB HASH TABLE SLOTS
//*   UHTS    XXXXX # OF SPQB HASH TABLE SLOTS
//*
//******** STORAGE POOL VALUES IN K, M OR G **********
//*
//*   FBP     XXXXXX MESSAGE BUFFER POOL SIZE
//*   PSB     XXXXXX PSB POOL SIZE - NON DLISAS
//*   DMB     XXXXXX DMB POOL SIZE
//*   CIOP    XXXXXX CIOP POOL UPPER LIMIT
//*   WKAP    XXXXXX WORKING STORAGE BUFFER POOL SIZE
//*   PSBW    XXXXXX PSB WORK POOL SIZE
//*   DBWP    XXXXXX DATABASE WORK POOL SIZE
//*   CSAPSB  XXXXXX DLISAS: CSA PSB POOL SIZE
//*   DLIPSB  XXXXXX DLISAS: DLI PSB POOL SIZE
//*   EPCB    XXXXXX EPCB POOL SIZE
//*   HIOP    XXXXXX HIOP POOL UPPER LIMIT
//*   FPWP    XXXXXX FPWP POOL UPPER LIMIT
//*   EMHB    XXXXXX EMHB POOL UPPER LIMIT
//*   LUMP    XXXXXX LUMP POOL UPPER LIMIT
//*   LUMC    XXXXXX LUMC POOL UPPER LIMIT
//*   DYNP    XXXXXX DYNP POOL UPPER LIMIT
//*   AOIP    XXXXXX AOI POOL UPPER LIMIT
//*   CMDP    XXXXXX CMDP POOL UPPER LIMIT
//*
//******** MEMBER SUFFIXES ***************************
//*
//*   SUF     X     LAST CHARACTER OF CTL PROGRAM LOAD
//*                 MODULE MEMBER NAME
//*   FIX     XX    2 CHARACTER FIX PROCEDURE MODULE SUFFIX
//*   PRLD    XX    2 CHARACTER PROCLIB MEMBER SUFFIX FOR PRELOAD
//*   VSPEC   XX    2 CHARACTER BUFFER POOL SPEC MODULE SUFFIX
//*   SPM     XX    STG POOL MGR PROCLIB MEMBER SUFFIX
//*   CSLG    XXX   CSL GLOBAL MEMBER (DFSCGXXX)
//*   DSCT    X     ETO USER DESCRIPTOR TABLE(DFSDSCTX)
//*   DFSDF   XXX   DRD, CSL, AND SQ MEMBER (DFSDFXXX)
//*
```

```
//*************************************************
//*                                               *
//*       DATA DEFINITION STATEMENTS FOLLOW       *
//*                                               *
//*************************************************
//*
//******** LIBRARY  STATEMENTS ********************
//*
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//*
//******** GENERIC START DEPENDANT REGION ***********
//*
//* IN ORDER TO START A DEPENDENT REGION, MODIFIED
//* START-UP JCL IS WRITTEN FROM INTERNAL STORAGE TO
//* THE INTERNAL READER.
//*
//IMSIRD   DD SYSOUT=(A,INTRDR)
//*
//******** DASD LOGGING STATEMENTS ******************
//*
//* THE FOLLOWING DD CARDS DESCRIBE THE DASD LOGGING
//* OLDS AND WADS. THESE CARDS ARE FOR EXAMPLE ONLY.
//* ALL OLDS AND WADS DATA SETS MAY BE DYNAMICALLY
//* ALLOCATED.  DD CARDS ARE NOT REQUIRED.
//* THE OLDS AND WADS TO BE USED DURING STARTUP MUST
//* BE SPECIFIED VIA OLDSDEF AND WADSDEF CONTROL
//* STATEMENTS IN THE DFSVSMXX MEMBER OF IMS PROCLIB.
//* THE ACTUAL SELECTION OF OLDS AND WADS MUST BE
//* TAILORED TO INSTALLATION REQUIREMENTS.  THE OLDS
//* AND WADS MUST BE PREDEFINED BY A SET UP JOB.
//* THE BLOCK SIZE OF ALL OLDS MUST BE THE SAME.
//* THE BLOCK SIZE AND DEVICE TYPE OF ALL WADS MUST
//* BE THE SAME.  AT LEAST 3 PRIMARY OLDS AND 1 WADS
//* MUST BE AVAILABLE FOR STARTUP.  THE BLOCK SIZE
//* SHOULD NOT BE SPECIFIED IN THIS JCL.  THE LOGGER
//* WILL GET THE BLOCK SIZE FROM THE VTOC.
//*
//DFSOLP00 DD DSN=IMS.&SYS.OLP00,DISP=SHR
//DFSOLP01 DD DSN=IMS.&SYS.OLP01,DISP=SHR
//DFSOLP02 DD DSN=IMS.&SYS.OLP02,DISP=SHR
//DFSOLP03 DD DSN=IMS.&SYS.OLP03,DISP=SHR
//DFSOLP04 DD DSN=IMS.&SYS.OLP04,DISP=SHR
//DFSOLP05 DD DSN=IMS.&SYS.OLP05,DISP=SHR
//*
//DFSOLS00 DD DSN=IMS.&SYS.OLS00,DISP=SHR
//DFSOLS01 DD DSN=IMS.&SYS.OLS01,DISP=SHR
//DFSOLS02 DD DSN=IMS.&SYS.OLS02,DISP=SHR
//DFSOLS03 DD DSN=IMS.&SYS.OLS03,DISP=SHR
//DFSOLS04 DD DSN=IMS.&SYS.OLS04,DISP=SHR
//DFSOLS05 DD DSN=IMS.&SYS.OLS05,DISP=SHR
//*
//DFSWADS0 DD DSN=IMS.&SYS.WADS0,DISP=SHR
//DFSWADS1 DD DSN=IMS.&SYS.WADS1,DISP=SHR
//*
//******** MONITOR STATEMENTS ***********************
//*
//* THE IMSMON DD STATEMENT MUST BE REMOVED IF
//* THIS DATA SET IS TO BE DYNAMICALLY ALLOCATED.
//*
//IMSMON   DD DSN=IMS.&SYS1.IMSMON,DISP=(,KEEP),
//         VOL=(,,,99),UNIT=(&LOGT,,DEFER)
//*
//******** MESSAGE QUEUE STATEMENTS *****************
//*
//QBLKS    DD DSN=IMS.&SYS1.QBLKS,DISP=OLD
//SHMSG    DD DSN=IMS.&SYS1.SHMSG,DISP=OLD
```

```
//LGMSG    DD DSN=IMS.&SYS1.LGMSG,DISP=OLD
//QBLKSL   DD DSN=IMS.&SYS1.QBLKSL,DISP=OLD
//SHMSGL   DD DSN=IMS.&SYS1.SHMSGL,DISP=OLD
//LGMSGL   DD DSN=IMS.&SYS1.LGMSGL,DISP=OLD
//*
//******* ONLINE CHANGE STATEMENTS ******************
//*
//* THE IMSACBA AND IMSACBB DD STATEMENTS MUST BE
//* REMOVED IF YOU WISH TO DYNAMICALLY ALLOCATE THE
//* ACBLIB DATA SETS THROUGH THE DFSMDA MEMBER.
//*
//IMSACBA  DD DSN=IMS.&SYS2.ACBLIBA,DISP=SHR
//IMSACBB  DD DSN=IMS.&SYS2.ACBLIBB,DISP=SHR
//MODBLKSA DD DSN=IMS.&SYS2.MODBLKSA,DISP=SHR
//MODBLKSB DD DSN=IMS.&SYS2.MODBLKSB,DISP=SHR
//MODSTAT  DD DSN=IMS.&SYS.MODSTAT,DISP=SHR
//MODSTAT2 DD DSN=IMS.&SYS.MODSTAT2,DISP=SHR
//******** FAST PATH STATEMENTS **********************
//*
//* THE MSDB DD STATEMENTS ARE BEING CREATED AS
//*  COMMENTS. THE ASTERISK IN IN COLUMN THREE NEEDS
//*  TO BE DELETED TO HAVE THE DD STATEMENT ACTIVATED
//*
//*MSDBCP1  DD DSN=IMS.&SYS.MSDBCP1,DISP=SHR
//*MSDBCP2  DD DSN=IMS.&SYS.MSDBCP2,DISP=SHR
//*MSDBCP3  DD DSN=IMS.&SYS.MSDBCP3,DISP=SHR
//*MSDBCP4  DD DSN=IMS.&SYS.MSDBCP4,DISP=SHR
//*MSDBDUMP DD DSN=IMS.&SYS1.MSDBDUMP,DISP=SHR
//*MSDBINIT DD DSN=IMS.&SYS.MSDBINIT,DISP=SHR
//*
//******** MFS STATEMENTS ****************************
//*
//FORMATA  DD DSN=IMS.&SYS2.FORMATA,DISP=SHR
//FORMATB  DD DSN=IMS.&SYS2.FORMATB,DISP=SHR
//IMSTFMTA DD DSN=IMS.&SYS2.TFORMAT,DISP=SHR
//         DD DSN=IMS.&SYS2.FORMATA,DISP=SHR
//IMSTFMTB DD DSN=IMS.&SYS2.TFORMAT,DISP=SHR
//         DD DSN=IMS.&SYS2.FORMATB,DISP=SHR
//*
//******** SYSTEM STATEMENTS *************************
//*
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=125,RECFM=FBA,BLKSIZE=3129),
//         SPACE=(6050,300,,ROUND)
//IMSRDS   DD DSN=IMS.&SYS.RDS,DISP=SHR
//IMSRDS2  DD DSN=IMS.&SYS.RDS2,DISP=SHR
//*DFSTCF  DD DSN=IMS.TCFSLIB,DISP=SHR
//PRINTDD  DD SYSOUT=&SOUT
//*
//******** TELEPROCESSING LINE STATEMENTS ************
//****** GENERATED FROM THE DB/DC DEFINITION *********
//*
//* *** THE GENERATED TELEPROCESSING LINE STATEMENTS WOULD BE HERE ***
//*
//******** EXTERNAL SUBSYSTEM STATEMENTS *************
//*
//* USER MAY OPTIONALLY ADD THE DFSESL DD CARD
//* FOR EXTERNAL SUBSYSTEM CONNECTION.
//*
//******** DATA BASE STATEMENTS **********************
//*
//* USER MAY OPTIONALLY SUPPLY THE DD STATEMENTS
//* FOR THE ON-LINE DATA BASES TO BE
//* INSERTED HERE PRIOR TO ATTEMPTING
//* AN ON-LINE SYSTEM EXECUTION USING
//* THIS PROCEDURE.
//* IF NO DD STATEMENTS ARE SUPPLIED FOR
```

```
//* A DATA BASE, IMS ASSUMES THAT THIS
//* DATA BASE HAS BEEN DESCRIBED THROUGH
//* THE DFSMDA MACRO.
//* IF THE USER WILL BE EXECUTING WITH THE DL/I
//* SAS OPTION, THESE DD STATEMENTS SHOULD BE ADDED
//* TO THE DLISAS PROCLIB MEMBER OR DESCRIBED
//* THROUGH THE DFSMDA MACRO.
```

IMSBATCH procedure

The IMSBATCH procedure executes an IMS online batch message processing address space.

The procedure shown in "Sample procedure to execute DB/DC or DBCTL online batch message processing region" on page 572 executes an IMS DB/DC or IMS DBCTL online batch message processing region. Parameters in parentheses are positional.

Parameters

The following parameters are valid for the IMSBATCH procedure.

AGN=
ALTID=
APARM=
CKPTID=
CPUTIME=
DIRCA=
ENVIRON=
IMSID=
IN=
JVMOPMAS=
LOCKMAX=
MBR=
OBA=
OPT=
OUT=
PARDLI=
PREINIT=
PRLD=
PSB=
RGN=
SOUT=
SPIE=
SSM=
STIMER=
SYS2=
TEST=

See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

DD statements

The following DD statements are valid for the IMSBATCH procedure.

DFSCTL DD (optional)
DFSESL DD (optional)
DFSHALDB DD (optional)
DFSSTAT DD (optional)

FPTRACE DD (optional)
IMS DD (optional)
IMSLOGR DD
PROCLIB DD
STEPLIB DD
SYSHALDB DD (optional)
SYSUDUMP DD

Note: If the checkpoint records that are required to restart the BMP do not exist in the online log data sets (OLDSn), an //IMSLOGR DD statement specifying a data set containing the checkpoint log records must be added to the BMP job's JCL.

See "DD statement descriptions" on page 502 for descriptions.

Sample procedure to execute DB/DC or DBCTL online batch message processing region

```
//       PROC MBR=TEMPNAME,PSB=,IN=,OUT=,
//            OPT=N,SPIE=0,TEST=0,DIRCA=000,
//            PRLD=,STIMER=,CKPTID=,PARDLI=,
//            CPUTIME=,NBA=,OBA=,IMSID=,AGN=,
//            SSM=,PREINIT=,RGN=56K,SOUT=A,
//             SYS2=,ALTID=,APARM=,LOCKMAX=,
//            ENVIRON=,JVMOPMAS=
//*
//G      EXEC PGM=DFSRRC00,REGION=&RGN,
//            PARM=(BMP,&MBR,&PSB,&IN,&OUT,
//            &OPT&SPIE&TEST&DIRCA,&PRLD,
//            &STIMER,&CKPTID,&PARDLI,&CPUTIME,
//            &NBA,&OBA,&IMSID,&AGN,&SSM,
//             &PREINIT,&ALTID,
//             '&APARM',&LOCKMAX,&ENVIRON,&JVMOPMAS)
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=VBA,BLKSIZE=3129),
//         SPACE=(125,(2500,100),RLSE,,ROUND)
```

IMSCOBGO procedure

The IMSCOBGO procedure is a three-step compile, bind, and go procedure combining the IMSCOBOL procedure with an execution step for a stand-alone DL/I batch address space.

JCL

In the sample IMSCOBGO procedure JCL:

- Parameters in parentheses are positional.
- The IMS PROCLIB data set should have the same block size as SYS1.PROCLIB, which should be 80, 400, or 3200.
- The IEFRDER and IEFRDER2 statements are not required in DB/DC or DBCTL environments if the job does not declare database update intent.
- Assumptions:
  - You supply source data from SYSIN.
  - Output class is A.
  - MBR=NAME, where NAME is the load module name for the program.
  - SYSDA is a generic device name.

If VSAM databases are used, see "IMS buffer pools" on page 179.

```
//       PROC MBR=TEMPNAME,PAGES=60,SYS2=,
//            LNGPRFX=IGY,
//            LIBPRFX=CEE,
//            SOUT=A,RGN=4M,LOGT=2400,
//            PSB=,BUF=7,SPIE=0,TEST=0,EXCPVR=0,
//            RST=0,PRLD=,SRCH=0,CKPTID=,MON=N,
//            LOGA=0,FMTO=T,IMSID=,SWAP=,DBRC=,IRLM=,
//            IRLMNM=,BKO=N,IOB=,SSM=,APARM=,
//            LOCKMAX=,IMSPLEX=
//C      EXEC PGM=IGYCRCTL,REGION=4M,
//        PARM='SIZE(832K),BUF(10K),LINECOUNT(50)'
//STEPLIB  DD DSN=&LNGPRFX..SIGYCOMP,
//         DISP=SHR
//SYSLIN   DD DSN=&&LIN,DISP=(MOD,PASS),UNIT=SYSDA,
//         DCB=(IMS.&SYS2.PROCLIB),
//         SPACE=(3520,(40,10),RLSE,,ROUND)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE,,ROUND)
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT4   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT5   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT6   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT7   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//L      EXEC PGM=IEWL,REGION=4M,
//            PARM='XREF,LET,LIST',
//            COND=(8,LT,C)
//SYSLIB   DD DSN=&LIBPRFX..SCEELKED,
//         DISP=SHR
//DFSRESLB DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//SYSLIN   DD DSN=&&LIN,DISP=(OLD,DELETE),
//         VOL=REF=*.C.SYSLIN
//         DD DISP=SHR,
//         DSN=IMS.&SYS2.PROCLIB(CBLTDLI)
//         DD DDNAME=SYSIN
//SYSLMOD  DD DISP=SHR,
//         DSN=IMS.&SYS2.PGMLIB(&MBR)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE,,ROUND)
//SYSUT1   DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),
//         DISP=(,DELETE),
//         SPACE=(3520,(100,10),RLSE,,ROUND)
//G      EXEC PGM=DFSRRC00,REGION=&RGN,TIME=2,
//            COND=((8,LT,C),(4,LT,L)),
//            PARM=(DLI,&MBR,&PSB,&BUF,
//            &SPIE&TEST&EXCPVR&RST,
//            &PRLD,&SRCH,&CKPTID,&MON,&LOGA,
//            &FMTO,&IMSID,&SWAP,
//            &DBRC,&IRLM,&IRLMNM,&BKO,&IOB,
//            &SSM,'&APARM',&LOCKMAX,,,&IMSPLEX)
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//         DD DSN=&LIBPRFX..SCEERUN,DISP=SHR
//DFSRESLB DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//IMS      DD DSN=IMS.&SYS2.PSBLIB,DISP=SHR
//         DD DSN=IMS.&SYS2.DBDLIB,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//IEFRDER  DD DSN=IMSLOG,DISP=(,KEEP),VOL=(,,,99),
//         UNIT=(&LOGT,,DEFER),
//         DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//IEFRDER2 DD DSN=IMSLOG2,DISP=(,KEEP),VOL=(,,,99),
//         UNIT=(&LOGT,,DEFER,SEP=IEFRDER),
```

```
//         DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//SYSOUT   DD SYSOUT=&SOUT,SPACE=(CYL,(1,1)),
//         DCB=(LRECL=133,RECFM=FBA,BLKSIZE=665)
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=FBA,BLKSIZE=3025),
//         SPACE=(3025,(200,100),RLSE,,ROUND)
//CEEDUMP  DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=FBA,BLKSIZE=3025),
//         SPACE=(3025,(200,100),RLSE,,ROUND)
```

Usage

For a job step declaring database-update intent, DD DUMMY can be specified if the job step is not using DBRC. This is valid where an image copy of the database is taken before the update job step.

Log initialization calculates the smallest value necessary for logical record length. If the JCL logical record length value is larger than the calculated value, the JCL value is used; otherwise, log initialization uses the calculated value for logical record length and adds 4 for the block size.

If multiple volumes are required for the system log, a volume count value should be specified in the VOL parameter of the DD statement.

If the IBM 3480 tape drive is used for the IMS log data set, IMS forces tape write mode (DCB=OPCD=W). The default on the 3480 is to hold the write in a buffer so that IMS cannot detect when the write is performed. If a power failure occurs after a log record is written to the 3480, and the database is updated but the log record is not yet written to tape, database integrity is lost. Tape write mode is forced for the log in batch and GSAM data sets.

Parameters

The following parameters are valid for the IMSCOBGO procedure.

BKO=
BUF=
CKPTID=
DBRC=
DBRCGRP=
EXCPVR=
FMTO=
IMSID=
IN=
IMSPLEX=
IRLM=
IRLMNM=
JOB=
LOGA=
LOGT=
MBR=
MON=
PRLD=
PSB=
RGN=
SOUT=
SPIE=
SRCH=

SWAP=
SYS2=
TEST=

See “Parameter descriptions for IMS procedures” on page 448 for descriptions.

DD statements

The following DD statements are valid for the IMSCOBGO procedure.

In addition to the following DD statements, add statements for data sets representing IMS databases.

DFSHALDB [1]
DFSRESLB DD
DFSSTAT DD
IEFRDER DD
IEFRDER2 DD
IMS DD
PROCLIB DD
RECONn DD
STEPLIB DD
SYSHALDB DD[1]
SYSLIB DD
SYSLIN DD
SYSLMOD DD
SYSOUT DD
SYSPRINT DD
SYSUDUMP DD
SYSUTn DD

See “DD statement descriptions” on page 502 for descriptions.

IMSCOBOL procedure

The IMSCOBOL procedure is a two-step compile and bind procedure for IMS applications that are written in COBOL.

Parameters

The following parameters are valid for the IMSCOBOL procedure. See “Parameter descriptions for IMS procedures” on page 448 for descriptions.

MBR=
SOUT=
SYS2=

DD statements

The following DD statements are valid for the IMSCOBOL procedure. See “DD statement descriptions” on page 502 for descriptions.

DFSRESLB DD
SYSLIB DD
SYSLIN DD
SYSLMOD DD
SYSPRINT DD
SYSUTn DD

JCL

The JCL in "Sample compile and bind procedure for IMS COBOL" assumes that:

- You supply source data from SYSIN.
- The output class is A.
- MBR=NAME, where NAME is the load module name for the program.
- SYSDA is a generic device name.

Sample compile and bind procedure for IMS COBOL

```
//       PROC MBR=TEMPNAME,PAGES=60,SYS2=,
//            LNGPRFX=IGY,
//            LIBPRFX=CEE,
//            SOUT=A
//C      EXEC PGM=IGYCRCTL,REGION=4M,
//        PARM='SIZE(832K),BUF(10K),LINECOUNT(50)'
//STEPLIB  DD DSN=&LNGPRFX..SIGYCOMP,
//         DISP=SHR
//SYSLIN   DD DSN=&&LIN,DISP=(MOD,PASS),UNIT=SYSDA,
//         DCB=(IMS.&SYS2.PROCLIB),
//         SPACE=(3520,(40,10),RLSE,,ROUND)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE,,ROUND)
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT4   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT5   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT6   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT7   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//L      EXEC PGM=IEWL,REGION=4M,
//            PARM='XREF,LET,LIST',
//            COND=(8,LT,C)
//SYSLIB   DD DSN=&LIBPRFX..SCEELKED,
//         DISP=SHR
//SDFSRESL DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//SYSLIN   DD DSN=&&LIN,DISP=(OLD,DELETE),
//         VOL=REF=*.C.SYSLIN
//         DD DISP=SHR,
//         DSN=IMS.&SYS2.PROCLIB(CBLTDLI)
//         DD DDNAME=SYSIN
//SYSLMOD  DD DISP=SHR,
//         DSN=IMS.&SYS2.PGMLIB(&MBR)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE,,ROUND)
//SYSUT1   DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),
//         DISP=(,DELETE),
//         SPACE=(3520,(100,10),RLSE,,ROUND)
```

IMSDALOC procedure

Use the IMSDALOC procedure to generate the list of databases, DEDB data areas, and data sets that are to be dynamically allocated.

Usage

The IMSDALOC procedure is created as a part of system definition and is placed into the IMS.PROCLIB library by stage two of IMS system definition. This is a three-step procedure for generating the list of databases and DEDB data areas that are to be dynamically allocated.

IMSDALOC assumes that:

- Input is read from SYSIN.
- Each database or DEDB data set described in the input has a corresponding module placed in the dynamic allocation member data set.
- The name given to each module is the name of the database or DEDB data area described in the input.

The dynamic allocation macro statements are supplied as input to the IMSDALOC procedure and executed as a z/OS job.

Step BLDMBR is used with the z/OS IEBUPDTE utility, which incorporates changes to sequential or partitioned data sets.

Refer to the IEBUPDTE utility in *z/OS DFSMSdfp Utilities* for information about this step.

Step LNKEDT is the bind step.

Refer to *MVS/DFP Linkage Editor and Loader* for information about linkage-editors.

Parameters

The following parameters are valid for the IMSDALOC procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

SOUT=
SYS2=

DD statements

The following DD statements are valid for the IMSDALOC procedure are shown below. See "DD statements for IMS procedures" on page 502.

OBJMOD DD
SYSIN DD
SYSLIB DD
SYSLIN DD
SYSLMOD DD
SYSPRINT DD
SYSPUNCH DD
SYSUT1 DD
SYSUT2 DD

JCL

The following code is the JCL for the IMSDALOC procedure.

```
//       PROC SOUT=A,SYS2=
//ASSEM  EXEC PGM=ASMA90,
//            PARM='ALIGN,DECK,NOOBJECT,NODBCS'
//SYSLIB   DD DSN=IMS.&SYS2.SDFSMAC,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
```

```
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(10,5))
//SYSPUNCH DD DSN=&OBJMOD,
//         DCB=(RECFM=FB,LRECL=80,BLKSIZE=400),
//         SPACE=(400,(100,100)),UNIT=SYSDA,
//         DISP=(NEW,PASS)
//SYSPRINT DD SYSOUT=&SOUT
//BLDMBR EXEC PGM=IEBUPDTE,PARM='NEW',
//            COND=(7,LT,ASSEM)
//SYSPRINT DD DUMMY
//SYSUT2   DD DSN=&TEMPPDS,UNIT=SYSDA,
//         DISP=(NEW,PASS,DELETE),
//         SPACE=(80,(1000,500,10)),
//         DCB=(RECFM=F,BLKSIZE=80)
//SYSIN    DD DSN=*.ASSEM.SYSPUNCH,
//         DISP=(OLD,DELETE,DELETE)
//LNKEDT EXEC PGM=IEWL,PARM='LIST,XREF,LET',
//            COND=((7,LT,ASSEM),(3,LT,BLDMBR))
//SYSUT1   DD UNIT=SYSDA,SPACE=(1024,(100,50))
//SYSLIB   DD DUMMY
//SYSPRINT DD SYSOUT=&SOUT
//SYSLMOD  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//OBJMOD   DD DSN=&TEMPPDS,DISP=(OLD,DELETE,DELETE)
//SYSLIN   DD DSN=&TEMPPDS(LNKCTL),
//         DISP=(OLD,DELETE,DELETE),
//         VOL=REF=*.OBJMOD
```

Example

The following JCL statement invokes the IMSDALOC procedure.

```
//DALOC    JOB
//*
//STEP    EXEC IMSDALOC
//*
//SYSIN     DD  *
   DFSMDA TYPE=
   END
/*
```

IMSFP procedure

The IMSFP procedure is an online procedure that initiates a Fast Path region where Fast Path applications run.

Non-message-driven Fast Path applications are not supported and should be changed to run as BMPs, using the IMSBATCH procedure. See "IMSBATCH procedure" on page 571.

Parameters

The following parameters are valid for the IMSFP procedure.

See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

AGN=
ALOT=
ALTID=
APARM=
CPUTIME=
DBLDL=
DIRCA=
ENVIRON=
IMSID=

JVMOPMAS=
LOCKMAX=
MBR=
NBA=
OBA=
OPT=
PARDLI=
PREINIT=
PRLD=
PSB=
RGN=
SOD=
SOUT=
SSM=
STIMER=
SYS2=
TLIM=

DD statements

The following DD statements are valid for the IMSFP procedure. See "DD statement descriptions" on page 502 for descriptions.

PROCLIB DD
STEPLIB DD
SYSUDUMP DD

Sample procedure to execute a Fast Path application program

The procedure shown here executes a Fast Path region where Fast Path applications run. The `IFP,&MBR,&PSB,&NBA,&OBA,&OPT` parameters are positional.

```
//IFPROC PROC MBR=TEMPNAME,SOUT=A,RGN=100K,OPT=N,
//            PSB=,NBA=000,OBA=000,TLIM=1,
//            DIRCA=000,PRLD=,STIMER=0,SOD=,DBLDL=,
//            CPUTIME=,IMSID=,AGN=,SSM=,PREINIT=,
//            SYS2=,ALTID=,APARM=,LOCKMAX=,
//            ENVIRON=,JVMOPMAS=,PARDLI=
//IFP   EXEC  PGM=DFSRRC00,REGION=&RGN,
//      TIME=1440,
//      PARM=(IFP,&MBR,&PSB,&NBA,&OBA,&OPT,
//      &TLIM,&DIRCA,&PRLD,&STIMER,&SOD,&DBLDL,
//      &CPUTIME,&IMSID,&AGN,&SSM,&PREINIT,
//      &ALTID,'&APARM',&LOCKMAX,&ENVIRON,
//      &JVMOPMAS,&PARDLI)
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=VBA,BLKSIZE=3129),
//         SPACE=(125,(2500,100),RLSE,,ROUND)
```

IMSMSG procedure

The IMSMSG job is used to execute an IMS message processing program.

JCL

The following job executes procedure DFSMPR, which is used to start an online message region. See "DFSMPR procedure" on page 547 for details.

```
//MESSAGE  JOB 1,IMS,MSGLEVEL=1,PRTY=11,CLASS=K,MSGCLASS=A,REGION=56K
//       EXEC DFSMPR,
//            IMSID=
```

IMSPLI procedure

The IMSPLI procedure is a two-step compile and bind procedure for IMS applications written in PL/I.

"Sample JCL for the IMSPLI procedure" demonstrates the IMSPLI procedure. This sample procedure assumes that:

- Your supply source data is from SYSIN.
- The output class is A.
- MBR=NAME, where NAME is the load module name for the program.
- SYSDA is a generic device name.

Parameters

The following parameters are valid for the IMSPLI procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

MBR=
SPIE=
SYS2=

DD statements

The following DD statements are valid for the IMSPLI procedure. See "DD statement descriptions" on page 502 for descriptions.

DFSRESLB DD
SYSLIB DD
SYSLIN DD
SYSLMOD DD
SYSPRINT DD
SYSUT1 DD

Sample JCL for the IMSPLI procedure

```
//       PROC MBR=TEMPNAME,PAGES=50,SYS2=,
//            LNGPRFX=IEL,
//            LIBPRFX=CEE,
//            SOUT=A
//C      EXEC PGM=IEL1AA,REGION=4M,
//            PARM=(XREF,A,OBJ,NODECK,NOMACRO,,
//            'OPT(TIME)')
//STEPLIB  DD DSN=&LNGPRFX..SIELCOMP,DISP=SHR
//         DD DSN=&LIBPRFX..SCEERUN,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,
//         SPACE=(1024,(200,50),RLSE,,ROUND),
//         DCB=BLKSIZE=1024,DISP=(,DELETE)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(LRECL=125,BLKSIZE=629,RECFM=VBA),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSLIN   DD UNIT=SYSDA,SPACE=(80,(250,100),RLSE),
//         DCB=(IMS.&SYS2.PROCLIB),
//         DISP=(,PASS)
//L      EXEC PGM=IEWL,PARM='XREF,LIST,LET',
//            COND=(9,LT,C),REGION=4M
//SYSLIB   DD DSN=&LIBPRFX..SCEELKED,DISP=SHR
//SDFSRESL DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//SYSLIN   DD DSN=*.C.SYSLIN,DISP=(OLD,DELETE)
```

```
//         DD DISP=SHR,
//         DSN=IMS.&SYS2.PROCLIB(PLITDLI)
//         DD DDNAME=SYSIN
//SYSLMOD  DD DISP=SHR,
//         DSN=IMS.&SYS2.PGMLIB(&MBR)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=FBA,BLKSIZE=605),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSUT1   DD UNIT=SYSDA,DISP=(,DELETE),
//         SPACE=(CYL,(5,1),RLSE)
```

IMSPLIGO procedure

The IMSPLIGO procedure is a three-step compile, bind, and go procedure combining the IMSPLI procedure with an execution step for a stand-alone DL/I batch processing region.

If VSAM databases are used, see "IMS buffer pools" on page 179.

JCL

The IMSPLIGO procedure is shown in the following sample. This sample procedure assumes that:

- You supply source data from SYSIN.
- The output class is A.
- MBR=NAME, where NAME is the load module name for the program.
- SYSDA is a generic device name.
- You add DD statements for data sets representing IMS databases.

```
//       PROC MBR=TEMPNAME,PAGES=50,SYS2=,
//            LNGPRFX=IEL,
//            LIBPRFX=CEE,
//            SOUT=A,RGN=4M,
//            PSB=,BUF=7,SPIE=0,TEST=0,EXCPVR=0,
//            RST=0,PRLD=,SRCH=,CKPTID=,MON=N,
//            LOGA=0,FMTO=T,IMSID=,SWAP=,RGN=4M,
//            LOGT=2400,
//            DBRC=,IRLM=,IRLMNM=,BKO=N,IOB=,
//            SSM=,APARM=,LOCKMAX=,IMSPLEX=
//C      EXEC PGM=IEL1AA,REGION=4M,
//            PARM=(XREF,A,OBJ,NODECK,NOMACRO,,
//            'OPT(TIME)')
//STEPLIB  DD DSN=&LNGPRFX..SIELCOMP,DISP=SHR
//         DD DSN=&LIBPRFX..SCEERUN,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,
//         SPACE=(1024,(200,50),RLSE,,ROUND),
//         DCB=BLKSIZE=1024,DISP=(,DELETE)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(LRECL=125,BLKSIZE=629,RECFM=VBA),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSLIN   DD UNIT=SYSDA,SPACE=(80,(250,100),RLSE),
//         DCB=(IMS.&SYS2.PROCLIB),
//         DISP=(,PASS)
//L      EXEC PGM=IEWL,PARM='XREF,LIST,LET',
//            COND=(9,LT,C),REGION=4M
//SYSLIB   DD DSN=&LIBPRFX..SCEELKED,DISP=SHR
//SDFSRESL DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//SYSLIN   DD DSN=*.C.SYSLIN,DISP=(OLD,DELETE)
//         DD DISP=SHR,
//         DSN=IMS.&SYS2.PROCLIB(PLITDLI)
//         DD DDNAME=SYSIN
//SYSLMOD  DD DISP=SHR,
//         DSN=IMS.&SYS2.PGMLIB(&MBR)
```

```
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,RECFM=FBA,BLKSIZE=605),
//         SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSUT1   DD UNIT=SYSDA,DISP=(,DELETE),
//         SPACE=(CYL,(5,1),RLSE)
//G      EXEC PGM=DFSRRC00,REGION=&RGN,TIME=5,
//            COND=(9,LT),
//            PARM=(DLI,&MBR,&PSB,&BUF,
//            &SPIE&TEST&EXCPVR&RST,
//            &PRLD,&SRCH,&CKPTID,&MON,&LOGA,
//            &FMTO,&IMSID,&SWAP,&DBRC,&IRLM,
//            &IRLMNM,&BKO,&IOB,&SSM,'&APARM',
//            &LOCKMAX,,,&IMSPLEX)
//STEPLIB  DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//         DD DSN=IMS.&SYS2.PGMLIB,DISP=SHR
//         DD DSN=&LIBPRFX..SCEERUN,DISP=SHR
//DFSRESLB DD DSN=IMS.&SYS2.SDFSRESL,DISP=SHR
//IMS      DD DSN=IMS.&SYS2.PSBLIB,DISP=SHR
//         DD DSN=IMS.&SYS2.DBDLIB,DISP=SHR
//PROCLIB  DD DSN=IMS.&SYS2.PROCLIB,DISP=SHR
//IEFRDER  DD DSN=IMSLOG,DISP=(NEW,KEEP),
//         VOL=(,,,99),UNIT=(&LOGT,,DEFER),
//         DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//IEFRDER2 DD DSN=IMSLOG2,DISP=(NEW,KEEP),
//         UNIT=(&LOGT,,DEFER,SEP=IEFRDER),
//         VOL=(,,,99),DCB=(RECFM=VB,BLKSIZE=4096,
//         LRECL=4092,BUFNO=2)
//SYSPRINT DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA),
//         SPACE=(605,(500,500),RLSE,,ROUND)
//SYSUDUMP DD SYSOUT=&SOUT,
//         DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA),
//         SPACE=(605,(500,500),RLSE,,ROUND)
```

Usage

In the JCL provided:

- The parameters are positional.
- The IMS PROCLIB data set should have the same block size as SYS1.PROCLIB, which should be 80, 400, or 3200.
- The IEFRDER and IEFRDER2 statements are not required in DB/DC or DBCTL environments if the job does not declare database update intent.

For a job step declaring database-update intent, DD DUMMY can be specified if the job step is not using DBRC. This is valid where an image copy of the database is taken before the update job step.

The log-initialization process calculates the smallest value necessary for logical record length. If the JCL logical record length value is larger than the calculated value, the JCL value is used; otherwise, the log-initialization process uses the calculated value for logical record length and adds 4 for the block size.

If multiple volumes are required for the system log, a volume count value should be specified in the VOL parameter of the DD statement.

If the IBM 3480 tape drive is used for the IMS log data set, IMS forces tape write mode (DCB=OPCD=W). The default on the 3480 is to hold the write in a buffer so that IMS cannot detect when the write is performed. If a power failure occurs after a log record is written to the 3480, and the database is updated but the log record

is not yet written to tape, database integrity is lost. Tape write mode is forced for the log in batch and GSAM data sets.

Parameters

The following parameters are valid for the IMSPLIGO procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

BKO=
BUF=
CKPTID=
DBRC=
DBRCGRP=
EXCPVR=
FMTO=
IMSID=
IMSPLEX=
IOB=
IRLM=
IRLMNM=
LOGA=
LOGT=
MBR=
MON=
PRLD=
PSB=
RGN=
RST=
SOUT=
SPIE=
SRCH=
SWAP=
SYS2=
TEST=

DD statements

The following DD statements are valid for the IMSPLIGO procedure. See "DD statement descriptions" on page 502 for descriptions.

DFSHALDB
DFSRESLB DD
DFSSTAT DD
IEFRDER DD
IEFRDER2 DD
IMS DD
PROCLIB DD
RECONn DD
SYSHALDB DD
SYSLIB DD
SYSLIN DD
SYSLMOD DD
SYSOUT DD
SYSPRINT DD
SYSUDUMP DD
SYSUT1 DD

IMSRDR procedure

The IMSRDR procedure is used to read an IMSMSG job into the operating system job stream from direct access storage devices (DASDs).

The JCL shown in "Sample DASD procedure to read IMSMSG job" reads an IMSMSG job into the operating system job stream from direct access storage devices.

Parameters

The following parameters are valid for the IMSRDR procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

CLASS=
MBR=
SYS2=

DD statements

The following DD statements are valid for the IMSRDR procedure. See "DD statement descriptions" on page 502 for descriptions.

IEFRDER DD
SYSIN DD
SYSPRINT DD
SYSUT1 DD
SYSUT2 DD

Sample DASD procedure to read IMSMSG job

```
//       PROC MBR=IMSMSG,CLASS=A,SYS2=
//IEFPROC EXEC PGM=IEBEDIT
//SYSPRINT DD DUMMY
//SYSUT1   DD DDNAME=IEFRDER
//SYSUT2   DD SYSOUT=(&CLASS,INTRDR),DCB=BLKSIZE=80
//SYSIN    DD DUMMY
//IEFRDER  DD DISP=SHR,
//         DSN=IMS.&SYS2.JOBS(&MBR)
```

RDIBATCH procedure

Use the RDIBATCH procedure to help maintain database availability in failure situations.

The RDI region is used in a block-level data-sharing environment to provide a path from the IRLM to DBRC, which guarantees database availability if a failure occurs. The RDI region accomplishes the following tasks:

- IDENTIFY to the IRLM
- SIGNON to DBRC
- WAIT for an IRLM failure

Usage

If an IRLM fails, its associated RDI region abends with a U3303 abend. The surviving IRLM partner is notified of the failure and notifies DBRC of the failure.

The RDI region can be terminated only by an IRLM failure or by an operator CANCEL command. You must ensure that the IRLM and DBRC are both being used to gain the benefits of data-sharing protection that this region offers.

If you are using IMS with IBM CICS Transaction Server for z/OS in an XRF environment, this procedure is essential. For more information about using IMS with CICS in an XRF environment, see the *CICS Extended Recovery Facility Guide*.

The procedure shown in "Sample RDIBATCH procedure" on page 586 is not placed in the IMS PROCLIB data set by IMS system definition. You must create it by modifying a copy of either the DBBBATCH or DLIBATCH procedure. The region type is RDI.

Parameters

The following parameters are valid for the RDIBATCH procedure. See "Parameter descriptions for IMS procedures" on page 448 for descriptions.

APARM=
BKO=
BUF=
CKPTID=
DBRC=
DBRCGRP=
EXCPVR=
FMTO=
IMSID=
IMSPLEX=
IOB=
IRLM=
IRLMNM=
LOCKMAX=
LOGA=
MBR=
MON=
PRLD=
PSB=
RGN=
RST=
SOUT=
SPIE=
SRCH=
SSM=
SWAP=
SYS=
SYS2=
TEST=

DD statements

The following DD statements are valid for the RDIBATCH procedure. See "DD statement descriptions" on page 502 for descriptions.

DFSRESLB DD
PROCLIB DD
RECONn DD
STEPLIB DD
SYSABEND DD

Sample RDIBATCH procedure

The following sample procedure demonstrates the RDIBATCH procedure.

In this sample:

1. MBR=xxxxxxxx must be specified but does not need to be valid. The specified member does not receive control but must be specified or parameter analysis fails.
2. DBRC=Y is required (specified or used by default).
3. IRLM=Y is required (specified or used by default).
4. IRLMNM=cccc is required (specified or used by default).
5. TIME=1440 is recommended.

```
//       PROC MBR=ANYNAME,1 PSB=,BUF=,
 //            SPIE=0,TEST=0,EXCPVR=0,RST=0,PRLD=,
 //            SRCH=0,CKPTID=,MON=,LOGA=0,FMTO=T,
 //            IMSID=,SWAP=,DBRC=Y,2 IRLM=Y,3
 //            IRLMNM=IRLM,4 BKO=N,IOB=,SSM=,APARM=,
 //            RGN=2048K,
 //            SOUT=A,
 //            SYS=,
 //            SYS2=,LOCKMAX=,IMSPLEX=
 //G      EXEC PGM=DFSRRC00,REGION=&RGN,TIME=1440,5
 //            PARM=(RDI,&MBR,&PSB,&BUF,
 //            &SPIE&TEST&EXCPVR&RST,&PRLD,
 //            &SRCH,&CKPTID,&MON,&LOGA,&FMTO,
 //            &IMSID,&SWAP,&DBRC,&IRLM,&IRLMNM,
 //             &BKO,&IOB,&SSM,
 //            '&APARM',&LOCKMAX,
 //            &GSGNAME,&TMINAME,&RRS,
 //            &IMSPLEX,&RGSUF,
 //            '&PARM1','&PARM2')
 //STEPLIB  DD DISP=SHR,DSN=IMS.&SYS2.SDFSRESL
 //DFSRESLB DD DISP=SHR,DSN=IMS.&SYS2.SDFSRESL
 //DFSVSAMP DD DISP=SHR,DSN=IMS.&SYS2.DFSVSAMP
 //PROCLIB  DD DISP=SHR,DSN=IMS.&SYS.PROCLIB
 //RECON1   DD DISP=SHR,DSN=IMS.&SYS.RECON1
 //RECON2   DD DISP=SHR,DSN=IMS.&SYS.RECON2
 //RECON3   DD DISP=SHR,DSN=IMS.&SYS.RECON3
 //SYSABEND DD SYSOUT=&SOUT.
            PEND
```

Chapter 17. Members of the IMS PROCLIB data set

These topics describe the IMS PROCLIB data set members that can be used in an IMS environment.

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

"Specifying comments in IMS PROCLIB data set members processed by the Syntax Checker" on page 291

BPE configuration parameter member of the IMS PROCLIB data set

Use the BPE configuration parameter member of the IMS PROCLIB data set to define BPE execution environment settings such as tracing, language, and statistics time interval settings for an address space that is being started.

You can use the BPE configuration parameter member of the IMS PROCLIB data set to specify parameters for:

- Whether trace entries are written to an external data set, and the external data set that trace entries are written to if so (EXTTRACE=)
- The language that is used for BPE and IMS component messages (LANG=)
- The trace level settings for BPE and IMS component internal trace tables (TRCLEV=)
- The name of a BPE exit list member of the IMS PROCLIB data set where configuration information for IMS component user exit routines is stored (EXITMBR=)
- The time interval between calls to the BPE statistics exit routines (STATINTV=)

Specify the member name by coding `BPECFG=`*member_name* on the EXEC PARM= statement in the address space startup JCL, as shown in this example:

```
EXEC CQSINIT0,PARM='BPECFG=BPECFGCQ'
```

Avoid coding statements in the BPE configuration member that specify definitions for the same resources multiple times. For example, avoid multiple TRCLEV statements for the same trace table type, or multiple EXITMBR statements for the same IMS component type. BPE uses the last statement that it encounters in the member. Any values that are specified on earlier duplicate statements are ignored. Message BPE0017I is issued for each duplicate statement found.

Environments

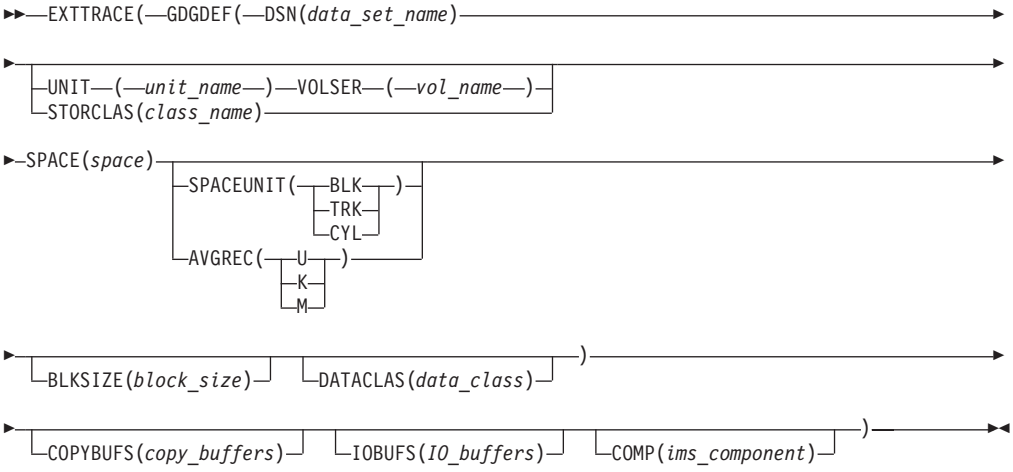
The BPE configuration parameter member of the IMS PROCLIB data set can be used whenever you are using an IMS address space that uses BPE, such as CQS, DBRC, ODBM, OM, RM, SCI, and IMS Connect.

BPECFG= EXTTRACE syntax

The EXTTRACE parameter defines the external trace data set to IMS.

© Copyright IBM Corp. 1974, 2015

EXTTRACE statement



Usage

If you do not include the COMP keyword, the EXTTRACE statement applies to all address spaces that use the current BPE configuration parameter member (BPECFG=) of the IMS PROCLIB data set and do not have a separate EXTTRACE statement. For example, you can define one EXTTRACE statement for CQS address spaces and another for all other address spaces:

```
EXTTRACE ( GDGDEF (...) )
EXTTRACE ( GDGDEF (...)  COMP(CQS) )
```

If you have an EXTTRACE statement that is shared by more than one address space, use dynamic system symbols to make the GDG name unique for each address space. If two or more address spaces use the same GDG name, only one of them can allocate and open the data set at a time. If a BPE address space attempts to allocate a GDG data set that is already in use by another address space, the dynamic allocation fails with a return code 00000004, reason code 02100000, indicating that the data set is allocated by another job. To allow the GDG data set to be used by the second address space, stop the first address space's external tracing. This causes the first address space to deallocate the GDG name so that the second address space can allocate it.

BPECFG= EXTTRACE parameters

BLKSIZE
Specifies the block size for the data set. The maximum allowable decimal value for block size is 32 760.

Recommendation: Although the minimum supported block size is 8340, use a large block size or the maximum block size. Using a large block size is more efficient (multiple records are written per block). In addition, your BPE external trace definitions do not have to be changed if the minimum supported block size is ever increased.

COMP
Specifies the IMS component to which the current EXTTRACE statement applies. Values include:

CQS
Specifies that the current EXTTRACE statement applies only to CQS address spaces.

DBRC
Specifies that the current EXTTRACE statement applies only to DBRC address spaces.

HWS
Specifies that the current EXTTRACE statement applies only to IMS Connect address spaces.

ODBM
Specifies that the current EXTTRACE statement applies only to ODBM address spaces.

OM Specifies that the current EXTTRACE statement applies only to OM address spaces.

RM Specifies that the current EXTTRACE statement applies only to RM address spaces.

SCI
Specifies that the current EXTTRACE statement applies only to SCI address spaces.

COPYBUFS
Specifies the number of buffers that are used to copy the trace table entries from the BPE tables. IMS copies the data from these buffers into the buffers that are specified by the IOBUFS keyword.

The values for this keyword can be 1 - 64000. The default value is 15.

DATACLAS
Specifies the data class for the data set.

If you specify a value for this parameter, the specified data class must have attributes DSORG=PS and RECFM=VB.

GDGDEF
Specifies the generation data group (GDG) data set for the external trace. For general information about GDGs, see *z/OS MVS JCL Reference*. For specific information about defining GDGs for BPE external tracing, see *IMS Version 11 System Administration*.

DSN
Specifies the GDG data set name. This data set name does not include absolute or relative generation numbers. You can create the data set on DASD or you can specify a system-managed storage (SMS) class to manage the data set.

Any static or dynamic z/OS system symbol can be specified in the data set name. This includes all static system symbols that you specify by using SYMDEF in an IEASYMxx PARMLIB member. For example, the dynamic system symbol `&JOBNAME.` can be used to include the job name of the BPE address space in the GDG data set name. Specifying `DSN(IMSTESTL.&JOBNAME..GDG)` results in a data set name `IMSTESTL.SCI1.GDG` for the BPE SCI1 address space. For more information about system symbols, see the topic "Sharing PARMLIB definitions" in *z/OS MVS Initialization and Tuning Reference*.

In addition to the z/OS system symbols, the BPE-defined symbols that are described in the following table can also be specified in the data set name.

Name of symbol	Length of symbol name	Symbol value	Length of symbol value	Example
&BPEUTYPE.	10	Component type	4 (maximum)	USRT001.GDG.SCI.USRT001.GDG.SCI
&BPESNAME.	10	Component system name	8 (maximum)	USRT001.GDG.CQS1CQS.USRT001.GDG.CQS

The value of the &BPEUTYPE. symbol is the name of the IMS component that is writing to the external trace data set. Possible values for &BPEUTYPE. are:

CQS
Common Queue Server address space

DBRC
Database Recovery Control address space

HWS
IMS Connect address space

ODBM
Open Database Manager address space

OM Operations Manager address space

RM Resource Manager address space

SCI
Structured Call Interface address space

The value of the &BPESNAME. symbol is the system name that is specified by the IMS component. This system name is appended to the end of most messages that are issued by most IMS components that run in a BPE address space. The system name for each IMS component is as follows:

CQS
For CQS address spaces, the system name is the CQS SSN appended with "CQS." For example, a CQS with SSN=CQS1 has a system name of CQS1CQS.

DBRC
For DBRC address spaces, the system name is "DBRC" appended with the IMSID of the DBRC. For example, a DBRC with an IMSID of "IMS1" has a system name of DBRCIMS1.

HWS
IMS Connect address spaces do not have a BPE system name. If you specify &BPESNAME. for an IMS Connect address space, the symbol is replaced with a null value. For example, the data set name USRT001.GDG.$&BPESNAME. resolves to USRT001.GDG.$ for an IMS Connect address space.

ODBM
For ODBM address spaces, the system name is the ODBMNAME, as specified on the CSLDIxxx initialization member of the IMS PROCLIB data set, which is appended with "OD." For example, an ODBM with ODBMNAME=ODBM1 has a system name of ODBM1OD.

OM For OM address spaces, the system name is the OMNAME appended with "OM." For example, an OM with OMNAME=OM1 has a system name of OM1OM.

RM For RM address spaces, the system name is the RMNAME appended with "RM." For example, an RM with RMNAME=RM1 has a system name of RM1RM.

SCI
For SCI address spaces, the system name is the SCINAME appended with "SCI." For example, an SCI with SCINAME=SCI1 has a system name of SCI1SCI.

You cannot override the data control block (DCB) attributes for the data set. The disposition for the data set is new (DISP=NEW).

IOBUFS
Specifies the number of buffers that IMS uses to copy data into the external trace data sets.

The values for this keyword can be 1 - 10. The default value is 2.

SPACE
Specifies the number of units of space to be allocated for the data set.

SPACEUNIT
Specifies the quantity of the space for the data in blocks (BLK), tracks (TRK), or cylinders (CYL).

AVGREC
Determines the size of the data set allocation. Values are:

U Use the space quantity that is specified on the SPACE operand.

K Multiply the space quantity that is specified on the SPACE operand by 1024 (1 KB).

M Multiply the space quantity that is specified on the SPACE operand by 1 048 576 (1 MB).

STORCLAS
Specifies the storage class for the data set.

UNIT
Specifies the unit or device type for the data set.

VOLSER
Specifies the volume for the data set.

BPECFG= LANG syntax

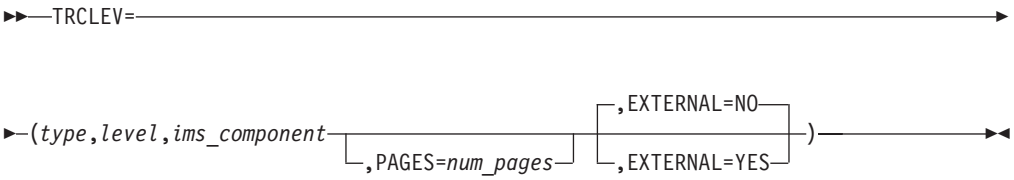
The LANG parameter specifies the language that is used for BPE and IMS component message text. ENU is for US English, which is the only supported language.

BPECFG= TRCLEV syntax

The TRCLEV parameter specifies:

- The trace level for a trace table.
- Optionally, the number of storage pages that are allocated for the trace table.
- Whether entries made to a trace table are written to an external data set.

TRCLEV controls the level of tracing (the amount of detail traced) for each specified trace table type. BPE-managed trace tables are areas in storage where BPE, and the IMS component that uses BPE, can trace diagnostic information about events occurring within the address space.



Usage

BPE-managed trace table entries are written either to memory only, or to both memory and external data sets. IMS writes trace entries to external data sets only if you define the data sets to IMS by using the EXTTRACE parameter.

Some trace table types are defined and owned by BPE itself. These are known as *system trace tables*, and are present in all IMS component address spaces that use BPE. The IMS component can also define its own trace tables. These are known as component trace tables or user-product trace tables, and are only present in address spaces of the defining IMS component. For example, trace table types that are defined by Common Queue Server (CQS) are only present in a CQS address space.

You can share one BPE configuration parameter member of the IMS PROCLIB data set among several different IMS component address spaces. Any TRCLEV statements you code for system trace tables apply identically to all the address spaces that share the member. TRCLEV statements for a particular IMS component trace table are processed only by address spaces running that component. For example, you could have a BPE configuration parameter member of the IMS PROCLIB data set containing TRCLEV statements for BPE, CQS, and Resource Manager (RM) trace table types. When you start a CQS address space, only the BPE and CQS TRCLEV statements are processed. When you start an RM address space, only the BPE and RM TRCLEV statements are processed.

BPECFG= TRCLEV parameters

type
Specifies the type of trace table. A trace table's type refers to the events that are traced into that table. For example, the BPE DISP trace table contains entries that are related to events in the BPE dispatcher.

Specifying an asterisk (*) as the type sets the default trace level (and optionally, the default number of pages per trace table) for all trace table types. If you use an asterisk (*) type, make sure that it is the first TRCLEV statement in your member. You can then code additional TRCLEV statements for specific trace table types to selectively override the defaults.

Recommendation: Code a TRCLEV statement with a type of asterisk (*), specifying a level of at least LOW as your first TRCLEV statement for trace table types. This ensures that at least some tracing is done for all trace tables. Specifying a TRCLEV of asterisk (*) also ensures that any new trace table types

that are activated in the future are turned on in your system, even if you have not modified your BPE configuration parameter member to explicitly add a TRCLEV statement.

Trace table types that require explicit commands are not processed if an asterisk (*) is specified. To configure trace table types that require explicit commands, include a separate TRCLEV statement that explicitly identifies the table. For example: `TRCLEV=(RCTR,MEDIUM,HWS)`.

The trace table types for each IMS component supported by BPE are shown below.

BPE trace table types

BPE provides a set of trace table types for tracing processing within BPE functions. These BPE trace table types are present in all IMS component address spaces. TRCLEV statements specifying a component of BPE are processed for all IMS component address space types.

***** Specifying an asterisk (*) sets the default trace level (and, optionally, the default number of pages per trace table) for all BPE-defined trace table types, except for those requiring explicit commands.

AWE
The asynchronous work element (AWE) services trace table shows details of AWE server creation and deletion and AWE processing requests. The default number of pages for this table is 6.

CBS
The control block services trace table traces requests for control block storage. The default number of pages for this table is 6.

CMD
The command trace table traces the first 48 characters of each command that is processed by BPE. The default number of pages for this table is 2.

DISP
The dispatcher trace table traces BPE dispatcher activity. The default number of pages for this table is 8.

ERR
The error trace table traces error events within a BPE address space. The default number of pages for this table is 2.

Restriction: You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure that error diagnostics are captured. Any level that you specify for the ERR trace table is ignored. You can, however, specify the number of pages for the ERR trace table on the TRCLEV statement.

ERRV
This table contains variable-length BPE error entries that occur within the BPE address space. The default number of pages for this table is 8.

HASH
The hash trace table traces events that are related to BPE hash table services. Currently, only OM, RM, and SCI address spaces request hash table services. For address spaces that do not use the BPE hash table services, this TRCLEV statement is ignored. The default number of pages for this table is 8.

LATC
The latch trace table traces BPE latch management (serialization) activity. The default number of pages for this table is 8.

SSRV
The system services trace table traces general BPE system service calls. The default number of pages for this table is 4.

STG
The storage service trace table traces storage service requests. The default number of pages for this table is 8.

USRX
The user exit routine trace table traces activity that is related to exit routines (for example, loads, calls, or abends). The default number of pages for this table is 4.

CQS trace table types

CQS provides a set of trace table types for tracing processing within the CQS address space. These CQS trace table types are present only in a CQS address space. TRCLEV statements specifying a component of CQS are ignored for any other address space type.

* Specifying an asterisk (*) sets the default trace level (and, optionally, the default number of pages per trace table) for all CQS-defined trace table types, except for those requiring explicit commands.

CQS
The CQS trace table traces general activity that is not related to a specific structure. The default number of pages for this table is 4.

ERR
The error trace table traces error events within a CQS address space. The default number of pages for this table is 4.

Restriction: You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure that error diagnostics are captured. Any level that you specify for the ERR trace table is ignored. You can, however, specify the number of pages for the ERR trace table on the TRCLEV statement.

INTF
The interface trace table traces activity in the interface between a CQS and its client. The default number of pages for this table is 8.

STR
The structure trace table traces activity that is related to a structure. CQS defines one STR trace table for each structure pair that is defined to CQS. The default number of pages for this table is 8.

DBRC trace table types

A DBRC address space that is using the Base Primitive Environment defines the following values for DBRC-owned trace tables:

* By specifying an asterisk (*), you can set the default trace level (and optionally, the default number of pages per trace table) for all DBRC-defined trace table types, except for those requiring explicit commands.

ERR
This table is used to trace errors that occur within the DBRC address space. The default number of pages for this table is 2.

Restriction: You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure that error diagnostics are captured. Any level that you specify for the ERR trace table is ignored. You can, however, specify the number of pages for the ERR trace table on the TRCLEV statement.

GRPS
This table is used for DBRC group services message and notification tracing. The default number of pages for this table is 8.

MODF
This table is used for DBRC module flow tracing. The default number of pages for this table is 8.

RQST
This table is used for general DBRC request processing. The default number of pages for this table is 8.

IMS Connect trace table types

IMS Connect defines its own trace tables by using a set of trace table types for tracing processing within IMS Connect functions. These tables are known as component trace tables or user-product trace tables. You can code the following values for IMS Connect-defined trace tables:

***** Specifying an asterisk (*) sets the default trace level (and optionally, the default number of pages per trace table) for all IMS Connect-defined trace table types, except for those requiring explicit commands.

ADPT
Adapter trace table.

CMDT
The command trace table traces IMS Connect command activity. The default number of pages for this table is 2.

ENVT
The interface trace table traces activity in the interface between an IMS Connect and its client. The default number of pages for this table is 2.

ERRV
This table contains variable-length IMS Connect error entries that occur within the BPE address space.

HWSI
The IMS Connect to OTMA driver trace table traces communication activity between IMS Connect and OTMA drivers. The default number of pages for this table is 2.

HWSN
The IMS Connect to local option driver trace table traces communication activity and event between local option driver and IMS Connect. The default number of pages for this table is 2.

HWSO
The IMSplex driver (IPDC) trace table traces communication activity and events between the IMSplex driver and IMS Connect. The default number of pages for this table is 2.

HWSW
The IMS Connect to TCP/IP driver trace table traces communication activity and events between TCP/IP drivers and IMS Connect. The default number of pages for this table is 2.

LEPS
Language Environment (LE) preinitialization services trace table.

ODBM
ODBM communication trace table.

ODDR
ODBM driver trace table.

OMDR
The IMSplex driver (IPDC) trace table traces communication protocol activity (SCI calls). The default number of pages for this table is 2.

OSDR
ODBM socket driver trace table.

OSOC
ODBM socket communication trace table.

OTMA
The OTMA communication driver trace table traces internal communication protocol activity (z/OS cross-system coupling facility calls). The default number of pages for this table is 2.

PCDR
The local option driver trace table traces local option communication protocol activity. The default number of pages for this table is 2.

PCIF
PC interface trace table.

RCTR
The Recorder trace table is used to reroute variable-length recorder trace data to the BPE external trace data set. All recorder trace entries are generated at LEVEL(MEDIUM). The Recorder trace table requires explicit commands.

RRSI
RRS interface trace table.

TCPI
The TCP/IP communication driver trace table traces communication protocol activity (TCP/IP calls). The default number of pages for this table is 2.

ODBM trace table types

An ODBM address space that is using the Base Primitive Environment defines the following values for ODBM-owned trace tables:

*****
Specifying an asterisk (*) sets the default trace level (and, optionally, the default number of pages per trace table) for all ODBM-defined trace table types, except for those requiring explicit commands.

CSL
The Common Service Layer (CSL) trace table is used for routines that are common to all CSL managers. The default number of pages for this table is 4.

ERR
This table is used to trace errors that occur within the ODBM address space. The default number of pages for this table is 4.

Restriction: You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure that error diagnostics are captured. Any level that you specify for the ERR trace table is ignored. You can, however, specify the number of pages for the ERR trace table on the TRCLEV statement.

ODBM
This table is used for general ODBM processing. The default number of pages for this table is 4.

PLEX
This table is used for ODBM processing for a specific IMSplex. The default number of pages for this table is 8.

OM trace table types

OM provides a set of trace table types for tracing processing within the OM address space. These OM trace table types are present only in an OM address space. TRCLEV statements specifying a component of OM are ignored for any other address space type.

***** Specifying an asterisk (*) sets the default trace level (and, optionally, the default number of pages per trace table) for all OM-defined trace table types, except for those requiring explicit commands.

CSL
The Common Service Layer (CSL) trace table is used for routines that are common to all CSL managers. The default number of pages for this table is 4.

ERR
The error trace table traces error events within an OM address space. The default number of pages for this table is 4.

Restriction: You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure that error diagnostics are captured. Any level that you specify for the ERR trace table is ignored. You can, however, specify the number of pages for the ERR trace table on the TRCLEV statement.

OM The Operations Manager (OM) trace table traces events that are related to general OM processes. The default number of pages for this table is 4.

PLEX
The IMSplex trace table traces OM processing for a specific IMSplex. The default number of pages for this table is 8.

RM trace tables

RM provides a set of trace table types for tracing processing within the RM address space. These RM trace table types are present only in an RM address space. TRCLEV statements specifying a component of RM are ignored for any other address space type.

***** Specifying an asterisk (*) sets the default trace level (and, optionally, the default number of pages per trace table) for all RM-defined trace table types, except for those requiring explicit commands.

CSL
The Common Service Layer (CSL) trace table is used for routines that are common to all CSL managers. The default number of pages for this table is 4.

ERR
The error trace table traces error events within an RM address space. The default number of pages for this table is 4.

Restriction: You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure that error diagnostics are captured. Any level that you specify for the ERR trace table is ignored. You can, however, specify the number of pages for the ERR trace table on the TRCLEV statement.

RM The Resource Manager (RM) trace table traces events that are related to general RM processes. The default number of pages for this table is 4.

PLEX
The IMSplex trace table traces RM processing for a specific IMSplex. The default number of pages for this table is 8.

SCI trace tables

SCI provides a set of trace table types for tracing processing within the SCI address space. These SCI trace table types are present only in a SCI address space. TRCLEV statements specifying a component of SCI are ignored for any other address space type.

***** Specifying an asterisk (*) sets the default trace level (and, optionally, the default number of pages per trace table) for all SCI-defined trace table types, except for those requiring explicit commands.

CSL
The Common Service Layer (CSL) trace table is used for routines that are common to all CSL address spaces. The default number of pages for this table is 8.

ERPL
The error parameter list trace table traces a copy of the interface parameter list when an error occurs processing an SCI request or message. The default number of pages for this table is 8.

ERR
The error trace table traces error events within an SCI address space. The default number of pages for this table is 4.

Restriction: You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure that error diagnostics are captured. Any level that you specify for the ERR trace table is ignored. You can, however, specify the number of pages for the ERR trace table on the TRCLEV statement.

INTF
The interface trace table traces IMSplex member interface activity (requests and messages). The default number of pages for this table is 8.

INTP
The interface parameter trace table traces a copy of the interface parameter list during SCI request and message processing. The default number of pages for this table is 16.

PLEX
The IMSplex trace table traces SCI processing for a specific IMSplex. The default number of pages for this table is 8.

SCI
The Structured Call Interface (SCI) trace table traces events that are related to general SCI processes. The default number of pages for this table is 8.

level
Controls how much tracing is done in the specified trace table. Each trace entry that is made has a level that is associated with the entry. Each trace table has a level setting that is controlled by the value for level that you specify on the TRCLEV statement for the table.

A trace entry is written only if the trace entry's level is less than or equal to the table's level setting. For example, if the trace entry level is MEDIUM, the trace entry is added to the trace table only if the table's level is MEDIUM or HIGH. Thus, the level you specify controls the volume (number) of trace entries that are written to a given table.

A low setting of the level parameter results in fewer trace entries being made to the table. The trace table does not wrap as quickly as with a higher setting (which means that diagnostic information remains available for a longer time), and the performance impact is minimized. However, the trace information is not as detailed as with higher settings, so the captured information might not be sufficient to solve a problem.

A high setting of the level parameter results in more trace entries being written to the table. This can provide additional diagnostic information for solving a problem; however, the trace table tends to wrap more frequently, and higher settings can cause additional CPU usage.

Choose one of the following for the level parameter:

NONE
No tracing.

Recommendation: Do not specify NONE because no tracing, not even tracing for error conditions, is done for the specified table.

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

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

MEDIUM
Medium-volume tracing (most component events).

HIGH
High-volume tracing (all component events).

ims_component
Specifies the IMS component that defines the trace table type. Possible values are:

BPE
Indicates that the table is a BPE-defined (system) trace table. BPE trace tables exist in all IMS component address spaces that run with BPE.

CQS
Indicates that the table is a Common Queue Server-defined trace table type.

DBRC
Indicates that the table is a DBRC-defined trace table type.

HWS
Indicates that the table is an IMS Connect-defined trace table type.

ODBM
Indicates that the table is an ODBM-defined trace table type.

OM Indicates that the table is an Operations Manager-defined trace table type.

RM Indicates that the table is a Resource Manager-defined trace table type.

SCI
Indicates that the table is a Structured Call Interface-defined trace table type.

PAGES=*num_pages*
An optional parameter that can be used to specify the number of 4 KB pages to be allocated for the table type.

Specify a value from 1 - 32767 pages for this parameter. If BPE is unable to get the amount of storage you requested for a trace table, it tries to get a smaller number of pages to enable some tracing to be done. You can see the actual number of pages BPE obtained for each trace by issuing the DISPLAY TRACETABLE command.

If you do not use this parameter, then the trace table has the default number of pages, as specified under the description of each trace table type.

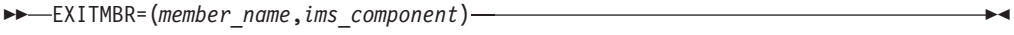
EXTERNAL=

YES
Specifies that trace entries are written to the external trace data set. However, IMS writes the trace entries only if you define the data set to IMS by using the EXTTRACE statement. If you use the BPE recorder trace facility, you must specify EXTERNAL=YES.

NO Specifies that trace entries are written only to memory and not to the external trace data set. NO is the default value.

BPECFG= EXITMBR syntax

The EXITMBR parameter specifies the exit list member name. You can specify one EXITMBR= parameter for each IMS component running with BPE, and one EXITMBR= parameter for BPE itself.



BPECFG= EXITMBR parameters

member_name
Specifies the 8-character exit list member name.

ims_component
Specifies the IMS component whose user exit routines are being defined. Possible values are:

BPE
Indicates the BPE exit routine member name.

CQS
Indicates the Common Queue Server exit routine member name.

DBRC
Indicates the DBRC exit routine member name.

HWS
Indicates the IMS Connect exit routine member name.

ODBM
Indicates the Open Database Manager (ODBM) exit routine member name.

OM Indicates the Operations Manager (OM) exit routine member name.

RM Indicates the Resource Manager (RM) exit routine member name.

SCI
Indicates the Structured Call Interface (SCI) exit routine member name.

BPECFG= STATINTV syntax

The optional STATINTV parameter specifies the time interval, in seconds, between calls to the BPE statistics exit or exit routines. You can set STATINTV from 1 to 2147483647 (2^{31}-1). The default STATINTV value is 600 (10 minutes).

Usage

Specify a STATINTV value of 60 or more to avoid possible performance problems due to frequent exit routine calls.

BPECFG= examples

A sample BPE configuration data set for BPE is shown in the example below. This example shows a BPE configuration data set that can be shared by IMS Connect, CQS, CSL, and DBRC. It contains definitions for traces for:

- BPE
- CQS
- DBRC
- IMS Connect
- ODBM
- OM
- RM
- SCI

It also contains user exit routine list member specifications.

```
********************************************************************
*  CONFIGURATION FILE FOR BPE                                      *
********************************************************************

LANG=ENU                        /* Language for messages       */
                                /* (ENU = U.S. English)        */
STATINTV=420                    /* STATS user exit interval    */
                                /* = 420 seconds (7 minutes)   */

#
```

```
# Definitions for BPE system traces
#

TRCLEV=(*,LOW,BPE)              /* Set default for all BPE     */
                                /* traces to LOW.              */
TRCLEV=(AWE,HIGH,BPE)           /* AWE server trace on high    */
TRCLEV=(CBS,MEDIUM,BPE)         /* Ctrl blk serv trc on medium */
TRCLEV=(DISP,HIGH,BPE,PAGES=12) /* Dispatcher trace on high    */
                                /* with 12 pages               */

#
# Definitions for IMS Connect traces
#
TRCLEV(RCTR,MEDIUM,HWS,EXT=YES)   /* Starts recorder trace      */
                                   /* facility                   */


# Definitions for CQS traces
#

TRCLEV=(*,MEDIUM,CQS)           /* Set default for all CQS     */
TRCLEV=(STR,HIGH,CQS)           /* traces to medium            */
                                /* but run STR trace on high   */
# DBRC:
|
TRCLEV=(*,LOW,DBRC)             /* DEFAULT DBRC TRACES TO LOW */
TRCLEV=(RQST,HIGH,DBRC)        /* DBRC GENERAL TRACE ON HIGH  */
TRCLEV=(MODF,MEDIUM,DBRC)       /* MODULE FLOW TRACE ON MEDIUM*/
#
#
# DEFINITIONS FOR ODBM TRACES - SET DEFAULT FOR ALL ODBM TRACES
# TO LOW, THEN SELETIVELY OVERRIDE THOSE THAT NEED A DIFFERENT
# LEVEL.
#

TRCLEV=(*,LOW,ODBM)                 /* DEFAULT ODBM TRACES TO LOW */
TRCLEV=(CSL,HIGH,ODBM)              /* CSL TRACE ON HIGH          */
TRCLEV=(ODBM,HIGH,ODBM)             /* ODBM GENERAL TRACE ON HIGH */
TRCLEV=(PLEX,HIGH,ODBM)             /* IMSPLEX TRACE ON HIGH      */
#
#
# Definitions for OM traces
#

TRCLEV=(*,MEDIUM,OM)            /* Set default for all OM      */
                                /* traces to medium            */

#
# Definitions for RM traces
#

TRCLEV=(*,MEDIUM,RM)            /* Set default for all RM      */
                                /* traces to medium            */

#
# Definitions for SCI traces
#

TRCLEV=(*,MEDIUM,SCI)           /* Set default for all SCI     */
                                /* traces to medium            */
TRCLEV=(INTF,HIGH,SCI)          /* Intf call trace on high     */
TRCLEV=(INTP,HIGH,SCI)          /* Intf parmlist trace on high */

#
# User exit list PROCLIB member specifications
#

EXITMBR=(BPEEXIT0,BPE)          /* BPE user exit definitions   */
```

```
EXITMBR=(DBREXIT0,DBRC)          /* DBRC user exit definitions  */
EXITMBR=(CQSEXIT0,CQS)           /* CQS user exit definitions   */
EXITMBR=(OMEXIT00,OM)            /* OM user exit definitions    */
EXITMBR=(RMEXIT00,RM)            /* RM user exit definitions    */
EXITMBR=(SCIEXIT0,SCI)           /* SCI user exit definitions   */
EXITMBR=(CSLEXDM0,ODBM)          /* SPECIFY PROCLIB DATASET     */
                                 /* MEMBER CSLEXDM0 AS ODBM'S   */
                                 /* USER EXIT LIST MEMBER       */
```

A sample member of the IMS PROCLIB data set with generic and specific EXTTRACE entries is shown in the example below.

You can use z/OS symbolic substitution parameters in the data set name of the BPE external trace GDG data set. This makes a single EXTTRACE specification unique to each address space. For example, you can use the symbol `&JOBNAME.` in the data set name; it is replaced by the actual address space job name when a BPE address space starts and allocates the external trace data set.

Sample BPE configuration member for a single IMS Connect address space

```
LANG=ENU           /* LANGUAGE FOR MESSAGES     */
                   /* (ENU = U.S. ENGLISH)      */
#
# DEFINITIONS FOR IMS CONNECT TRACES
# HWS:
TRCLEV=(ENVT,MEDIUM,HWS,EXTERNAL=YES)
TRCLEV=(HWSO,HIGH,HWS,EXTERNAL=NO)
TRCLEV=(PLEX,HIGH,ODBM)
TRCLEV=(OMDR,LOW,HWS,EXTERNAL=YES)
TRCLEV=(OTMA,HIGH,HWS,EXTERNAL=YES)

# External trace definitions:
EXTTRACE(GDGDEF(DSN(IMSTESTL.BPETHWS) UNIT(SYSDA) VOLSER(000000)
SPACE(5) ) COMP(HWS) )  /* EXTTRACE specific to IMS Connect address space */
```

Sample BPE configuration with multiple EXTTRACE statements

A sample BPE member of the IMS PROCLIB data set with multiple EXTTRACE statements is shown in the example below.

```
LANG=ENU           /* LANGUAGE FOR MESSAGES     */
                   /* (ENU = U.S. ENGLISH)      */

#
# DEFINITIONS FOR SYSTEM TRACES
#
TRCLEV=(*,LOW,BPE,EXTERNAL=NO)

# CQS:

TRCLEV=(CQS,MEDIUM,CQS,EXTERNAL=YES)
TRCLEV=(ERR,HIGH,CQS,EXTERNAL=NO)
TRCLEV=(INTF,LOW,CQS,EXTERNAL=YES)
TRCLEV=(STR,HIGH,CQS,EXTERNAL=YES)
TRCLEV=(OFLW,HIGH,CQS,EXTERNAL=YES)
TRCLEV=(SEVT,HIGH,CQS,EXTERNAL=YES)

# SCI:

TRCLEV=(ERR,HIGH,SCI,EXTERNAL=NO)
TRCLEV=(INTF,LOW,SCI,EXTERNAL=YES)
TRCLEV=(INTP,HIGH,SCI,EXTERNAL=YES)
TRCLEV=(SCI,MEDIUM,SCI,EXTERNAL=YES)
```

```
# OM:

TRCLEV=(*,LOW,OM,EXTERNAL=YES)     /* SET ALL TABLES TO LOW TRACING */

# ODBM:

TRCLEV=(*,LOW,ODBM,EXTERNAL=YES)   /* SET ALL TABLES TO LOW TRACING */

# DBRC:

TRCLEV=(*,MEDIUM,DBRC,EXTERNAL=YES) /* SET ALL TABLES TO MEDIUM TRACING */

# External trace definitions:

EXTTRACE(GDGDEF(DSN(IMSTESTL.BPETCQS) UNIT(SYSDA) VOLSER(000000)
SPACE(5) ) COMP(CQS) )  /* <------ EXTTRACE for CQS                */

EXTTRACE(GDGDEF(DSN(IMSTESTL.BPETDBRC) UNIT(SYSDA) VOLSER(000000)
SPACE(5) ) COMP(DBRC) ) /* <------ EXTTRACE for DBRC               */

EXTTRACE(GDGDEF(DSN(IMSTESTL.BPETSCI) UNIT(SYSDA) VOLSER(000000)
SPACE(5) ) COMP(SCI) )  /* <------ EXTTRACE for SCI                */

EXTTRACE(GDGDEF(DSN(IMSTESTL.BPETOM) UNIT(SYSDA) VOLSER(000000)
SPACE(5) ) COMP(OM) )   /* <------ EXTTRACE for OM                 */

EXTTRACE(GDGDEF(DSN(IMSTESTL.BPETODBM) UNIT(SYSDA) VOLSER(000000)
SPACE(5) ) COMP(ODBM) )   /* <------ EXTTRACE for ODBM             */
```

Sample BPE configuration using symbolic parameters for EXTTRACE statements

A sample BPE member of the IMS PROCLIB data set with EXTTRACE statements using symbolic parameters is shown in the example below.

```
#
# DEFINITIONS FOR SYSTEM TRACES
#
TRCLEV=(*,LOW,BPE,EXTERNAL=NO)

# SCI:

TRCLEV=(ERR,HIGH,SCI,EXTERNAL=NO)
TRCLEV=(INTF,LOW,SCI,EXTERNAL=YES)
TRCLEV=(INTP,HIGH,SCI,EXTERNAL=YES)
TRCLEV=(SCI,MEDIUM,SCI EXTERNAL=YES)

#OM:

TRCLEV=(*,LOW,OM,EXTERNAL=YES)

#ODBM:

TRCLEV=(*,LOW,ODBM,EXTERNAL=YES)

#DBRC:
TRCLEV=(*,MEDIUM,DBRC,EXTERNAL=YES)

#RM:

TRCLEV=(CSL,HIGH,RM,EXTERNAL=YES)
TRCLEV=(ERR,HIGH,RM,EXTERNAL=NO)
TRCLEV=(PLEX,LOW,RM,EXTERNAL=YES)
TRCLEV=(RM,MEDIUM,RM,EXTERNAL=YES)

# External trace definitions:
# Use symbolic symbols in the DSN, SMS testing
```

```
/* EXTTRACE with symbolic data set name BPE&JOBNAME     */
EXTTRACE(GDGEF(DSN(IMSTESTL.BPE&JOBNAME) STORCLAS(SNPSTOR)
SPACE(20) AVGREC(K) DATACLAS(SNPDATA) ) )

/* EXTTRACE statement specifically for RM address space */
/* to allow it to have a name and characteristics       */
/* different from the generic specification, above.     */
EXTTRACE(GDGDEF(DSN(IMSTESTL.RMTRACE) STORCLAS(SNPSTOR)
SPACE(30) AVGREC(K) DATACLAS(SNPDATA)) COMP(RM)  )
```

Related tasks:

"Setting up tracing for BPE-managed address spaces" on page 244

Related reference:

Base Primitive Environment commands (Commands)

BPE UPDATE TRACETABLE command (Commands)

RM trace record example (Diagnosis)

"Trace environment - conservative" on page 239

"Trace environment - more aggressive" on page 239

BPE exit list members of the IMS PROCLIB data set

Use the members of the IMS PROCLIB data set specified by the EXITMBR= parameter in the BPE configuration parameter member of the IMS PROCLIB data set to define user exit routines to BPE.

BPE exit list members of the IMS PROCLIB data set are IMS-component specific. In the BPE configuration parameter member of the IMS PROCLIB data set, specify one EXITMBR statement for each IMS component that provides user exit routines through BPE services. Each EXITMBR statement specifies the name of an IMS PROCLIB data set member that contains the definitions for exit routines for that IMS component. You can have a separate exit list member for each IMS component, or you can share one exit list member among several IMS components.

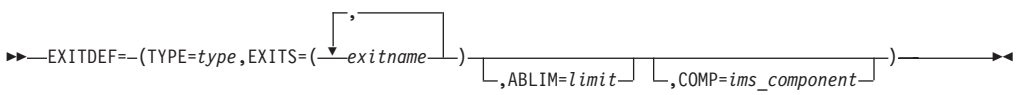
A BPE exit list member associates a user exit routine type with a list of one or more user exit routines. Use the EXITDEF statement to define the exit routine modules to be called for a particular exit routine type. The BPE exit list member is processed by BPE during address space initialization. It is also processed when you enter a `REFRESH USEREXIT` command (see *IMS Version 11 Commands, Volume 2: IMS Commands N-V* for more information about BPE USEREXIT commands).

Environments

The BPE exit list member of the IMS PROCLIB data set can be used whenever you are using an IMS address space that uses BPE, such as CQS, DBRC, ODBM, OM, RM, SCI, and IMS Connect.

BPE EXITMBR= EXITDEF syntax

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.



Usage

The EXITDEF statement associates an exit routine type with a list of one or more exit routine modules to be called. The modules are called in the order listed. The EXITDEF statement consists of a sublist (enclosed in parentheses) containing the keywords TYPE, EXITS, ABLIM, and COMP.

Recommendation: Avoid coding statements in the BPE exit routine list member that specify definitions for the same exit routine type multiple times. BPE always uses the last statement it encounters in the member for a particular exit routine. Any earlier statements for the same exit routine are ignored. Message BPE0017I is issued for each duplicate statement found.

If you code the same user exit routine name more than once in the exit routine list (EXIT=) of any single EXITDEF= statement, BPE always uses the first occurrence of the exit routine module name to determine the order for calling the exit routines. Duplicate names are ignored, and a message, BPE0018I is issued for each duplicate name.

BPE EXITMBR= EXITDEF parameters

TYPE=*type*

Specifies the type of exit routine. The IMS component defined on COMP= determines the types of exit routines that are supported. Refer to *IMS Version 11 Exit Routines* for more information about these types of exit routines.

BPE exit routine types

All BPE-owned user exit routines are available to all IMS address spaces running with BPE.

INITTERM
An exit routine that is called once during early BPE initialization, and once during normal termination.

STATS
An exit routine that is called periodically (it is timer-driven), and is called once during normal address space shut down, with statistics about BPE system functions. Optionally, the IMS component that is running on BPE can provide statistics specific to its operation.

CQS exit routine types

CLNTCONN
An exit routine that is called during client connect and disconnect processing.

INITTERM
An exit routine that is called during various phases of initialization and termination.

OVERFLOW
An exit routine that is called during overflow threshold processing to select queue names for overflow processing.

STRSTAT
An exit routine that is called during checkpoint processing to allow you to gather structure statistics.

STREVENT
An exit routine that is called for various structure events. For certain structure events, it also allows you to gather structure statistics like the STRSTAT exit routine.

DBRC exit routine types

RECONIO
An exit routine that is called to allow for auditing of DBRC RECON data set I/O. If this exit routine is specified, the standard RECON I/O exit routine, DSPCEXT0, is not called.

REQUEST
An exit routine that is called at start and end of DBRC request processing.

SECURITY
An exit routine that is called to allow user security checking before command execution. If this exit routine is specified, the standard DBRC command authorization exit routine, DSPDCAX0, is called.

IMS Connect exit routine types

ODBMAUTH
IMS Connect DB security user exit, which enables users to access IMS database resources via ODBA from outside an IMS dependent region.

ODBMROUT
IMS Connect Routing exit for ODBM, which enables users to override the IMS alias or to select an ODBM.

POR*nnnnn*
IMS Connect Port Message Edit exit. *nnnnn* in the exit type name is the port number in decimal—for example, POR01234. The Port Message Edit exit is a BPE type-2 exit.

XMLADAP
An exit routine that is called to perform XML-to-COBOL data conversion in IMS Connect.

ODBM exit routine types

CLNTCONN
An exit routine that is called during client command registration and deregistration processing.

INITTERM
An exit routine that is called during various phases of initialization and termination.

INPUT
An exit routine that is called to view DL/I calls that are issued to IMS databases. This exit routine can either modify the command before execution or reject the command before it is processed.

OUTPUT
An exit routine that is called to view output (for example, ODBA call output) from ODBM to an ODBM client. The exit routine can modify the output before it is returned to the originator of the command.

OM exit routine types

CLNTCONN
An exit routine that is called during client command registration and deregistration processing.

INITTERM
An exit routine that is called during various phases of initialization and termination.

INPUT
An exit routine that is called to view command input to the Operations Manager. This exit routine can either modify the command before execution or reject the command before it is processed.

OUTPUT
An exit routine that is called to view output (for example, command response) from Operations Manager to an automation client. The exit routine can modify the output before it is returned to the originator of the command.

SECURITY
An exit routine that is called to allow user security checking before command execution.

RM exit routine types

CLNTCONN
An exit routine that is called during client connect and disconnect processing.

INITTERM
An exit routine that is called during various phases of initialization and termination.

SCI exit routine types

CLNTCONN
An exit routine that is called during client connect and disconnect processing.

INITTERM
An exit routine that is called during various phases of initialization and termination.

EXITS=(*exitname***,...)**
Specifies a list of one or more exit routine module names. The position of the exit routine in the list determines the order in which the exit routine is driven. The default order is first to last (the first exit listed on the EXITS= parameter is the first exit to be called). However, for some exit types, the order is reversed (the last exit listed is the first to be called). Exit types that use a reverse call order explicitly state this in their documentation. Refer to *IMS Version 11 Exit Routines* for the specific exit type you are writing to determine the exit call order.

When an exit routine returns to its caller, it indicates whether additional exit routines are to be called.

ABLIM=*limit*
A number from 0 to 2 147 483 647 that specifies the abend limit for the type of exit routine being defined. If the number of abends for an exit routine

module reaches the abend limit for the exit routine type, the module is removed from the exit routine list and is not called until the exit routine type is refreshed.

This parameter is optional; the default is 1. If you specify a value of 0, there is no abend limit.

COMP=*ims_component*
An optional parameter that specifies the type of the IMS component that owns the exit routine being defined. Possible values are:

BPE
Base Primitive Environment

CQS
Common Queue Server

DBRC
Database Recovery Control

HWS
IMS Connect

ODBM
Open Database Manager

OM Operations Manager

RM Resource Manager

SCI
Structured Call Interface

BPE processes only EXITDEF statements that:

- Do not have COMP coded
- Have COMP=*ims_component* coded (where *ims_component* matches the IMS component specified on the EXITMBR statement that points to the BPE user exit PROCLIB member currently being processed).

  For example, if BPE were processing the BPEEXIT0 PROCLIB member specified on the EXITMBR=(BPEEXIT0,BPE) statement, it would only process EXITDEF statements that had no COMP= specified, and those that had COMP=BPE specified. If BPE were processing the CQSEXIT0 PROCLIB member specified on the EXITMBR=(CQSEXIT0,CQS) statement, it would only process EXITDEF statements that had no COMP= specified, and those that had COMP=CQS specified.

  For any given IMS component address space, BPE only processes BPE user exit PROCLIB members for EXITMBR statements that specify BPE, and those that specify the IMS component name of the address space that is running (for example, CQS, DBRC, HWS, ODBM, OM, RM, or SCI).

Examples

Several sample user exit list members of the IMS PROCLIB data set are provided.

Sample BPE user exit list member of the IMS PROCLIB data set

The sample BPE user exit list member of the IMS PROCLIB data set shown below defines:

- One BPE init/term exit routine

- One BPE Statistics exit routine

```
*********************************************************************
* BPE USER EXIT LIST PROCLIB MEMBER                                 *
*********************************************************************

#-------------------------------------------------------------------#
# Define one BPE init/term exit: MYINIT00.                          #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(MYINIT00))

#-------------------------------------------------------------------#
# Define 1 BPE Statistics exit: HHGSTAT0 with an abend limit of 42#
#-------------------------------------------------------------------#
EXITDEF(TYPE=STATS,EXITS=(HHGSTAT0),ABLIM=42)

#-------------------------------------------------------------------#
# Define 1 statistics exit for DBRC statistics : STATS01            #
#-------------------------------------------------------------------#
EXITDEF(TYPE=STATS,EXITS=(STATS01),COMP=BPE)
```

Sample CQS user exit list member of the IMS PROCLIB data set

The sample CQS user exit list member of the IMS PROCLIB data set shown below defines:

- One client connection exit routine
- Two INITTERM user exit routines
- Four overflow exit routines
- One structure statistic exit routine
- One CQS structure event user exit routine

```
*********************************************************************
* CQS USER EXIT LIST PROCLIB MEMBER                                 *
*********************************************************************

#-------------------------------------------------------------------#
# DEFINE 1 CLIENT CONNECTION EXIT: CLCONX00                         #
#------------------------------------------------------------------ #
EXITDEF(TYPE=CLNTCONN,EXITS=(CLCONX00))

#-------------------------------------------------------------------#
# DEFINE 2 INITTERM USER EXITS:  MYCQSIT0 AND OEMCQIT0              #
# WITH AN ABEND LIMIT OF 8.
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(MYCQSIT0,OEMCQIT0),ABLIM=8)

#-------------------------------------------------------------------#
# DEFINE 4 OVERFLOW EXITS: OVERFL01, OVERFL02, OVERFL03, OVERFL04 #
#-------------------------------------------------------------------#
EXITDEF(TYPE=OVERFLOW,EXITS=(OVERFL01,
                             OVERFL02,
                             OVERFL03,
                             OVERFL04))

#-------------------------------------------------------------------#
# DEFINE 1 STRUCTURE STATISTIC EXIT: STRSTAT0                       #
#-------------------------------------------------------------------#
EXITDEF(TYPE=STRSTAT,EXITS=(STRSTAT0))

#-------------------------------------------------------------------#
# DEFINE 1 CQS STRUCTURE EVENT USER EXIT (STREVNT0) WITH            #
# NO ABEND LIMIT                                                    #
#-------------------------------------------------------------------#
EXITDEF(TYPE=STREVENT,EXITS=(STREVNT0),ABLIM=0)
```

Sample DBRC user exit list member of the IMS PROCLIB data set

The sample DBRC user exit list member of the IMS PROCLIB data set shown below defines:

- One RECON I/O exit routine
- An abend limit of 8.
- Three DBRC security exit routines

```
/*****************************************************************/
/* DBRC USER EXIT LIST PROCLIB MEMBER                            */
/*****************************************************************/
#---------------------------------------------------------------#
# Define 1 DBRC RECON I/O exit: RECONIO1                        #
#---------------------------------------------------------------#
EXITDEF(TYPE=RECONIO,EXITS=(RECONIO1),ABLIM=8,COMP=DBRC)

#---------------------------------------------------------------#
# Define 3 DBRC security exits: SECUIRE01, SECURE02, and SECURE03 #
#---------------------------------------------------------------#
EXITDEF(TYPE=SECURITY,EXITS=(SECURE03,SECURE01,SECURE02),COMP=DBRC)
```

Sample ODBM user exit list member of the IMS PROCLIB data set

The sample DBRC user exit list member of the IMS PROCLIB data set shown below defines:

- One ODBM initialization/termination exit routine
- An abend limit of 8.
- One ODBM output exit routine

```
*********************************************************************
* ODBM USER EXIT LIST PROCLIB MEMBER                                *
*********************************************************************

#---------------------------------------------------------------#
# DEFINE 1 ODBM INIT/TERM USER EXIT: ZDINTM00                   #
#---------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZDINTM00),COMP=ODBM)

#---------------------------------------------------------------#
# DEFINE 1 ODBM INPUT USER EXIT: ZINPUT00                       #
# WITH AN ABEND LIMIT OF 8.                                     #
#---------------------------------------------------------------#
EXITDEF(TYPE=INPUT,EXITS=(ZINPUT00),ABLIM=8,COMP=ODBM)

#---------------------------------------------------------------#
# DEFINE 1 ODBM OUTPUT USER EXIT: ZOUTPUT0                      #
#---------------------------------------------------------------#
EXITDEF(TYPE=OUTPUT,EXITS=(ZOUTPUT0),COMP=ODBM)
```

Sample IMS Connect user exit list member of the IMS PROCLIB data set

The sample IMS Connect user exit list member of the IMS PROCLIB data set shown below defines the XML adapter system routine.

```
*********************************************************************
* HWS USER EXIT LIST PROCLIB MEMBER *
*********************************************************************

#---------------------------------------------------------------#
# Define XML Adapter system routine: HWSXMLA0 #
#---------------------------------------------------------------#
EXITDEF(TYPE=XMLADAP,EXITS=(HWSXMLA0),ABLIM=8,COMP=HWS)
```

Sample OM user exit list member of the IMS PROCLIB data set

The sample OM user exit list member of the IMS PROCLIB data set shown below defines:

- One OM init/term exit routine
- Two OM client connection exit routines
- One OM command input exit routine
- One OM command output exit routine
- Three OM security exit routines

```
*********************************************************************
* OM USER EXIT LIST PROCLIB MEMBER                                  *
*********************************************************************

#-------------------------------------------------------------------#
# Define one OM init/term exit: OMINITRM.                           #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(OMINITRM))

#-------------------------------------------------------------------#
# Define 2 OM client connection exits: OMCLCN00 and OEMCLI00        #
# with an abend limit of 2.                                         #
#-------------------------------------------------------------------#
EXITDEF(TYPE=CLNTCONN,EXITS=(OMCLCN00,OEMCLI00),ABLIM=2)

#-------------------------------------------------------------------#
# Define one OM command input exit: MYCMI000                        #
# with no abend limit.                                              #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INPUT,EXITS=(MYCMI000),ABLIM=0)
#-------------------------------------------------------------------#
# Define one OM command output exit: MYCMO000                       #
# with no abend limit.                                              #
#-------------------------------------------------------------------#
EXITDEF(TYPE=OUTPUT,EXITS=(MYCMO000),ABLIM=0)

#-------------------------------------------------------------------#
# Define 3 OM security exits: OMSEC000,OMSEC001, and ZZZSEC00       #
#-------------------------------------------------------------------#
EXITDEF(TYPE=SECURITY,EXITS=(OMSEC000,
                             OMSEC001,
                             ZZZSEC00))
```

Sample RM user exit list member of the IMS PROCLIB data set

The sample RM user exit list member of the IMS PROCLIB data set shown below defines:

- One RM init/term exit routine
- Two RM client connection exit routines

```
*********************************************************************
* RM USER EXIT LIST PROCLIB MEMBER                                  *
*********************************************************************

#-------------------------------------------------------------------#
# Define one RM init/term exit: RMINITRM.                           #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(RMINITRM))

#-------------------------------------------------------------------#
# Define 2 RM client connection exits: RMCLCN00 and XYZCLCN0        #
```

```
# with an abend limit of 6.                                    #
#-------------------------------------------------------------#
EXITDEF(TYPE=CLNTCONN,EXITS=(RMCLCN00,XYZCLCN0),ABLIM=6)
```

Sample SCI user exit list member of the IMS PROCLIB data set

The sample SCI user exit list member of the IMS PROCLIB data set is shown below defines:

- One SCI init/term exit routine
- Three SCI client connection exit routines

```
********************************************************************
* SCI USER EXIT LIST PROCLIB MEMBER                                *
********************************************************************

#-------------------------------------------------------------#
# Define one SCI init/term exit: SCINITRM.                    #
#-------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(SCINITRM))

#-------------------------------------------------------------#
# Define 3 SCI client connection exits: SCCLCN00, SCCLCN10,   #
# and SCCLCN20 with an abend limit of 9.                      #
#-------------------------------------------------------------#
EXITDEF(TYPE=CLNTCONN,EXITS=(SCCLCN00,SCCLCN10,SCCLCN20),ABLIM=9)
```

Sample combined user exit list member of the IMS PROCLIB data set

You can combine all the preceding user exit list members into a single shared member by using the COMP keyword on the EXITDEF statements, as shown below. The sample defines:

- BPE exit routines
- CQS user exit routines
- DBRC user exit routines
- IMS Connect user exit routines
- OM user exit routines
- RM user exit routines
- SCI user exit routines

```
********************************************************************
*  BPE EXIT DEFINITIONS                                            *
********************************************************************
EXITDEF=(TYPE=INITTERM,EXITS=(MYINIT00),COMP=BPE)
EXITDEF=(TYPE=STATS,EXITS=(HHGSTAT0),ABLIM=42,COMP=BPE)
********************************************************************
* CQS USER EXIT ROUTINE DEFINITIONS                                *
********************************************************************
EXITDEF=(TYPE=CLNTCONN,EXITS=(CLCONX00),COMP=CQS)
EXITDEF=(TYPE=INITTERM,EXITS=(MYCQSIT0,OEMCQIT0),ABLIM=8,COMP=CQS)
EXITDEF=(TYPE=OVERFLOW,EXITS=(OVERFL01,
                              OVERFL02,
                              OVERFL03,
                              OVERFL04),COMP=CQS)
EXITDEF=(TYPE=STRSTAT,EXITS=(STRSTAT0),COMP=CQS)
EXITDEF=(TYPE=STREVENT,EXITS=(STREVENT),ABLIM=0,COMP=CQS)

/*****************************************************************/
/* DBRC USER EXIT LIST PROCLIB MEMBER                           */
/*****************************************************************/
#-------------------------------------------------------------#
# DEFINE 1 RECON I/O EXIT: ZDBRCIO0                            #
# WITH AN ABEND LIMIT OF 8.                                   #
```

```
#-----------------------------------------------------------------#
EXITDEF(TYPE=RECONIO,EXITS=(ZDBRCIO0),ABLIM=8,COMP=DBRC)
#-----------------------------------------------------------------#
# DEFINE 1 DBRC SECURITY EXIT: ZDBRCSE0             #
#-----------------------------------------------------------------#
EXITDEF(TYPE=SECURITY,EXITS=(ZDBRCSE0),COMP=DBRC)

*******************************************************************
* HWS USER EXIT LIST PROCLIB MEMBER *
*******************************************************************

EXITDEF(TYPE=XMLADAP,EXITS=(HWSXMLA0),ABLIM=8,COMP=HWS)

*******************************************************************
* OM USER EXIT ROUTINE DEFINITIONS                                *
*******************************************************************
EXITDEF=(TYPE=INITTERM,EXITS=(OMINITRM),COMP=OM)
EXITDEF=(TYPE=CLNTCONN,EXITS=(OMCLCN00,OEMCLI00),ABLIM=2,COMP=OM)
EXITDEF=(TYPE=INPUT,EXITS=(MYCMI000),ABLIM=0,COMP=OM)
EXITDEF=(TYPE=OUTPUT,EXITS=(MYCMO000),ABLIM=0,COMP=OM)
EXITDEF=(TYPE=SECURITY,EXITS=(OMSEC000,
                              OMSEC001,
                              ZZZSEC00),COMP=OM)

*******************************************************************
* RM USER EXIT ROUTINE DEFINITIONS                                *
*******************************************************************
EXITDEF=(TYPE=INITTERM,EXITS=(RMINITRM),COMP=RM)
EXITDEF=(TYPE=CLNTCONN,EXITS=(RMCLCN00,XYZCLCN0),ABLIM=6,COMP=RM)

*******************************************************************
* SCI USER EXIT ROUTINE DEFINITIONS                               *
*******************************************************************
EXITDEF=(TYPE=INITTERM,EXITS=(SCINITRM),COMP=SCI)
EXITDEF=(TYPE=CLNTCONN,EXITS=(SCCLCN00,SCCLCN10,SCCLCN20),ABLIM=9,
         COMP=SCI)
```

Note: If you use a single shared user exit list member of the IMS PROCLIB data set, change the EXITMBR statements in the BPE configuration member of the IMS PROCLIB data set to point to the shared user exit list member. Here is an example of how you can change the EXITMBR statements:

```
#
# User exit list PROCLIB member specifications
#
EXITMBR=(SHREXIT0,BPE)              /* BPE user exit definitions  */
EXITMBR=(SHREXIT0,CQS)              /* CQS user exit definitions  */
EXITMBR=(SHREXIT0,OM)               /* OM user exit definitions   */
EXITMBR=(SHREXIT0,RM)               /* RM user exit definitions   */
EXITMBR=(SHREXIT0,SCI)              /* SCI user exit definitions  */
```

Related concepts:

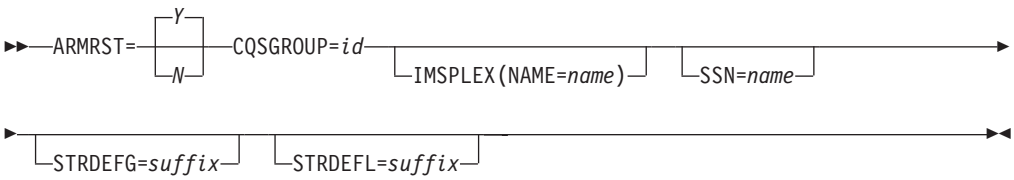
Chapter 14, “IMS Syntax Checker,” on page 283

CQSIPxxx member of the IMS PROCLIB data set

Use the CQSIPxxx member of the IMS PROCLIB data set to specify parameters that are related to initialization of the CQS address space. You can use CQS execution parameters to override certain parameters within CQSIPxxx.

Syntax

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.



Usage

The following rules apply to the format of the CQSIPxxx member:

- The execution member consists of one or more fixed-length character records. (The configuration data set can be of any logical record length (LRECL) greater than eight, but it must be of fixed-record format.)
- The rightmost eight columns of each record are ignored and you can use them for sequence numbers or any other notation. In the remaining columns, you code the keyword parameters. For example, if your record size is 80, you use columns 1 through 72 for your configuration data. You can use columns 73 through 80 for sequence numbers.
- Keywords can contain leading and trailing blanks.
- Each record can contain multiple keywords.
- Use commas or spaces to delimit keywords.
- Use an asterisk (*) or pound sign (#) in column one to begin a comment. You can include a comment anywhere within a statement by enclosing it between a slash-asterisk and an asterisk-slash pair.

  ```
  /*This is an example of a comment within a statement*/
  ```

- Values coded in this member of the IMS PROCLIB data set are case sensitive.

Parameters

ARMRST= Y | N

Specifies whether the z/OS Automatic Restart Manager (ARM) is used to restart the CQS address space after an abend. If you specify `Y` (yes), ARM restarts the CQS address space after most system failures. If you specify `N` (no), ARM does not restart the CQS address space after any system failure.

ARM does not restart the CQS address space if the CQS abends before restart is complete.

To restart the CQS when it has been canceled by z/OS, you must specify the ARMRESTART option of either the z/OS CANCEL or FORCE command.

Related reading: For information about the CANCEL and FORCE commands, see *MVS/ESA System Commands*.

This parameter can be specified as an execution parameter on the CQS procedure to override the value in CQSIPxxx.

CQSGROUP=

Specifies a 1- to 5-character identifier. CQS concatenates this identifier to the characters `CQS` to create the group name of the z/OS cross-system coupling facility CQS shared queues. You must use the same identifier for all CQS address spaces that share the same set of structures. You can also use the same identifier for the SQGROUP= parameter in the DFSSQxxx member of the IMS PROCLIB data set.

This parameter can be specified as an execution parameter on the CQS procedure to override the value in CQSIPxxx.

IMSPLEX()
Specifies the IMSplex to which CQS joins. IMSPLEX is an optional parameter. IMSPLEX does not have a default value. Only one IMSPLEX keyword can be specified. The IMSPLEX definition parameter follows:

NAME=
A 1- to 5-character user-specified identifier that is concatenated to 'CSL' to create the cross-system coupling facility (XCF) CSL IMSPLEX group name. The value specified here must match the IMSPLEX NAME= value specified in the SCI startup procedure. All OM, RM, SCI, IMS, CQS and similar address spaces must specify the same name to be part of the same IMSplex. The same identifier must also be used for the IMSPLEX= parameter in the CSLSIxxx, CSLOIxxx, CSLRIxxx, and DFSCGxxx members of the IMS PROCLIB data set.

SSN=
Specifies the name for the CQS address space. The value must be 1 - 4 alphanumeric characters. The SSN= parameter can be specified either in the CQSIPxxx PROCLIB member or as a CQS execution parameter. If the SSN= parameter is specified in both the CQSIPxxx PROCLIB member and as an execution parameter, the value specified for the execution parameter overrides the value specified in the CQSIPxxx PROCLIB member. The name specified is used to create the CQSID, which is used in CQS processing. The CQSID is the SSN followed by the characters CQS. Every CQS within the same CQSGROUP must have a unique name. Additionally, every CQS running on the same LPAR must have a unique name, regardless of what CQSGROUP it is in. Recommendation: Give unique names to all CQSs across your sysplex. This helps to avoid name conflicts if CQSs are ever moved among LPARs.

Every CQS within the same CQSGROUP must have a unique name. Additionally, every CQS running on the same LPAR must have a unique name, regardless of what CQSGROUP it is in.

Recommendation: Give unique names to all CQSs across your sysplex. This helps to avoid name conflicts if CQSs are ever moved among LPARs.

Example: If SSN=ABC, CQSID=ABCCQS.

Trailing blanks are deleted and the CQSID is padded with blanks.

Note: CQS does not register as a IBM z/OS subsystem.

STRDEFG=
Specifies a 3-character suffix for the CQS global structure definition member of the IMS PROCLIB data set, CQSSGxxx. This member contains the parameters related to the coupling facility structures that are common to all CQS address spaces that are sharing the queues. If you specify this as an execution parameter on the CQS startup procedure, it overrides the value specified in the CQSIPxxx member of the IMS PROCLIB data set. The default suffix is 000.

STRDEFL=
Specifies a 3-character suffix for the CQS local structure definition member of the IMS PROCLIB data set, CQSSLxxx. This member contains the parameters that are related to the coupling facility structures and that are unique to an individual CQS address space. If you specify this as an execution parameter on the CQS startup procedure, it overrides the value specified in the CQSIPxxx member of the IMS PROCLIB data set. The default suffix is 000.

Sample CQSIPxxx member of the IMS PROCLIB data set

A sample CQSIPxxx member of the IMS PROCLIB data set is shown here:

```
*********************************************************************
* CQS INITIALIZATION PROCLIB MEMBER                                 *
*********************************************************************

ARMRST=Y                /* ARM SHOULD RESTART CQS ON FAILURE        */
CQSGROUP=GRUP1          /* GROUP NAME (XCF GROUP = GRUP1CQS)        */
SSN=CQS1                /* CQS ADDRESS SPACE  (CQSID = CQS1CQS)     */
STRDEFG=190             /* GLOBAL STR DEFINITION MEMBER = CQSSG190  */
STRDEFL=191             /* LOCAL STR DEFINITION MEMBER  = CQSSL191  */
IMSPLEX(NAME=PLEX1)     /* IMSPLEX NAME(CSLPLEX1)                   */
```

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

Related tasks:

"Preparing to start the CQS address space" on page 202

CQSSLxxx member of the IMS PROCLIB data set

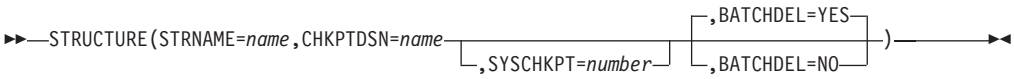
Use the CQSSLxxx member of the IMS PROCLIB data set to define local CQS parameters that are related to one or more coupling facility structures.

Each CQS should point to a different CQSSLxxx member. CQS connects to each defined structure in the member. The structures defined in the CQSSLxxx member must also be defined in the CQSSGxxx member of the IMS PROCLIB data set.

Important: The CQSSLxxx member of the IMS PROCLIB data set applies to queue structures only, not resource structures. If you do not define queue structures, you do not need to define the CQSSLxxx member of the IMS PROCLIB data set.

Syntax

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.



Restrictions

The CQSSLxxx member of the IMS PROCLIB data set applies to queue structures only, not resource structures. If you do not define queue structures, you do not need to define the CQSSLxxx member of the IMS PROCLIB data set.

Usage

The following rules apply to the format of the CQSSLxxx member:

- The execution member consists of one or more fixed-length character records. (The configuration data set can be of any LRECL greater than eight, but it must be of fixed-record format.)
- The rightmost eight columns of each record are ignored and can be used for sequence numbers or any other notation. In the remaining columns, you code

the keyword parameters. For example, if your record size is 80, you use columns 1 through 72 for your configuration data. You can use columns 73 through 80 for sequence numbers.

- Keywords can contain leading and trailing blanks.
- Each record can contain multiple keywords.
- Commas or spaces delimit keywords.
- A comment begins with an asterisk (*) or pound sign (#) in column one. You can include a comment anywhere within a statement by enclosing it between a slash-asterisk and an asterisk-slash pair.

  ```
  /*This is an example of a comment within a statement*/
  ```

- Values coded in this member of the IMS PROCLIB data set are case sensitive.

If the STRUCTURE statement for an EMHQ structure is deleted from the CQSSLxxx member of the IMS PROCLIB data set, resources for the EMHQ structure and its associated CQS data sets are not allocated.

Parameters

Use the following keyword parameters to define a structure to CQS.

STRUCTURE()
The statement with which you define the structure definition parameters for the CQS. The structure definition parameters must be enclosed within parentheses. The STRUCTURE keyword must precede the left parenthesis.

Example: STRUCTURE (STRNAME=*strname*, CHKPTDSN=*chkptdsn*, ...)

STRNAME=
The required 1- to 16-character name of the primary coupling facility structure to which CQS connects.

The installation must have defined the structure in the coupling facility resource management (CFRM) administrative policy. The structure name must follow the naming rules of the CFRM. If the name has fewer than 16 characters, CQS pads the name with blanks. The valid characters are A-Z, 0-9, and the following special characters:

```
$ @ # _
```

Names must be uppercase and start with an alphabetic character.

Restriction: Avoid using names that IBM uses for its structures. Do not begin structure names with the letters A-I, or with the character string SYS. If you do name a structure beginning with the letters A-I or SYS, your name might conflict with an existing or future IBM-defined structure name.

CHKPTDSN=
The required 1- to 44-character data set name of the cataloged VSAM data set that is used for the checkpoint data set for the indicated structure. The data set is dynamically allocated by CQS during CQS initialization. Each structure defined in `CQSSLxxx` must have a unique `CHKPTDSN`.

SYSCHKPT=
Specifies the number of log records CQS writes between system checkpoints. This value can be from 200 to 2 147 483 647. Each CQS address space that is connected to a queue structure can specify a different system checkpoint log record count. This value is not shared between CQS address spaces.

This parameter has no default. If you do not specify a value, automatic system checkpoints are only taken during restart, normal shut down, and after a structure checkpoint.

Sample CQSSLxxx member of the IMS PROCLIB data set

The following is a sample CQSSLxxx member of the IMS PROCLIB data set.

```
************************************************************************
* LOCAL STRUCTURE DEFINITION PROCLIB MEMBER                            *
************************************************************************

*----------------------------------------------*
* DEFINITION FOR IMS MESSAGE QUEUE STRUCTURE     *
*----------------------------------------------*
STRUCTURE (
 STRNAME=QMSGIMS01,  CHKPTDSN=CQSA.QMSG.IMS01.CHKPT, SYSCHKPT=50000)

*----------------------------------------------*
* DEFINITION FOR IMS EMH QUEUE STRUCTURE         *
*----------------------------------------------*
STRUCTURE (
 STRNAME=QEMHIMS01,  CHKPTDSN=CQSA.QEMH.IMS01.CHKPT, SYSCHKPT=50000)
```

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

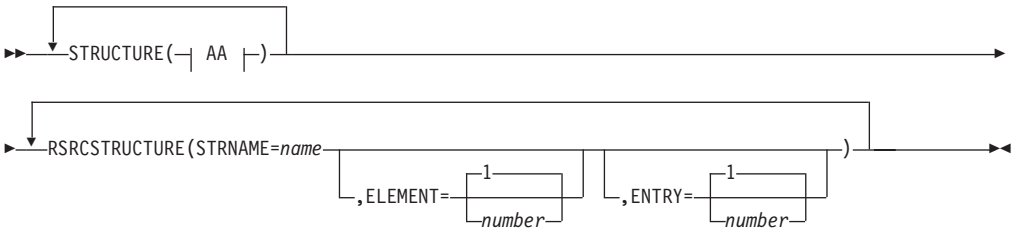
CQSSGxxx member of the IMS PROCLIB data set

Use the CQSSGxxx member of the IMS PROCLIB data set to define global CQS parameters that are related to one or more coupling facility structures. These parameters are shared by all CQS address spaces that share the structures.

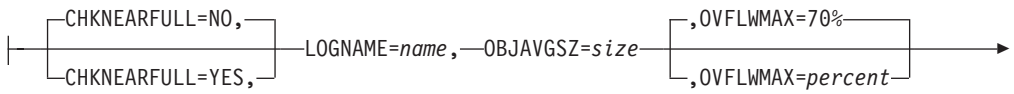
A particular CQS can support queue structures, resource structures, or a combination of both queue structures and resource structures. Each CQS sharing a structure must point to a CQSSGxxx member containing identical structure definition parameters.

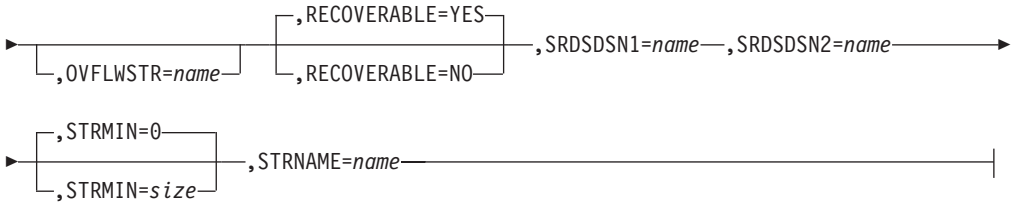
Syntax

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.



AA:





Usage

Point all CQSs to the same CQSSGxxx member to avoid parameter mismatches. CQS connects to each structure that is defined in the member. The structures defined in the CQSSGxxx member must also be defined in the CQSSLxxx member of the IMS PROCLIB data set.

If you are using queue structures, define an overflow structure name OVFLWSTR= if there is a possibility that you use an overflow structure. If you have to add an overflow structure later, the structure and all CQSs must be cold started.

If the STRUCTURE statement for an EMHQ structure is deleted from the CQSSGxxx member of the IMS PROCLIB data set, resources for the EMHQ structure are not allocated. These resources include the EMHQ structure's associated overflow structure, structure recovery data sets, and CQS log.

If you use RECOVERABLE=NO, consider that:

- RECOVERABLE=NO structures are truly not recoverable. If the structure fails, the data in it is lost. Before using RECOVERABLE=NO, carefully consider whether you can tolerate the loss of data in your CQS shared queues structure if the structure or the coupling facility hardware fails. For example, a nonrecoverable queue might be acceptable for a test or development system; it might be unacceptable for a production system. Consider using a duplexed structure for a RECOVERABLE=NO structure in a production environment to provide redundancy.
- The failure of a nonrecoverable structure causes all CQSs that are connected to that structure to abend (ABENDU0373). If the structure was in overflow mode and the primary structure failed, the overflow structure is deleted. If the structure was in overflow and the overflow structure failed, the primary structure is cold started, and the overflow structure is not reallocated. All objects in the structures are lost. When the CQSs are restarted, they connect to an empty structure, resynchronize with their clients, and proceed with an empty queue. This processing is required to ensure that work that was in-flight when the structure failed is correctly resolved during the resynchronization at CQS restart.
- Changing between RECOVERABLE=NO and RECOVERABLE=YES requires a cold start of the CQS queue structure. You must delete the structure and the z/OS log stream, and scratch and reallocate the SRDS and system checkpoint data sets.
- Nonrecoverable structures require that SRDS data sets (SRDSDSN1=, SRDSDSN2=) be specified and allocated. Although the SRDS data sets are not used for recovery, client-initiated structure checkpoints (including the CQSCHKPT macro, the /CQCHKPT IMS command, and CQS normal shut down checkpoints requested through the IMS /CQSET command) are still allowed. These checkpoints can be useful for diagnostics, as an alternative to taking a memory dump of the structure. Internally generated CQS structure checkpoints are skipped for nonrecoverable structures.

- RECOVERABLE=NO structures require that a z/OS log stream name be specified on the LOGNAME= parameter; however, the log stream does not have to be defined, and it is neither opened nor written to.
- When you start CQS with a RECOVERABLE=NO structure, message CQS0370I is issued, indicating that the structure is not recoverable.
- When you restart a CQS with a nonrecoverable structure, all restarts are cold starts (of the CQS, not the structure). When CQS connects to a structure that is not empty, message CQS0370I is issued, indicating that CQS is forcing a cold start for the structure.

The following rules apply to the format of the CQSSGxxx member:

- The execution member consists of one or more fixed-length character records. (The configuration data set can be of any LRECL greater than eight, but it must be of fixed-record format.)
- The rightmost eight columns of each record are ignored and can be used for sequence numbers or any other notation. In the remaining columns, you code the keyword parameters. For example, if your record size is 80, you use columns 1 through 72 for your configuration data. You can use columns 73 through 80 for sequence numbers.
- Keywords can contain leading and trailing blanks.
- Each record can contain multiple keywords.
- Commas or spaces delimit keywords.
- A comment begins with an asterisk (*) or pound sign (#) in column one. You can include a comment anywhere within a statement by enclosing it between a slash-asterisk and an asterisk-slash pair.

  ```
  /*This is an example of a comment within a statement*/
  ```
- Values coded in this member of the IMS PROCLIB data set are case sensitive.

Parameters

Use the following parameters to define a structure to CQS. At least one STRUCTURE or RSRCSTRUCTURE definition is required.

STRUCTURE ()

Defines a queue structure to CQS. This keyword can be repeated. Keyword parameters must be enclosed within parentheses.

Example: STRUCTURE (STRNAME=*strname*, SRDSDSN1=*srsdsn1*, ...)

The following keyword parameters are available to the STRUCTURE definition:

CHKNEARFULL

Optional parameter specifying whether (YES) or not (NO) CQS automatically takes a structure checkpoint when the log stream near-full conditions are detected. The default is NO.

The log stream near-full conditions happen in one of the following situations:

- Log stream usage reaches the 1/3 point between the high offload threshold percentage and 100% full (0.33% of the delta).
- Log stream usage reaches the 2/3 point between the high offload threshold percentage and 100% full (0.67% of the delta).

If you specify CHKNEARFULL=YES, CQS will take additional structure checkpoints when the z/OS log stream structure usage crosses the near-full

thresholds. During the CQS message queue structure checkpoint process, CQS issues a log record delete request to the z/OS logger to relieve the log stream full condition. Using CHKNEARFULL=YES can help avoid log stream structure full situations; however, it might also result in additional CQS structure checkpoints.

If you specify CHKNEARFULL=NO, CQS will not take any additional structure checkpoints when log stream usage crosses the near-full thresholds. Specify CHKNEARFULL=NO or omit this parameter if you do not want CQS to take additional structure checkpoints when the z/OS log stream structure gets close to full.

LOGNAME=

Is the required 1- to 26-character name of the z/OS log stream that CQS uses to record all information related to the structure. The installation must have previously defined this name to the z/OS system logger.

All CQS address spaces that connect to a queue structure must use the same value for this parameter. The value specified by the CQS that initially allocates the structure is the value that is used for the life of the structure.

OBJAVGSZ=

Specifies the average size of a data object that is written to a queue on this structure. This value can range from 128 bytes to 61312 bytes or from 1K to 59K. The following list defines some IMS object sizes:

IMS client
The object size is the size of the IMS message plus some control information.

IMS queue manager messages
If the user message and the message queue prefix both fit completely into one queue buffer, the object size is the sum of the user message and the message queue prefix. If both parts do not fit into one queue buffer, the object size is the size of the portion of the message and the message queue prefix that do fit into one queue buffer. The size of an IMS message queue buffer is specified to the IMS control region by the `QBUFSZ` execution parameter.

IMS expedited message handler messages
The object size is the size of the user message plus 240 bytes (the size of the EMHB global header).

Each object in a CQS queue structure is stored using one entry and one or more 512-byte elements. The entry contains control data about the object on the queue. The elements contain the message data (IMS and CQS prefixes, and client data). The OBJAVGSZ parameter is used by CQS to determine the ratio of the number of entries to the number of elements within the coupling facility structure. OBJAVGSZ applies only to the initial allocation of the structure; it cannot be used to change the entry-to-element ratio of a structure that is already allocated.

For a structure defined with ALLOWAUTOALT(NO) in the CFRM policy, you must cold start the structure to change the entry-to-element ratio. This is because once CQS connects to a structure, it preserves the actual number of entries and elements obtained. If a structure rebuild occurs, CQS uses the actual entry-to-element ratio from these saved values when it connects to the rebuild structure, and not the values from OBJAVGSZ. This is done to ensure that the rebuild structure has a similar number of entries and elements as the old structure, to reduce the chance of a rebuild failure.

For a structure defined with ALLOWAUTOALT(YES), you still must cold start the structure to change the entry-to-element ratio to a specific value (for example, to set the ratio to a value you have computed based on your message sizes). However, z/OS dynamically changes the entry-to-element ratio if it is not correct for the size of the objects in the structure.

Recommendation: Define your queue structures with the ALLOWAUTOALT(YES) parameter, to allow z/OS to dynamically adjust the entry-to-element ratio as necessary.

Recommendation: Specify the OBJAVGSZ to be the average of the sizes of all the objects passed to CQS by a CQSPUT request. CQS adds its own prefix containing control information to every object placed on the structure. CQS adds the length of its prefix to the OBJAVGSZ value that you specify to get the true average object size. Therefore, OBJAVGSZ should reflect only the average size of the objects as they are passed to CQS, not the average size of the object on the coupling facility.

If the OBJAVGSZ is too small, too much space in the structure is allocated for control information. The structure becomes full when all the space for data is used up, even though space for control information is still available.

If the OBJAVGSZ is too large, too much space in the structure is allocated for data. The structure becomes full when all the control space is used up, even though space for data is still available.

Example: Five objects are put on the structure by a CQSPUT request. The sizes of the objects are:

object 1
134 bytes

object 2
1066 bytes

object 3
3200 bytes

object 4
172 bytes

object 5
345 bytes

The average object size is calculated to be 983 bytes.

```
(134 + 1066 + 3200 + 172 + 345)/5 = 983
```

OVFLWMAX=
Specifies the maximum threshold percentage for overflow processing. This value indicates the percentage of the structure that must be in use before CQS goes into overflow mode. This value can be from 50 to 100. For example, if OVFLWMAX=75, the structure is put into overflow mode when the structure usage reaches 75% of the structure size. The default is 70%.

The value specified by the CQS that initially allocates the structure is used for the life of the structure.

Recommendation: If your structure is defined with ALLOWAUTOALT(YES) in the CFRM policy, you should also code the FULLTHRESHOLD parameter in the policy, and specify a value that is at least five percent lower than the value specified on OVFLWMAX. This is to

allow the z/OS auto alter function to adjust the entry-to-element ratio and the structure size before CQS enters overflow processing.

OVFLWSTR=

Is the 1- to 16-character name of the optional coupling facility structure to which CQS connects for structure overflow processing. The name must follow the same naming convention as the structure name specified by the STRNAME= parameter. When CQS is processing in overflow mode, selected queues are written to this structure instead of to the primary structure.

If an overflow structure is not specified and an overflow condition is detected, CQS rejects requests to add data objects to those queues that were selected for overflow.

If an overflow structure is specified, CQS connects to the overflow structure during CQS initialization and then again during phase one of overflow threshold processing. If CQS detects that the overflow structure size is less than 30% of the primary structure size, the overflow structure is considered to be too small and CQS issues the CQS0268W message. CQS is allowed to initialize even though the overflow structure is too small. CQS disconnects from and deletes the overflow structure at the end of CQS initialization.

CQS does not attempt to connect to the overflow structure again until the overflow threshold is reached. If at that time the overflow structure size is still less than 30% of the primary structure size, CQS again issues the CQS0268W message. CQS goes into overflow mode, but the overflow structure is not used. Requests to add data objects to those queues that were selected for overflow are rejected.

Recommendation: Define the size of the overflow structure in the CFRM policy to be at least X% of the primary structure size, where X is the value specified for the OVFLWMAX= parameter. The value specified for the OVFLWMAX= parameter indicates the percentage of the primary structure that must be in use before CQS goes into overflow mode, the overflow threshold. For example, if the overflow threshold was defined with the OVFLWMAX= parameter to be 75% of the primary structure size, the size of the overflow structure should be at least 75% of the primary structure size. If a value is not specified for the OVFLWMAX= parameter, the overflow threshold defaults to 70% and the size of the overflow structure should be at least 70% of the primary structure size.

An overflow structure name can be defined only when the structure is cold started. Once structures have been allocated, an overflow structure cannot be added unless the structure and all CQSs are cold started.

All CQS address spaces that connect to a queue structure must use the same value for this parameter. The value specified by the CQS that initially allocates the structure is used for the life of the structure.

RECOVERABLE=

Optional parameter specifying whether (YES) or not (NO) the queue structure is recoverable. The default is YES.

By default, CQS queue structures are defined as recoverable. When data on a queue structure is changes, CQS records the change in a log record in the z/OS system log stream that is associated with the structure. Periodically, the contents of the queue structure are written to a structure recovery data set (SRDS). If a queue structure fails, CQS reallocates the structure and

recovers it by first restoring the data from the last SRDS, and then applying any subsequent changes using the data from the z/OS log stream. This is the default for CQS queue structures (RECOVERABLE=YES).

You can optionally define a CQS queue structure as nonrecoverable (RECOVERABLE=NO). In this case, CQS does not write log records to a z/OS log stream. CQS does write structure checkpoint to an SRDS if one is requested; however, the checkpoints are not used for recovery. Using a nonrecoverable structure saves the overhead of using the z/OS logger. If the structure fails, however, all objects on the structure are lost.

SRDSDSN1=

Is a required 1- to 44-character data set name of the cataloged VSAM data set that is used for the first structure recovery data set. The data set name is used to dynamically allocate the data set when a structure checkpoint is requested. For a given structure checkpoint request, CQS uses either structure recovery data set 1 or data set 2. CQS alternates between the two data sets for structure checkpoint processing.

All CQS address spaces that connect to a queue structure must use the same value for this parameter. The value specified by the CQS that initially allocates the structure is the value that is used for the life of the structure.

SRDSDSN2=

Is a required 1- to 44-character data set name for the cataloged VSAM data set that is used for the second structure recovery data set. The data set name is used to dynamically allocate the data set when a structure checkpoint is requested. For a given structure checkpoint request, CQS uses either structure recovery data set 1 or data set 2. CQS alternates between the two data sets for structure checkpoint processing.

All CQS address spaces that connect to a queue structure must use the same value for this parameter. The value specified by the CQS that initially allocates the structure is the value that is used for the life of the structure.

STRMIN=

Specifies the value for the minimum primary structure size to which CQS can connect. This value is specified in units of 4 KB blocks and can be any value from 0 to the maximum structure size of 524288 (a 2-GB structure).

The default value is 0, indicating that CQS accepts the size as allocated by the coupling facility. If the coupling facility is constrained, the structure can be allocated to something smaller than that defined by the CFRM policy. Depending on the size, the structure might overflow sooner than expected.

Recommendation: Specify a value for STRMIN= that is less than the structure size that is defined in the policy.

The value specified by the CQS that initially allocates the structure is used for the life of the structure.

When the first CQS connects to an empty structure, that structure is allocated on the coupling facility. After it is allocated, the structure remains on the coupling facility regardless of whether a CQS is connected to it.

If, during connection to a structure, CQS determines that the size of the structure is smaller than the minimum size and the structure is empty, CQS terminates. In this case, the installation needs to redefine the use of the coupling facility to ensure that the required size can be allocated. If CQS connects to a structure that is smaller than the minimum size, but the

structure contains data objects, CQS does not terminate. CQS attempts to use the smaller structure because it already contains data. In this case, CQS issues a message that allows an operator to initiate a structure rebuild in order to increase the structure size.

STRNAME=
The required 1- to 16-character name of the primary coupling facility structure to which CQS connects.

The installation must have defined the structure name in the CFRM administrative policy. The structure name must follow the naming rules of the CFRM. For names with fewer than 16 characters, CQS pads the name with blanks. The valid characters are A-Z, 0-9, and the following special characters:

```
$ @ # _
```

Names must be uppercase and start with an alphabetic character.

Restriction: Avoid using names that IBM uses for its structures. Do not begin structure names with the letters A-I, or with the character string SYS. If you do name a structure beginning with the letters A-I or SYS, your name might conflict with an existing or future IBM-defined structure name.

RSRCSTRUCTURE ()
Defines a resource structure to CQS. An IMSplex can define only one resource structure; name uniqueness is within one resource structure. Keyword parameters must be enclosed within parentheses.

If you use shared queues, and you want IMS to manage serial programs across the IMSplex, you must define a resource structure. You must also identify the resource structure on the CSLRIxxx member of the IMS PROCLIB data set RSRCSTRUCTURE parameter.

Example: RSRCSTRUCTURE (STRNAME=*strname*)

The following keyword parameters are available:

STRNAME=
The required 1- to 16-character name of the coupling facility list structure to which CQS connects. This parameter defines the name of the resource structure used by RM to keep IMS resource information.

The installation must have defined the structure name in the CFRM administrative policy. The structure name must follow the naming rules of the CFRM. For names with fewer than 16 characters, CQS pads the name with blanks. The valid characters are A-Z, 0-9, and the characters $, &, # and _. Names must be uppercase and start with an alphabetic character.

Restriction: Avoid using names IBM uses for its structures. Do not begin structure names with the letters A-I, or with the character string SYS.

ELEMENT=
Specifies the data element value of the entry-to-element ratio for resources that are used to allocate the resource structure. The number of data elements depends upon the resource type. To calculate the number of data elements needed, use the formulas in the topic “Calculating resource structure entry and element values”. Valid values are from 1 to 65,535. The default is 1.

ENTRY=
Specifies the entry value of the entry-to-element ratio for resources that are

used to allocate the resource structure. Each resource is stored on the resource structure using one entry and either zero, one, or more elements. The number of entries is equal to the number of resources. To calculate the number of entries needed, use the formulas in the topic "Calculating resource structure entry and element values". Valid values are from 1 to 65,535. The default is 1.

For a structure defined with ALLOWAUTOALT(NO) in the CFRM policy, you must cold start the structure to change the entry-to-element ratio. This is because once CQS connects to a structure, it preserves the actual number of entries and elements obtained. If a structure rebuild occurs, CQS uses the actual entry-to-element ratio from these saved values when it connects to the rebuild structure, and not the values from ELEMENT or ENTRY. This is done to ensure that the rebuild structure has a similar number of entries and elements as the old structure, to reduce the chance of a rebuild failure.

For a structure defined with ALLOWAUTOALT(YES), you still must cold start the structure to change the entry-to-element ratio to a specific value (for example, to set the ratio to a value you have computed based on your message sizes). However, z/OS dynamically changes the entry-to-element ratio if it is not correct for the size of the objects in the structure.

Recommendation: Define your resource structure with the ALLOWAUTOALT(YES) parameter, to allow z/OS to dynamically adjust the entry-to-element ratio as necessary.

Sample CQSSGxxx member of the IMS PROCLIB data set

A sample CQSSGxxx member of the IMS PROCLIB data set that defines both message queue and resource structures is shown here:

```
************************************************************
* GLOBAL STRUCTURE DEFINITION PROCLIB MEMBER
************************************************************

*-------------------------------------------------*
* DEFINITION FOR IMS MESSAGE QUEUE STRUCTURES     *
*-------------------------------------------------*
STRUCTURE (
  STRNAME=QMSGIMS01,
  OVFLWSTR=QMSGIMS01OFLW,
  SRDSDSN1=CQS.QMSG.IMS01.SRDS1
  SRDSDSN2=CQS.QMSG.IMS01.SRDS2,
  LOGNAME=SYSLOG.QMSG01.LOG,
  OBJAVGSZ=1024,
  CHKNEARFULL=YES)

*-------------------------------------------------*
* DEFINITION FOR IMS EMH QUEUE STRUCTURES         *
*-------------------------------------------------*
STRUCTURE (
  STRNAME=QEMHIMS01,
  OVFLWSTR=QEMHIMS01OFLW,
  SRDSDSN1=CQS.QEMH.IMS01.SRDS1,
  SRDSDSN2=CQS.QEMH.IMS01.SRDS2,
  LOGNAME=SYSLOG.QEMH01.LOG,
  OBJAVGSZ=1024,
  CHKNEARFULL=YES)
```

```
*-----------------------------------------------*
* DEFINITION FOR IMS RESOURCE STRUCTURE         *
*-----------------------------------------------*
RSRCSTRUCTURE (STRNAME=QRSCIMS01)
```

Related concepts:

Using structure alter for CQS (System Administration)

Chapter 14, "IMS Syntax Checker," on page 283

CSLDCxxx member of the IMS PROCLIB data set

Use the Open Database Manager (ODBM) configuration member of the IMS PROCLIB data set (CSLDCxxx) to define the data store connections between one or more ODBM instances and one or more IMS systems.

Each IMS system that ODBM can connect to is defined as a *data store* in the CSLDCxxx member of the IMS PROCLIB data set. Each data store has one or more alias names and several modifiable connection attributes.

Each instance of ODBM in an IMSplex can use a separate, local CSLDCxxx member of the IMS PROCLIB data set, or all the instances of ODBM in an IMSplex can share one CSLDCxxx member. The shared CSLDCxxx member of the IMS PROCLIB data set must be stored in a data set on shared DASD and concatenated to the local PROCLIB data set member of each ODBM instance that is sharing it. Sharing the CSLDCxxx member simplifies the management of data store connections across multiple instances of ODBM.

The parameters in the CSLDCxxx member of the IMS PROCLIB data set are organized into sections: a global section and a local section. The parameters in the global section of CSLDCxxx apply to all the data stores defined in the CSLDCxxx member of the IMS PROCLIB data set. The parameters in the local section of the CSLDCxxx member of the IMS PROCLIB data set apply only to specific data store connections of specific instances of ODBM. The parameters in the local section override the same parameters specified in the global section.

Both the global and local sections of the CSLDCxxx member of the IMS PROCLIB data set are identified by a predefined section header. These section headers are required in the CSLDCxxx member of the IMS PROCLIB data set even if no parameters are specified in the global section, as might be the case if all the default values are used.

The section header for the global section is: `<SECTION=GLOBAL_DATASTORE_CONFIGURATION>`. The section header for the local section is: `<SECTION=LOCAL_DATASTORE_CONFIGURATION>`.

The global data store configuration section defines attributes that are used by all the data store connections defined on this CSLDCxxx member of the IMS PROCLIB data set. You can specify the following attributes globally:

- IDRETRY: the number of times ODBM tries to connect to an IMS data store if the first attempt is unsuccessful
- TIMER: the amount of time to wait between connection attempts
- MAXTHRDS: the number of concurrent active threads that an IMS data store can have
- FPBUF: the number of Fast Path DEDB buffers to allocate and fix per thread
- FPBOF: the number of Fast Path DEDB overflow buffers to allocate per thread

- CNBA: the total number of Fast Path normal buffer allocation (NBA) buffers that an instance of ODBM can use for all its data store connections
- SOD: the system output class of a SNAP dump that is produced by the ODBM address space.

The local data store configuration section defines attributes for specific data store connections of specific instances of ODBM and overrides the same values specified in the global section:

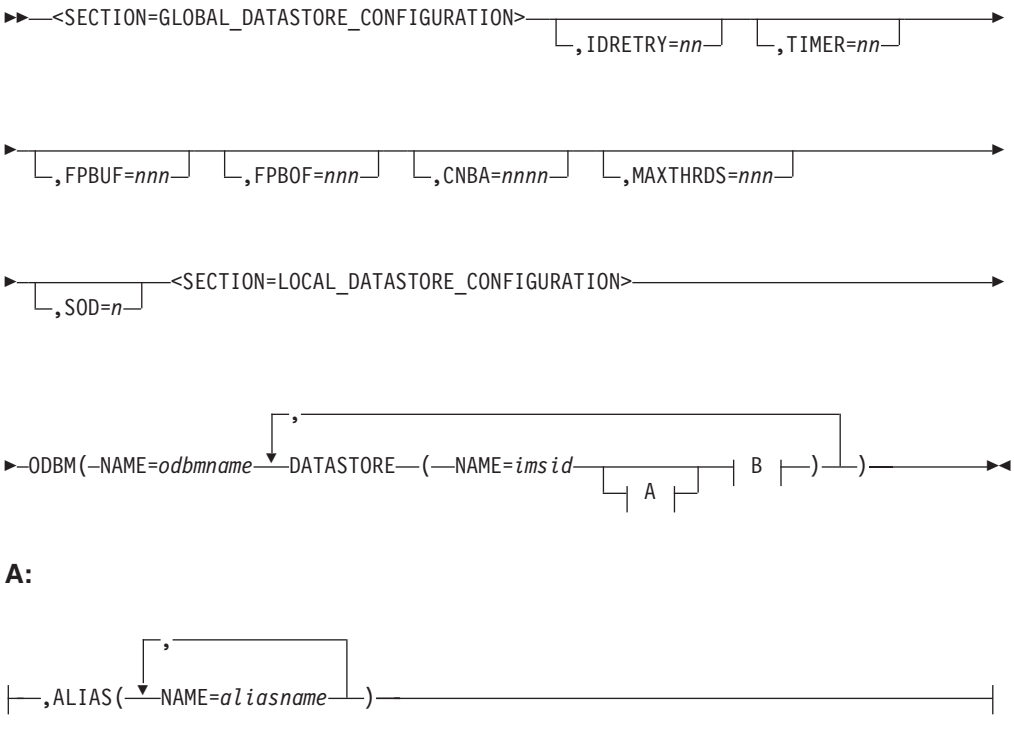
- ALIAS(NAME=): the IMS alias names
- MAXTHRDS: the number of concurrent active threads that an IMS data store can have
- FPBUF: the number of Fast Path DEDB buffers to allocate and fix per thread
- FPBOF: the number of Fast Path DEDB overflow buffers to allocate per thread
- CNBA: the total number of Fast Path NBA buffers that an instance of ODBM can use for all its data store connections
- SOD: the system output class of a SNAP dump that is produced by the ODBM address space.

An example of the CSLDCxxx member of the IMS PROCLIB data set, is shown in "Examples" on page 632.

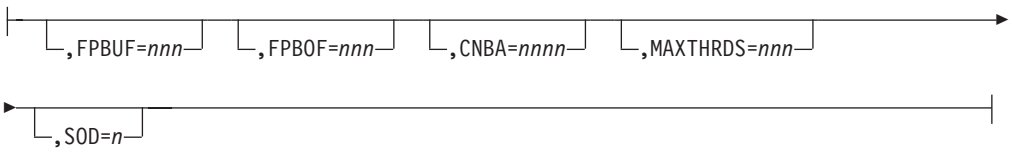
You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.

Syntax

The following diagram shows the syntax for the parameters specified in the CSLDCxxx member of the IMS PROCLIB data set.



B:



Usage

A CSLDCxxx member of the IMS PROCLIB data set consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and they can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a "*" or "#" in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between " /* "and "*/", for example, /* PROCLIB comments */. Values coded in this member of the IMS PROCLIB data set are case sensitive. In general, you should use uppercase for all parameters.

Parameters

The parameters on the CSLDCxxx member of the IMS PROCLIB data set include those that can be specified only in the global section, those that can be specified only in the local section, and those that can be specified in either the global section, the local section, or both.

Parameters that can be specified as a global default and as an instance-specific override of the default value

The following parameters can be specified in both the global and the local sections of CSLDCxxx. When specified in the global section, these parameters define the default value for the parameter on all data store connections defined on the CSLDCxxx member of the IMS PROCLIB data set. When specified in the local section, these parameters apply only to the specified data store connection of a specified instance of ODBM and override the global specifications.

MAXTHRDS=*nnn*
Specifies the maximum number of concurrent active threads for an individual IMS data store. Valid values for MAXTHRDS are from 1 to 999. The default value for MAXTHRDS is 1.

Values for MAXTHRDS specified in the local section of CSLDCxxx override the value of MAXTHRDS from the global section.

FPBUF=*nnn*
Specifies the number of Fast Path DEDB buffers that are allocated and fixed per thread. Valid values for FPBUF are from 0 to 999. The default value for FPBUF is 0.

Values for FPBUF specified in the local section of CSLDCxxx override the value of FPBUF from the global section.

FPBOF=*nnn*
Specifies the number of Fast Path DEDB overflow buffers to be allocated per thread. Valid values for FPBOF are from 0 to 999. The default value for FPBOF is 0.

Values for FPBOF specified in the local section of CSLDCxxx override the value of FPBOF from the global section.

CNBA=*nnnn*
Specifies the total number of Fast Path NBA buffers for ODBM use. Valid values for CNBA are from 0 to 9999. The default value for CNBA is 0.

Values for CNBA specified in the local section of CSLDCxxx override the value of CNBA from the global section.

You can determine a starting value for CNBA by using the following formula: (FPBUF+FPBOF) x MAXTHRDS = CNBA. If needed, adjust the value of CNBA to meet the performance and storage requirements of your installation.

SOD=*n*
Specifies the system output class of SNAP dumps that can be produced by the ODBM address space. Valid values for SOD are any single alphanumeric character. The default value is A.

Values for SOD specified in the local section of CSLDCxxx override the value of SOD from the global section.

Parameters that can be specified only as global defaults

The following parameters can be specified only in the global section of the CSLDCxxx member of the IMS PROCLIB data set. Their values apply to all data store statements included in the CSLDCxxx member of the IMS PROCLIB data set.

<SECTION=GLOBAL_DATASTORE_CONFIGURATION>
Required header for the global section of the CSLDCxxx member of the IMS PROCLIB data set.

IDRETRY=*nnn*
Specifies the number of times that ODBM attempts to connect to an IMS data store if the first connection attempt fails. Valid values for IDRETRY are from 0 to 255. The default value is 0. IDRETRY is an optional parameter.

TIMER=*nn*
Specifies the number of seconds that ODBM waits between connection attempts if the first attempt to connect to an IMS data store by ODBM fails and the value of IDRETRY is greater than 0. Valid values for TIMER are from 1 to 99. The default value for TIMER is 60.

Parameters that can be specified only as instance-specific values

<SECTION=LOCAL_DATASTORE_CONFIGURATION>
Required header for the local section of the CSLDCxxx member of the IMS PROCLIB data set.

ODBM( )

Contains the attribute specifications for the data store connections of a specific instance of ODBM.

NAME=*odbmname*
Identifies the ODBM instance to which the included data store statements apply.

DATASTORE( )
Defines an IMS data store that can be connected to by this ODBM. You must include at least one DATASTORE statement in the CSLDCxxx member of the IMS PROCLIB data set.

NAME=*imsid*
Specifies the 1- to 4-character alphanumeric IMSID of an IMS data store that ODBM can connect to. This parameter is required for each DATASTORE parameter. There is no default for this parameter.

ALIAS( )
Specifies one or more ALIAS names for the data store defined by the DATASTORE statement. Application programs must use an alias name to access IMS data stores and do not need to know the actual IMSID of the IMS data store. If an application program wants to access an IMS data store using the DATASTORE name, that name must be specified as an ALIAS or be an internal alias.

For a DATASTORE statement, an associated ALIAS parameter is not required. If no ALIAS parameter is specified, ODBM generates an internal alias, giving it the data store name specified in the DATASTORE NAME parameter. This allows application programs to use the data store name when making calls to that data store.

These internally generated aliases cannot be started or stopped by using the following commands:

- UPD ODBM START(CONNECTION) ALIAS(*internal_alias_name*)
- UPD ODBM STOP(CONNECTION) ALIAS(*internal_alias_name*)

Starting or stopping a connection for a data store with an internally generated alias can only be achieved by starting or stopping the data store itself.

When displaying ODBM configuration and status, none of the internally generated aliases appear in any output resulting from a QRY ODBM command.

NAME=*aliasname*
Specifies a 4-character alphanumeric alias name.

You can specify multiple alias names on a DATASTORE statement, which allows different application programs to access the same IMS data store by using different alias names. For each instance of ODBM, many alias names may be associated with a data store, and an alias name may be associated with more than one data store.

For more information about defining Fast Path buffers, see the information about designing and tuning Fast Path databases in *IMS Version 11 Database Administration*.

Examples

An example of the CSLDCxxx members of the IMS PROCLIB data set is shown below.

```
*********************************************************************
* CSLDC000 ODBM CSL PROCLIB MEMBER                                  *
*********************************************************************

<SECTION=GLOBAL_DATASTORE_CONFIGURATION>
IDRETRY=5                    /* Retry connection 5 times before quit    */
MAXTHRDS=10                  /* 10 threads max to any IMS Datastore     */
```

```
TIMER=30                      /* 30 seconds between ID retry attempts    */
FPBUF=10                      /* 10 DEDB buffers per thread              */
FPBOF=10                      /* 10 Overflow buffers per thread          */
CNBA=200                      /* (FPBUF+FPBOF)*MAXTHRDS <= CNBA          */
/*******************************************************************/
/* Define DATASTORE properties for ODBM01                          */
/*******************************************************************/

<SECTION=LOCAL_DATASTORE_CONFIGURATION>
ODBM(NAME=ODBM01,                    /* Define parms for ODBM01    */
  DATASTORE(NAME=IMS1,               /* IMSID on LPAR A            */
    ALIAS(NAME=IO1A,NAME=IO1B),      /* Names for APPL sets 1 & 2  */
    FPBUF=0,FPBOF=0,CNBA=0           /* No FastPath on this IMS    */
      )
  DATASTORE(NAME=IMS2,               /* IMSID on LPAR A            */
    ALIAS(NAME=IO2A,NAME=IO2B),      /* Names for DEDB apps        */
    FPBUF=50,FPBOF=50,CNBA=500,      /* FastPath on this IMS       */
    MAXTHRDS=5                       /* Throttle down threads      */
      )
    )
/*******************************************************************/
/* Define DATASTORE properties for ODBM02                          */
/*******************************************************************/
ODBM(NAME=ODBM02,                   /* Define parms for ODBM02    */
  DATASTORE(NAME=IMS3,               /* IMSID on LPAR B            */
    ALIAS(NAME=IO3A,NAME=IO3B)       /* Names for APPL sets 3 & 4  */
      )                              /* Take globals for the rest  */
    )
```

Related concepts:

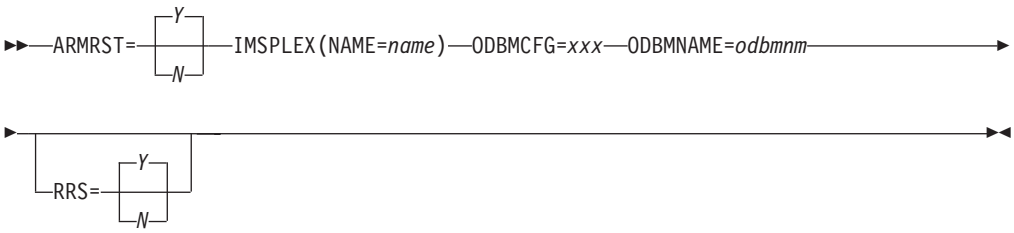
Chapter 14, “IMS Syntax Checker,” on page 283

CSLDIxxx member of the IMS PROCLIB data set

Use the ODBM initialization member of the IMS PROCLIB data set (CSLDIxxx) to specify parameters that initialize the ODBM address space.

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.

Syntax



Usage

A CSLDIxxx member consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a “*” or “#” in column 1 are comment lines

and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between " /* " and "*/", for example, `/* PROCLIB comments */`. Values coded in this member of the IMS PROCLIB data set are case sensitive. In general, you should use uppercase for all parameters.

The ARMRST, ODBMCFG, ODBMNAME, and RRS parameters can also be specified as execution parameters in the CSLODBM procedure. When they are specified as execution parameters, the values of the execution parameters override the values of the parameters specified in the CSLDIxxx member of the IMS PROCLIB data set. For more information about the ODBM execution parameters, see "CSLODBM procedure" on page 515.

Note that, in addition to determining whether ODBM uses the z/OS Resource Recovery Services (RRS), the RRS parameter also determines whether ODBM uses the Open Database Access (ODBA) interface or the database resource adapter (DRA) interface. When RRS=Y, ODBM uses RRS and the ODBA interface.

BPE considerations

Use the ODBM BPE user exit list member of the IMS PROCLIB data set to define ODBM user exits to BPE. This is the member of the IMS PROCLIB data set specified by the EXITMBR= parameter in the BPE configuration parameter member of the IMS PROCLIB data set.

Use the user exit list member of the IMS PROCLIB data set to specify the modules to be called for specific exit types. Each user exit type can have one or more exit modules associated with it. Use the EXITDEF statement to define the user exit modules to be called for a given exit type.

Parameters

ARMRST= Y | N

Specifies whether the z/OS Automatic Restart Manager (ARM) is to be used to restart the ODBM address space after an abend. If you specify Y (yes), ARM restarts the ODBM address space after most system failures. If you specify N (no), ARM does not restart the ODBM address space after any system failure.

ARM does not restart the ODBM address space if ODBM abends before restart is complete. For more information about ARM, see The z/OS Automatic Restart Manager (ARM) (System Administration) in *IMS Version 11 System Administration*.

IMSPLEX()

Specifies definitions for the IMSplex in which ODBM is a member. IMSPLEX is a required parameter. There is no default. Only one IMSPLEX keyword can be specified. The IMSPLEX keyword must precede the left parenthesis. The IMSPLEX definition parameters follow:

NAME=

A 1- to 5-character user-specified identifier that is concatenated to 'CSL' to create the cross-system coupling facility (z/OS cross-system coupling facility) CSL IMSPLEX group name. The value specified here must match the IMSPLEX NAME= value specified in the SCI startup procedure. All IMS address spaces, such as OM, RM, SCI, IMS, ODBM, and so on, that are in the same IMSplex sharing group, sharing either databases or message queues, must specify the same identifier. The same identifier must

also be used for the IMSPLEX= parameter in the CSLSIxxx, CSLRIxxx, CSLOIxxx, and DFSCGxxx members of the IMS PROCLIB data set.

ODBMCFG= 000 | *xxx*
Specifies the 3-character suffix for the ODBM configuration member of the IMS PROCLIB data set, CSLDCxxx, that contains definitions for ODBA connection initialization parameters and any ODBM configuration statements. The default suffix is 000.

ODBMNAME= *odbmnm*
Specifies the 1- to 6-character name of the ODBM address space. You can specify the ODBMNAME parameter on either the CSLODBM startup procedure or in the CSLDIxxx member of the IMS PROCLIB data set. Each instance of ODBM in an IMSplex must have a unique ODBMNAME.

For each ODBMNAME, ODBM creates an eight-character ODBMID that identifies the instance of ODBM within the IMSplex. The ODBMID is the ODBMNAME followed by the characters "OD" and any blank spaces that ODBM needs to add to make the ODBMID eight characters in length.

For example, if you specify an ODBMNAME that is three characters long, ODBM creates the eight-character ODBMID by appending the characters OD to the ODBMNAME and then padding the ODBMID with three blank spaces. If ODBMNAME=ABC, ODBM creates an ODBMID of "ABCOD␣␣␣", in which ␣ represents a blank space.

If you specify the ODBMNAME parameter in the CSLDIxxx member of the IMS PROCLIB data set, you must also either define a separate CSLDIxxx member for each instance of ODBM in an IMSplex or specify ODBMNAME in the startup procedure for each instance of ODBM in the IMSplex.

RRS= Y | N

An optional keyword that specifies both whether ODBM uses RRS and whether ODBM uses the Open Database Access (ODBA) interface or the DRA for communications with IMS DB.

If you specify Y, ODBM uses RRS and the ODBA interface. If ODBM cannot register with RRMS during initialization, ODBM issues message CSL4001A and suspends initialization until the operator responds to the message.

If you specify N, ODBM does not use RRS and uses the DRA interface for communications with IMS DB. When RRS=N is specified, ODBM interfaces with IMS DB in a similar manner as a CCTL.

For more information, see ODBM and RRS (System Administration) in *IMS Version 11 System Administration*.

Examples

A sample ODBM initialization member of the IMS PROCLIB data set is shown in the following example. The sample, called CSLDI000, is provided as part of the IMS PROCLIB data set.

```
*-------------------------------------------------------------------*
* Sample ODBM initialization PROCLIB member CSLDI000.
*-------------------------------------------------------------------*

ARMRST=N               /* ARM should not restart ODBM on failure    */
ODBMNAME=ODBM1         /* ODBM Name (ODBMID = ODBM1OD)              */
IMSPLEX(NAME=PLEX1)    /* IMSplex Name (CSLPLEX1)                   */
ODBMCFG=000            /* Configuration member (CSLDC000)           */
RRS=Y                  /* RRS is the ODBM sync coordinator          */
```

```
*-------------------------------------------------------------------*
* End of Member CSLDI000                                            *
*-------------------------------------------------------------------*
```

A sample ODBM BPE user exit list member of the IMS PROCLIB data set is shown in the following example.

```
*********************************************************************
* ODBM USER EXIT LIST PROCLIB MEMBER                                *
*********************************************************************

#-------------------------------------------------------------------#
# DEFINE 1 ODBM INIT/TERM USER EXIT: ZDINTM00                       #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZDINTM00),COMP=ODBM)

#-------------------------------------------------------------------#
# DEFINE 1 ODBM INPUT USER EXIT: ZINPUT00                           #
# WITH AN ABEND LIMIT OF 8.                                         #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INPUT,EXITS=(ZINPUT00),ABLIM=8,COMP=ODBM)

#-------------------------------------------------------------------#
# DEFINE 1 ODBM OUTPUT USER EXIT: ZOUTPUT0                          #
#-------------------------------------------------------------------#
EXITDEF(TYPE=OUTPUT,EXITS=(ZOUTPUT0),COMP=ODBM)
```

Related concepts:

Chapter 14, “IMS Syntax Checker,” on page 283

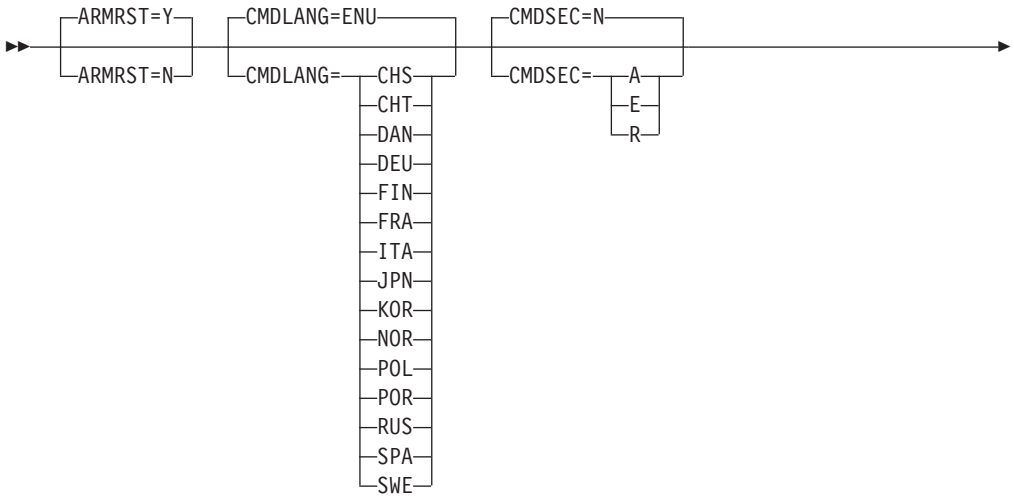
CSLOIxxx member of the IMS PROCLIB data set

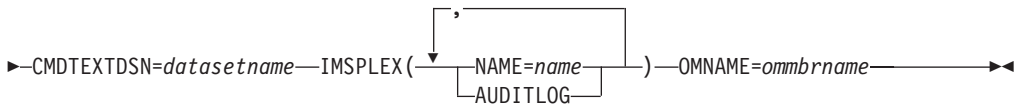
Use the CSLOIxxx member of the IMS PROCLIB data set to specify parameters that initialize the Operations Manager (OM) address space.

Some parameters within CSLOIxxx can be overridden with the OM execution parameters.

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.

Syntax





Usage

A CSLOIxxx member consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a * or # in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between /* and */, for example,

```
/* PROCLIB comments */
```

Values coded in this member of the IMS PROCLIB data set are case-sensitive. In general, use uppercase for all parameters.

BPE considerations

Use the OM BPE user exit list member of the IMS PROCLIB data set to define OM user exits to BPE. This is the member of the IMS PROCLIB data set specified by the EXITMBR= parameter in the BPE configuration parameter member of the IMS PROCLIB data set.

Use the user exit list member of the IMS PROCLIB data set to specify the modules to be called for specific exit types. Each user exit type can have one or more exit modules associated with it. Use the EXITDEF statement to define the user exit modules to be called for a given exit type.

A sample OM BPE user exit list member of the IMS PROCLIB data set is shown in "Sample OM user exit list member of the IMS PROCLIB data set" on page 640.

A sample CSLOIxxx member of the IMS PROCLIB data set is shown in "Sample CSLOIxxx member of the IMS PROCLIB data set" on page 640. The sample, called CSLOI000, is provided as part of the IMS PROCLIB data set.

Parameters

ARMRST= Y | N

An optional keyword that you use to specify whether the z/OS Automatic Restart Manager (ARM) is to be used to restart the OM address space after an abend. If you specify Y (yes), ARM restarts the OM address space after most system failures. If you specify N (no), ARM does not restart the OM address space after any system failure.

ARM does not restart the OM address space if OM ends abnormally before restart is complete.

CMDLANG=

An optional keyword that you use to specify the language to be used for IMS command text that is distributed to OM automation clients upon request. This

affects only the command descriptions that are displayed on a workstation SPOC that requests command text from OM. This value defaults to ENU for US English.

The value is not validated at OM initialization time. It is only validated when a `CSLOMQRY QUERY TYPE(CMDSYNTAX)` request is issued. OM attempts to read a PDS member in the data set specified by the CMDTEXTDSN= with a member name of "CSLOT" concatenated with the 3-character CMDLANG= value. This is the member that contains the command syntax translatable text. The CMDLANG= value can be overridden on the CSLOMQRY request.

This parameter can be specified as an execution parameter on the OM procedure to override the value in CSLOIxxx.

The possible values for this parameter are:

CHS
Simplified Chinese

CHT
Traditional Chinese

DAN
Danish

DEU
German

ENU
English

FIN
Finnish

FRA
French

ITA
Italian

JPN
Japanese

KOR
Korean

NOR
Norwegian

POL
Polish

POR
Portuguese

RUS
Russian

SPA
Spanish

SWE
Swedish

CMDSEC=
An optional keyword that you use to specify the security method to be used for OM command security.

A Specifies that both RACF (or an equivalent security product) and the exit are to be called (options E and R). RACF is called first. Then the security authorization facility (SAF) return code, RACF return code, and RACF reason code are passed to the exit. These return codes are decoded into a security code, which is also passed to the exit for processing.

E Specifies that the OM Security user exit routine is to be called for command authorization.

N Specifies that no authorization checking is to be done. This is the default.

R Specifies that RACF (or an equivalent security product) is to be called for command authorization.

CMDTEXTDSN=

Specifies the data set name for the PDS that contains the command syntax translatable text. This keyword is required. The parameter value is the 1-44 character data set name. The data set must be a PDS with fixed-length record members.

If you specified, for example, CMDTEXTDSN=IMSBLD.IMS11R.SDFSDATA and CMDLANG=SPA (for Spanish language command text), the text would be located in PDS member IMSBLD.IMS11R.SDFSDATA(CSLOTSPA).

IMSPLEX()

Specifies definitions for an IMSplex managed by OM. IMSPLEX is a required parameter. There is no default. Only one IMSPLEX keyword can be specified. The IMSPLEX keyword must precede the left parenthesis. The IMSPLEX definition parameters follow:

NAME=

Specifies a 1-5 character identifier that specifies the IMSplex group name. OM concatenates this identifier to "CSL" to create the IMSplex group name. All IMSplex member address spaces that are in the same IMSplex group sharing either databases or message queues must specify the same identifier. The same identifier must also be used for the IMSPLEX= parameter in the CSLSIxxx, CSLRIxxx, and DFSCGxxx members of the IMS PROCLIB data set.

AUDITLOG=

An optional 1- to 26-character name of the z/OS log stream to which OM writes input and output information about commands, command responses, and unsolicited messages. The information in this log stream tells you where commands originate, what the command processed, and the result of that processing. You cannot issue this parameter as an OM execution parameter.

The audit trail log stream SYSLOG.OM2Q01.LOG is defined as part of the IVP and is then associated with the AUDITLOG= parameter of the IVP sample job.

Requirement: The following SAF security rules must be defined to support the use of the OM audit trail:

- The OM address space has FAC (facility) access to the structure.
- The OM address space has access to the log stream. This access can be accomplished using an SAF rule.
- Any user reading the log stream must have SAF read access.

OMNAME=*ommbrname*

Specifies the name for the OM address space. This is an optional 1-6 character name. Specify this parameter either as an execution parameter or in the CSLOIxxx member of the IMS PROCLIB data set. This name is used to create the OMID which is used in OM processing. The 8-character OMID is the OMNAME followed by the characters "OM". Trailing blanks in the OMNAME are deleted and the OMID is padded with blanks. For example, if OMNAME=ABC then OMID="ABCOM ".

Sample OM user exit list member of the IMS PROCLIB data set

```
*********************************************************************
* OM USER EXIT LIST PROCLIB MEMBER                                  *
*********************************************************************

#-------------------------------------------------------------------#
# DEFINE 1 OM CLIENT CONNECTION USER EXIT: ZOCLNCN0                 #
#-------------------------------------------------------------------#
EXITDEF(TYPE=CLNTCONN,EXITS=(ZOCLNCN0),COMP=OM)

#-------------------------------------------------------------------#
# DEFINE 1 OM INIT/TERM USER EXIT: ZOINTM00                         #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZOINTM00),COMP=OM)

#-------------------------------------------------------------------#
# DEFINE 1 OM INPUT USER EXIT: ZINPUT00                             #
# WITH AN ABEND LIMIT OF 8.                                         #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INPUT,EXITS=(ZINPUT00),ABLIM=8,COMP=OM)

#-------------------------------------------------------------------#
# DEFINE 1 OM OUTPUT USER EXIT: ZOUTPUT0                            #
#-------------------------------------------------------------------#
EXITDEF(TYPE=OUTPUT,EXITS=(ZOUTPUT0),COMP=OM)

#-------------------------------------------------------------------#
# DEFINE 1 OM SECURITY USER EXIT: ZSECURE0                          #
#-------------------------------------------------------------------#
EXITDEF(TYPE=SECURITY,EXITS=(ZSECURE0),COMP=OM)
```

Sample CSLOIxxx member of the IMS PROCLIB data set

```
*-------------------------------------------------------------------*
* Sample OM Initialization PROCLIB member.
*-------------------------------------------------------------------*

ARMRST=Y,                   /* ARM should restart OM on failure  */
CMDLANG=ENU,                /* Use English for Command Desc      */
CMDSEC=N,                   /* No Command Security               */
OMNAME=OM1,                 /* OM Name (OMID = OM1OM)            */
IMSPLEX(NAME=PLEX1)         /* IMSplex Name (CSLPLEX1)           */
CMDTEXTDSN=IMSTESTG.DUMMY.TRNTBL  /* CMD Syntax Translation Table */

*-------------------------------------------------------------------*
* End of Member CSLOI000                                           *
*-------------------------------------------------------------------*
```

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

The z/OS Automatic Restart Manager (ARM) (System Administration)

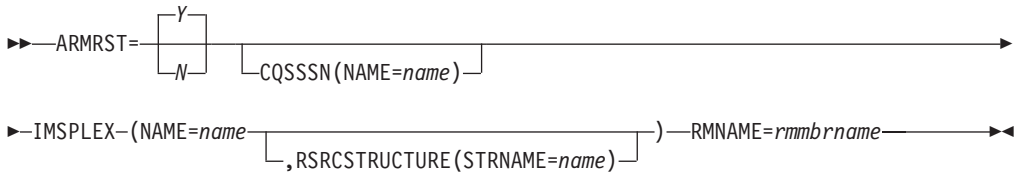
CSLRIxxx member of the IMS PROCLIB data set

Use the CSLRIxxx member of the IMS PROCLIB data set to specify parameters that initialize RM.

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.

Syntax

Some parameters within CSLRIxxx can be overridden with RM execution parameters.



Usage

The CSLRIxxx member of the IMS PROCLIB data set consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a "*" or "#" in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between " /* "and "*/", for example, `/* PROCLIB comments */`. Values coded in this member of the IMS PROCLIB data set are case sensitive. In general, you should use uppercase for all parameters.

A sample CSLRIxxx member of the IMS PROCLIB data set is shown in "Sample CSLRIxxx member of the IMS PROCLIB data set" on page 643.

BPE considerations

Use the RM BPE user exit list member of the IMS PROCLIB data set to define RM user exits to BPE. This is the member of the IMS PROCLIB data set specified by the EXITMBR= parameter in the BPE configuration parameter member of the IMS PROCLIB data set.

Use the user exit list member of the IMS PROCLIB data set to specify the modules to be called for specific exit types. Each user exit type can have one or more exit modules associated with it. Use the EXITDEF statement to define the user exit modules to be called for a given exit type.

A sample RM user exit list member of the IMS PROCLIB data set is shown in "Sample RM user exit member of the IMS PROCLIB data set" on page 643.

Parameters

ARMRST= Y | N

Specifies whether the z/OS Automatic Restart Manager (ARM) should be used to restart RM after an abend. **Y** (yes) specifies that ARM should be used. The RM address space is restarted by ARM after most system failures. **N** (no) specifies that ARM should not be used. RM is not restarted by ARM after any failures. For information about ARM, see The z/OS Automatic Restart Manager (ARM) (System Administration) in *IMS Version 11 System Administration*.

CQSSSN=

Specifies the one- to four-character subsystem name of the CQS. RM uses this name to connect to the proper CQS. When connecting RM to CQS, you must specify the same value on CQSSSN= and on the SSN= parameter of the CQSIPxxx member of the IMS PROCLIB data set for the target CQS. The parameter is optional, and no default exists. Both CQSSSN and RSRCSTRUCTURE must be specified together, or neither must be specified. CQSSSN and RSRCSTRUCTURE must be specified to use RM's global resource services.

IMSPLEX()

Specifies definitions for an IMSplex managed by RM. IMSPLEX is a required parameter. There is no default. Only one IMSPLEX keyword can be specified. The IMSPLEX keyword must precede the left parenthesis. The IMSPLEX definition parameters follow:

NAME=

Specifies a 1-5 character identifier that specifies the IMSplex group name. This defines the IMSplex to which the resource structure is defined. NAME is required and no default exists. RM concatenates this identifier to "CSL" to create the IMSplex group name. All OM, RM, SCI, IMS, and IMSplex members that are in the same IMSplex sharing group sharing either databases or message queues must specify the same identifier. The same identifier must also be used for the IMSPLEX= parameter in the CSLSIxxx, CSLOIxxx, and DFSCGxxx members of the IMS PROCLIB data set.

RSRCSTRUCTURE()

Specifies definitions for a resource structure managed by RM. This keyword construct is optional. RSRCSTRUCTURE must be specified to use RM's global resource services. Only one resource structure can be defined. The resource structure definitions must be enclosed within parentheses and separated by commas. The RSRCSTRUCTURE keyword must precede the left parenthesis.

STRNAME=

Specifies the 1- to 16-character name of a resource structure that IMS connects to, which contains IMS resource information. If the RSRSTRUCTURE construct is specified, then STRNAME is required within the RSRCSTRUCTURE construct.

The installation must have defined the structure name in the CFRM administrative policy. The structure name must follow the naming rules as allowed by the CFRM. The structure name must be from 1- to 16-characters long. For names with less than 16 characters, CQS pads the name with blanks. The valid characters are A-Z, 0-9 and the following special characters:

```
$ @ # _
```

Names must be uppercase and start with an alphabetic character. This resource structure must also be defined in the CQS global structure definition member of the IMS PROCLIB data set (CQSSGxxx) of the CQS in the same IMSplex sharing group. This resource structure must also be defined in the CFRM policy.

Restriction: Avoid using names that IBM uses for its structures. Do not begin structure names with the letters A-I, or with the character string SYS. If you do name a structure beginning

with the letters A-I or SYS, your name might conflict with an existing or future IBM-defined structure name.

RMNAME=*rmmbrname*

Specifies the name for the RM address space. This is an optional 1- to 6-character name. You must specify this parameter either as an execution parameter or in the CSLRIxxx member of the IMS PROCLIB data set. This name is used to create the RMID, which is used in RM processing. The 8-character RMID is the RMNAME followed by the characters "RM". Trailing blanks in the RMNAME are deleted, and the RMID is padded with blanks. For example, if RMNAME=ABC then RMID="ABCRM ".

Sample CSLRIxxx member of the IMS PROCLIB data set

```
*-------------------------------------------------------------------*
* Sample RM Initialization PROCLIB member.                          *
*-------------------------------------------------------------------*
ARMRST=Y,                     /* ARM should restart RM on failure  */
CQSSSN=CQS1,                  /* CQS to manage Resource Structure  */
IMSPLEX(
  NAME=PLEX1,                 /* IMSplex name (CSLPLEX1)           */
  RSRCSTRUCTURE(
    STRNAME=IMSRSRC01)),      /* RESOURCE STRUCTURE NAME           */
RMNAME=RM1                    /* RM Name (RMID = RM1RM)            */
*-------------------------------------------------------------------*
* End of Member CSLRI000                                            *
*-------------------------------------------------------------------*
```

Sample RM user exit member of the IMS PROCLIB data set

```
*********************************************************************
* RM USER EXIT LIST PROCLIB MEMBER                                  *
*********************************************************************

#-------------------------------------------------------------------#
# DEFINE 1 RM CLIENT CONNECTION USER EXIT: ZRCLNCN0                 #
# WITH AN ABEND LIMIT OF 8.                                         #
#-------------------------------------------------------------------#
EXITDEF(TYPE=CLNTCONN,EXITS=(ZRCLNCN0),ABLIM=8,COMP=RM)

#-------------------------------------------------------------------#
# DEFINE 1 RM INIT/TERM USER EXIT: ZRINTM00                         #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZRINTM00),COMP=RM)
```

Related concepts:

Authorizing connections to CQS structures (System Administration)

Chapter 14, "IMS Syntax Checker," on page 283

Related reference:

"COMMON_SERVICE_LAYER section of the DFSDFxxx member" on page 673

"DFSCGxxx member of the IMS PROCLIB data set" on page 649

"CSLRM procedure" on page 518

"CSLSIxxx member of the IMS PROCLIB data set"

"CSLOIxxx member of the IMS PROCLIB data set" on page 636

"CQSSGxxx member of the IMS PROCLIB data set" on page 619

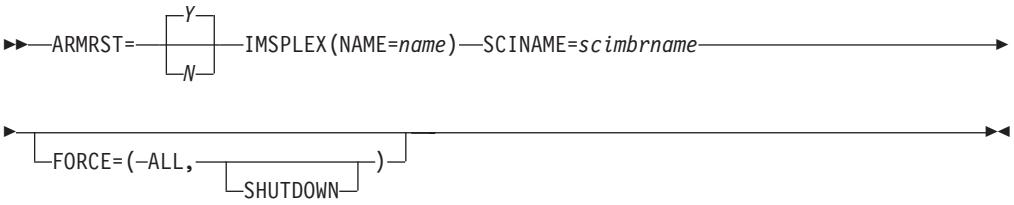
CSLSIxxx member of the IMS PROCLIB data set

Use the CSLSIxxx member of the IMS PROCLIB data set to specify parameters related to initialization of the SCI address space.

Some parameters within CSLSIxxx can be overridden using SCI execution parameters.

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.

Syntax



Usage

This member of the IMS PROCLIB data set consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a "*" or "#" in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between " /* "and "*/", for example, `/* PROCLIB comments */`. Values coded in this member of the IMS PROCLIB data set are case sensitive. In general, you should use uppercase for all parameters.

BPE considerations

Use the SCI BPE user exit list member of the IMS PROCLIB data set to define SCI user exits to BPE. This is the member specified by the EXITMBR= parameter in the BPE configuration parameter member of the IMS PROCLIB data set.

Use the user exit list member of the IMS PROCLIB data set to specify the modules to be called for specific exit types. Each user exit type can have one or more exit modules associated with it. Use the EXITDEF statement to define the user exit modules to be called for a given exit type.

A sample SCI user exit list member of the IMS PROCLIB data set is shown in "Sample SCI user exit list member of the IMS PROCLIB data set" on page 646.

Parameters

The parameters associated with the syntax diagram are added next.

ARMRST= Y | N
Specifies whether the z/OS Automatic Restart Manager (ARM) should be used to restart the SCI address space after an abend. **Y** (yes) specifies that ARM should be used. The SCI address space is restarted by ARM after most system failures. **N** (no) specifies that ARM should not be used. The SCI address space

is not restarted by ARM after any failures. For more information about ARM, see The z/OS Automatic Restart Manager (ARM) (System Administration) in *IMS Version 11 System Administration*.

IMSPLEX()
Specifies definitions for an IMSplex managed by SCI. IMSPLEX is a required parameter. There is no default. Only one IMSPLEX keyword can be specified. The IMSPLEX keyword must precede the left parenthesis. The IMSPLEX definition parameters follow:

NAME=
Specifies a 1- to 5-character name that specifies the IMSplex group name. SCI concatenates this name to "CSL" to create the IMSplex group name. All OM, RM, SCI, IMS, and other address spaces that are in the same IMSplex must specify the same name. This is done by specifying the same name for the IMSPLEX= parameter in the CSLOIxxx, CSLRIxxx, CSLSIxxx, and DFSCGxxx members of the IMS PROCLIB data set.

SCINAME=*scimbrname*
Specifies the name for the SCI address space. This is an optional 1-6 character name. You must specify this parameter either as an execution parameter or in the CSLSIxxx member of the IMS PROCLIB data set. This name is used to create the SCIID which is used in SCI processing. The 8-character SCIID is the SCINAME followed by the characters "SC". Trailing blanks in the SCINAME are deleted and the SCIID is padded with blanks. For example, if SCINAME=ABC then SCIID="ABCSC ".

FORCE=()
Specifies that SCI is to clean up the global interface storage. FORCE is an optional parameter and has no default. The keywords are:

ALL SCI should delete all the global storage, including control blocks and routines. This keyword is required.

SHUTDOWN
SCI should shut down after cleaning the global storage. This keyword is optional.

No local IMSplex members can be active when the FORCE keyword is used. If a member is active, results are unpredictable. Use the FORCE keyword in the following situations:

- When an IMSplex managed by an SCI on one image is managed by a different SCI. For example, PLEX1 is managed by SCI1. If SCI1 becomes inactive, PLEX1 is managed by SCI2. Before SCI2 is started, use FORCE(ALL,SHUTDOWN) on SCI1 to clean the global storage.
- When an SCI is not reactivated on an image. To clean the global storage, reactive that SCI one final time using the FORCE(ALL, SHUTDOWN) keyword.

Sample CSLSIxxx member of the IMS PROCLIB data set

A sample CSLSIxxx member of the IMS PROCLIB data set is shown in "Sample CSLSIxxx member of the IMS PROCLIB data set."

```
*********************************************************************
* SCI INITIALIZATION PROCLIB MEMBER                                 *
*********************************************************************
```

```
ARMRST=Y                /* ARM should restart SCI on failure        */
IMSPLEX(NAME=PLEX1)     /* IMSplex name (CSLPLEX1)                  */
SCINAME=SCI1            /* SCI name (SCIID = SCI1SC)                */
```

Sample SCI user exit list member of the IMS PROCLIB data set

```
*********************************************************************
* SCI USER EXIT LIST PROCLIB MEMBER                                 *
*********************************************************************
#-------------------------------------------------------------------#
# DEFINE 1 SCI CLIENT CONNECTION USER EXIT: ZSCLNCN0                #
# WITH AN ABEND LIMIT OF 8.                                         #
#-------------------------------------------------------------------#
EXITDEF(TYPE=CLNTCONN,EXITS=(ZSCLNCN0),ABLIM=8,COMP=SCI)

#-------------------------------------------------------------------#
# DEFINE 1 SCI INIT/TERM USER EXIT: ZSINTM00                        #
#-------------------------------------------------------------------#
EXITDEF(TYPE=INITTERM,EXITS=(ZSINTM00),COMP=SCI)
```

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

DBFMSDBx member of the IMS PROCLIB data set

Use the DBFMSDBx member of the IMS PROCLIB data set to specify the Main Storage Databases (MSDBs) to be loaded at startup.

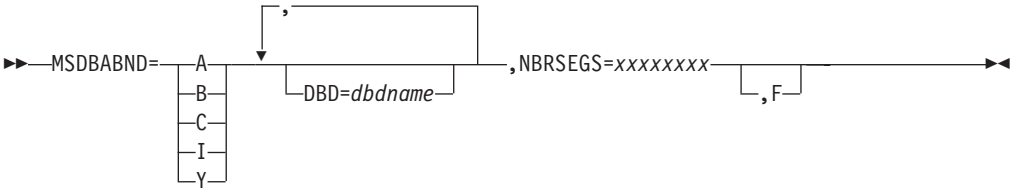
You must have previously placed DBFMSDBx in the library whose DD name is PROCLIB. Several DBFMSDBx members can be present in the library; differentiate them by varying the last character (the suffix, x) of the name. The MSDB= keyword parameter of the IMS procedure specifies the suffix of the particular member to be used at startup.

Environments

This member is not applicable in the DBCTL environment.

Syntax

A DBFMSDBx member consists of one or more 80-character records, each specifying an MSDB to be loaded. The format is:



Usage

You can separate keyword parameters with either commas or blanks. Positions 73 through 80 are ignored but can be used to sequence the records. Syntax errors are identified by messages.

At startup, the MSDBs specified by a DBFMSDBx member are loaded from one of the sequential data sets identified as MSDBINIT, MSDBCP1, or MSDBCP2. You

cannot, therefore, use the DBFMSDBx member to specify MSDBs that are not included in these data sets; an abend occurs if you use DFSMSDBx in this manner.

Parameters

MSDBABND=
Specifies that the IMS control region is to abend if an error occurs during MSDB loading at system initialization. The valid values for the MSDBABND= parameter are:

A Valid for both I and C options.

B Valid for both Y and C options.

C Abend if initial checkpoint for MSDBs cannot occur.

I Abend for one or more of the following reasons:

- No segments exist in the MSDBINIT data set for at least one defined MSDB.
- For the reasons described in option Y.

Y Abend if the MSDBs cannot be loaded due to errors with the MSDBINIT data set.

DBD=
Specifies the same one- to eight-character DBD name as was specified at DBDGEN. Multiple DBD= parameters are allowed.

NBRSEGS=
Specifies the number of database records expected for this MSDB. You must specify a one- to eight-digit number greater than or equal to the number of MSDB segments loaded at restart. By specifying a number greater than the number of segments to be loaded, the parameter can be used to reserve space for a terminal-related dynamic MSDB.

F Optionally specifies that the MSDB is to be page-fixed.

DFS62DTx member of the IMS PROCLIB data set

Use the DFS62DTx member of the IMS PROCLIB data set to store the LU 6.2 device descriptors that are built during IMS initialization.

In the DFS62DTx member of the IMS PROCLIB data set, *x* is the IMS nucleus suffix. LU 6.2 device descriptors specify an LTERM that associates an output destination with an LU 6.2 device. The system administrator can additionally change applications that use alternate PCBs into applications that use LU 6.2 devices, without application coding changes.

No IMS definition or system definition changes are required to use an LU 6.2 device.

The record format is:

Position
Contents

1 Descriptor type (“U” for user descriptor)

Although this column is not required for an LU 6.2 descriptor, it is included for consistency with ETO descriptors. It is not checked, and no error message is issued if it is omitted.

2 Blank

3-10 Contains the eight-character LTERM name, left-justified and padded with blanks, if necessary. This positional parameter is required, and is used as the descriptor name. A descriptor definition can be continued by repeating the descriptor name in subsequent records. Each individual descriptor can contain up to 50 records (excluding comment records).

11 Blank

12-72 Contains the parameter sets for the descriptor.

73-80 Ignored

DFS62DTx descriptor format

This is the format for the DFS62DTx descriptor. Do not code a parameter in the descriptor and then leave it blank such as (SIDE=b). If you do this, an error message is issued. Instead, omit it completely if you do not need to specify a DFS62DTx descriptor. A descriptor definition can be continued by repeating the descriptor name in subsequent records.

```
U ltermname parm1 parm2 parmn
```

In this example:

U Indicates user descriptor.

ltermname
The LTERM name.

parm (1...n)
Can be any of the following parameters: SIDE=, MODE=, TPNAME=, SYNCLEVEL=, CONVTYPE= , OUTBND=.

The following is an example of a DFS62DTx descriptor:

```
Column   Column                                                  Column
1        12                                                         72
U L62TERM1 LUNAME=L62IMS1 TPNAME=CPICTRN1 MODE=L62MDE02
U L62TERM1 SYNCLEVEL=N OUTBND=MYLU02
```

Separate the parameters with a blank.

Keyword descriptions

CONVTYPE=
Specifies whether the conversation type is basic (B) or mapped (M). The default value is M.

LUNAME=
Is a 1- to 17-alphanumeric character (uppercase) name of the destination LU 6.2 application program, which can be a 1- to 17-byte network-qualified LU name. An example of a network-qualified LU name is *netid1.luname1*. If SIDE= is specified, LUNAME overrides the LUNAME in the side information entry. The default is DFSLU.

Do not specify a network name for a dependent LU.

MODE=
Is a one- to eight-character name of the VTAM mode table entry to be used. If SIDE is specified, MODE overrides the mode name in the side information entry. The default is DFSMODE.

OUTBND=
Identifies a local LU to be used for outbound message processing.

SIDE=
Is a one- to eight-character alphanumeric name identifying side information entry. (Side information refers to the system-defined values for CPI communications initialization.) If you specify an entry name on this keyword, the LUNAME, TPNAME, and MODE keywords are blank unless you also specify an overriding LUNAME, TPNAME, or MODE. If you do not specify an entry name, the default values apply to the LUNAME, TPNAME, and MODE keywords.

SYNCLEVEL=
Specifies whether the APPC/IMS sync level is confirmed (C) or not (N). The default value is C.

TPNAME=
Is a 1- to 64-character name of the transaction program to be scheduled. The default is DFSASYNC. If SIDE= is specified, TPNAME overrides the TPNAME in the side information entry.

Recommendation: Use upper-case characters for TPNAMES so that the `/ALLOCATE` and `/CHANGE` commands can operate on the resulting messages and descriptors.

DFSCGxxx member of the IMS PROCLIB data set

Use the DFSCGxxx member of the IMS PROCLIB data set to specify parameters that are related to the Common Service Layer (CSL), including the Operations Manager (OM), the Resource Manager (RM), and the Structured Call Interface (SCI).

All of the parameters that are valid in the DFSCGxxx member of the IMS PROCLIB data set are valid in the DFSDFxxx member of the IMS PROCLIB data set. If you specify values in both the DFSCGxxx member of the IMS PROCLIB data set and the CSL section of the DFSDFxxx member of the IMS PROCLIB data set, the values specified in the DFSCGxxx member override the values specified in the DFSDFxxx member.

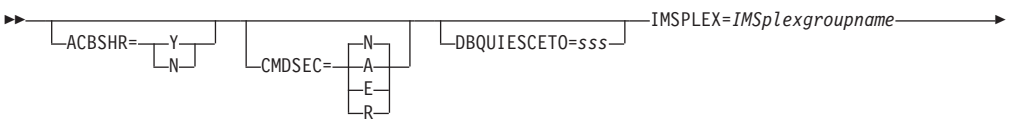
The suffix for DFSCGxxx, *xxx*, is specified on the CSLG= parameter. All IMSplex members (OM, RM, SCI, IMS, and others) that are in the same IMSplex sharing group, sharing either databases or message queues, must specify the same values.

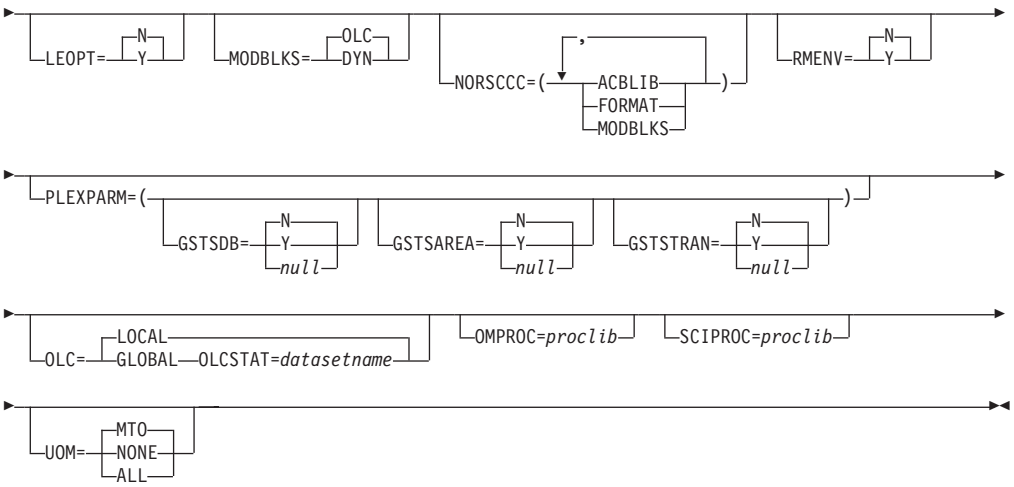
Environments

The DFSCGxxx member of the IMS PROCLIB data set can be used in the IMS DB/DC and DBCTL environments.

Syntax

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.





Usage

A DFSCGxxx member consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a "*" or "#" in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between " /* "and "*/", for example, /* PROCLIB comments */. Values that are coded in this member of the IMS PROCLIB data set are case-sensitive. In general, use uppercase for all parameters.

For installations that do not require Resource Manager services, the DFSCGxxx member of the IMS PROCLIB data set includes parameters (RMENV, OMPROC, and SCIPROC) for specifying that a Resource Manager environment is not required. These parameters are useful if you do not use shared queues or a sysplex, but you would like to issue IMS type-1 and type-2 commands through the OM API.

Table 66 identifies the DFSCGxxx parameters and their applicability based on whether RM is used.

The DFSCGxxx parameters can also be specified on the DFSDFxxx member of the IMS PROCLIB data set.

Table 66. Applicability of DFSCGxxx parameters based on resource manager environment

Parameter	RMENV=Y	RMENV=N
ACBSHR	optional	not applicable
CMDSEC	optional	optional
DBQUIESCETO	optional	optional
IMSPLEX	required	required
LEOPT	optional	optional
MODBLKS	optional	optional
NORSCCC	ignored	ignored

Table 66. Applicability of DFSCGxxx parameters based on resource manager environment (continued)

Parameter	RMENV=Y	RMENV=N
OLC	optional	optional
OLCSTAT	optional	optional
OMPROC	not applicable	optional
RMENV	optional	required
PLEXPARM	optional	not applicable
SCIPROC	not applicable	optional
UOM	optional	optional

Important: When RMENV=N and OLC=GLOBAL is specified, ensure that no other IMS attempts to use the same OLCSTAT data set. If another IMS uses the same OLCSTAT data set, or if the OLCSTAT data set utility (DFSUOLC0) is used to initialize the data set for another member, an additional IMSID is added to the OLCSTAT data set. If this occurs and IMS (running with RMENV=N and OLC=GLOBAL specified) attempts to initiate an online change by issuing the INIT | TERM OLC command, the command is rejected because an IMSID other than the one executing the command resides in the OLCSTAT data set. The OLCSTAT data set must then be corrected before an INIT | TERM OLC can complete successfully.

Parameters

ACBSHR=

Specifies whether the ACB libraries are shared among the IMS systems that are listed in the OLCSTAT data set. This parameter can be specified in both the DFSCGxxx and DFSDFxxx members of the IMS PROCLIB data set. If it is specified in both members, IMS uses the ACBSHR value that is specified in the DFSCGxxx member of the IMS PROCLIB data set.

Y Specifies that all the IMS systems that are listed in the OLCSTAT data set use the same ACB library.

N Specifies that each IMS that is listed in the OLCSTAT data set uses its own dedicated ACB library.

Each IMS system sharing the OLCSTAT data set must have the same value for ACBSHR=. If the ACBSHR= values are not the same, the `INIT OLC PREPARE TYPE(ACBMBR)` command is rejected.

CMDSEC=

Specifies whether IMS should perform security checking on commands that are routed from OM, and if so, whether RACF or the Command Authorization exit routine (or both) should be used. If IMS performs the security checking on commands, the CMDSEC keyword also specifies whether RACF, the IMS Command Authorization exit routine (DFSCCMD0), or both should be used. This option applies only to IMS type-1 commands that can be issued through the OM API.

For type-2 commands (that are only allowed through the OM API, not allowed from the IMS master terminal), all security checking is performed by OM and is controlled by the CMDSEC= parameter on the OM startup JCL or on the OM initialization member of the IMS PROCLIB data set (CSLOIxxx).

Related reading: For all the details of the CSLOIxxx member of the IMS PROCLIB data set, see "CSLOIxxx member of the IMS PROCLIB data set" on page 636.

Recommendation: Use OM command security instead of IMS security. By allowing OM to perform the security checks, commands which fail security authorization are not routed to IMS, which reduces processing overhead and network traffic. When IMS command security is used, you must ensure that all IMS systems use the same security profiles and user exits. If IMS systems in the same IMSplex use different security rules, the results of command security checking may be unpredictable.

A Specifies that both RACF and DFSCCMD0 are to be called (options E and R). RACF is called first. Then the security authorization facility (SAF) return code, RACF return code, and RACF reason code are passed to DFSCCMD0.

Related reading: For complete information about the Command Authorization exit routine, see *IMS Version 11 Exit Routines*.

E Specifies that the Command Authorization exit routine is to be called for command authorization.

N Specifies that no authorization checking is to be done. Security checking can still be performed by OM depending on the specification of the OM CMDSEC= parameter.

R Specifies that RACF is to be called for command authorization.

DBQUIESCETO=
The amount of time, in seconds, that the QUIESCE command waits for currently running applications to commit before the QUIESCE command is aborted. This value applies only when the quiesce function is initiated. The actual time required for the QUIESCE command varies and might exceed this timeout value. The default value is 30 seconds. *sss* must be a non-zero value 1 - 999. You can also specify this timeout value in the COMMON_SERVICE_LAYER section of the DFSDFxxx member of the IMS PROCLIB data set, or override the value using the appropriate UPDATE command for the resource type, for example, UPDATE DATAGRP NAME(name) START(QUIESCE) SET(TIMEOUT(10)).

IMSPLEX=
Specifies a one- to five-character identifier. IMS appends this identifier to "CSL" to create the IMSplex group name. All IMSplex members (OM, RM, IMS, CQS) that are in the same IMSplex sharing group sharing either databases or message queues must specify the same identifier. IMS has no way to enforce this; the user is responsible for ensuring that the IMSplex name is correctly specified. The same identifier must also be used for the IMSplex= parameter in the CSLSIxxx, CSLOIxxx, CSLRIxxx, and CQSIPxxx members of the IMS PROCLIB data set.

When RMENV=N, IMSPLEX= specifies a one- to five-character name of the IMSplex that OM, SCI, and the IMS control regions join. This IMSplex is simply a grouping of all address spaces that are managed by one or more OMs.

This parameter is required.

LEOPT=Y | N
Specifies whether IMS should allow IBM Language Environment for z/OS (LE) dynamic runtime parameter overrides.

MODBLKS=

Specifies whether resources in the IMS.MODBLKS data set are defined dynamically or by online change. These resources include databases, programs, routing codes, and transactions. This attribute can only be changed on an IMS cold start. If the value is changed for the next IMS warm or emergency restart, IMS restart terminates, and abend U0168 is issued.

DYN

Enables dynamic definition for resources in the IMS.MODBLKS data set. Resource definitions can be dynamically added, changed, or deleted by using online commands, including CREATE, DELETE, and UPDATE. Online change for resources in the IMS.MODBLKS data set that is initiated by a /MODIFY PREPARE MODBLKS or INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS) command is not allowed.

OLC

Enables online change for resources in the IMS.MODBLKS data set. Resource definitions can be added, changed, or deleted by using online change. Online change commands for resources in the IMS.MODBLKS data set include /MODIFY PREPARE MODBLKS or INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS). When MODBLKS=OLC, UPDATE commands that update runtime values (such as a transaction class) are permitted. This is the default.

NORSCCC=()

Resource consistency checking is no longer performed for ACBLIB, FMTLIB, and MODBLKS libraries, regardless of whether there is a resource structure or if values are specified on NORSCCC. The NORSCCC keyword is supported for compatibility, but its values are ignored.

Specifies that no resource consistency checking is to be performed for the specified resources. The resources to not check for consistency must be enclosed within parentheses and separated by commas. The NORSCCC keyword must precede the left parenthesis. NORSCCC is an optional parameter.

If a resource structure is defined for the IMSplex, by default, consistency checking is performed for ACBLIB, FORMAT, and MODBLKS. Consistency checking is performed on the data set names; it is not performed on the resources that reside in the libraries. Resource consistency checking is useful for cloned systems that either share all these data sets or use the same data set names on each IMS.

If no resource structure is defined, no resource definition consistency checking is performed.

When RMENV=N, the NORSCCC parameter is ignored.

Specify one or more of the following parameters:

ACBLIB

The ACBLIB data set names are not checked for consistency. The IMS systems in the IMSplex do not need to define the same data sets for the ACBLIB library. ACBLIB applies only if OLC=GLOBAL is specified.

FORMAT

The FORMAT data set names are not checked for consistency. The IMS systems in the IMSplex do not need to define the same data sets for the FORMAT library. FORMAT applies only if OLC=GLOBAL is specified.

MODBLKS
The MODBLKS data set names are not checked for consistency. The IMS systems in the IMSplex do not need to define the same data sets for the MODBLKS library. MODBLKS applies only if OLC=GLOBAL is specified.

OLC=LOCAL | GLOBAL
Specifies the scope of the online change. OLC=LOCAL means that the online change applies locally to each IMS. Local online change is prepared and committed by using the `/MODIFY PREPARE` and `/MODIFY COMMIT` commands on each local IMS. The local online changes must be manually coordinated across an IMSplex. OLC=GLOBAL means that the online change is coordinated across the IMSplex; ACBLIB member OLC requires OLC=GLOBAL. Global online change is prepared and committed by using the `INITIATE OLC` command. If global online change is enabled and a resource structure is defined, the MODBLKS, FORMAT, and ACBLIB data sets must be consistent across the IMSplex, unless resource consistency checking is omitted by the NORSCCC keyword.

Online change across an IMSplex (OLC=GLOBAL) is not supported for RSR tracking IMS systems. If specified, IMS abends during restart with ABENDU2801 subcode X'0013'.

OLCSTAT=
Specifies the 1 - 44 character data set name of the OLCSTAT data set. OLCSTAT is ignored if OLC=LOCAL is specified. OLCSTAT is required if OLC=GLOBAL is defined. The OLCSTAT data set is a cataloged BSAM data set that contains global online change information and status. The data set name is used to dynamically allocate the data set when an IMS initialized, restarts, or is master of an online change phase.

All IMS systems in an IMSplex must refer to the same physical OLCSTAT data set. If a resource structure is defined to the IMSplex, IMS ensures that the OLCSTAT data set name is defined consistently for the IMSplex. If the OLCSTAT data set name definition is not consistent with the OLCSTAT defined to other IMS systems in the IMSplex, IMS initialization fails.

RMENV=Y | N
Indicates whether (Y) or not (N) IMS requires an RM environment.

Y Indicates that IMS requires an RM environment in order to use Resource Manager services. IMS initialization does not complete until IMS successfully registers with RM. This is the default.

N Indicates that IMS does not require an RM environment and does not use Resource Manager services. IMS does not attempt to register with RM even if an RM address space is active. When RMENV=N, you do not need to define or start an RM address space. IMS commands and functions that require RM are not available in this environment.

When RMENV=N, online change is as follows:

- If OLC=LOCAL, use the `/MODIFY` command to initiate online change. The MODSTAT data set must be defined.
- If OLC=GLOBAL, use the `INIT OLC` command to initiate online change. The OLCSTAT data set must be defined; however, it cannot be shared between IMS systems.

You can change the value of RMENV= across IMS warm or cold starts.

OMPROC=
Specifies the one- to eight-character name of the member of the IMS PROCLIB

data set that contains the procedure for the OM address space. This parameter is only valid when RMENV=N is specified. If RMENV=Y is specified, this parameter is ignored. This parameter is optional.

IMS does not start the OM address space specified by this parameter if another OM address space is already active in the IMSplex during IMS initialization.

If OMPROC= is not specified, start the OM address space before starting IMS. This start sequence can be performed through an automation program or by another IMS control region (if RMENV=N is specified).

PLEXPARM=()

PLEXPARM is optional. If you do not specify it, either the default value or a NULL value is used. Refer to the specific subparameter to determine whether the default or NULL value is used.

The PLEXPARM parameter defines how global status is maintained for resources in an IMSplex. An IMS system that is initialized in an IMSplex checks to see if a global PLEXPARM entry exists in the RM resource structure. If the entry exists, the IMS system uses the values. If no entry exists, the initializing IMS writes its resource definition values to the RM resource structure global PLEXPARM entry. Any IMS system that initializes in that IMSplex after that uses the values stored by the first IMS system. If the subsequent IMS system has PLEXPARM values that are different from the values in the global PLEXPARM entry, message DFS3425I is issued, and values from the global PLEXPARM entry are used.

To view the current PLEXPARM values, use the `QUERY IMS` command. To update PLEXPARM values, use the `UPDATE IMS` command.

If an IMS system is running in an IMSplex that does not include an RM, or the RM in the IMSplex does not use a resource structure, the PLEXPARM parameter is ignored. No message is issued indicating that the parameter is ignored. All PLEXPARM values are set to no (N).

In an RSR or FDBR system, the PLEXPARM parameter is ignored. No message is issued indicating that the parameter is ignored. All PLEXPARM values are set to no (N). However, an FDBR system and an XRF alternate system do maintain global PLEXPARM values internally. You can use the QUERY IMS SHOW(PLEXPARM) command to display PLEXPARM values in an FDBR or XRF alternate system.

GSTSDB=

Specifies how global status for databases is kept in the IMSplex.

N No global status is maintained for databases resources in the RM. This is the default value for DB/DC and DBCTL regions.

Y Global status for databases is maintained in the RM.

null
No global status is maintained for databases resources in the RM. This is the default value for DCCTL regions.

A DCCTL system that initializes this value in the global PLEXPARM entry writes a null value. If a DB/DC or DBCTL system later joins that IMSplex, that system updates the GSTSDB value in the global PLEXPARM entry with the value from its PLEXPARM initialization parameter.

GSTSAREA=

Specifies how global status for DEDB areas is kept in the IMSplex.

N No global status is maintained for DEDB area resources in the RM. This is the default value for DB/DC and DBCTL regions.

Y Global status for DEDB areas is maintained in the RM.

null
No global status is maintained for DEDB area resources in the RM. This is the default value for DCCTL regions.

A DCCTL system that initializes this value in the global PLEXPARM entry writes a null value. If a DB/DC or DBCTL system later joins that IMSplex, that system updates the GSTSDB value in the global PLEXPARM entry with the value from its PLEXPARM initialization parameter.

GSTSTRAN=
Specifies how global status for transactions is kept in the IMSplex.

N No global status is maintained for transaction resources in the RM. This is the default value for DB/DC and DCCTL regions.

Y Global status for transactions is maintained in the RM.

null
No global status is maintained for transaction resources in the RM. This is the default value for DBCTL regions.

A DBCTL system that initializes this value in the global PLEXPARM entry writes a null value. If a DB/DC or DCCTL system later joins that IMSplex, that system updates the GSTSDB value in the global PLEXPARM entry with the value from its PLEXPARM initialization parameter.

SCIPROC=
Specifies the one- to eight-character name of the member of the IMS PROCLIB data set that contains the procedure for the SCI address space. This parameter is only valid when RMENV=N. If RMENV=Y is specified, this parameter is ignored. This parameter is optional.

If SCIPROC= is not specified, start the SCI address space before starting IMS. This start sequence can be performed through an automation program or by another IMS control region (if RMENV=N).

UOM=
Specifies which unsolicited output messages are sent to the Operations Manager:

MTO
Send only unsolicited output messages that are destined for the MTO, unsolicited output messages that are destined for the system console, or both, to OM. This is the default.

NONE
Do not send any unsolicited output messages to OM.

ALL
Send all unsolicited output messages to OM.

DFSDCxxx member of the IMS PROCLIB data set

Use the DFSDCxxx member of the IMS PROCLIB data set to define data communication options.

You can find an example of the DFSDCxxx member of the IMS PROCLIB data set, DFSDC000, in library SDFSSLIB.

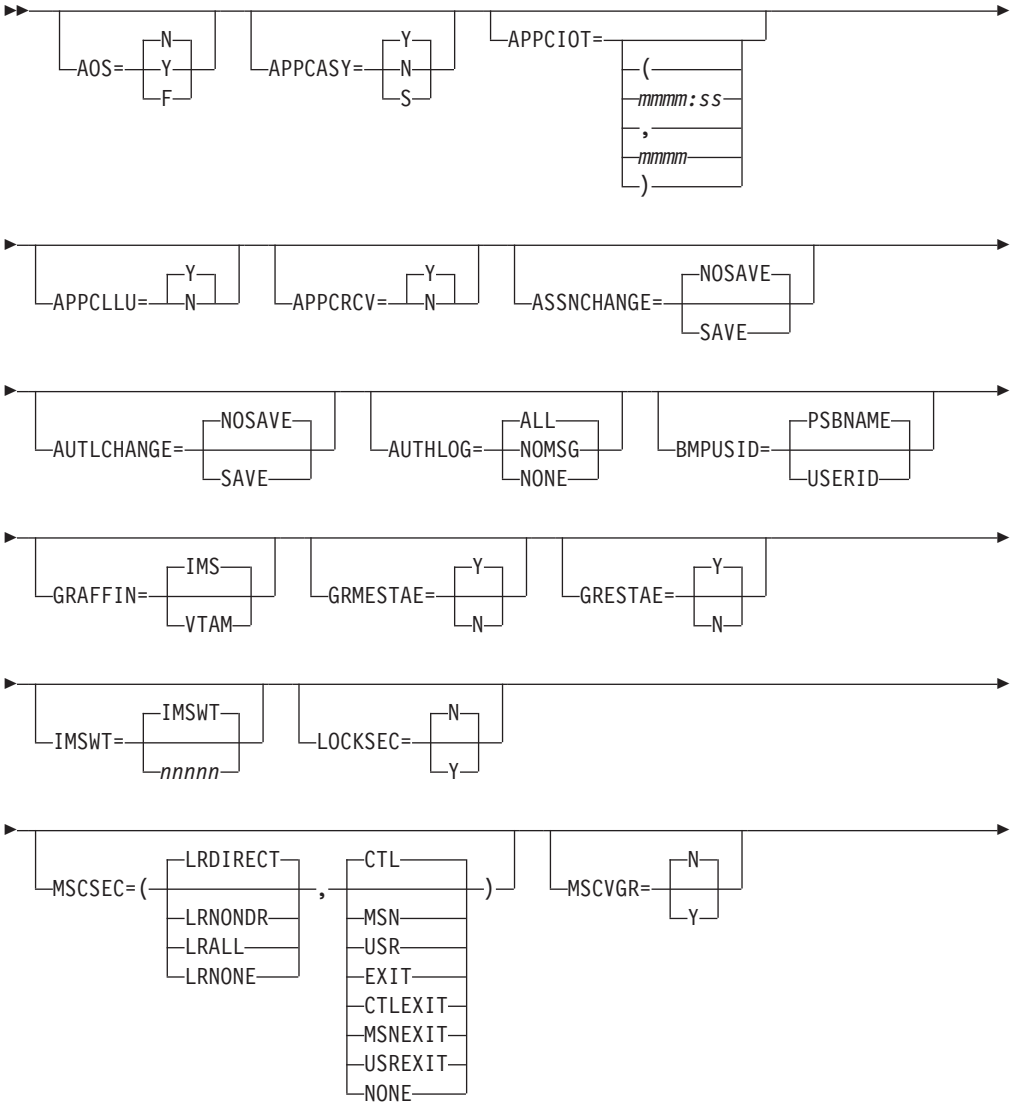
The DC= parameter in the IMS or DCC startup procedure defines the DFSDCxxx PROCLIB member that is used. The default suffix is 000.

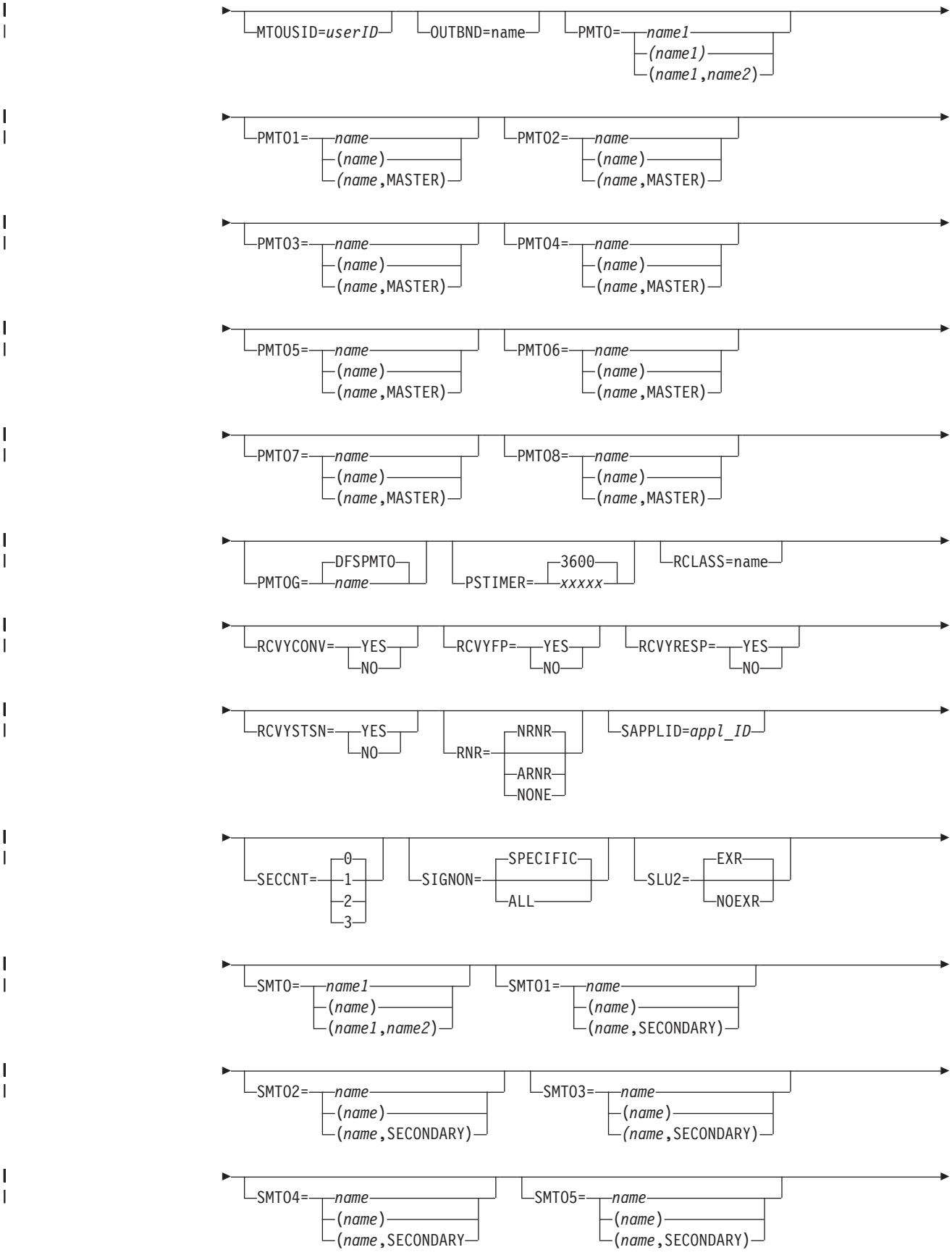
Environments

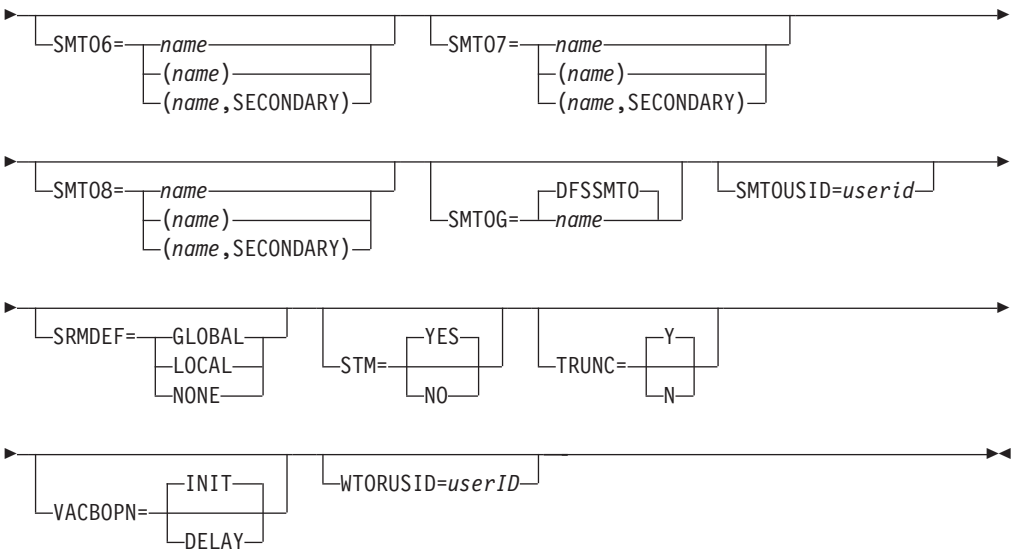
The DFSDCxxx member of the IMS PROCLIB data set applies to the DB/DC and DCCTL environments.

Syntax

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.




MTOUSID=userID
OUTBND=name
PMTO=
name1
(name1)
(name1,name2)
PMTO1=
name
(name)
(name,MASTER)
PMTO2=
name
(name)
(name,MASTER)
PMTO3=
name
(name)
(name,MASTER)
PMTO4=
name
(name)
(name,MASTER)
PMTO5=
name
(name)
(name,MASTER)
PMTO6=
name
(name)
(name,MASTER)
PMTO7=
name
(name)
(name,MASTER)
PMTO8=
name
(name)
(name,MASTER)
PMTOG=
DFSPMTO
name
PSTIMER=
3600
xxxxx
RCLASS=name
RCVYCONV=
YES
NO
RCVYFP=
YES
NO
RCVYRESP=
YES
NO
RCVYSTSN=
YES
NO
RNR=
NRNR
ARNR
NONE
SAPPLID=appl_ID
SECCNT=
0
1
2
3
SIGNON=
SPECIFIC
ALL
SLU2=
EXR
NOEXR
SMTO=
name1
(name)
(name1,name2)
SMTO1=
name
(name)
(name,SECONDARY)
SMTO2=
name
(name)
(name,SECONDARY)
SMTO3=
name
(name)
(name,SECONDARY)
SMTO4=
name
(name)
(name,SECONDARY
SMTO5=
name
(name)
(name,SECONDARY)



Restrictions

If you use the DFSDCxxx member to override a default name (from system definition), do not specify a statically defined lterm or node name.

If you use the DFSDCxxx member to override the master or secondary master node names, do not specify a statically defined node name.

If you use the DFSDCxxx member to override the master or secondary master LTERM names, do not specify a statically defined LTERM name.

If you use the DFSDCxxx member to override the generic LTERM name of the master LTERM or secondary master LTERM, do not specify a statically defined LTERM name.

The node that you specify using the PMTO or SMTO parameters must have the same physical characteristics as the default node that you are overriding. For example, you cannot override a SLU2 device with a non-SNA device.

If you change the PMTO=, SMTO=, PMOTn=, or SMTOn= control statements in the DFSDCxxx member, you must perform a cold start of IMS. You can perform a warm start for changes to any other control statements in the DFSDCxxx member.

Usage

A DFSDCxxx member consists of one or more 80-character records. You code keyword parameters in positions 1-71. Positions 72-80 are ignored. Keywords can contain leading and trailing blanks. If you specify multiple keywords in a record, use commas as delimiters.

Parameters

AOS=

AOS=N
Specifies that shared message queue support for synchronous APPC/OTMA is not active. The default is inactive (N).

AOS=Y
Specifies that shared message queue support for synchronous APPC/OTMA is active (Y). Shared message queue (SMQ) support for APPC/OTMA enables IMSplex users to execute transactions that are originated by APPC or OTMA on a back-end system.

AOS=F
Activates the function by force (F) even if another member in the IMSplex cannot activate the function.

APPC/OTMA SMQ enablement uses z/OS Resource Recovery Services (RRS) multiSystem cascaded transactions to synchronize IMS systems. If RRS is not active on one system, the systems that have specified F (force) still queue incoming transactions without any affinity. Therefore, if a system without RRS tries to process one of these transactions, IMS abends the application with a U711 code.

Important: By default, APPC/OTMA shared message queue support is deactivated for all the members within the IMSplex, except for members where the F (force) option is specified.

APPCASY=
Specifies whether (Y) or not (N) a non-response transaction originating from a program-to-program switch should be scheduled asynchronously or synchronously (S).

If you specify APPCASY=S, the first program-to-program switch that can continue the synchronous conversation is synchronous. Every further program switch is asynchronous.

If you specify APPCASY=S, the OTMAASY value is S. If you specify OTMAASY=S, the APPCASY value is S.

In a shared-queues environment, if you specify APPCASY=S, the program-to-program switch transactions do not have any affinity.

The default is APPCASY=Y.

Recommendation: Use the default specification of APPCASY=Y.

APPCIOT=*(mmmm:ss,mmmm)*
Specifies the APPC timeout values in minutes and seconds. Valid values for *mmmm* are 00 to 1440. Valid values for *ss* are 00 to 59. If APPCIOT=00, there is no timeout detection.

The first parameter specifies the APPC/MVS timeout value. This is the number of minutes or seconds an APPC/MVS service is allowed to wait for completion.

The second parameter specifies the APPC/IMS timeout expression. This is the number of minutes an application is allowed to be inactive and is for synchronous conversations only. Inactive means that the application was not able to respond within the timeout limit.

Note: In some cases, IMS cannot determine if an APPC conversation is synchronous or asynchronous. This situation occurs in program-to-program switches, where IMS cannot predict which program continues the synchronous APPC conversation.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must restart IMS, however, to allow the new specification to take effect.

APPCLLU=
Identifies a local LU to be used for outbound message processing.

N The base LU is used for asynchronous message responses to the IOPCB. This is the default.

Y The incoming local LU is used as the local LU for asynchronous message responses to the IOPCB.

APPCRCV=
Specifies whether IMS should pass the first segment to the DFSNPTR0 or DFSMSCE0 exit. The default is Y. If N is specified, IMS calls the exits before receiving the first segment.

ASSNCHANGE=NOSAVE | SAVE
Sets the default value for the /ASSIGN LTERM|USER TO USER command. SAVE specifies that LTERM assignments (made in ETO) are retained across session and IMS restarts (the user and LTERM control blocks are not deleted). The default is NOSAVE, which indicates that the assignment is not retained and the user and LTERM control blocks are deleted.

You can override the ASSNCHANGE control statement by specifying SAVE or NOSAVE on the /ASSIGN command. If you issue the /ASSIGN command without specifying SAVE or NOSAVE, IMS uses the default value that is specified on the ASSNCHANGE control statement.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

AUTHLOG=ALL | NOMSG | NONE
Determines whether to suppress a RACF ICH408I message to the system console, and whether to turn on SMF logging. This determination takes place after an AUTH call issued by an IMS application program fails authorization.

ALL
IMS turns on SMF logging, and the RACF ICH408I message is issued. ALL is the default.

NOMSG
IMS turns on SMF logging when appropriate, and the RACF ICH408I message is suppressed.

NONE
SMF logging is not turned on, and the RACF ICH408I message is suppressed.

AUTLCHANGE= NOSAVE | SAVE
Sets the default value for the /CHANGE USER AUTOLOGON command. SAVE specifies that autologon changes made in ETO are retained across session and IMS restarts (the user control block are not deleted and remain in effect across a restart or XRF takeover until another /CHANGE command with the NOSAVE parameter is issued). The default is NOSAVE, which indicates that the autologon changes are not retained across session and IMS restarts (the user control block is deleted).

You can override the AUTLCHANGE control statement by specifying SAVE or NOSAVE on the /CHANGE command. If you issue the /CHANGE command without specifying SAVE or NOSAVE, IMS uses the default value that is specified on the ASSNCHANGE control statement.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

BMPUSID=PSBNAME | USERID
Specifies the system option for the value to be placed in the user ID field of the message prefix for a message generated by a non-message driven BMP and the value to be used for the userid for CHNG and AUTH calls made by a non-message-driven BMP.

- If BMPUSID=USERID is specified, the value from the USER= keyword on the JOB statement is used.
- If USER= is not specified on the JOB statement, the program's PSB name is used.
- If BMPUSID=PSBNAME is specified, or if BMPUSID= is not specified at all, the program's PSB name is used.

Note: This parameter is not valid for DBCTL. The value from the USER= keyword on the JOB statement is used for a JBP.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to enable the new specification to take effect.

GRAFFIN=IMS | VTAM
This keyword is obsolete and is ignored.

GRESTAE=Y | N
Specifies whether IMS should bypass the VTAM generic resource logic in the IMS ESTAE exit.

Y indicates that IMS should continue the existing ESTAE logic to delete affinity for all nodes where no status remains before closing the ACF/VTAM ACB.

N indicates that IMS should close the ACF/VTAM ACB immediately to expedite IMS termination, and leave affinity for all nodes set.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

GRMESTAE=Y | N
Indicates whether MSC link affinity should be deleted.

Y During an IMS abend, IMS calls the ESTAE routine in the control region and deletes the affinity setting of any MSC link that does not have a significant status requiring the link session to connect to the same IMS in the IMSplex. This is the default.

N IMS does not delete MSC link affinity in the ESTAE routine.

Significant status for MSC means that the link is in ERE mode. Use the /DIS LINK command to display status.

IMSWT=IMSWT | *nnnnn*
Specifies up to 5 characters that are appended with a 3-character line number

yyy to make up the full JOBNAME (nnnnnyyy) in the /STA REGION jobname for the auto-scheduled spool print utility. If IMSWT= is not specified or if nnnnn is blank, IMS uses the default value of IMSWT (IMSWTyyy).

Requirement: Changing this parameter after a previous IMS start requires a cold start to take effect.

LOCKSEC=Y | N
When password protection is used for the /LOCK and /UNLOCK commands, specifies whether (Y) or not (N) calls to RACF security and the Transaction Authorization exit routine (DFSCTRN0) are made. When LOCKSEC=Y is specified, the command is processed if either of the following is true:

- The resource specified by the command is defined to RACF and the user is authorized to use the resource.
- The resource is not defined to RACF.

If the resource is defined to RACF, but the user is not authorized to use the resource, the command is rejected with the message DFS3689W.

MSCSEC=
Specifies the security options for direct and non-direct routed transactions received from an MSC link.

LRDIRECT
Specifies that Link Receive routing use RACF, the Transaction Authorization exit routine (DFSCTRN0), or both to check security for direct routed transactions. This is the default.

LRNONDR
Specifies that Link Receive routing use RACF, the Transaction Authorization exit routine (DFSCTRN0), or both to check security for non-direct routed transactions. No security checking is performed for direct routed transactions.

LRALL
Specifies that Link Receive routing use RACF, the Transaction Authorization exit routine (DFSCTRN0), or both to check security for direct and non-direct routed transactions.

LRNONE
Specifies that the IMS subsystem does not request any security checking for either type of transaction.

CTL
Specifies that RACF security is to authorize the use of the transaction based upon the user ID of the IMS control region.

MSN
Specifies that RACF security is to authorize the use of the transaction based upon the user ID of the MSNAME.

USR
Specifies that RACF security is to authorize the use of the transaction based upon the user ID of the inputting terminal.

EXIT
Specifies that authorization is performed by the Transaction Authorization exit routine (DFSCTRN0).

CTLEXIT
Specifies that both RACF security and the Transaction Authorization exit routine (DFSCTRN0) are used to authorize the IMS control region user ID to use the transaction.

MSNEXIT
Specifies that both RACF security and the Transaction Authorization exit routine (DFSCTRN0) are used to authorize the MSNAME to use the transaction.

USREXIT
Specifies that both RACF security and the Transaction Authorization exit routine (DFSCTRN0) are used to authorize the user ID of the inputting terminal to use the transaction.

NONE
Specifies that no security authorization checking is performed.

MSCVGR=N | Y
Specifies whether MSC links use the GRSNAME or the APPLID name in an IMS VGR environment.

Y IMS attempts to use the GRSNAME first to establish the session with the remote IMS. The remote IMS must use the value of the MSPLINK NAME=nodename for the IMSplex GRSNAME.

N The APPLID name, not the GRSNAME, is used in an IMS VGR environment. This is true also if the remote IMS uses the IMSplex nodename. This is the default.

MTOUSID=
Specifies a user ID that IMS uses if the primary MTO does not sign on for transaction authorization checking.

If you do not specify MTOUSID, the control region's user ID is used for any other terminal that does not sign on.

OUTBND=
Specifies a different outbound LU. The specified LU must be one of the APPC LUs defined in the APPCPMxx member of the SYS1.PARMLIB library. The default outbound LU is BASE LU.

PMTO=*name1* **|** **(***name1***)** **|** **(***name1***,***name2***)**
name1 specifies the node name of the primary master terminal (PMTO). This name overrides the PMTO node name that is specified during IMS system definition.

name2 specifies the name of the additional PMTO resource that exists in an XRF environment. If *name2* is specified in a non-XRF environment, it is ignored.

If the PMTO parameter is not specified, the default PMTO name (either one or both) that was specified during IMS system definition is used.

Requirement: Changing this parameter after a previous IMS start requires a cold start to take effect.

PMTO1-8=*name* **|** **(***name***)** **|** **(***name***,MASTER)**
Specifies the first through the eighth LTERM names of the PMTO. The names you specify here override the first through the eighth LTERM names specified during IMS system definition. The LTERM name specified with the MASTER

designation becomes the PMTO LTERM. These LTERM names must be unique for each IMS system within a shared IMS environment.

Requirement: Changing this parameter after a previous IMS start requires a cold start to take effect.

PMTOG=DFSPMTO | *name*
Specifies the generic LTERM name of the PMTO. The default is DFSPMTO. Use the same generic LTERM name for each IMS system within a shared IMS environment.

Requirement: Changing this parameter after a previous IMS start requires a cold start to take effect.

PSTIMER=3600 | *xxxxx*
When Rapid Network Reconnect (RNR) is used, the PSTIMER= parameter specifies a one- to five-digit number indicating the number of seconds IMS can wait pending recovery for persistent sessions following an IMS or ACF/VTAM failure. The restarted IMS must successfully issue an OPEN ACB before the timer expires or the sessions are terminated. Valid values are from zero (no timer) to 86,400 seconds (24 hours). The IMS default is 3600 seconds (1 hour). If a null or invalid value is specified, IMS uses 3600.

When an ACB name is specified on the MNPS= keyword (in the PROCLIB member DFSHSBxx or DFSPBxxx, or in the IMS procedure), specifies the amount of time before the XRF alternate must successfully open the MNPS ACB. This applies to an XRF takeover only. It does not apply to an emergency restart because persistent sessions are not restored following a restart in an XRF system using MNPS.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

RCLASS=
Specifies an identifier of 1 - 7 alphanumeric characters that is to be used to identify the IMS system as a resource class to RACF for transaction authorization and user ID verification. The default identifier is “IMS”.

IMS class names do not have to be unique. If you do make them unique, you can, for example, define the same transaction name on a production subsystem and a test subsystem, with different access lists for each subsystem, with no ambiguity.

The specification is valid only if one of the following statements is true:

- TYPE=RACFTERM is specified on the stage-1 system definition SECURITY macro; or RCF=A, RCF=T, or RCF=Y is specified on the IMS procedure.
- TYPE=RASRACF or TYPE=RAS is specified on the SECURITY macro; or ISIS=R or ISIS=A is specified on the IMS procedure.

The RCLASS= specification on the DFSPBxxx PROCLIB member overrides the RCLASS= specification on both the SECURITY macro and the DFSDCxxx member. The RCLASS= specification on the DFSDCxxx PROCLIB member overrides the RCLASS= specification on the SECURITY macro.

RCVYCONV=
Specifies whether the status of a conversation can be recovered (YES) or not (NO). RCVYCONV applies to conversation status, not to output messages.

Even if the conversation status is not recovered, conversation output continues to be recoverable and is delivered asynchronously.

The following table describes the default values for RCVYCONV as they relate to the values specified on the SRMDEF keyword.

Table 67. RCVYCONV values related to SRMDEF values

SRMDEF value	RCVYCONV=YES	RCVYCONV=NO
SRMDEF=GLOBAL	Valid (default)	Valid
SMRDEF=LOCAL	Valid (default)	Valid
SRMDEF=NONE	Invalid	Valid (default)

If RCVYCONV is specified incorrectly, message `DFS1920I` is issued and IMS uses the appropriate default from Table 67.

RCVYFP=

Specifies whether the status of Fast Path can be recovered (YES) or not (NO). RCVYFP applies to Fast Path status and output.

The following table describes the default values for RCVYFP as they relate to the values specified on the SRMDEF keyword.

Table 68. RCVYFP values related to SRMDEF values

SRMDEF value	RCVYFP=YES	RCVYFP=NO
SRMDEF=GLOBAL	Valid (default)	Valid
SRMDEF=LOCAL	Valid (default)	Valid
SRMDEF=NONE	Invalid	Valid (default)

Restrictions: If SRMDEF=LOCAL, STSN recoverability (RCVYSTSN) and fast path recoverability (RCVYFP) must be the same. They must both specify YES or both specify NO.

If RCVYFP is specified incorrectly, message `DFS1920I` is issued and IMS uses the appropriate default from Table 68.

Related reading: For a discussion on the recoverability of Fast Path status, see Fast Path recovery (Communications and Connections).

RCVYRESP=

Specifies whether the status of full-function response mode can be recovered (YES) or not (NO). RCVYRESP applies to full-function response mode status, not to output messages. Even if full-function response mode status is not recovered, response mode output continues to be recoverable and is delivered asynchronously.

The following table describes the default values for RCVYRESP as they relate to the values specified on the SRMDEF keyword.

Table 69. RCVYRESP values related to SRMDEF values

SRMDEF value	RCVYRESP=YES	RCVYRESP=NO
SRMDEF=GLOBAL	Invalid	Valid (default)
SRMDEF=LOCAL	Valid	Valid (default)

Table 69. RCVYRESP values related to SRMDEF values (continued)

SRMDEF value	RCVYRESP=YES	RCVYRESP=NO
SRMDEF=NONE	Valid. However, only those terminals or users that are assigned SRM=LOCAL as an override to SRMDEF=NONE (through user descriptor or user exit) are eligible to recover full-function response mode. All other terminals and users use RCVYRESP=NO.	Valid (default)

If RCVYRESP is specified incorrectly, message DFS1920I is issued, and IMS uses the appropriate default from Table 69 on page 666.

Restrictions:

- RCVYRESP=YES is invalid when IMS is using an RM resource structure to maintain TM resources (that is, when STM=YES in the DFSDCxxx member of the IMS PROCLIB data set).
- RCVYRESP=YES applies only to terminals or users that have a status recovery mode of LOCAL. If a terminal or user has a status recovery mode of GLOBAL or NONE, the terminal or user is automatically assigned RCVYRESP=NO.
- Full-function response mode is not recovered following an IMS shut down or failure. After an IMS cold start, warm start, or emergency restart, the node or user is no longer in full-function response mode. Full-function response mode is recovered following an XRF takeover for class-1 terminals only.
- The RCVYRESP value specified cannot be overridden for individual users or nodes using either ETO user descriptors, the user logon exit (DFSLGNX0), or the user signon exit (DFSSGNX0). Therefore, RCVYRESP=YES automatically applies to all terminals and users with a LOCAL status recovery mode.

RCVYSTSN=

Specifies whether the status of STSN terminals (SLUP, FINANCE, and ISC) can be recovered (YES) or not (NO).

The following table describes the default values for RCVYSTSN as they relate to the values specified on the SRMDEF keyword.

Table 70. RCVYSTSN values related to SRMDEF values

SRMDEF value	RCVYSTSN=YES	RCVYSTSN=NO
SRMDEF=GLOBAL	Valid (default)	Valid
SRMDEF=LOCAL	Valid (default)	Valid
SRMDEF=NONE	Invalid	Valid (default)

Restrictions: If SRMDEF=LOCAL, STSN recoverability (RCVYSTSN) and fast path recoverability (RCVYFP) must be the same. They must both specify YES or both specify NO. If SRMDEF=GLOBAL, Fast Path is always nonrecoverable, so if there is a failure during the execution of a fast path transaction, the STSN session must be restarted cold.

If RCVYSTSN is specified incorrectly, message DFS1920I is issued and IMS uses the appropriate default from Table 70.

RNR=NRNR | ARNR | NONE
Specifies that IMS should enable Rapid Network Reconnect (RNR) support for ACF/VTAM persistent sessions as defined on the ACF/VTAM APPL statement for IMS. This parameter is used as a system default for an unspecified RNR value for the OPTIONS= keyword at the terminal level.

ARNR
Specifies that a basic session reconnect should be done automatically by IMS following an IMS or ACF/VTAM restart. This level of session reconnect is to a data traffic reset point equivalent to logon without available signon data. A DFS3649 signon required message can be sent as appropriate for the terminal type and definition.

NRNR
Specifies that automatic session reconnect should not be done. The session is terminated following restart of IMS or ACF/VTAM.

NONE
Specifies that the RNR function should not be activated. This option can also be used to deactivate the RNR function if it was activated before the IMS restart.

Requirement: Changing this parameter after a previous IMS start does not require a cold start, although a cold start is recommended in order to clear persistent session data in VTAM and terminate IMS sessions that are waiting for an RNR reconnection. You must perform at least a warm start to allow the new specification to take effect.

SAPPLID=*appl_ID*
Specifies the application ID to be used when calling RACF, or equivalent product, during sign on. If SAPPLID= is not specified, the IMSID of that IMS is the default application ID. The application ID becomes important when Passtickets are used instead of passwords because the application ID is used in the RACF creation and verification of the Passticket.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

SECCNT=0 | 1 | 2 | 3
Specifies the maximum number of terminal and password security violations to be accepted per physical terminal and the number of transaction command violations per transaction prior to master terminal notification of such violations. The default is 0, which nullifies notification to the master terminal. The number specified must be 0, 1, 2, or 3.

SIGNON=SPECIFIC | ALL
Specifies that either all (ALL) or specific (SPECIFIC) static terminals are required to sign on.

ALL
Specifies that all static terminals (except the master terminal) are required to sign on. The following devices are not required to sign on when SIGNON=ALL is specified:

- LU6.1 devices
- 3284/3286 devices
- SLU1 printer-only devices
- Master terminals

To require master terminals to sign on, specify the OPTIONS=SIGNON on the TYPE or TERMINAL macro.

SPECIFIC
Specifies that individual static terminals might be required to sign on. These terminals are specified in system definition with the OPTIONS=SIGNON parameter. SPECIFIC is the default.

SLU2=EXR | NOEXR
Specifies whether all IMS-defined SLU2 terminals should be treated as with or without DFT architecture.

EXR Specifies that all IMS-defined SLU2 terminals are treated as with DFT architecture. EXR is the default.

NOEXR
Suppresses the SNA exception response that proceeds an IMS error message. As a result of no exception response, the terminal PROG check is prevented, the keyboard is locked, and IMS is allowed to send the error message.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

SMTO=*name1* | (*name1*) | (*name1*,*name2*)
name1 specifies the node name of the secondary master terminal (SMTO). This name overrides the SMTO node name that is specified during IMS system definition.

name2 specifies the name of the additional SMTO resource that exists in an XRF environment. If *name2* is specified in a non-XRF environment, it is ignored.

If the PMTO parameter is not specified, the default SMTO name (either one or both) that was specified during IMS system definition is used.

Requirement: Changing this parameter after a previous IMS start requires a cold start to take effect.

SMTOUSID=*userid*
Specifies a user ID that IMS will use if the secondary MTO does not sign on for transaction authorization checking.

SMTO1-8=*name* | (*name*) | (*name*,MASTER)
Specifies the first through the eighth LTERM names of the SMTO. The names you specify here override the first through the eighth LTERM names specified during IMS system definition. The LTERM name specified with the SECONDARY designation becomes the SMTO LTERM. These LTERM names must be unique for each IMS system within a shared IMS environment.

Requirement: Changing this parameter after a previous IMS start requires a cold start to take effect.

SMTOG=DFSSMTO | *name*
Specifies the generic LTERM name for the secondary master terminal. The default generic SMTO LTERM name is DFSSMTO. Use the same generic LTERM name for all IMS systems within a shared IMS environment.

Requirement: Changing this parameter after a previous IMS start requires a cold start to take effect.

SRMDEF=

Specifies the status recovery mode for user and node resources. Valid values are GLOBAL, LOCAL, and NONE.

Related reading: In order to understand completely what the GLOBAL, LOCAL, and NONE values mean, you have to first understand how IMS classifies resource status. For a discussion of these concepts, see *IMS Version 11 System Administration*.

The default value of SRMDEF depends on the operating environment:

- GLOBAL is the default if a resource structure and shared queues are used.
- LOCAL is the default if a resource structure and shared queues are not used.

The following list describes the valid values for SRMDEF.

GLOBAL

The significant status of a resource is saved globally in the coupling facility resource structure every time the significant status changes, along with all other recoverable status for that resource. The resource status is restored at the next logon or sign on and is available from any IMS system in the IMSplex. Resource status is copied to the local system when that resource becomes active, but is deleted from the local system when it becomes inactive. Status is not recovered in local IMS log records.

Requirement: Using a status recovery mode of GLOBAL requires a Resource Manager and a resource structure in the coupling facility. If STM=NO is specified, SRMDEF=GLOBAL is invalid.

LOCAL

The significant status is saved in local control blocks and log records, along with all other recoverable resource status. The resource status is restored at the next logon or sign on if the user or node returns to the same local IMS system. This status is not available on any other IMS system in the IMSplex.

LOCAL mode enforces an affinity for the user or node to the IMS system where the local status exists. This is referred to as RM affinity. IMS prevents the user or node from accessing any other IMS while local status exists. If the IMS system that has local status fails, the node and user are allowed access to another IMS system. In this case, the node or user does not recover any status and when the failed IMS restarts, the local status that existed is deleted.

Using a status recovery mode of LOCAL does not require a Resource Manager. If RM is not used to manage node or user resources, then no RM affinity is enforced.

NONE

The significant status is not saved by RM or in local log records. When the user signs on or the node logs on, the corresponding significant status does not exist.

STM=YES | NO

Specifies whether (YES) or not (NO) IMS is to use the RM resource structure to manage TM resources. The resources that are managed include APPC descriptors, VTAM LTERMS, MSNAMEs, VTAM terminal nodes, users, and user IDs. The default is YES.

A cold start is required if STM=NO is specified for installations that were previously using the resource structure.

Any change to the STM= specification requires a cold start of the TM component (COLDCOMM). This restriction is enforced during restart processing through the U233 ABENDU0233.

Note that transactions (both statically defined and CPI-C) are maintained in the resource structure independent of TM resources, regardless of the STM= specification.

If STM= is specified, but IMS is not using a resource structure, the specification is ignored.

TRUNC=Y | N

Specifies the system default for the truncated data options used for IMS conversational transactions. If a transaction definition (TRANSACT) does not specify a truncated data option, the system default is used when a conversation starts for the first time.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

VACBOPN=INIT | DELAY

Specifies that the opening of the IMS VTAM ACB is delayed until a /START DC command is issued. This parameter is optional. The following is an example of using the VACBOPN parameter.

- An IMS system is restarted (after an emergency restart) and this system includes many Automated Teller Machines (ATMs). All the ATMs attempt to connect immediately. The logon attempts are rejected by VTAM until IMS opens the VTAM ACB.

  If, however, the VTAM ACB is opened during initialization, VTAM begins queuing the logon requests. The programs in the ATMs wait for a short period of time for a response and then time out. When the timeout occurs, an UNBIND request is sent and this is also queued. This scenario continues until the logon is successful. Meanwhile, IMS has to complete its initialization and then process all the log records for the emergency restart.

  After the successful completion of the emergency restart, the operator can issue a /START DC command to tell VTAM to start accepting logon requests. When this finally happens, there are many UNBIND requests on the queue. At this point, the logon requests are queued behind enough error processing that they also time out.

  In some situations, apparently the IMS system is looping (because no sessions can get connected) and the operator is tempted to terminate IMS. Using the VACBOPN=DELAY parameter alleviates this problem by delaying the queuing of the logon requests until IMS is ready to start accepting the logon requests (when the /START DC command is issued).

INIT Specifies that the IMS VTAM ACB is opened when IMS initializes.

DELAY
Specifies whether the IMS VTAM ACB is opened when a /START DC command is issued.

Requirement: Changing this parameter after a previous IMS start does not require a cold start. You must perform a warm start, however, to allow the new specification to take effect.

WTORUSID=

Specifies a user ID that IMS uses if the WTOR (system console) does not sign on for transaction authorization checking.

If you do not specify WTORUSID, the control region's user ID is used for any other terminal that does not sign on.

If both SIGNON=ALL and WTORUSID= are specified, the specification for WTORUSID= overrides the SIGNON=ALL specification for the system console.

Related concepts:

Chapter 14, “IMS Syntax Checker,” on page 283

DFSDFxxx member of the IMS PROCLIB data set

The DFSDF*xxx* member of the IMS PROCLIB data set consolidates the specification of processing options for many different IMS components and functions.

The parameters in the DFSDF*xxx* member are organized into sections. Each section has a different purpose and begins with a predefined header that identifies the section. Each section supports different parameters and syntax.

In the sections of the DFSDF*xxx* member you can specify processing options for the following IMS components and functions:

- Databases
- Dynamic resource definition (DRD)
- Exit routines
- The Fast Path 64-bit buffer manager
- The IMS abend search and notification procedure
- The IMS Common Service Layer (CSL)
- Shared queues

All the sections within the DFSDF*xxx* member of the IMS PROCLIB data set are optional. If you include a section for any of the functions, the section must begin with the appropriate predefined header.

A DFSDF*xxx* member of the IMS PROCLIB data set consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be specified in the remaining columns in free format, and they can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a “*” or “#” in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between “ /* ”and “*/”, for example, /* PROCLIB comments */. Values that are specified in this member of the IMS PROCLIB data set are case-sensitive. In general, use uppercase for all parameters.

You can use the IMS Syntax Checker to modify the DFSDF*xxx* member of the IMS PROCLIB data set.

Related concepts:

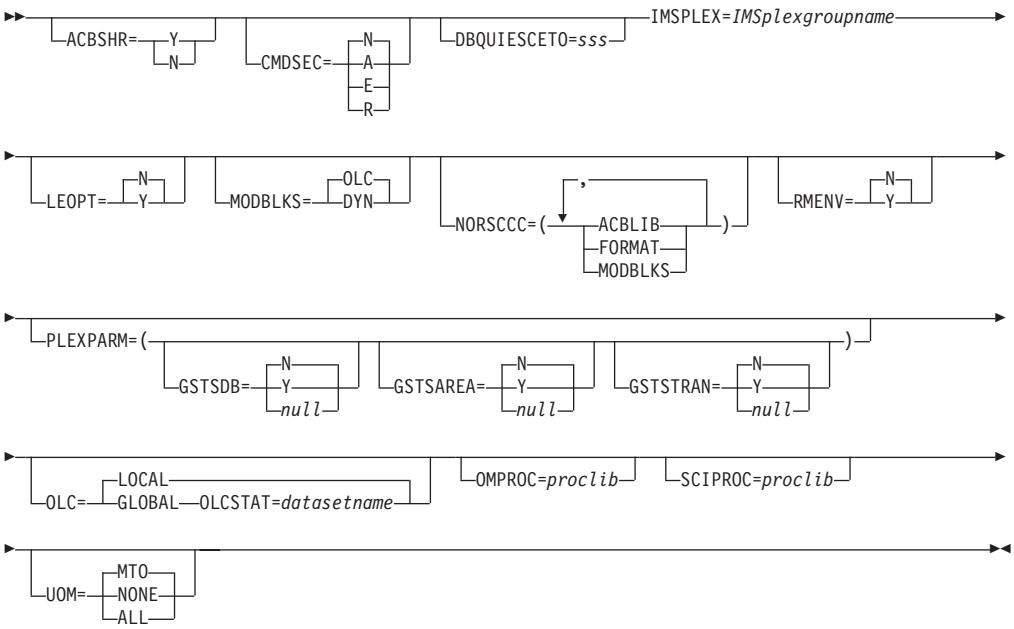
Chapter 14, “IMS Syntax Checker,” on page 283

COMMON_SERVICE_LAYER section of the DFSDFxxx member

The COMMON_SERVICE_LAYER section of the DFSDFxxx member of the IMS PROCLIB data set specifies options for the Common Service Layer (CSL), such as IMSplex name, global online change, command authorization checking, OLCSTAT, DRD, and global resource status. The section is defined by the header <SECTION=COMMON_SERVICE_LAYER>.

Syntax

The syntax of the parameters that are specified for the CSL in the DFSDFxxx member of the IMS PROCLIB data set is the same as the syntax of the parameters specified for CSL in the DFSCGxxx member of the IMS PROCLIB data set.



Parameters

All of the parameters that are valid in the DFSCGxxx member of the IMS PROCLIB data set are valid in the DFSDFxxx member of the IMS PROCLIB data set. If you specify values in both the DFSCGxxx member of the IMS PROCLIB data set and the CSL section of the DFSDFxxx member of the IMS PROCLIB data set, the values specified in the DFSCGxxx member override the values specified in the DFSDFxxx member.

ACBSHR=

Specifies whether the ACB libraries are shared among the IMS systems that are listed in the OLCSTAT data set. This parameter can be specified in both the DFSCGxxx and DFSDFxxx members of the IMS PROCLIB data set. If it is specified in both members, IMS uses the ACBSHR value that is specified in the DFSCGxxx member of the IMS PROCLIB data set.

Y Specifies that all the IMS systems that are listed in the OLCSTAT data set use the same ACB library.

N Specifies that each IMS that is listed in the OLCSTAT data set uses its own dedicated ACB library.

Each IMS system sharing the OLCSTAT data set must have the same value for ACBSHR=. If the ACBSHR= values are not the same, the `INIT OLC PREPARE TYPE(ACBMBR)` command is rejected.

CMDSEC=
Specifies whether IMS should perform security checking on commands that are routed from OM, and if so, whether RACF or the Command Authorization exit routine (or both) should be used. If IMS performs the security checking on commands, the CMDSEC keyword also specifies whether RACF, the IMS Command Authorization exit routine (DFSCCMD0), or both should be used. This option applies only to IMS type-1 commands that can be issued through the OM API.

For type-2 commands (that are only allowed through the OM API, not allowed from the IMS master terminal), all security checking is performed by OM and is controlled by the CMDSEC= parameter on the OM startup JCL or on the OM initialization member of the IMS PROCLIB data set (CSLOIxxx).

Related reading: For all the details of the CSLOIxxx member of the IMS PROCLIB data set, see "CSLOIxxx member of the IMS PROCLIB data set" on page 636.

Recommendation: Use OM command security instead of IMS security. By allowing OM to perform the security checks, commands which fail security authorization are not routed to IMS, which reduces processing overhead and network traffic. When IMS command security is used, you must ensure that all IMS systems use the same security profiles and user exits. If IMS systems in the same IMSplex use different security rules, the results of command security checking may be unpredictable.

A Specifies that both RACF and DFSCCMD0 are to be called (options E and R). RACF is called first. Then the security authorization facility (SAF) return code, RACF return code, and RACF reason code are passed to DFSCCMD0.

Related reading: For complete information about the Command Authorization exit routine, see *IMS Version 11 Exit Routines*.

E Specifies that the Command Authorization exit routine is to be called for command authorization.

N Specifies that no authorization checking is to be done. Security checking can still be performed by OM depending on the specification of the OM CMDSEC= parameter.

R Specifies that RACF is to be called for command authorization.

DBQUIESCETO=
The amount of time, in seconds, that the QUIESCE command waits for currently running applications to commit before the QUIESCE command is aborted. This value applies only when the quiesce function is initiated. The actual time required for the QUIESCE command varies and might exceed this timeout value. The default value is 30 seconds. *sss* must be a non-zero value 1 - 999. You can also specify this timeout value in the COMMON_SERVICE_LAYER section of the DFSDFxxx member of the IMS PROCLIB data set, or override the value using the appropriate UPDATE command for the resource type, for example, UPDATE DATAGRP NAME(name) START(QUIESCE) SET(TIMEOUT(10)).

IMSPLEX=
Specifies a one- to five-character identifier. IMS appends this identifier to

"CSL" to create the IMSplex group name. All IMSplex members (OM, RM, IMS, CQS) that are in the same IMSplex sharing group sharing either databases or message queues must specify the same identifier. IMS has no way to enforce this; the user is responsible for ensuring that the IMSplex name is correctly specified. The same identifier must also be used for the IMSplex= parameter in the CSLSIxxx, CSLOIxxx, CSLRIxxx, and CQSIPxxx members of the IMS PROCLIB data set.

When RMENV=N, IMSPLEX= specifies a one- to five-character name of the IMSplex that OM, SCI, and the IMS control regions join. This IMSplex is simply a grouping of all address spaces that are managed by one or more OMs.

This parameter is required.

LEOPT=Y | N
Specifies whether IMS should allow IBM Language Environment for z/OS (LE) dynamic runtime parameter overrides.

MODBLKS=
Specifies whether resources in the IMS.MODBLKS data set are defined dynamically or by online change. These resources include databases, programs, routing codes, and transactions. This attribute can only be changed on an IMS cold start. If the value is changed for the next IMS warm or emergency restart, IMS restart terminates, and abend U0168 is issued.

DYN
Enables dynamic definition for resources in the IMS.MODBLKS data set. Resource definitions can be dynamically added, changed, or deleted by using online commands, including CREATE, DELETE, and UPDATE. Online change for resources in the IMS.MODBLKS data set that is initiated by a /MODIFY PREPARE MODBLKS or INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS) command is not allowed.

OLC
Enables online change for resources in the IMS.MODBLKS data set. Resource definitions can be added, changed, or deleted by using online change. Online change commands for resources in the IMS.MODBLKS data set include /MODIFY PREPARE MODBLKS or INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS). When MODBLKS=OLC, UPDATE commands that update runtime values (such as a transaction class) are permitted. This is the default.

NORSCCC=()

Resource consistency checking is no longer performed for ACBLIB, FMTLIB, and MODBLKS libraries, regardless of whether there is a resource structure or if values are specified on NORSCCC. The NORSCCC keyword is supported for compatibility, but its values are ignored.

Specifies that no resource consistency checking is to be performed for the specified resources. The resources to not check for consistency must be enclosed within parentheses and separated by commas. The NORSCCC keyword must precede the left parenthesis. NORSCCC is an optional parameter.

If a resource structure is defined for the IMSplex, by default, consistency checking is performed for ACBLIB, FORMAT, and MODBLKS. Consistency checking is performed on the data set names; it is not performed on the

resources that reside in the libraries. Resource consistency checking is useful for cloned systems that either share all these data sets or use the same data set names on each IMS.

If no resource structure is defined, no resource definition consistency checking is performed.

When RMENV=N, the NORSCCC parameter is ignored.

Specify one or more of the following parameters:

ACBLIB
The ACBLIB data set names are not checked for consistency. The IMS systems in the IMSplex do not need to define the same data sets for the ACBLIB library. ACBLIB applies only if OLC=GLOBAL is specified.

FORMAT
The FORMAT data set names are not checked for consistency. The IMS systems in the IMSplex do not need to define the same data sets for the FORMAT library. FORMAT applies only if OLC=GLOBAL is specified.

MODBLKS
The MODBLKS data set names are not checked for consistency. The IMS systems in the IMSplex do not need to define the same data sets for the MODBLKS library. MODBLKS applies only if OLC=GLOBAL is specified.

OLC=LOCAL | GLOBAL
Specifies the scope of the online change. OLC=LOCAL means that the online change applies locally to each IMS. Local online change is prepared and committed by using the `/MODIFY PREPARE` and `/MODIFY COMMIT` commands on each local IMS. The local online changes must be manually coordinated across an IMSplex. OLC=GLOBAL means that the online change is coordinated across the IMSplex; ACBLIB member OLC requires OLC=GLOBAL. Global online change is prepared and committed by using the `INITIATE OLC` command. If global online change is enabled and a resource structure is defined, the MODBLKS, FORMAT, and ACBLIB data sets must be consistent across the IMSplex, unless resource consistency checking is omitted by the NORSCCC keyword.

Online change across an IMSplex (OLC=GLOBAL) is not supported for RSR tracking IMS systems. If specified, IMS abends during restart with ABENDU2801 subcode X'0013'.

OLCSTAT=
Specifies the 1 - 44 character data set name of the OLCSTAT data set. OLCSTAT is ignored if OLC=LOCAL is specified. OLCSTAT is required if OLC=GLOBAL is defined. The OLCSTAT data set is a cataloged BSAM data set that contains global online change information and status. The data set name is used to dynamically allocate the data set when an IMS initialized, restarts, or is master of an online change phase.

All IMS systems in an IMSplex must refer to the same physical OLCSTAT data set. If a resource structure is defined to the IMSplex, IMS ensures that the OLCSTAT data set name is defined consistently for the IMSplex. If the OLCSTAT data set name definition is not consistent with the OLCSTAT defined to other IMS systems in the IMSplex, IMS initialization fails.

RMENV=Y | N
Indicates whether (Y) or not (N) IMS requires an RM environment.

Y Indicates that IMS requires an RM environment in order to use Resource

Manager services. IMS initialization does not complete until IMS successfully registers with RM. This is the default.

N Indicates that IMS does not require an RM environment and does not use Resource Manager services. IMS does not attempt to register with RM even if an RM address space is active. When RMENV=N, you do not need to define or start an RM address space. IMS commands and functions that require RM are not available in this environment.

When RMENV=N, online change is as follows:

- If OLC=LOCAL, use the /MODIFY command to initiate online change. The MODSTAT data set must be defined.
- If OLC=GLOBAL, use the INIT OLC command to initiate online change. The OLCSTAT data set must be defined; however, it cannot be shared between IMS systems.

You can change the value of RMENV= across IMS warm or cold starts.

OMPROC=

Specifies the one- to eight-character name of the member of the IMS PROCLIB data set that contains the procedure for the OM address space. This parameter is only valid when RMENV=N is specified. If RMENV=Y is specified, this parameter is ignored. This parameter is optional.

IMS does not start the OM address space specified by this parameter if another OM address space is already active in the IMSplex during IMS initialization.

If OMPROC= is not specified, start the OM address space before starting IMS. This start sequence can be performed through an automation program or by another IMS control region (if RMENV=N is specified).

PLEXPARM=()

PLEXPARM is optional. If you do not specify it, either the default value or a NULL value is used. Refer to the specific subparameter to determine whether the default or NULL value is used.

The PLEXPARM parameter defines how global status is maintained for resources in an IMSplex. An IMS system that is initialized in an IMSplex checks to see if a global PLEXPARM entry exists in the RM resource structure. If the entry exists, the IMS system uses the values. If no entry exists, the initializing IMS writes its resource definition values to the RM resource structure global PLEXPARM entry. Any IMS system that initializes in that IMSplex after that uses the values stored by the first IMS system. If the subsequent IMS system has PLEXPARM values that are different from the values in the global PLEXPARM entry, message DFS3425I is issued, and values from the global PLEXPARM entry are used.

To view the current PLEXPARM values, use the QUERY IMS command. To update PLEXPARM values, use the UPDATE IMS command.

If an IMS system is running in an IMSplex that does not include an RM, or the RM in the IMSplex does not use a resource structure, the PLEXPARM parameter is ignored. No message is issued indicating that the parameter is ignored. All PLEXPARM values are set to no (N).

In an RSR or FDBR system, the PLEXPARM parameter is ignored. No message is issued indicating that the parameter is ignored. All PLEXPARM values are set to no (N). However, an FDBR system and an XRF alternate system do

maintain global PLEXPARM values internally. You can use the QUERY IMS SHOW(PLEXPARM) command to display PLEXPARM values in an FDBR or XRF alternate system.

GSTSDB=
Specifies how global status for databases is kept in the IMSplex.

N No global status is maintained for databases resources in the RM. This is the default value for DB/DC and DBCTL regions.

Y Global status for databases is maintained in the RM.

null
No global status is maintained for databases resources in the RM. This is the default value for DCCTL regions.

A DCCTL system that initializes this value in the global PLEXPARM entry writes a null value. If a DB/DC or DBCTL system later joins that IMSplex, that system updates the GSTSDB value in the global PLEXPARM entry with the value from its PLEXPARM initialization parameter.

GSTSAREA=
Specifies how global status for DEDB areas is kept in the IMSplex.

N No global status is maintained for DEDB area resources in the RM. This is the default value for DB/DC and DBCTL regions.

Y Global status for DEDB areas is maintained in the RM.

null
No global status is maintained for DEDB area resources in the RM. This is the default value for DCCTL regions.

A DCCTL system that initializes this value in the global PLEXPARM entry writes a null value. If a DB/DC or DBCTL system later joins that IMSplex, that system updates the GSTSDB value in the global PLEXPARM entry with the value from its PLEXPARM initialization parameter.

GSTSTRAN=
Specifies how global status for transactions is kept in the IMSplex.

N No global status is maintained for transaction resources in the RM. This is the default value for DB/DC and DCCTL regions.

Y Global status for transactions is maintained in the RM.

null
No global status is maintained for transaction resources in the RM. This is the default value for DBCTL regions.

A DBCTL system that initializes this value in the global PLEXPARM entry writes a null value. If a DB/DC or DCCTL system later joins that IMSplex, that system updates the GSTSDB value in the global PLEXPARM entry with the value from its PLEXPARM initialization parameter.

SCIPROC=
Specifies the one- to eight-character name of the member of the IMS PROCLIB data set that contains the procedure for the SCI address space. This parameter is only valid when RMENV=N. If RMENV=Y is specified, this parameter is ignored. This parameter is optional.

If SCIPROC= is not specified, start the SCI address space before starting IMS. This start sequence can be performed through an automation program or by another IMS control region (if RMENV=N).

UOM=
Specifies which unsolicited output messages are sent to the Operations Manager:

MTO
Send only unsolicited output messages that are destined for the MTO, unsolicited output messages that are destined for the system console, or both, to OM. This is the default.

NONE
Do not send any unsolicited output messages to OM.

ALL
Send all unsolicited output messages to OM.

Example of the COMMON_SERVICE_LAYER section of the DFSDF*xxx* member

```
/*******************************************************************/
/* Common Service Layer Section                                    */
/*******************************************************************/
<SECTION=COMMON_SERVICE_LAYER>
ACBSHR=Y                                  /* Share ACB libraries */
CMDSEC=N                          /* No cmd authorization checking */
IMSPLEX=PLEX1                                  /* IMSplex name */
OLC=GLOBAL                                /* GLOBAL online change */
OLCSTAT=IMSTESTS.IMS01.OLCSTAT           /* OLCSTAT data set name */
MODBLKS=DYN                            /* DRD ENABLED; OLC DISABLED */
PLEXPARM=()                              /* GLOBAL resource status  */
UOM=MTO                     /* Unsolicited output message support */
/*******************************************************************/
/*                                                                 */
/*******************************************************************/
```

Related tasks:

"Importing resource and descriptor definitions by using the automatic import function" on page 61

DATABASE section of the DFSDFxxx member

The DATABASE section of the DFSDFxxx member specifies options for databases, such as the size of storage pools for ACBLIB members. The section is defined by the header <SECTION=DATABASE>.

Syntax

Parameters

ACBIN64=*nnn*
Specifies the amount of storage to allocate in 64-bit memory for non-resident PSB and DMB ACB members. Only online IMS regions types (not DCCTL or batch) are supported. The specification is in gigabytes, where *nnn* can be a value 1 - 999. You can specify the 64-bit storage pool in all IMS region types.

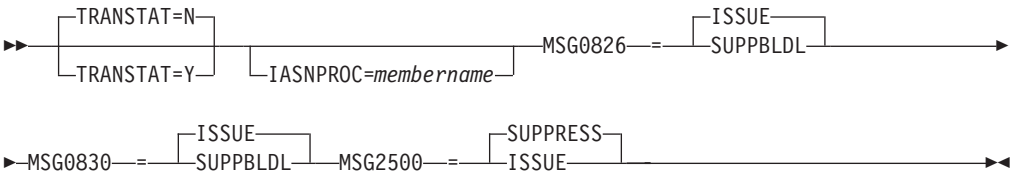
Example of the DATABASE section of the DFSDFxxx member

```
/*****************************************************************/
/* Database Section                                              */
/*****************************************************************/
<SECTION=DATABASE>
ACBIN64=8                          /* Create 64-bit storage pool */
/*****************************************************************/
/*                                                               */
/*****************************************************************/
```

DIAGNOSTICS_STATISTICS section of the DFSDFxxx member

The DIAGNOSTICS_STATISTICS section of the DFSDFxxx member specifies options for the IMS Abend Search and Notification procedure (DFSIASN0) and for obtaining database and transaction statistics. The section is defined by the header <SECTION=DIAGNOSTICS_STATISTICS>.

Syntax



Parameters

IASNPROC=*membername*
Specifies the 8-character name of the IMS abend search and notification procedure. The IMS abend search and notification function is enabled only if the IASNPROC parameter is specified and defines a member name; otherwise, it is disabled.

MSG0826=
Specifies which forms of the DFS826I message are issued or suppressed.

ISSUE
All forms of the message can be issued. ISSUE is the default.

SUPPBLDL
Suppresses the following forms of the message:

- DFS826I BLDL FAILED FOR FOLLOWING DBDs
- DFS826I xxx DBD ERRORS SENT TO JOB LOG

The following message is issued to indicate how many failure messages were suppressed: DFS826I *xxx* DBD ERRORS SUPPRESSED.

MSG0830=
Specifies which forms of the DFS830I message are issued or suppressed.

ISSUE
All forms of the message can be issued. ISSUE is the default.

SUPPBLDL
Suppresses the following forms of the message:

- DFS830I BLDL FAILED FOR FOLLOWING PSBs
- DFS830I xxx PSB ERRORS SENT TO JOB LOG

The following message is issued to indicate how many failure messages were suppressed: DFS830I *xxx* PSB ERRORS SUPPRESSED.

MSG2500=

Specifies whether the dynamic allocation and deallocation message DFS2500I is issued or suppressed when an UPDATE DB command is issued for HALDBs.

SUPPRESS

Suppresses dynamic allocation and deallocation message DFS2500I from being issued by IMS for HALDB partitions. SUPPRESS is the default.

ISSUE

IMS can issue the dynamic allocation and deallocation message DFS2500I for HALDB partitions.

TRANSTAT=

Specifies whether transaction level statistics are logged. If Y is specified, transaction level statistics are written to the log in an X'56FA' log record.

The value specified for this parameter is taken into consideration when IMS is cold started and transaction and program definitions are created from definitions contained in the MODBLKS data set or when creating new programs and transactions by using the CREATE PGM and CREATE TRAN commands and the TRANSTAT value for the new resource is obtained from the system default descriptor.

The TRANSTAT= parameter is ignored when transaction and program definitions are created from definitions imported from an RDDS.

N Transaction-level statistics are not logged for all transactions and all programs. Transaction-level statistics can be logged for individual transactions or programs by defining the APPLCTN macro or TRANSACT macro with TRANSTAT=Y, or issuing an UPDATE PGM command or UPDATE TRAN command. N is the default.

Y Transaction-level statistics are logged for all transactions and all programs, regardless of the transaction statistics value defined with the APPLCTN macro or TRANSACT macro. An UPDATE PGM command or UPDATE TRAN command can be issued to reset the TRANSTAT value to N for a particular transaction or program.

If dynamic resource definition (DRD) is enabled and resource definitions are imported from an RDDS during cold start, the UPDATE TRAN NAME(*) SET(TRANSTAT(Y)) and UPDATE PGM NAME(*) SET(TRANSTAT(Y)) commands can be used to turn on transaction level statistics for all programs and transactions once IMS is up. To ensure that the updated TRANSTAT values persist across a cold start, all resource and descriptor definitions should be exported to a system RDDS.

Example of the DIAGNOSTICS_STATISTICS section of the DFSDFxxx member

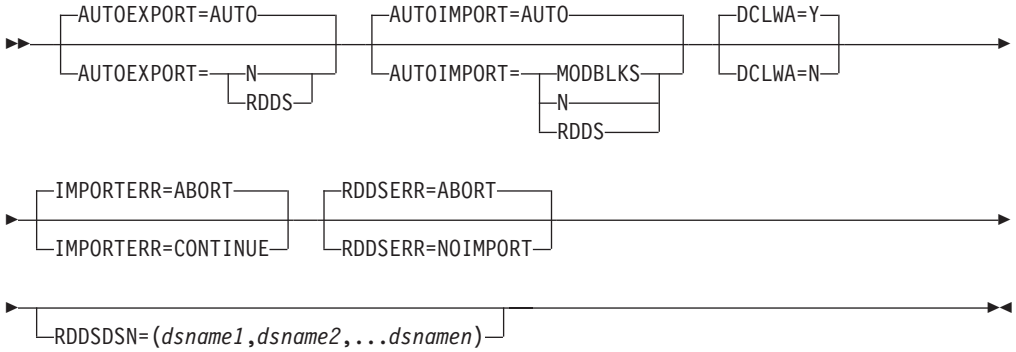
```
/*********************************************************************/
/* DFSDFXXX MEMBER                                                   */
/* DIAGNOSTIC SECTION                                                */
/*********************************************************************/
<SECTION=DIAGNOSTICS_STATISTICS>
IASNPROC=procname  /* IMS abend search and notification PROCLIB member */
MSG2500=ISSUE          /* Allow message DFS2500I for HALDB partitions */
/*********************************************************************/
/*                                                                   */
/*********************************************************************/
```

Related reference:
"DFSIASN0 procedure" on page 541

DYNAMIC_RESOURCES section of the DFSDFxxx member

The DYNAMIC_RESOURCES section of the DFSDFxxx member specifies options for import and export, log write ahead, and system resource definition data sets. This section is processed only if DRD is enabled (MODBLKS=DYN). If DRD is not enabled, this section is ignored. The section is defined by the header <SECTION=DYNAMIC_RESOURCES>.

Syntax



Parameters

These parameters are applicable only in a DRD environment; they are ignored in a non-DRD environment.

AUTOEXPORT=AUTO | NO | RDDS
Specifies whether all resource and descriptor definitions are exported at checkpoint time (simple or normal shutdown).

The automatic export takes place only if definitional changes were made to any resource since the time of the last checkpoint, except for restart checkpoint. After IMS completes restart processing, a restart checkpoint is taken. Export occurs after the restart checkpoint if automatic export is enabled.

AUTO
IMS determines whether to enable automatic export. Automatic export is enabled if two or more system resource definition data sets are defined and accessible.

AUTOEXPORT=AUTO is the default.

NO Disables automatic export. No resource or descriptor definitions are exported at checkpoint time.

RDDS
Automatic export is enabled if two or more system resource definition data sets are defined and accessible. All resource and descriptor definitions are exported to the oldest system resource definition data set at checkpoint time.

AUTOIMPORT=AUTO | MODBLKS | NO | RDDS

Specifies whether resource and descriptor definitions are automatically imported during IMS cold start.

If AUTOIMPORT=AUTO is specified, these conditions determine the source of the imported resource and descriptor definitions:

If all of the following conditions are true, automatic import from an RDDS is enabled:

- Two or more system RDDSs are defined in the DFSDFxxx member of the IMS PROCLIB data set.
- All the defined RDDSs can be allocated and read.
- At least one of the RDDSs contains valid resource and descriptor definitions.

If all of the following conditions are true, automatic import from the MODBLKS data set is enabled:

- The MODBLKS data set exists.
- No RDDSs are defined in the DFSDFxxx member of the IMS PROCLIB data set.
- The MODBLKS data set exists, and all the defined system RDDSs are empty.

In addition, only if no errors occur trying to access one or more of the data sets, automatic import is accomplished.

AUTO
IMS determines whether to enable automatic import processing. If IMS enables automatic import, it also determines the data source from which to import the definitions. AUTOIMPORT=AUTO is the default.

MODBLKS
Resource definitions are imported from the MODBLKS data set if one is defined. If definitions are imported from the MODBLKS data set, the values specified for the IMPORTERR= and RDDSERR= parameters do not apply and are ignored if specified. If an error occurs, IMS cold start is canceled.

NO No resource or descriptor definitions are imported when IMS cold starts.

RDDS
Resource and descriptor definitions are imported from the most recently updated system resource definition data set (RDDS).

DCLWA=Y | N

Specifies the default log write ahead option for transactions that are defined with the CREATE TRAN command. DCLWA= specifies whether (Y) or not (N) IMS performs log write-ahead for recoverable nonresponse-mode input messages and transaction output messages. The default is Y. The IMS transaction descriptor DFSDSTR1 is defined with the DCLWA value that was defined in this DFSDFxxx member of the IMS PROCLIB data set. Any subsequent CREATE TRAN commands that are issued either without the LIKE keyword, or with LIKE(DESC(DFSDSTR1)), are defined with this DCLWA value.

Y Specifies that:

- Information in the log buffers is written to the IMS log before the associated input acknowledgment or output reply is sent to the terminal.
- A nonresponse-input transaction is made recoverable across IMS failures before IMS acknowledges receipt of the input.

- Database changes are made recoverable before IMS sends associated output reply messages.

N Specifies that input message integrity and the consistency of output messages with associated database updates is not required. DCLWA does not apply to response mode or Fast Path input processing. If it is specified in these situations, is ignored during IMS execution.

IMPORTERR=ABORT | CONTINUE

Specifies which action to take if an error occurs during automatic import processing due to an invalid resource or descriptor definition. This parameter does not pertain to errors that occur while trying to access an RDDS; use the RDDSERR= parameter to choose processing options following RDDS-related errors.

ABORT
Automatic import processing is canceled, and IMS cold start terminates abnormally with a U3397 abend. ABORT is the default.

CONTINUE
Automatic import processing continues. The resource in error is marked and given a not-initiated status (NOTINIT). If the resource is required, create or update the resource after cold start completes.

RDDSERR=ABORT | NOIMPORT

Specifies which action to take if an error occurs when a resource definition data set is accessed during automatic import processing.

ABORT
Automatic import processing is canceled, and IMS cold start terminates abnormally with a U3368 abend. ABORT is the default.

NOIMPORT
Automatic import processing is canceled. The IMS system is started with no database resources or descriptors (DDIRs), program resources or descriptors (PDIRs), routing code resources or descriptors (RCTEs), or transaction resources or descriptors (SMBs) defined.

RDDSDSN=(*dsname1, dsname2,...dsnamen***)**
Specifies the 1- to 44-character names of the system resource definition data sets. The data sets must be cataloged BSAM data sets. The data sets are dynamically allocated and opened in the order in which they are specified on the RDDSDSN= parm. Each IMS in the IMSplex must have its own set of system resource definition data sets. At least two data set names must be specified. Although any number of data sets can be specified, three RDDSs are recommended. When exporting to a system RDDS, IMS alternates exporting between each of the data sets specified. All resource and descriptor definitions for the local IMS are exported to the oldest data set specified on the RDDSDSN= parameter. When importing from a system RDDS, IMS uses the RDDS with the most current data.

IMS always attempts to write to the RDDS containing the oldest data. If a failure occurs while IMS attempts to write to the data set with the oldest data, the data set with the next oldest data is selected. The RDDS with the most current data is preserved. For example, if two RDDS data sets are defined, and RDDS1 contains the most current resource and descriptor definitions, automatic export attempts to write to RDDS2 first. If that fails, automatic export does not attempt to write to RDDS1. The automatic export function is suspended until the error is resolved.

When creating and allocating the RDDS data sets, be sure to place an end-of-file (EOF) mark at the beginning of the data set. Otherwise, unpredictable results can occur. To ensure that an EOF mark is placed at the beginning of the data set, program IEBGENER can be used with a DUMMY SYSUT1 DD statement that contains the parameters BLKSIZE=*xxxxx* and RECFM=VB, where *xxxxx* is a value 4096 - 32760, inclusive. The SYSUT2 DD statement:

- Points to the DSN and VOLSER of the RDDS to be created.
- Specifies LRECL=*yyyyy*, BLKSIZE=*xxxxx*, and RECFM=VB, where *xxxxx* is the value of the block size specified on the DUMMY SYSUT1 DD statement, and *yyyyy* is (*xxxxx* – 4).

Recommendation: Set BLKSIZE to 32760.

Example of the DYNAMIC_RESOURCES section of the DFSDFxxx member

```
/*****************************************************************/
/* Dynamic Resource Definition Section                           */
/*****************************************************************/
 <SECTION=DYNAMIC_RESOURCES>
RDDSDSN=(IMSTESTL.IMS1.RDDS1,
         IMSTESTL.IMS1.RDDS2,
         IMSTESTL.IMS1.RDDS3,)
AUTOIMPORT=AUTO
AUTOEXPORT=AUTO
IMPORTERR=ABORT
RDDSERR=ABORT
/*****************************************************************/
```

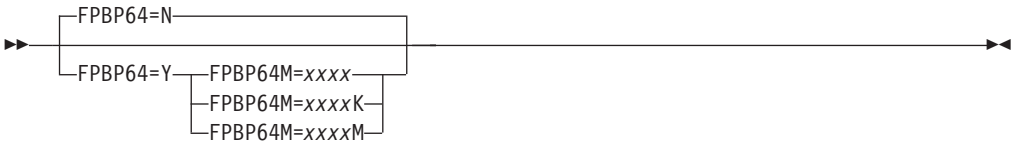
Related tasks:

"Importing resource and descriptor definitions by using the automatic import function" on page 61

FASTPATH section of the DFSDFxxx member

The FASTPATH section of the DFSDFxxx member specifies options for Fast Path, such as the use of the Fast Path 64-bit buffer manager. The section must begin with the header <SECTION=FASTPATH>.

Syntax



Parameters

FPBP64=N | Y

Specifies whether to enable the Fast Path 64-bit buffer manager. If you choose to use the Fast Path 64-bit buffer manager (FPBP64=Y), the DBBF, DBFX, and BSIZ parameters that define Fast Path buffers are ignored. If you specify FPBP64=N, Fast Path buffer management is defined by the DBBF, DBFX, and BSIZ parameters on the startup procedure. FPBP64=N is the default.

If Fast Database Recovery (FDBR) is being used, and the Fast Path 64-bit buffer manager is enabled, you must specify the DFSDF= parameter to reference this

DFSDFxxx member on the FDR procedure. This action ensures that the same Fast Path 64-bit buffer manager is used by both IMS and FDBR.

FPBP64M=
Use this parameter to set a maximum limit on how much 64-bit storage the Fast Path 64-bit buffer pool uses for DEDB data buffers. If FPBP64=Y is specified, this parameter is required. If FPBP64=Y is not specified, FPBP64M is ignored.

The value is specified as a decimal value, in bytes, kilobytes, or megabytes, where:

xxxxx Bytes. Maximum value is 2147483647.

*xxxxx***K**
Kilobytes. Maximum value is 2097151K.

*xxxxx***M**
Megabytes. Maximum value is 2047M.

The valid range is 1M - 2047M. The minimum value for this parameter is the amount of 64-bit storage that is initially allocated for the subpools (without any extents).

Once the specified limit is exceeded, it disallows the schedule of new threads. Currently active threads are unaffected and can force the limit past the threshold established by FPBP64M. In other words, if the amount of required 64-bit storage is greater than the value specified for FPBP64M, FPBP64M is increased and a DFS3299I message is issued.

The value for this parameter can be updated with the UPDATE POOL TYPE(FPBP64) SET(LIMIT(*xxxxx*)) command.

Example of the FASTPATH section of the DFSDFxxx member

```
/*****************************************************************/
/* Fast Path Section                                             */
/*****************************************************************/
<SECTION=FASTPATH>
FPBP64=Y                      /* Fast Path 64-bit buffer manager */
FPBP64M=2047M                 /* FP 64-bit buffer pool max limit */
/*****************************************************************/
/*                                                               */
/*****************************************************************/
```

SHARED_QUEUES section of the DFSDFxxx member

The SHARED_QUEUES section of the DFSDFxxx member of the IMS PROCLIB data set specifies options for queue names, structure names, and group names. The section must begin with the header <SECTION=SHARED_QUEUES>.

Syntax

The syntax of the parameters that are specified for shared queues in the DFSDFxxx member are the same as the syntax and parameters specified for shared queues in the DFSSQxxx member of the IMS PROCLIB data set.

Parameters

All of the parameters that are valid in the DFSSQxxx member of the IMS PROCLIB data set are valid in the DFSDFxxx member of the IMS PROCLIB data set.

If you specify values in both the DFSSQxxx member and the shared-queues section of the DFSDFxxx member, the values specified in the DFSSQxxx member of the IMS PROCLIB data set override the values specified in the DFSDFxxx member. If you specify all your shared-queues values in the DFSDFxxx member, you can delete the DFSSQxxx member.

Example DFSDFxxx member for shared queues definitions

```
//*****************************************************************/
/* Shared Queues Section                                         */
/*****************************************************************/
<SECTION=SHARED_QUEUES>
CQS=CQSCW1                                          /* CQS name */
CQSSSN=CQS1                                /* CQS subsystem name */
EMHQ=IMSEMHQ01                            /* EMHQ structure name */
MSGQ=IMSMSGQ01                            /* MSGQ structure name */
SQGROUP=GRUP1                                 /* XCF group name */
/*****************************************************************/
/*                                                               */
/*****************************************************************/
```

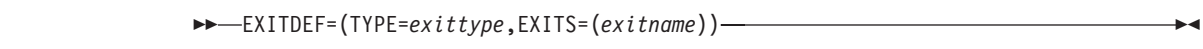
Related reference:

"DFSSQxxx member of the IMS PROCLIB data set" on page 736

USER_EXITS section of the DFSDFxxx member

The USER_EXITS section of the DFSDFxxx member specifies the user exits to be called. The section must begin with the header <SECTION=USER_EXITS>.

Syntax



Parameters

EXITDEF()
Associates an exit routine type with a list of one or more exit routine modules to be called.

TYPE=*exittype*
Specifies the user exit type. The types of user exits that can be specified include:

- Build Security Environment User Exit (BSEX)
- IMS CQS Event user exit (ICQSEVNT)
- IMS CQS Structure Event user exit (ICQSSTEV)
- Initialization/Termination user exit (INITTERM)
- Log Edit user exit (LOGEDIT)
- Logger user exit (LOGWRT)
- Non-Discardable Messages user exit (NDMX)
- OTMA Input/Output Edit user exit (OTMAIOED)
- OTMA Destination Resolution user exit (OTMAYPRX)
- OTMA Resume TPIPE Security user exit (OTMARTUX)
- Partner Product user exit (PPUE)
- Resource Access Security user exit (RASE)
- Restart user exit (RESTART)

EXITS=*exitnames*
Specifies a list of one or more exit routine names. The position of the exit routine in the list determines the order in which the exit routine is driven.

Example of the USER_EXITS section of the DFSDFxxx member (Single exit)

```
//*****************************************************************/
/* Restart exit section                                          */
/*****************************************************************/
<SECTION=USER_EXITS>
  EXITDEF=(TYPE=RESTART,               /* Restart User Exit      */
          EXITS=(UEXIT1))              /* Single exit              */
/*****************************************************************/
/*                                                               */
/*****************************************************************/
```

Example of the USER_EXITS section of the DFSDFxxx member (Multiple exits)

```
//*****************************************************************/
/* Restart exit section                                          */
/*****************************************************************/
<SECTION=USER_EXITS>
  EXITDEF=(TYPE=RESTART,               /* Restart User Exit      */
          EXITS=(UEXIT1,UEXIT2,UEXIT3)) /* Exit list             */
/*****************************************************************/
/*                                                               */
/*****************************************************************/
```

Related reference:

QUERY USEREXIT command (Commands)

DFSDRFnn member of the IMS PROCLIB data set

Use the DFSDRFnn member of the IMS PROCLIB data set to specify that portions of the control region be placed in disabled reference (DREF) storage during initialization.

With this member, VTAM terminal blocks (VTCBs) in DREF storage are never paged out to auxiliary storage; they remain in real or expanded storage. DFSDRFnn requires that the system have sufficient expanded storage available for the VTCBs. If you choose the DREF option for VTCBs during installation, but do not have expanded storage, the VTCBs DREF storage are page fixed.

Environments

This member of the IMS PROCLIB data set is not applicable in the DBCTL environment.

Syntax

Usage

If the VTCBs have been paged out of real storage when a checkpoint occurs, DFSDRFnn can improve the time to bring them back into real storage.

The *nn* suffix of DFSDRFnn is taken from the IMS procedure FIX=nn initialization parameter, the same one used for the page fixing member, DFSFIXnn (see "DFSFIXnn member of the IMS PROCLIB data set" on page 704).

If you specify either DFSDRFnn or DFSFIXnn, define the one you do not choose to use as a null member. For example, if you specify FIX=55, define both members (DFSDRF55 and DFSFIX55), and make the member you do not need a null member containing a blank control statement. Without both defined, IMS issues error message DFS0579, indicating a member could not be found. If DFSFIXnn is defined as a null member, message DFS0757 is issued, reminding you that no fixing requests were made.

If BLOCK=VTCB is requested in DFSFIXnn and in DFSDRFnn, the page-fix request in DFSFIXnn is ignored.

Parameters

The control information is contained in 80-character records. Continuation or sequencing must not be entered. The format is as follows:

BLOCKS=

VTCB
Specifies that VTCBs are loaded into DREF storage, but are never to page to auxiliary storage.

DFSDSCMx member of the IMS PROCLIB data set

When the Extended Terminal Option (ETO) is enabled, IMS generates ETO descriptors during stage 1 system definition and stores them in the DFSDSCMx member of the IMS.PROCLIB data set.

IMS generates both default descriptors and descriptors that are based on any static definitions in the IMS stage-1 system definition macros. The types of descriptors that IMS can generate include logon descriptors, MFS device descriptors, MSC descriptors, and user descriptors.

Recommendation: If you code your own descriptors, do not store them in the DFSDSCMx member. The contents of the DFSDSCMx member are deleted and regenerated each time a stage 1 system definition is performed. To preserve descriptors that you code across a stage-1 system definition, store them in the DFSDSCTy member of the IMS.PROCLIB data set.

The *x* on DFSDSCMx is the IMS nucleus suffix. The suffix must match a suffix that your installation specifies on the SUFFIX= parameter of the IMSGEN system definition macro.

To use IMS ETO Support, at least one user descriptor and one logon descriptor must exist. If at least one of each of these descriptors does not exist or cannot be found when you enable IMS ETO Support, IMS abends with U0015 issuing message DFS3652.

The following rules apply to creating ETO descriptors:

- Separate one keyword or parameter set from another with one or more blanks.
- Do not include embedded blanks within a keyword and its parameters.
- Separate a keyword from its parameters with an equal sign (=).

- Do not abbreviate keywords.

You can continue a keyword and its parameter set to the next statement if no intervening blanks appear at the end of the first statement or at the beginning of the parameters of the next statement. A continued statement still has the same descriptor type and name in columns 1-10; the continued specification begins in column 12. If you specify keywords, they must be accompanied by parameters. Keywords followed by blanks or commas are invalid.

The format and parameters of the ETO descriptors are described in the following topics:

Related concepts:

"ETO descriptors" on page 186

Related reference:

"DFSDSCTy member of the IMS PROCLIB data set" on page 701

"IMSGEN macro" on page 351

Coding ETO descriptors (Communications and Connections)

Common format of ETO descriptor statements

Each descriptor statement must be 80 characters in length.

All statements are translated to uppercase. The basic format of each of the descriptor types is shown in the following table.

Table 71. Descriptor type formats

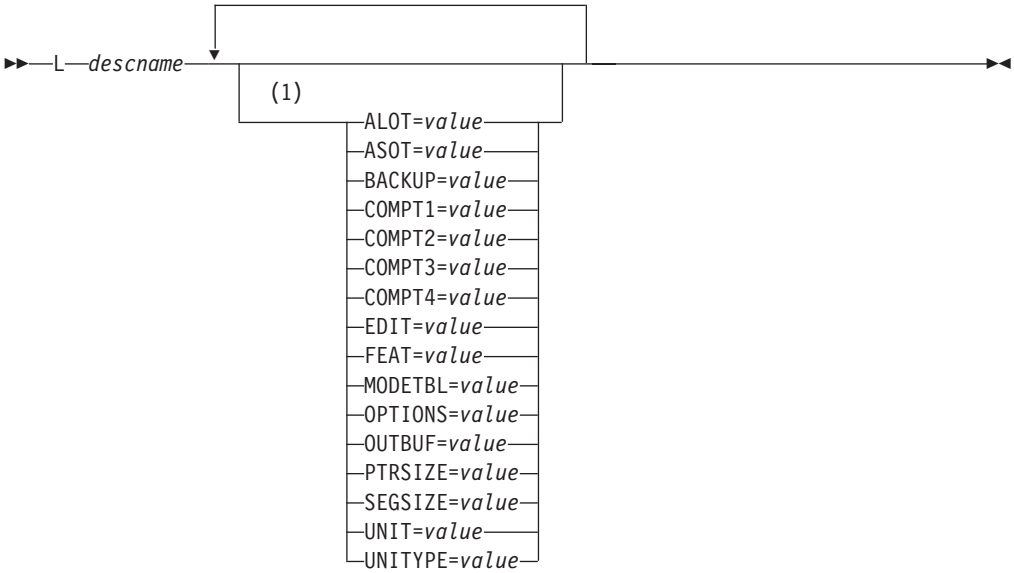
Column number	Contains	Considerations
01	The descriptor type indicator: L (logon) M (MSC) U (user) D (MFS device)	If an asterisk (*) is in column 01, the record is considered a comment record and is ignored.
03-10	The descriptor name	• All descriptor names must be 1-8 alphanumeric characters. • The first character for logon and user descriptors must be alphabetic (A-Z, #, $, or @). • Duplicate descriptor names are allowed if they are grouped, because more parameters might be required than can exist in an 80-byte record. • For MSC descriptors, this column contains the link name.
12-72	Parameters for each descriptor	• Parameters vary by type. • They are delineated by blanks. • They are typically in the same format as the equivalent parameters specified in the stage 1 input deck.
73-80	Can contain sequence numbers	IMS ignores these columns.

Logon descriptor format and parameters

Because logon descriptors are built from equivalent TYPE and TERMINAL macros, most definitions of logon descriptor parameters match the parameters found on those macro statements.

Because you do not need to define the maximum number of ETO-ISC sessions, the SESSION keyword is not supported on the ETO descriptor. You can continue to add sessions while storage and CPC capacity exist. The UNIT keyword is valid for non-SNA 3270 terminals only.

Logon descriptor syntax



Notes:

1 Each parameter can be specified only once. Parameters can be specified in any order.

Keyword parameters

Except for the descriptor type, L, and the descriptor name, all keyword parameters are specified in columns 12 through 72.

L Specified in column 1, indicates that the descriptor type is logon.

descname
Specified in columns 03-10, the name of the descriptor. Descriptor names can be an IMS default descriptor name, the name of a TERMINAL macro, or a unique name you create.

Default names include:

- DFS3270
- DFS327P
- DFSFIN
- DFSLU61
- DFSNTO
- DFSSLU1

- DFSSLU2
- DFSSLUP

ALOT=

Specifies the auto logoff time in minutes. Valid values are 0 and from 10 to 1440. If the ALOT value is not specified, the value from the JCL member is used except for FINANCE, SLU P, and ISC. If ALOT is not specified on the logon descriptor or overridden by the logon exit (DFSLGNX0) for FINANCE, SLU P, and ISC, a value of 1440 is used (the value from the JCL member is ignored).

- ALOT= 0

  The terminal is logged-off immediately when no signon is in effect. This specification is normally used in terminal sessions when the user is signed-on automatically during the logon process. During autologon, signon data can be provided in one of the following ways:

  – Signon data supplied by the `IMS /OPNDST` command
  – Signon data supplied by logon userdata (BIND)
  – Signon data supplied by logon exit (DFSLGNX0)

  There are two modes of operation for using ALOT=0, either of which can be set using the DFSINTX0 User Initialization Exit parameter list.

  In default mode, when signon errors are encountered, the session is automatically signed off and then logged off; no message is sent. If you do not supply the DFSINTX0 exit, or you supply the exit and indicate default mode for ALOT=0, then signon data must be supplied during the logon process. All the following error conditions result in automatic logoff:

  1. A non-signon, or errors detected during signon or input processing, result in immediate logoff.
  2. `/SIGNOFF` results in immediate logoff.
  3. `/SIGNON` signs off the current user and signs on a new user. However, errors encountered during the signon process, such as detection of an incorrect or expired password, result in immediate logoff.

  Restriction: Default mode should not be used for interactive terminal sessions that require a response to the DFS3649 message; these sessions do not wait for input signon and log off immediately.

  In alternate mode, when signon errors are encountered, the session is automatically signed off, a message is sent, and the session is logged off. Signon data can be supplied but is not required. All the following error conditions result in automatic logoff:

  1. A non-signon error detected during input processing results in immediate logoff.
  2. No signon data has been provided by the logon userdata (BIND) or the Logon Exit (DFSLGNX0).
  3. A `/SIGNOFF`, or errors resulting from a `/SIGNON`, cause message DFS3649(A) (Signon Required) to be sent, and a fixed ten-minute timer set to wait for a new signon. If no signon occurs during that interval, then the session is logged off.

- ALOT=(10-1439)

  The session is terminated after the specified number of minutes have elapsed without a signed-on user.

- ALOT=1440

  The session is never automatically terminated.

ASOT=

Specifies the auto signoff time, in minutes. Valid values are 0 and from 10 to 1440. You can override this value during signon with the user descriptor. If ASOT is not specified on either the user descriptor, the logon descriptor, or overridden by either the logon (DFSLGNX0) or signon (DFSSGNX0) exits, the value from the JCL member is used except for FINANCE, SLU P, and ISC. If ASOT is not specified on either the user or logon descriptor for FINANCE, SLU P, and ISC, a value of 1440 is used (the value from the JCL member is ignored).

- ASOT=0

  The user is signed off immediately when no output is available to be sent. ASOT=0 is normally specified when no IMS input or output is available, or after the last available output message completes.

  Restriction: Do not specify ASOT=0 for interactive terminals, for example 3270s or SLU2.

- ASOT=(10-1439)

  The user is signed off after the specified number of minutes have elapsed without terminal activity, irrespective of message status.

- ASOT=1440

  The user is never automatically signed off.

BACKUP=

Specifies the control of session switching (VTAM) after takeover. Use only when HSB=YES on the IMSCTRL macro.

This keyword is the same as the BACKUP= keyword in the TYPE macro. See "TYPE macro" on page 441 for more information about keyword values.

COMPTn=

Where *n* is 1-4, specifies the first through fourth components of this logon descriptor. This keyword is only valid for the following device types: 3270, FINANCE, SLU 1, SLU P, and LU6.1 (ISC). For UNITYPE=FINANCE, these four keywords are the same as the four operands on the TERMINAL macro's COMPT keyword. The parameters for this keyword are positional. See "TERMINAL macro" on page 391 for more information about this keyword for other device types.

COMPT=PTR1 for UNITYPE=3270, UNIT=3275 should be specified as COMPT2=PTR1.

EDIT=

Specifies the name of a user-supplied physical terminal output and input edit routine for the terminals in this communication description set.

This keyword is the same as the EDIT= keyword in the TYPE macro. See "TYPE macro" on page 441 for more information about keyword values.

FEAT=

The parameters for this keyword are not positional.

See the description of the FEAT= keyword in "TERMINAL macro" on page 391 and "TYPE macro" on page 441 for more information about this keyword.

MODETBL=

Specifies the name of the VTAM logon mode table entry (logon mode name) that contains the SNA bind parameters to be used when a session is initiated by the MTO or through the `/OPNDST` command.

This function allows a system definition specification for referencing an entry other than the default entry in the user's VTAM logon mode table. Normally, the terminal operator specifies this mode table entry name when logging on to a terminal; this is not possible, however, from the IMS master terminal because IMS initiates the session.

With this function, if MODETBL= is not specified at system definition, no functional or operational change affects the user.

If MODETBL= is specified at system definition, the specified entry name is used. The MODETBL can be overridden by:

- The LOGON APPLID entry by the remote terminal operator
- VARY ACT, LOGON= command by the network terminal operator
- /OPNDST, /RST, or /CHANGE command by the MTO

The MODETBL= parameter is required for the IMS master terminal if the VTAM default mode table is not configured specifically for the device to be used as the IMS master terminal.

With MODETBL=, you do not need to specify the mode table entry when logging on to terminals that always require specification of the same mode table entry name.

OPTIONS=
Specifies certain communication options associated with this terminal. Parameters for this keyword are not positional.

The ARNR|NRNR option specifies whether (ARNR) or not (NRNR) IMS should enable Rapid Network Reconnect (RNR) support for ACF/VTAM persistent sessions as defined in the ACF/VTAM APPL statement.

Unless otherwise noted, the communications options described in the OPTIONS= keyword of the TYPE and TERMINAL macros are supported for logon descriptors. See "TERMINAL macro" on page 391 and "TYPE macro" on page 441 for more information about this keyword.

The following options are not supported for logon descriptors:

- NOSHARE|SHARE
- ACK|OPTACK
- NORESP|FORCRESP|TRANRESP
- SIGNON|NOSIGNON

NORESP|FORCRESP|TRANRESP can be specified on the user descriptor.

The COPY|NOCOPY option is supported for logon descriptors for dynamic SLU 2 terminals, but not for dynamic non-SNA VTAM 3270s.

OUTBUF=
Specifies the size of the IMS output buffer to be used for workstations.

This keyword is the same as the OUTBUF= keyword on the TERMINAL macro. See "TERMINAL macro" on page 391 for more information about this keyword.

PTRSIZE=
Specifies the number of print positions of the 3284 or 3286 printer. The default is 120.

If PTRSIZE=IGNORE is specified with FEAT=IGNORE in its DEV statement, MFS always uses the device output format (DOF) when editing output for this device. The existing parameters and control block values, including default parameters, are unchanged.

SEGSIZE=
Specifies segment size. Acceptable values are from 256 bytes to 32000 bytes. The default is 256. For information about calculating segment size, see the description of the SEGSIZE= keyword in "TERMINAL macro" on page 391.

UNIT=
Specifies the terminal for the previously defined line group.

This keyword is the same as the UNIT= keyword on the TERMINAL macro. See "TERMINAL macro" on page 391 for more information about this keyword.

UNITYPE=
Specifies the UNITYPE of the terminal. If not specified, a default UNITYPE of SLUTYPE2 is used.

3601 is not a valid UNITYPE.

The following values are valid for the UNITYPE keyword:

- 3270
- FINANCE
- LUTYPE6
- NTO
- SLUTYPE1
- SLUTYPE2
- SLUTYPEP

Examples of logon descriptors

An example of a logon descriptor is:

```
Column     Column
1          12
L MKT01LU2 UNITYPE=SLUTYPE2 ALOT=30
```

You can use multiple records to create a single descriptor name (for example, if the number of parameters causes the descriptor to exceed one 80-byte record).

The following is an example of a logon descriptor with multiple records:

```
L MKT01LU2 UNITYPE=SLUTYPE2
L MKT01LU2 ALOT=30
```

Related concepts:

"Logon descriptors" on page 188

Related tasks:

Creating logon descriptors (Communications and Connections)

MFS device descriptor format and parameters

MFS device descriptors are used by the MFS Device Characteristics Table (MFSDCT) utility.

The MFSDCT utility updates screen information, for example 3270 screen sizes and feature information, in the DCT and generates new MFS default formats without system definition.

You use MFS device descriptors to define characteristics of terminals added dynamically that have different characteristics from statically defined terminals.

Related reading: See *IMS Version 11 Database Utilities* for more information about how to use the MFSDCT utility.

Syntax of the MFS device descriptor

```
►►—D—descname—FEAT=value—SIZE=value—TYPE=value————————————————————►◄
```

Keyword parameters of the MFS descriptor

The following list describes each element of an MFS device descriptor statement.

Except for the descriptor type, D, and the descriptor name, all keyword parameters are specified in columns 12 through 72.

D Indicates that the descriptor type is MFS (device).

descname
The name of the descriptor.

FEAT=*value*
For a description of this parameter, see TERMINAL macro (System Definition).

SIZE=*value*
For a description of this parameter, see TERMINAL macro (System Definition).

TYPE=*value*
For a description of this parameter, see TERMINAL macro (System Definition).

Example

An example of an MFS device descriptor is:

```
Column      Column
1           12
D IMSTYP22  TYPE=3270-A04 SIZE=(43,80) FEAT=IGNORE
```

Related concepts:

"MFS device descriptors" on page 189

Related tasks:

Creating MFS device descriptors (Communications and Connections)

Related reference:

"TERMINAL macro" on page 391

MSC descriptor format and parameters

MSC descriptors relate remote NAME macros to MSC links defined during system definition.

When you specify with ETOFEAT that descriptors are to be created during system definition, an MSC descriptor is created for each MSNAME macro statement.

Syntax of the MSC descriptor

Keyword parameters of the MSC descriptor

The descriptions of the variables in the MSC descriptor are:

M Indicates the descriptor type is MSC (device).

linkname
The name of the link path from an MSNAME statement.

ltermname
Specifies the name of a logical terminal associated with a physical terminal that is defined in a remote IMS system.

Examples

An example of an MSC descriptor is:

```
Column     Column
1          12
M REMSYS01 REM01AAA REM01BBB REM01CCC REM01DDD REM01EEE REM01FFF
M REMSYS02 REM01GGG REM01HHH REM0III
```

Related concepts:

"MSC descriptors" on page 189

Related tasks:

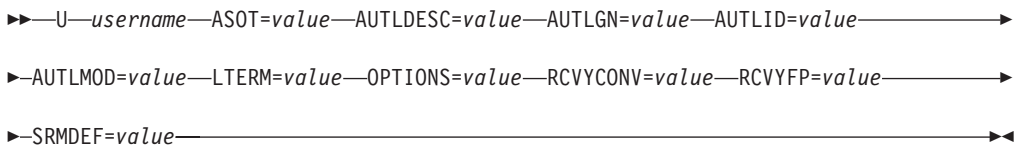
Creating MSC descriptors (Communications and Connections)

User descriptor syntax and parameters

User descriptors provide information relating to user options and user structure names.

Unless noted, the definition of a given user descriptor parameter is the same as that found for the equivalent NAME ("NAME macro" on page 380) and SUBPOOL macro statement ("SUBPOOL macro" on page 390). Also, unless noted, the general rules for creating descriptors apply to the following user descriptors.

User descriptor syntax



Keyword parameters of the user descriptor

The values in the user descriptor statement are described in the following list.

Except for the descriptor type, U, and the user name, all keyword parameters are specified in columns 12 through 72.

U Indicates the descriptor type is user.

username
The name of the descriptor, which can be a unique user ID you create, a node user name assigned to a TERMINAL or SUBPOOL macro, or the IMS default descriptor name, DFSUSER.

ASOT=*a*
a is 0 or a value from 10 to 1440. The default comes first from the logon descriptor parameter ASOT. If ASOT is not specified on either the user or logon descriptor, the value from the JCL member is used except for FINANCE, SLU P, and ISC. If ASOT is not specified on either the user or logon descriptor for FINANCE, SLU P, and ISC, or overridden by the logon (DFSLGNX0) or (DFSSGNX0) exits, then a value of 1440 is used, the value from the JCL member is ignored.

AUTLDESC=*d*
d is a one- to eight-byte alphanumeric logon descriptor name, the first character of which is alphabetic (A-Z, $, #, @). AUTOLDESC=d defines the characteristics of the terminal to be autologged on. This parameter is ignored for LU 6.1 devices.

AUTLGN=*b*
b is a one- to eight-byte alphanumeric LU name for auto logon, the first character of which is alphabetic (A-Z, $, #, @). Autologon allows IMS to logon and signon your terminal automatically. If you specify the autologon option for a user, the queuing of data to any of the user queues causes IMS to establish a session.

Related reading: For more information, see Autologon (Communications and Connections).

AUTLID=*g*
g is a one- to eight-byte alphanumeric ISC other system half session qualifier, of which the first character is alphabetic (A-Z, $, #, @). If the other system is IMS, then this is the name of an ISC user in that system. The AUTLGN parameter must be used with the AUTLID parameter when requesting autologon for an ISC parallel session.

AUTLMOD=*e*
e is a one- to eight-byte alphanumeric mode table for the autologon terminal.

LTERM=(*f*,*h*,*i*,*j***)**
f,*h*,*i*, and *j* are all positional parameters. *f* is a one- to eight-byte alphanumeric LTERM name. *h* is either ULC or UC; the default is ULC. *i* is the COMPT parameter, and *j* is the ICOMPT parameter. COMPT and ICOMPT are the same as defined by keyword parameters on the NAME macro. If *f* is the only parameter supplied, the parentheses can be omitted.

A maximum of eight queues can be specified for each user descriptor.

If no LTERM keyword is specified, the user descriptor control block structure is built with one queue. The name used as the descriptor name is the default.

The queue names specified on the LTERM parameter keyword for the user descriptors must be unique. That is, two user descriptors cannot be created with the same queue name. If duplicate LTERM names are specified, message DFS3669 is issued.

If the LTERM keyword is specified on the user descriptor, the queue name must also be specified. The DFSUSER descriptor must also follow this convention (for example, with queue name DFSUSER).

The queue names (and Remote LTERM names) must follow the same naming conventions as the user names because they can, in some circumstances, take on the same name as the user name. The user name has strict naming conventions because it must follow RACF naming conventions. Therefore, these names must be alphanumeric (A-Z, 0-9, $, #, @), but the first character in the name must not be 0-9.

OPTIONS=*j*,*k*

j is a response mode of FORCRESP, TRANRESP, or NORESP. The default response mode is dependent upon the device type. If the response mode is not explicitly set in the user descriptor, the default cannot be set until signon, when the device type is known. The default setting for each device type, during signon, is indicated in the following table.

Table 72. Default response mode values

Device type	Default response mode
3270	NORESP
FINANCE	NORESP
LU 6	TRANRESP
NTO	TRANRESP
SLU 1	TRANRESPX
SLU 2	NORESP
SLU P	NORESP

k is an MSGDEL option of SYSINFO, NONIOPCB, or NOTERM. The default is SYSINFO.

If OPTIONS=NOTERM is specified for a user descriptor, when that user descriptor is used for a sign on of a FINANCE, SLU P, or ISC terminal, SYSINFO issued instead of NOTERM.

The OPTION parameters are not positional. Refer to the MSGDEL and OPTIONS parameters of the TERMINAL macro for descriptions.

RCVYCONV=

Specifies whether the status of a conversation can be recovered (Y) or not (N). RCVYCONV applies to conversation status, not to output messages. Even if the conversation status is not recovered, conversation output continues to be recoverable and is delivered asynchronously.

The following table shows the default values for RCVYCONV as they relate to the values specified on the SRMDEF keyword.

Table 73. RCVYCONV values related to SRMDEF values

	RCVYCONV=YES	RCVYCONV=NO
SRMDEF=GLOBAL	Valid (default)	Valid
SRMDEF=LOCAL	Valid (default)	Valid
SRMDEF=NONE	Invalid	Valid (default)

If RCVYCONV is specified incorrectly, message DFS1920I is issued and IMS uses the appropriate default from the preceding table.

RCVYFP=

Specifies whether the status of Fast Path can be recovered (YES) or not (NO). RCVYFP applies to Fast Path status and output.

The following table shows the default values for RCVYFP as they relate to the values specified on the SRMDEF keyword.

Table 74. RCVYFP values related to SRMDEF values

	RCVYFP=YES	RCVYFP=NO
SRMDEF=GLOBAL	Valid (default)	Valid
SRMDEF=LOCAL	Valid (default)	Valid
SRMDEF=NONE	Invalid	Valid (default)

Restriction: If SRMDEF=LOCAL, STSN recoverability (RCVYSTSN) and fast path recoverability (RCVYFP) must be the same. They must both specify YES or both specify NO.

If RCVYFP is specified incorrectly, message DFS1920I is issued and IMS uses the appropriate default from the preceding table.

RCVYSTSN=

Specifies whether the status of STSN terminals (SLUP, FINANCE, and ISC) can be recovered (Y) or not (N).

The following table shows the default values for RCVYSTSN as they relate to the values specified on the SRMDEF keyword.

Table 75. RCVYSTSN values related to SRMDEF values

	RCVYFP=YES	RCVYFP=NO
SRMDEF=GLOBAL	Valid (default)	Valid
SRMDEF=LOCAL	Valid (default)	Valid
SRMDEF=NONE	Invalid	Valid (default)

Restrictions: If SRMDEF=LOCAL, STSN recoverability (RCVYSTSN) and fast path recoverability (RCVYFP) must be the same. They must both specify YES or both specify NO. If SRMDEF=GLOBAL, Fast Path is always nonrecoverable, so if there is a failure during the execution of a fast path transaction, the STSN session must be restarted cold.

If RCVYSTSN is specified incorrectly, message DFS1920I is issued and IMS uses the appropriate default from the preceding table.

SRMDEF=

Specifies the status recovery mode for user and node resources. Valid values are GLOBAL, LOCAL, and NONE.

Related reading: In order to understand what the GLOBAL, LOCAL, and NONE values mean, you have to first understand how IMS classifies resource status. For a discussion of these concepts, see *IMS Version 11 System Administration*.

The following list describes the valid values for SRMDEF.

GLOBAL

The significant status of a resource is saved globally in the coupling facility resource structure every time the significant status changes, along with all other recoverable status for that resource. The resource status is restored at the next logon or signon and is available from any IMS system in the IMSplex. Resource status is copied to the local

system when that resource becomes active, but is deleted from the local system when it becomes inactive. Status is not recovered in local IMS log records.

Requirement: Using a status recovery mode of GLOBAL requires a Resource Manager and a resource structure in the coupling facility.

LOCAL
The significant status is saved in local control blocks and log records, along with all other recoverable resource status. The resource status is restored at the next logon or signon if the user or node returns to the same local IMS system. This status is not available on any other IMS system in the IMSplex.

LOCAL mode enforces an affinity for the user or node to the IMS system where the local status exists. This is referred to as RM affinity. IMS prevents the user or node from accessing any other IMS while local status exists. If the IMS system that has local status fails, the node and user are allowed access to another IMS system. In this case, the node or user does not recover any status and when the failed IMS restarts, the local status that existed is deleted.

Using a status recovery mode of LOCAL does not require a Resource Manager. If RM is not used to manage node or user resources, then no RM affinity is enforced.

NONE
The significant status is not saved by RM or in local log records. When the user signs on or the node logs on, the corresponding significant status does not exist.

Examples

An example of a user descriptor specification is:

```
Column     Column
1          12
U SMITH    ASOT=20 LTERM=(SECLT1) AUTLGN=SEC01LU2
```

Related concepts:

"User descriptors" on page 190

Related tasks:

Creating user descriptors (Communications and Connections)

Related reference:

"TERMINAL macro" on page 391

DFSDSCTy member of the IMS PROCLIB data set

Use the DFSDSCTy member of the IMS PROCLIB data set to specify override descriptors for the Extended Terminal Option (ETO), including logon descriptors, MFS device descriptors, MSC descriptors, and user descriptors.

The descriptors that you can specify in the DFSDSCTy member, as well as the syntax and parameters of those descriptors, are exactly the same as the descriptors that you can specify in the DFSDSCMx member. However, if a descriptor is specified on both members, the specifications coded in the DFSDSCTy member for the descriptor override the specifications coded in the DFSDSCMx member.

The ETO descriptors supported by the DFSDSCTy and DFSDSCMx members include:

- Logon descriptors
- MFS device descriptors
- MSC descriptors
- User descriptors

To use IMS ETO Support, at least one user descriptor and one logon descriptor must exist in either the DFSDSCMx or the DFSDSCTy member. If one or both of the logon and user descriptors do not exist when you enable IMS ETO Support, IMS abends with U0015 issuing message DFS3652.

By default, the suffix *y* on DFSDSCTy is 0. You can define a different value as the suffix by specifying the value on the DSCT= parameter in the IMS or DCC startup procedures.

The syntax and parameters of the ETO descriptors are documented under DFSDSCMx member of the IMS PROCLIB data set (System Definition)

Related concepts:

"ETO descriptors" on page 186

Related tasks:

Creating logon descriptors (Communications and Connections)

Related reference:

"DFSDSCMx member of the IMS PROCLIB data set" on page 689

"Parameter descriptions for IMS procedures" on page 448

DFSFDRxx member of the IMS PROCLIB data set

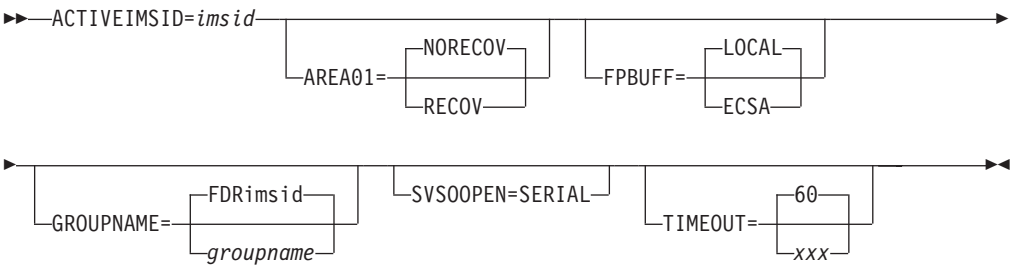
Use the DFSFDRxx member of IMS PROCLIB data set to specify the Fast Database Recovery (FDBR) options used by the FDR.

Multiple DFSFDRxx members can be present in the IMS PROCLIB data set. The FDRMBR= parameter in the FDR, IMS, or DBC procedure identifies the member currently in effect.

Environments

This member is applicable to the FDBR region and the FDBR-capable IMS or DBC regions.

Syntax



Usage

The record format is

Position
Contents

1-71 Option parameters separated by commas

72-80 Ignored

The first parameter in a record can have leading blanks. The last parameter in a record is denoted by a comma, followed by one or more blanks. The last parameter in the last record is followed by one or more blanks.

Parameters can occur in any order in the control statement. Multiple control statements can be provided.

Both the 64-bit pools and non-resident 31-bit pools in IMS active systems and their associated FDBR and XRF alternate systems should have identical contents. Whatever parameters and values you specify on the active IMS system should be the same on the alternate IMS system. For example, the parameters specified in the DFSVSMxx, DFSDFxxx, and DFSPBxxx members of the IMS PROCLIB data set must be the same; the sizes of the PSB and DMB pools must be the same.

Parameters

ACTIVEIMSID=
Specifies the IMS ID for the subsystem that the active FDBR region tracks. For IMS and DBC procedures (FDBR capable regions), ACTIVEIMSID must match the IMS ID defined in the procedure. This parameter is required.

AREA01=
Specifies whether Fast DB Recovery applies to DEDB areas defined as SHARELVL= 0|1. This parameter applies to FDR procedures. IMS and DBC procedures ignore it if it is specified.

RECOV
Specifies that Fast DB Recovery applies to SHARELVL= 0|1 areas.

NORECOV
Specifies that SHARELVL= 0|1 areas are not recovered by FDBR. Succeeding IMS emergency restarts recover the areas after FDBR completion.

The default is NORECOV.

FPBUFF=
Specifies where the control blocks for DEDB processing are allocated. This keyword is optional. If you do not specify this parameter, control blocks for DEDB processing are allocated in the FDBR control region private storage.

LOCAL
DEDB control blocks and buffer pools for shared VSO are allocated in the FDBR control region private storage. This is the default for IMS Version 11.

ECSA
DEDB control blocks and buffer pools for shared VSO are allocated in ECSA storage.

GROUPNAME=
Specifies the z/OS cross-system coupling facility group name for the active

IMS system and the tracking FDBR region. This parameter is optional. If you do not specify GROUPNAME=, a default name is created by using the ACTIVEIMSID= parameter and the prefix FDR. For example, if the active IMSID is IMS1, the default XCF group name is FDRIMS1.

SVSOOPEN=SERIAL
Specifies that all areas requiring redo processing in an FDBR system are serially processed. This option is ignored for emergency restart and XRF takeover processing. Use this option to reduce the number of structures that are allocated by FDBR for redo processing.

TIMEOUT=
Specifies the number of seconds that FDBR waits before determining a timeout status for the tracked IMS. Valid values are from 3-999.

The default is 60.

If a value less than 3 is specified, the value 3 is used. If a value greater than 999 is specified, 999 is used.

This parameter is applicable to IMS and DBC procedures.

The value you specify with this parameter applies only to IMS timeout status from XCF. For log surveillance, FDBR uses a five-second timeout value, and a three-second delay interval before calling DBRC to check whether an OLDS switch has occurred.

Related tasks:

"Enabling an IMS subsystem for FDBR" on page 106

DFSFIXnn member of the IMS PROCLIB data set

Use the DFSFIXnn member of the IMS PROCLIB data set to specify that portions of the control region (for example, certain control blocks, buffer pools, loaded modules, and part of the IMS nucleus) be fixed in address space during initialization.

Place control information in the IMS PROCLIB data set (typically named IMS.PROCLIB), as part of member DFSFIXnn, where nn is a two-character field supplied in the PARM field of the EXEC statement for the control region. See "Environments that support IMS system definition-supplied procedures" on page 446.

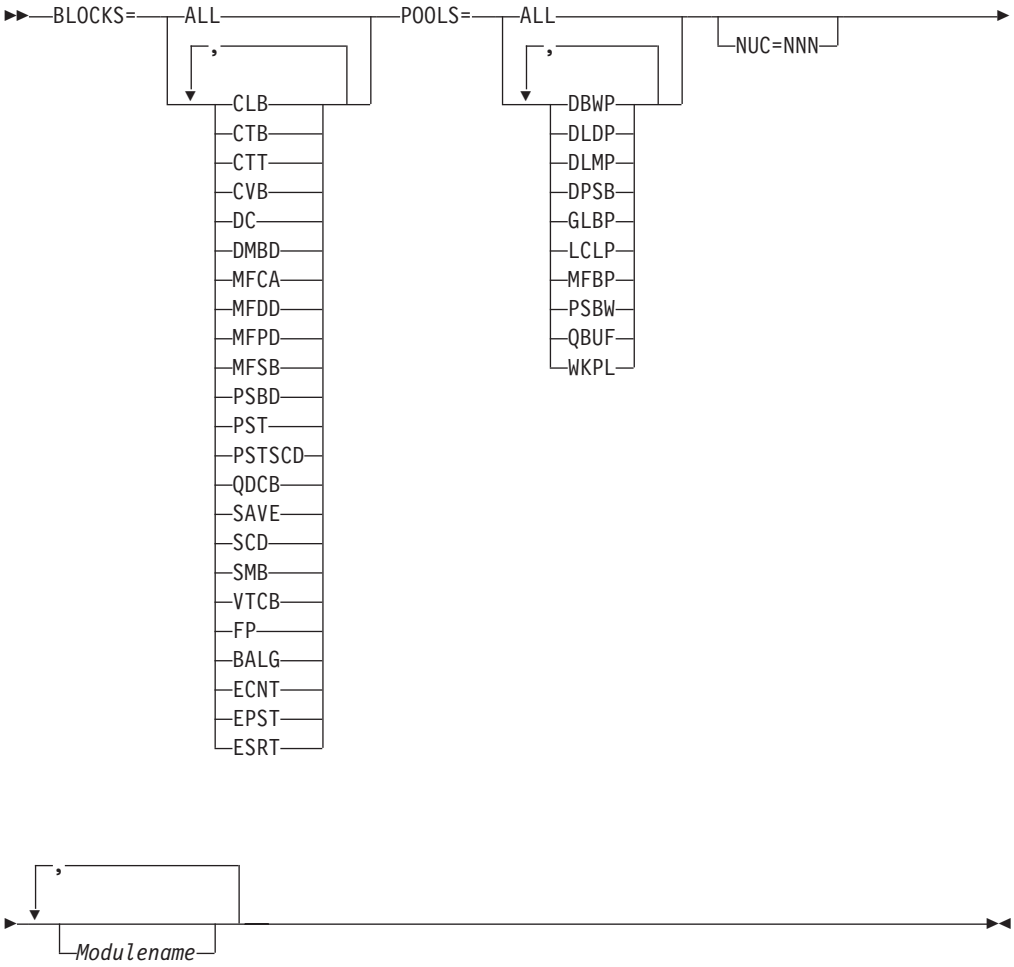
If you are running XRF, you can define an additional DFSFIXnn member. This member is identified by the DEFERFIX=xx parameter in the DFSHSBxx member of the IMS PROCLIB data set and can contain only non-Fast Path page-fix options for the control region. In the active system, both page-fix lists are processed. In the alternate system, the page fix list specified by the DEFERFIX=xx parameter is not processed until takeover. This allows you to control how much page fixing takes place in the alternate system while it is in tracking mode.

Environments

This member can be used in the IMS DB/DC and DBCTL environments.

Syntax

Recommendation: Specify BLOCKS=FP if your system uses Fast Path. If you do not specify BLOCKS=, either CPU performance can be seriously affected or system abends can be received, or both.



Usage

The control information is contained in 80-character records. Continuation or sequencing must not be entered. In addition:

- Either commas or blanks can be used to separate parameters, except BLOCKS, POOLS, and NUC, which are followed by an equal (=) sign.
- The *nnn* subparameter of NUC= can be any one, two, or three-digit number from 1 to 100.
- Following the BLOCKS or POOLS parameters, either ALL or any subset of the list of blocks or pools, respectively, can be coded.
- Parameters (NUC, BLOCKS, POOLS, Modulename) can be entered in any order and can be repeated as desired. If NUC is repeated, the percentage specified on the last NUC= is used. If BLOCKS or POOLS is repeated, a block or pool is fixed if it is specified in any of the BLOCKS= or POOLS= specifications.
- The subparameters of blocks and pools can be specified in any order.

- Invalid parameters are ignored, an error message is issued, and scanning continues.
- An invalid format or an invalid word (one not beginning with an alphabetic character) causes all scanning to stop and an error message to be issued. No fixing is done.
- All 80 characters in each record are available for control information.
- The modulename must be specified as the full eight-character name of a loaded module.

Parameters

BLOCKS=

ALL
All blocks listed this list.

CLB[1]
Non-VTAM communication line block

CTB[1]
Non-VTAM communication terminal block

CTT[1]
Communication translate table

CVB
Communication verb block

DC[1]
Non-VTAM CLB and non-VTAM CTB

DMBD or DMBDIR
Database directories

MFCA[1]
Message format buffer pool control area

MFDD[1]
MFS dynamic directory, (including hash table, prime area, and up to 10 additional areas)

MFPD[1]
MFS PDS directory indexes, (single area including one index per concatenation)

MFSB[1]
MFS staging buffers

PSBD or PSBDIR
Program directories

PST
Dependent region PSTs

PSTSCD
Blocks PST and SCD

QDCB[1]
Message queue DCBs

SAVE
Save area prefixes

SCD
System Contents Directory

SMB[1]
Scheduler message blocks

VTCB[1]
VTAM terminal control blocks

FP BALG, ECNT, EPST, and ESRT

BALG[1]
Load balancing group control blocks

ECNT[1]
Extension to the Communication Node Tables (CNTs)

EPST
Extension to the Partition Specification Tables (PSTs)

ESRT[1]
Message retrieve buffers

Note: [1]This keyword is not applicable in the DBCTL environment.

The following Fast Path control blocks, which are always page fixed, are not included in the preceding list:

BHDR
Main storage database headers

DMHR
Database buffer headers

LBUF
Synchronization point log buffer

DEDB
Control blocks for data entry databases: data management control blocks (DMCBs) and data management area control blocks (DMACs)

ESCD
Extension to the System Content Directory (SCD)

OTHR
Output thread control blocks

POOLS
This is the dynamic area acquired by IMS during initialization and used for various buffer pools.

ALL
All pools listed in this list.

DBWP
DMB work pool

DLDP
DMB pool

DLMP
With the DL/I address space option, that portion of the PSB pool in the z/OS common area.

DPSB
With the DL/I address space option, the portion of the PSB pool in DL/I local storage. Specification is ignored if the DL/I address space option is not in effect.

GLBP
Following storage pools are included: WKPL, DLMP, PSBW, DLDP, and DBWP. If the DL/I address space is used, DLDP and DBWP are not included.

LCLP[1]
Following storage pools are included: QBUF and MFBP. If the DL/I address space is used, DPSB, DLDP, and DBWP are also included.

MFBP[1]
Message format buffer pool

PSBW
PSB work pool

QBUF[1]
Message queue buffer pool

WKPL
General working pool

Note: [1]This keyword is not applicable in the DBCTL environment.

The parameters GLBP and LCLP can represent large amounts of storage. The terms GLBP and LCLP should not be interpreted as global and local storage, because the location of these pools can vary, based on whether the local storage option or the DL/I address space option is in effect. You should avoid using GLBP and LCLP; specify the pools to be fixed by name.

NUC=*nnn*
Is the percentage (specified by *nnn*) of the IMS load module (DFSVNUCx) to be page fixed.

If NUC=*nnn* is specified, a percentage of the nucleus, starting at location 0, is fixed regardless of what is placed there. IMS system definition places all the DC control blocks at the low end of the nucleus.

Modulename
The name of a loaded module. The specification of a module name does not cause that module to be fixed unless it has already been loaded into storage. (Module DFSVNUC0 cannot be fixed in this manner. See the NUC= keyword.)

If the DL/I address space option is being used, an attempt is made to fix a module in both the control and DL/I address spaces. If the module is loaded in only one of these address spaces, a message indicating that the module cannot be fixed is issued, in addition to the successful fix message.

Examples

The example in "Example of DFSFIXnn" on page 709 causes:

- Module DFSDLR00 to be fixed
- 76% of the IMS nucleus to be fixed
- The control program nucleus blocks DC, SMB, PST, and SCD to be fixed
- The message format buffer pool to be fixed

The member specified in the EXEC PARM field is read and processed during the IMS initialization procedure.

A list of the fixed pages is maintained during the execution of IMS. When IMS terminates, this list is used to free the fixed pages before returning to z/OS.

Example of DFSFIXnn

```
DFSDLR00 NUC=76 BLOCKS=DC,SMB,PSTSCD POOLS=MFBP
```

DFSHSBxx member of the IMS PROCLIB data set

Use the DFSHSBxx member of the IMS PROCLIB data set to specify the XRF options used by the active and alternate subsystems in an XRF complex.

Multiple DFSHSBxx members can be present in IMS PROCLIB data set. The HSBMBR= parameter in the IMS procedure identifies the member currently in effect.

An XRF complex has two DFSHSBxx members in effect at any time, one for the active system, and one for the alternate subsystem. The two members need not be identical, but the USERVAR and RSENAME parameters must be the same, and the surveillance options should be the same. If MNPS is used for terminal switching, then the USERVAR parameter is not applicable. Instead, the MNPS parameters must be the same in the two members.

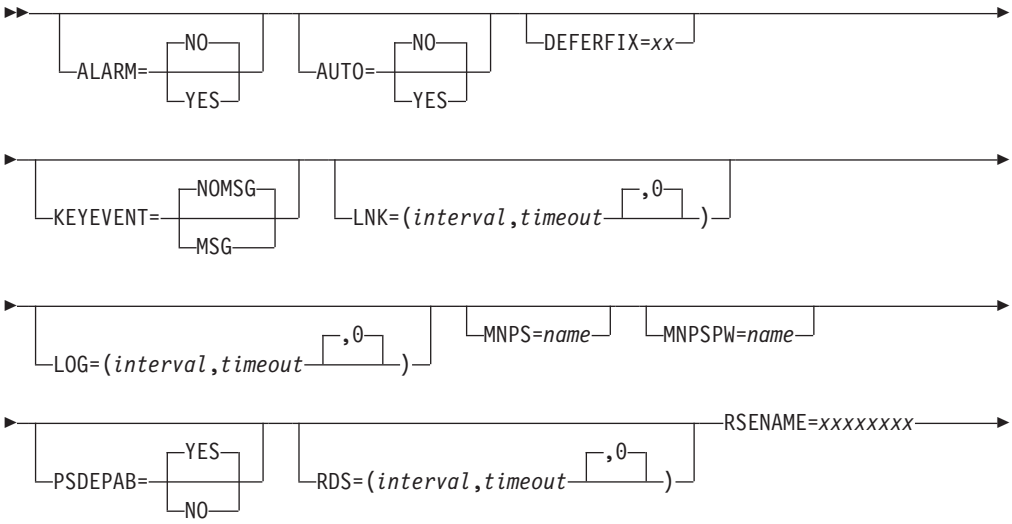
Environments

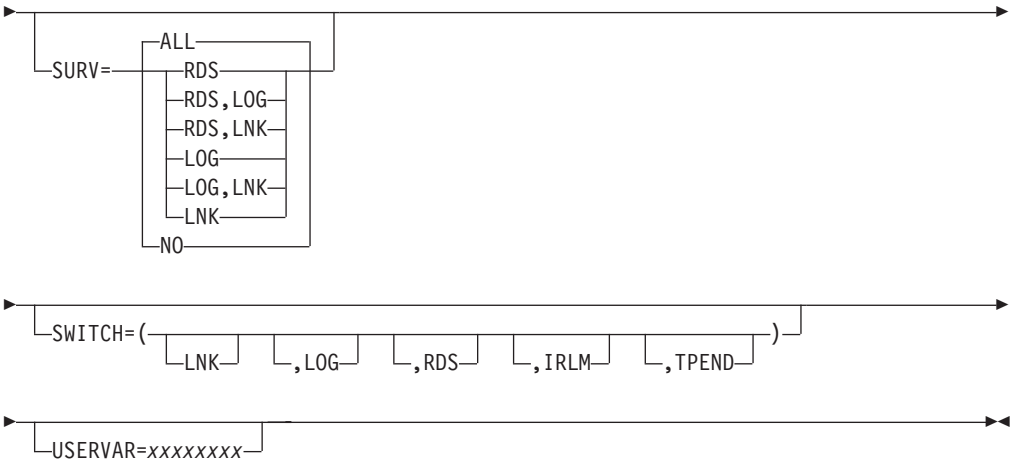
This member is not applicable in the DBCTL environment.

Syntax

The XRF option parameters can occur in any order in the control statement. Multiple control statements can be provided.

RSENAME= is required for all XRF complexes. Either USERVAR= or MNPS= is required if you use VTAM for XRF terminal switching.





Restrictions

Usage

The parameters in this member can have different meanings, depending on whether the subsystem is the active or the alternate subsystem.

The record format is:

Position	Contents
1-71	Option parameters separated by commas
72-80	Ignored

The first parameter in a record can have leading blanks. The last parameter in a record is denoted by a comma, followed by one or more blanks. The last parameter in the last record is followed by one or more blanks.

Parameters

ALARM=
Specifies whether (YES) or not (NO) the service processor alarm on the alternate subsystem sounds at takeover request. The default is NO.

AUTO=
Specifies whether (YES) or not (NO) a takeover is to proceed automatically. The default is NO.

DEFERFIX=xx
Identifies the DFSFIXxx member of IMS.PROCLIB containing the non-Fast Path page fixing options for the control region, which are processed during the restart of an active subsystem, and when the alternate subsystem becomes the active during takeover.

KEYEVENT=
Controls whether your IMS operator receives all XRF- related messages.

- MSG indicates that noncritical-event messages and critical-event messages are sent.
- NOMSG indicates that only critical-event messages are sent.

The default is NOMSG.

LNK=(interval,timeout)
Specifies the timing values for the ISC link surveillance. The default is (3,9).

interval
In the active subsystem, specifies how often (1 to 99 seconds) IMS sends signals across the ISC link.

In the alternate subsystem, specifies how often (1 to 99 seconds) IMS checks for those signals.

See Table 76 for additional information about coding the interval value.

timeout
In the alternate subsystem, specifies how long (1 to 999 seconds) IMS waits for a signal before considering a takeover.

See Table 77 on page 712 for additional information about coding the timeout value.

LOG=(interval,timeout)
Specifies the timing values for the log surveillance. It also determines the frequency with which the alternate subsystem accesses the log, after it has caught up. The default is (1,3).

interval
In the alternate subsystem, specifies how often (1 to 99 seconds) IMS checks to see if a new log record is received from the active subsystem.

This parameter also determines the frequency with which the alternate subsystem attempts to read a new log record, after the alternate has caught up with the active. If you do not specify LOG surveillance or do not specify the interval value, the alternate subsystem waits one second between reads to the log (only if the alternate has caught up with the active).

See Table 76 for additional information about coding the interval value.

Table 76. Initializing the INTERVAL value

Subsystem	Requirement	Exception action
Alternate	Alternate interval value must be greater than or equal to active interval value.	DFS3812 issued. Alternate interval value forced to active interval value.
Alternate	Alternate interval value must be greater than or equal to alternate LOG interval value (LNK and RDS).	DFS3833 issued. Alternate interval value forced to alternate log interval value.
Alternate	Alternate interval value less than or equal to alternate RDS and LNK interval value (LOG only).	DFS3833 issued. Alternate interval value forced to alternate RDS then LNK interval value.
Alternate	Twice the interval value must be less than or equal to the alternate timeout value.	DFS3832 issued. Alternate timeout value forced to twice the interval value.
Active	Twice the interval value must be less than or equal to the active timeout value.	DFS3832 issued. Active timeout value forced to twice the interval value.

timeout
In the alternate subsystem, specifies how long (1 to 999 seconds) IMS waits for a new record, before considering a takeover.

When the alternate subsystem catches up with the active, the timeout and interval values are also used to determine a delay interval, which limits the frequency of calls to DBRC to check if an OLDS switch has occurred.

See Table 77 for additional information about coding the timeout value.

Table 77. Initializing the TIMEOUT value

Subsystem	Requirement	Exception action
Alternate	Timeout value must be greater than or equal to twice the alternate interval value.	DFS3832 issued. Alternate timeout value forced to twice the alternate interval value.
Active	Timeout value must be greater than or equal to twice the active interval value.	DFS3832 issued. Active timeout value forced to twice the active interval value.

MNPS=*name*
Specifies the name of the MNPS ACB to use for terminal recovery. If specified, the USERVAR specification is ignored. This keyword is optional. If this specification is omitted, IMS uses the traditional XRF support and the USERVAR= specification for recovery.

MNPSPW=*name*
Specifies the password to use for the MNPS ACB. This password is checked by VTAM. If VTAM requires a password during VTAM system definition and no password is specified, then the MNPS ACB cannot be opened.

PSDEPAB=
Specifies whether (YES) or not (NO) prestarted BMP and IFP regions on the XRF alternate IMS are to be abnormally terminated with a U0454 abend during `/STO BACKUP` command processing on the alternate IMS, or `/CHE FREEZE` processing for the active IMS.

If NO is specified, the prestarted BMP and IFP regions are terminated with return code 0. The default is YES.

RDS=(interval,timeout)
Specifies the timing values for the restart data set (RDS) surveillance. The default is (1,3).

interval
In the active subsystem, specifies how often (1 to 99 seconds) IMS writes a time stamp on the RDS.

In the alternate subsystem, specifies how often (1 to 99 seconds) IMS checks for time stamps on the RDS.

See Table 76 on page 711 for additional information about coding the interval value.

timeout
In the alternate subsystem, specifies how long (1 to 999 seconds) IMS waits for a time stamp before considering a takeover.

See Table 77 for additional information about coding the timeout value.

RSENAME=xxxxxxxx
Specifies the name of the recoverable service element (RSE) containing the active and alternate IMS subsystems in an XRF complex. The RSENAME= specifications must be the same in DFSHSBxx members for the active and alternate subsystems.

The name used for the RSENAME parameter should not be the same name used for the IMSID parameter in the IMSCTRL macro for either the active or the alternate IMS subsystems in the XRF complex.

RSENAME= is required.

SURV=

Specifies the surveillance mechanisms to run in an XRF complex. The SURV= parameters in the DFSHSBxx members for the active and alternate subsystems must be the same. The default is RDS,LOG,LNK.

RDS
The alternate subsystem is to periodically check for time stamps in the RDS.

LOG
The alternate subsystem is to periodically check for new log records in the IMS log.

LNK
The alternate subsystem is to periodically check for signals coming across the ISC link.

ALL
All three surveillance mechanisms are to be used.

SWITCH=

Controls if the failure of a surveillance mechanism, VTAM, or the IRLM should cause the alternate subsystem to consider a takeover.

LNK
The absence of signals across the ISL link after a timeout period elapses causes IMS to consider a takeover.

LOG
The absence of a new log record in the IMS log after a timeout period elapses causes IMS to consider a takeover.

RDS
The absence of a new time stamp in the RDS after a timeout period elapses causes IMS to consider a takeover.

IRLM
An IRLM failure causes IMS to request a takeover.

TPEND
A VTAM failure that invokes the IMS TPEND exit routine causes IMS to request a takeover.

Specifying SWITCH=(LNK,LOG,RDS),(IRLM),(TPEND) causes IMS to request a takeover for any one of the following events:

- The LNK, LOG, and RDS surveillance mechanisms all indicate a failure.
- The IRLM fails.
- VTAM fails and causes the IMS TPEND exit routine to be invoked.

USERVAR=xxxxxxxx

Specifies the USERVAR that VTAM uses to associate a logon message with the VTAM application name (APPLID) of the current active IMS subsystem in an XRF complex. The USERVAR= in DFSHSBxx members for the active and alternate subsystem must be the same.

USERVAR= is required if you have VTAM and are not using MNPS for XRF terminal switching.

When MNPS=*name* is specified, the USERVAR specification is ignored, and IMS uses MNPS for XRF terminal switching.

DFSINTxx member of the IMS PROCLIB data set

Use the DFSINTxx member of the IMS PROCLIB data set to identify the preinitialization modules to receive control before MPR, IFP, BMP, JMP, and JBP dependent regions are initialized.

Control statement

Usage

The record format is:

Position
Contents

1-71 Module names, separated by a comma. The last name on a record is denoted by a comma followed by one or more blanks; the last name on the last record is followed by one or more blanks.

72-80 Ignored

A separate control statement for each module is required.

DFSJVMAP member of the IMS PROCLIB data set

Use the DFSJVMAP member of the IMS PROCLIB data set to map uppercase Java application names that are specified to IMS.

This member maps all the 8-byte or less uppercase Java application names that are specified to IMS in one of the following ways with the OMVS path name for the IMS solutions for Java development application .class file:

- LANG=JAVA,GPSB= parm on the APPLCTN macro
- LANG=JAVA,PSB= parm on the PSBGEN macro
- MBR= parm on the DFSJBP procedure

DFSJVMAP pertains to JBP and JMP regions only. The member name must be DFSJVMAP.

DFSJVMAP is a member contained in a data set concatenated in the PROCLIB DD. A sample member DFSJVMAP is supplied in the IMS sample library. This member is optional and is read during dependent region application scheduling, thus, making this member dynamic. You do not need to bring down IMS when adding application name mappings to this member for the changes to take effect.

Comments are supported for this member and must begin with an asterisk (*) in the first column. Each line in this member must not be longer than 72 bytes. There must be no space between the equal sign (=) and the beginning of the pathname.

DFSJVMAP example 1

In this example, APPLCTN GPSB=JAVAPL1 was specified during IMS system generation, and javapl1.class exists in /usr/IMSjava/applications. You would specify the following in DFSJVMAP:

```
***********************************************
* Path name for JAVAPL1                       *
***********************************************

JAVAPL1=/usr/IMSjava/applications/javapl1
```

or

```
JAVAPL1=javapl1
```

DFSJVMAP example 2

In this example, APPLCTN PSB=JAVAPL2 was specified during IMS system definition and its corresponding PSBGEN macro was specified as PSBGEN LANG=JAVA,PSBNAME=JAVAPL2 and javapl2.class exists in /usr/IMSjava/applications. But javapl2.java contains the package statement `package IMSjava.applications;`. You would specify the following in DFSJVMAP:

```
***********************************************
* Path name for JAVAPL2                       *
***********************************************
JAVAPL2=IMSjava/applications/javapl2
```

You must code the following required JVM option in the correct JVMOPMAS= member:

```
-Djava.class.path=/usr:...
```

DFSJVMMS member of the IMS PROCLIB data set

Use the DFSJVMMS member of the IMS PROCLIB data set to specify JVM options for the standalone JVM for JBP regions.

This is a sample member that demonstrates how to specify JVM options for the standalone JVM for JBP regions.

To define the JVM options:

1. Specify the -Xoptionsfile=<UNIX System Services file system JVM properties file>. The -Xoptionsfile allows you to specify pathnames that are greater than 255 characters in length on the -Djava.class.path option.
2. Specify the -Djava.class.path=<application class path> option in the options file.

Alternatively, you can specify the following directly in the member:

- -Djava.class.path=<application class path>.

Specify the path name (or path names) of your Java application class files. If your .class files are contained in a .jar file, the path name to the .jar file must be fully qualified, including the name of the .jar file.

Comments are supported for this options member. The comments begin with an asterisk (*) in the first column.

Each line in the options file must not be longer than 72 bytes, including the continuation mark. Use a greater-than symbol (>) at the end of the line as a continuation character.

If you do not use the -Xoptionsfile JVM option, path strings can be a maximum of 255 bytes (any characters over 255 bytes are ignored). A path string can be one path name or several path names. If you are specifying multiple path names, separate each by a colon (:).

For information about the format of this member, see the JVMOPMAS= parameter in "Parameter descriptions for IMS procedures" on page 448.

DFSMPLxx member of the IMS PROCLIB data set

Use the DFSMPLxx member of the IMS PROCLIB data set to make z/OS, IMS, or user-written program modules that are not automatically preloaded into the IMS control region resident in IMS regions.

Making some modules resident can improve throughput and response time for transactions frequently referred to if sufficient virtual storage is available with high-performance paging DASD. Ordinarily, preloading modules is advantageous only for MPPs and for message-driven Fast Path regions, and only when the preloaded modules are reentrant.

Program modules can be made resident in either an IMS region or LINKPACK. This topic explains how to make program modules permanently resident and provide some rules for doing so.

Making modules resident in a region

When making modules resident in a region, consider the following:

- Only serially reusable z/OS, IMS, or user-written program modules can be resident in a region.
- Only program modules used for transactions serviced by the region involved, should be made resident in that region. Program modules should be resident for only the duration of the region.
- z/OS and IMS modules that execute in a dependent region can reside in that dependent region.

Modules made resident in a region are in the region JOBPACK and are called without repeating the overhead of searching STEPLIB/JOBLIB, LINKPACK, and SYS1.LINKLIB. The overhead of fetching the module into virtual storage is encountered only at region initialization time.

Making modules resident in LINKPACK

In an operating environment where there are several batch regions or a combination of online and batch regions, it can be advantageous to place some of the frequently used IMS and access method modules in the operating system LPA. Programs made resident in LPA should be those that can be shared among all regions. This saves virtual storage space. The modules to be included must exist in either a LNKLSTxx member of SYS1.PARMLIB (or its concatenations, one of which should be IMS.SDFSRESL) or SYS1.SVCLIB.

Initial access of LPA resident program modules can be slow, because the region JOBPACK and STEPLIB/JOBLIB are searched before LPA is searched. Subsequent access can be at CPU speeds if the region JOBPACK has not been purged by z/OS virtual storage management (as when sufficient virtual storage is not available to satisfy a user GETMAIN for space). Although the modules are physically residing in LPA (and are shared among multiple regions), the overhead involved in searching program libraries and LPA is only experienced at region initialization time.

Although some real storage can be saved by putting the DL/I action modules into LPA, it is not recommended that an installation place IMS modules into the LPA. Doing so makes it difficult to execute different versions of IMS concurrently. Also, when maintenance is applied to modules in LPA, another initial program load of the z/OS system is required.

The IMS module preload function

The z/OS task under which modules are preloaded varies based on the IMS region type. Table 78 shows IMS region types and associated z/OS tasks.

Table 78. IMS region types and associated z/OS tasks

IMS region type	z/OS task
Control (CTL)	Physical Log
Message (MSG) non-reentrant modules	Program Control
Message MSG reentrant modules	Region Control
Batch Message (BMP)	Region/Program Control
Batch (DLI)	Region/Program Control
Fast Path (IFP)	Region/Program Control

Control statement

Complete the following steps to use this function to make program modules resident in a region or LINKPACK.

1. Use the IMS PROCLIB data set allocated before Stage 2 of IMS system definition and the z/OS utility IEBUPDTE to create members of the IMS PROCLIB data set that identify modules to be preloaded.

   Related reading: See *z/OS MVS Data Administration: Utilities*.

   Member names must be DFSMPLxx. The record format is:

   Position
   Contents

   1-71 Module names, separated by a comma

   72-80 Ignored

   The first name on a record can have leading blanks. The last name on a record is denoted by a comma followed by one or more blanks; the last name on the last record is followed by one or more blanks.

   No limit exists for the number of module names that can be specified. If a preloaded module has ALIAS names that are ordinarily invoked by z/OS LINK (that is, application programs in DLI BMPs and MPPs), those ALIAS names should also be specified for preload.

2. Bind the application program modules using the IMS reentrant DL/I language interface. The true attributes must be specified; for example, if reusable only, do not specify "RENT."

   The IMS DL/I language interface is not reentrant. Any IMS application programs that were designed to be reentrant or serially reusable can use the module preload function after being bound with the IMS language interface.
3. In the step execution JCL, do both of the following:
   - Insert a DD statement for IMS.PROCLIB with the ddname PROCLIB.
   - Specify the correct member of the IMS.PROCLIB data set in the EXEC statement parameter.

If an invalid member name is specified, or if the record format is incorrect, no modules are loaded but a message is issued and initialization continues.

The member name should be unique unless overridden by the IMS procedure.

For a description of the EXEC statement parameters, see "DFSMPR procedure" on page 547.

DFSORSxx member of the IMS PROCLIB data set

Use the DFSORSxx member of the IMS PROCLIB data set to define system-related startup parameters for the recovery manager in a DBCTL or DB/DC online environment.

This member is identified by the ORSMBR parameter in the EXEC statement. If the DFSORSxx member of the IMS PROCLIB data set is not provided, the installed recovery service is initiated with default values.

Related reading:

- For information about tailoring the DFSORSxx member for use with the IMS Online Recovery Service product, see *IMS Online Recovery for z/OS User's Guide*.
- For information about tailoring the DFSORSxx member for use with the IMS Database Recovery Facility V2 product, see *IMS Database Recovery Facility for z/OS User's Guide*.

DFSPBxxx member of the IMS PROCLIB data set

Use the three forms of the DFSPBxxx member of the IMS PROCLIB data set, DFSPBDBC, DFSPBDCC, and DFSPBIMS, to specify execution parameters for the DBCTL, DCCTL, and DB/DC control regions, respectively.

Each DFSPBxxx member of the IMS PROCLIB data set controls a different IMS environment:

DFSPBDBC
: DBCTL environment

DFSPBDCC
: DCCTL environment

DFSPBIMS
: DB/DC environment

| You can use the IMS Syntax Checker to modify these members of the IMS
| PROCLIB data set.

The DFSPBxxx member consists of one or more 80-character records. Only characters 1-71 are used as input. Data in characters 72-80 is ignored. A parameter must be specified in its entirety on a single line. You cannot continue to the next line to complete the parameter specification.

Each record contains either:

- A comment, marked by an '*' in statement column 1.
- One or more execution parameters in the following form:

  ```
  KEYWORD=value
  ```

  The parameters are separated by commas (,). Each parameter can be preceded by a space or the comma following the prior parameter value. You can have blanks on either, both, or neither side of the comma that separates keyword strings.
- One execution parameter

  ```
  KEYWORD=value
  ```

  followed by a comment. The comment is separated from the parameter by one or more blanks. A comment may contain any characters except the equal sign (=).

Example:

```
ARC=,
AUTO=,
```

DFSPBDBC

This member applies only to a DBCTL environment. It allows you to specify DBCTL control region execution parameters. The parameters that you specify in this member override those specified in the stage 1 macros.

You can place several DFSPBDBC members in the IMS PROCLIB data set by replacing the member name DFSPBDBC with DFSPBxxx, where *xxx* must be three alphanumeric characters. The RGSUF= keyword in the DBC procedure specifies the *xxx* suffix to be used during startup of the DBCTL control region.

For a list of parameters associated with DFSPBDBC, see "DBC procedure" on page 523. For a list of parameter descriptions, see "Parameter descriptions for IMS procedures" on page 448.

DFSPBDCC

This member applies only to a DCCTL environment. It allows you to specify DCCTL control region execution parameters. The parameters that you specify in this member override those specified in the stage 1 macros.

You can place several DFSPBDCC members in the IMS PROCLIB data set by replacing the member name DFSPBDCC with DFSPBxxx, where xxx must be three alphanumeric characters. The RGSUF= keyword of the DCC procedure provides the xxx suffix to be used during startup of the DCCTL control region.

For a list of parameters associated with DFSPBDCC, see "DCC procedure" on page 532. For a list of parameter descriptions, see "Parameter descriptions for IMS procedures" on page 448.

DFSPBIMS

This member applies only to a DB/DC environment. It allows you to specify DB/DC control region execution parameters. The parameters that you specify in this member override those specified in the stage 1 macros.

You can place several DFSPBIMS members in the IMS PROCLIB data set by replacing the member name DFSPBIMS with DFSPBxxx, where *xxx* must be three alphanumeric characters. The RGSUF= keyword of the IMS procedure provides the *xxx* suffix to be used during startup of the DB/DC control region.

For a list of parameters associated with DFSPBIMS, see "IMS procedure" on page 560. For a list of parameter descriptions, see "Parameter descriptions for IMS procedures" on page 448.

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

DFSRSRxx member of the IMS PROCLIB data set

Use the DFSRSRxx member of the IMS PROCLIB data set to specify the remote site recovery (RSR) options used by the online active and tracking subsystems in an RSR complex.

Multiple DFSRSRxx members can be present in the IMS PROCLIB data set.

The RSRMBR= parameter in the IMS, DBC, and DCC procedures identifies the member currently in effect. IMS batch jobs do not use the DFSRSRxx member.

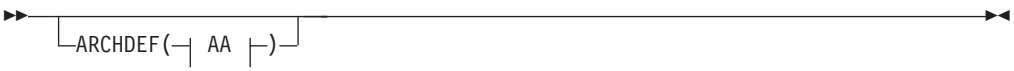
If the RSR feature is installed and you do not want to use it, you must override RSR enablement by specifying RSR(NO) in the DFSRSRxx member of the IMS PROCLIB data set.

An RSR complex has at least two DFSRSRxx members in effect at any time, one for the online active system and one for the tracking subsystem. The two members need not be identical, but the GSGNAME parameters must be the same. The parameters can have different meanings, depending on whether the subsystem is the active or the tracker.

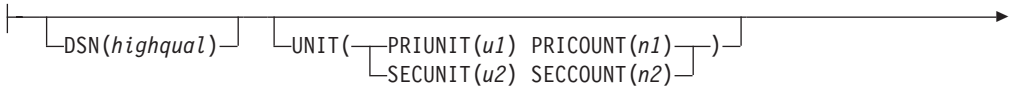
If RSRFEAT=RLT or RSRFEAT=DLT is not specified in the IMSCTRL macro during IMS system definition, the specification of a GSGNAME and TMINAME in the DFSRSRxx member is not accepted.

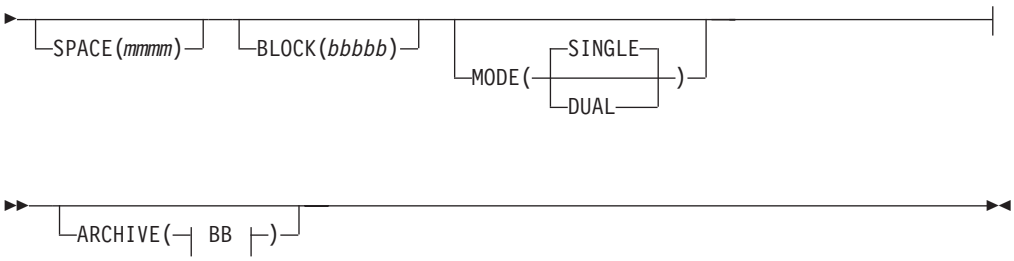
Syntax

All the RSR option parameters can occur in any order in the control statement. Multiple control statements can be provided.

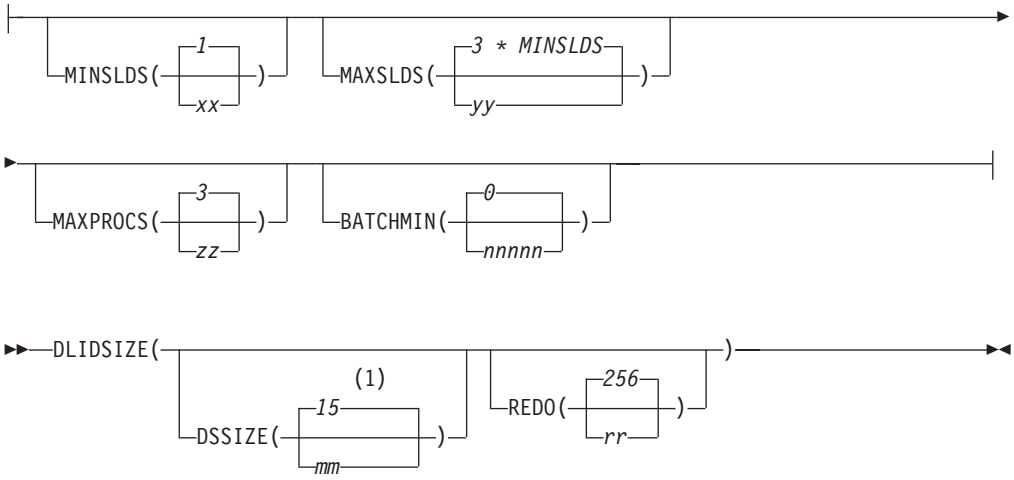


AA:



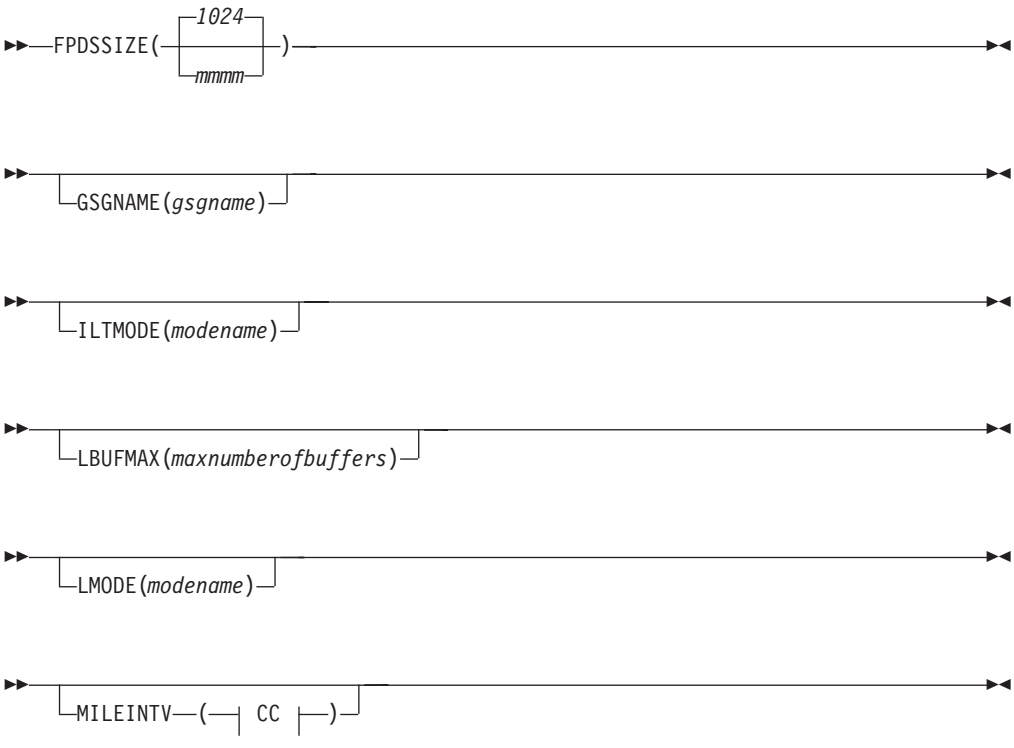


BB:

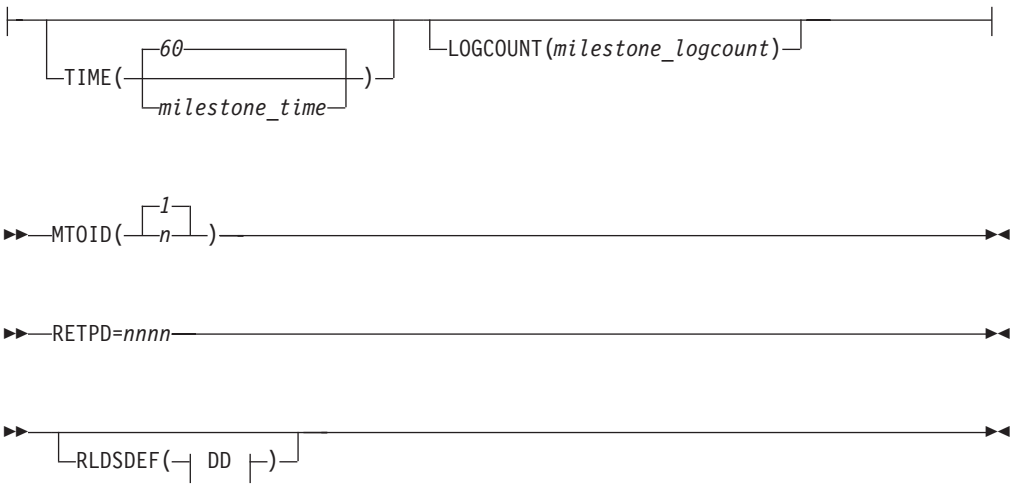


Notes:

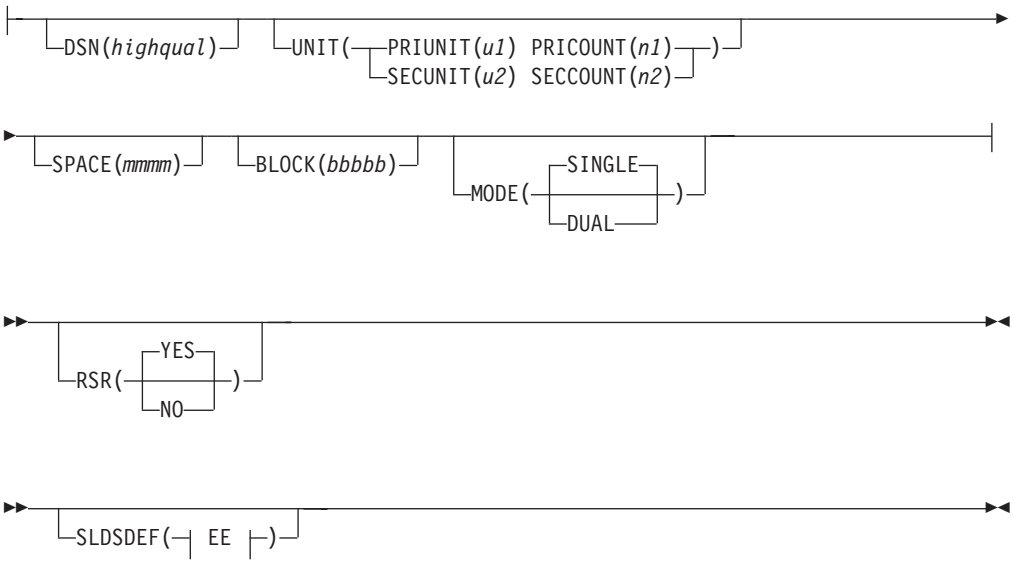
1 Or the number of PSTs, whichever is greater.



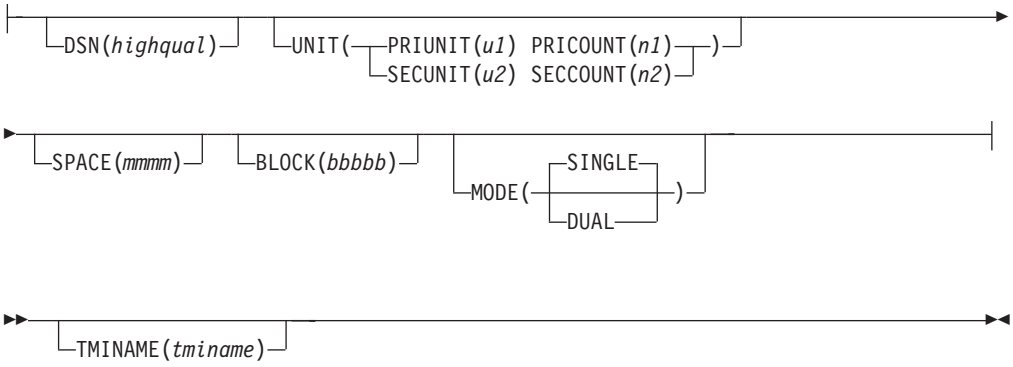
CC:


TIME(60 milestone_time)
LOGCOUNT(milestone_logcount)
MTOID(1 n)
RETPD=nnnn
RLDSDEF(DD)

DD:


DSN(highqual)
UNIT(PRIUNIT(u1) PRICOUNT(n1) SECUNIT(u2) SECCOUNT(n2))
SPACE(mmmm)
BLOCK(bbbbb)
MODE(SINGLE DUAL)
RSR(YES NO)
SLDSDEF(EE)

EE:


DSN(highqual)
UNIT(PRIUNIT(u1) PRICOUNT(n1) SECUNIT(u2) SECCOUNT(n2))
SPACE(mmmm)
BLOCK(bbbbb)
MODE(SINGLE DUAL)
TMINAME(tminame)

Usage

Because the parameters in the DFSRSRxx member are parsed by the TSO parser, the parameters must follow TSO parameter syntax rules. Any parsing errors are recorded in the SYSTSPRT data set. The record format is:

Position
Contents

1-80 Option parameters separated by commas

The first parameter in a record can have leading blanks. Delimiters between parameters can be spaces or commas. The last parameter in a record is denoted by a comma, followed by one or more blanks. The last parameter in the last record is followed by one or more blanks. Comments are delimited by `/*` and `*/`.

Parameters

ARCHDEF=
Defines the archived secondary system log data set (SLDS) characteristics for the tracking subsystem.

If ARCHDEF is not specified, no automatic archiving occurs.

DSN
Specifies the initial qualifiers of the data set name to be used for archive data sets.

No default data set name exists.

The character string specified must satisfy data set name syntax rules and be less than or equal to 27 characters in length. The actual data set name generated looks like: `highqual.ARCHc.Nnnnnnnn`, where `c` is `1` (one) for the primary SLDS archive and `2` (two) for the secondary SLDS archive, and `nnnnnnn` is a number generated to identify uniquely the data set name.

UNIT=
Specifies the primary and secondary unit types to be used when creating or reading SLDS archive data sets.

PRIUNIT=
Specifies the unit type to be used for the data set (if `MODE(SINGLE)` is specified) or for the primary data set (if `MODE(DUAL)` is specified). Example unit types are: SYSDA, TAPE, 3380, 3390.

No default unit type exists. You cannot specify different device types for the two units used in dual logging of a tracker log data set type.

PRICOUNT=
Specifies the number of devices to be allocated for the data set or primary data set. This is most meaningful for tape devices, because multiple tapes can be mounted, thus eliminating the need to wait for tape rewind and new tape mount.

The default for PRICOUNT is zero. Zero specifies a data set consisting of up to five volumes for tape data sets. Specifying any other value for PRICOUNT specifies multi-volume data sets in a maximum of 255 volumes per data set.

Recommendation: Specify `PRICOUNT(0)` and `SECCOUNT(0)` to limit the number of volumes to five or less when allocating archived SLDS during restart. Ensure the amount of data being archived fits into five volumes if tape data sets are being used.

SECUNIT=
Specifies the unit type to be used for the secondary data set. Example unit types are: SYSDA, TAPE, 3380, 3390.

No default unit type exists. You cannot specify different device types for the two units used in dual logging of a tracker log data set type.

SECCOUNT=
Specifies the number of devices to be allocated for the secondary data set. This is most meaningful for tape devices, because multiple tapes can be mounted, thus eliminating the need to wait for tape rewind and new tape mount.

The default for SECCOUNT is 0 (zero). A specification of 0 results in data sets consisting of up to five volumes for tape data sets. Any other value for SECCOUNT results in a maximum of 255 volumes per data set.

SPACE=
Specifies the space allocation in megabytes.

No default space allocation exists.

For DASD, the value specified is converted to blocks to give an integral number of cylinders. For tape units, the value specified controls switching from one data set to the next.

For tape units, if your space allocation is small enough to cause tape end-of-volume before the physical end of the tape, the data set overflows to another volume. The space allocation must be greater than the product of the values specified in SLDSDEF SPACE and ARCHIVE MINSLDS.

```
((SLDSDEF SPACE) * MINSLDS <= (ARCHDEF SPACE))
```

BLOCK=
Specifies the block size to be used for all SLDS archive data sets. No default block size exists.

This number should be a fairly large number to ensure good RSR performance. A block size of 32 KB is recommended for tape and half-track for DASD (22 KB for 3380 and 26 KB for 3390 devices). The block size must be at least as large as the largest incoming IMS log record. The range of valid values is 4096 to 32760.

MODE
Specifies whether SINGLE or DUAL archive SLDS logging is required. The default is SINGLE.

ARCHIVE
Controls automatic archiving of SLDS data sets created by a tracking subsystem. If ARCHIVE is not specified, no automatic archiving occurs.

Related reading: See *IMS Version 11 System Administration* for more information about how to use ARCHIVE to control automatic archiving of tracking SLDSs.

MINSLDS
Specifies the minimum number of tracking SLDSs to be archived for each active subsystem.

The range of valid values is 1 to 999. The default is 1.

MAXSLDS
Specifies the maximum number of tracking SLDSs to be maintained for each active subsystem before an archive operation is triggered.

The range of valid values is the MINSLDS value to 999. The default is 3 times MINSLDS.

MAXPROCS
Controls the number of concurrent archive operations.

The range of valid values is 1 to 99. The default is 3.

BATCHMIN
Specifies the minimum number of log records required before the system initiates automatic archiving. BATCHMIN is valid for log records that originate from active site batch IMS jobs. If a batch IMS job generates fewer log records than specified by the BATCHMIN parameter, the tracking SLDS is not archived.

The range of valid values is 1 to 999999.

If BATCHMIN is not specified, all tracking SLDS from batch IMS jobs are automatically archived when automatic archive is active.

DLIDSIZE=
Specifies how the DL/I database tracker uses its data space. The specification indicates the amount of virtual storage for the DL/I tracking data space that can be backed by real and expanded storage for those times when the database tracker cannot keep up with the active subsystems.

If you specify a number larger than the available storage for the database tracker, the machine can experience excessive paging, thus reducing the efficiency of the tracker. If you specify a number smaller than the available storage for the database tracker, the tracker might need to perform catch-up processing more often than is truly necessary.

DSSIZE=
Specifies the maximum number of megabytes for DL/I database tracking data space usage.

The range of valid values is 15 to 1600. The default is either 15 or the number of PSTs (set by the PST parameter on the EXEC PARM= statement), whichever is greater.

REDO=
Specifies an estimated average size, in bytes, for the active subsystem's DL/I redo (type X'50') log records. The number specified is rounded up to the nearest power of two.

The range of valid values is 128 to 4096. The default is 256.

FPDSSIZE
Specifies the threshold value size of storage, in megabytes, used for Fast Path database tracking data spaces. When data space usage reaches the value specified, the stored update data is written to disk. Actual storage size can be larger than the specified value, because additional data is normally processed while data is written to disk.

The range of valid values is 1 to 1024. The default is 1024.

GSGNAME=
Specifies the one- to eight-character global service group name to be used for the particular RSR complex. This GSG name overrides any specified during system definition (in the IMSCTRL macro).

Both the active and tracking subsystems must specify the same GSG name, either during system definition or in the DFSRSRxx member.

No default GSG name exists.

ILTMODE=
Specifies the one- to eight-character mode name to be used by the Transport Manager Subsystem (TMS) for isolated log sender conversations.

If ILTMODE is not specified, the default logon mode name defined in the DLOGMOD of the TMS's VTAM APPL definition is used.

LBUFMAX=
Specifies the maximum number of buffers that the active TMS obtains for transmitting log data. TMS buffers are 52 bytes larger than IMS log buffers and are obtained from IMS private storage. When LBUFMAX is not specified or cannot be specified (such as with IMS batch jobs), a limit of 10 times the number of IMS log buffers is used.

Restriction: The LBUFMAX parameter applies only to the active system.

LMODE=
Specifies the one- to eight-character mode name to be used by the TMS for the logger and log router conversations.

If LMODE is not specified, the TMS-implementation defined default logon mode name, TMDEFLT, is used.

MILEINTV=
Specifies the interval between milestones for tracking subsystems. Milestones are used by the tracking subsystem for tracker restart. Tracker restart is not a remote takeover. They keep track of the current positions on the SLDSs containing active IMS log records and of which log records have been applied to the shadow databases.

The milestone can be triggered either by time interval or by number of log records received from all active subsystems. You can specify either or both of these methods for triggering milestones, but the timer (default or specified value) is used in all cases. If both are specified, the first one reached (timer or log record count) triggers the milestone and both counters are reset to zero.

TIME=
Specifies the maximum elapsed time between milestones. The range of valid values is 10 to 600 seconds. The default is 60 seconds.

LOGCOUNT=
Specifies the maximum amount of log data that can be received from all active subsystems between milestones.

The value can be specified as either `nnnK` (indicating the number or records of data in thousands) or as `nnnM` (indicating the number of records of data in millions). The range of valid values is 1K to 2047M (or 2096128K) records. No default milestone log count exists.

MTOID=
Specifies which terminals to use for the IMS master terminal and secondary

master terminal. The terminals specified must be listed on the NAME= parameter of the TERMINAL macro. MTOID also controls which password is used, The password must be included on the list in the PASSWD= parameter of the COMM macro.

This parameter is only valid for tracking subsystems.

Valid values are 1, 2 or 3, depending on the number of names specified in the TERMINAL and COMM macros. The default is 1. The VTAM APPLID is overridden by the APPLIDn= startup parameter and APPLID3 is always used for the tracking subsystem.

You must take care not to cause the same IMS VTAM APPLID to be opened for more than one IMS subsystem. If necessary, the APPLID1, APPLID2, and APPLID3 parameters should all be overridden to ensure unique APPLIDs.

RETPD=

Specifies the retention period, *nnnn*, for a tracking log data set, where *nnnn* is a one- to four-digit value representing a number of days. The system adds the number you specify to the current date to arrive at an expiration date. If you do not specify RETPD, the default is either the system default for this class of data set or 0. Use of this optional parameter helps reduce the possibility of accidental deletion of the tracking log data set. After the retention period expires, another data set can write over the tracking log data set. You cannot use the RETPD parameter to change the expiration date of existing tracking log data sets if they are managed by SMS.

You can also specify the retention period of the tracking log data set using the EXPDT parameter on standard JCL. Code the RETPD parameter when you want to specify a retention period for the tracking log data set or when you want to override the retention period defined in the data class for new tracking log data sets.

The RETPD parameter cannot have a null value.

RLDSDEF=

Defines the recovery log data set (RLDS) characteristics for the tracking subsystem.

No default RLDS definition exists. If RLDSDEF is not specified, no RLDS is created at the tracking site.

DSN=

Specifies the initial qualifiers of the data set name to be used for RLDS data sets.

No default data set name exists.

The character string specified must satisfy data set name syntax and be less than or equal to 27 characters in length. The actual data set name generated looks like: `highqual.RLDSc.Nnnnnnnn`, where `c` is `1` for the primary RLDS data set and `2` for the secondary RLDS, and `nnnnnnn` is a number generated to uniquely identify the data set name.

UNIT=

Specifies the primary and secondary unit types to be used when creating or reading RLDS data sets.

PRIUNIT=

Specifies the unit type to be used for the data set (if `MODE(SINGLE)` is specified) or for the primary data set (if `MODE(DUAL)` is specified). Example unit types are: SYSDA, TAPE, 3380, 3390.

No default unit type exists. You cannot specify different device types for the two units used in dual logging of a tracker log data set type.

PRICOUNT=
Specifies the number of devices to be allocated for the data set or primary data set. Specifying PRICOUNT is most useful for tape devices, so that multiple tapes can be mounted, thus eliminating the need to wait for tape rewind and new tape mount.

The default for PRICOUNT is 0 (zero). A specification of 0 results in data sets consisting of up to five volumes for tape data sets. Any other value for SECCOUNT results in a maximum of 255 volumes per data set.

Recommendation: Specify `PRICOUNT(0)` and `SECCOUNT(0)` to limit the number of volumes to five or less when allocating archived RLDS during restart. Ensure the amount of data being archived fits into five volumes if tape data sets are being used.

SECUNIT=
Specifies the unit type to be used for the secondary data set. Example unit types are: SYSDA, TAPE, 3380, 3390.

There is no default unit type. You cannot specify different device types for the two units used in dual logging of a tracker log data set type.

SECCOUNT=
Specifies the number of devices to be allocated for the secondary data set. This is most meaningful for tape devices because multiple tapes can be mounted, thus eliminating the need to wait for tape rewind and new tape mount.

The default for SECCOUNT is zero. A specification of zero results in data sets consisting of up to five volumes for tape data sets. Any other value for SECCOUNT results in a maximum of 255 volumes per data set.

SPACE=
Specifies the space allocation, in megabytes, for each RLDS data set.

The range of valid values is 1 to 4095. No default space allocation exists.

BLOCK=
Specifies the block size to be used for all RLDS data sets. No default block size exists.

This value should be a fairly large number to ensure good RSR performance. A block size of 32 KB is recommended for tape archiving and half-track for DASD (22 KB for 3380 and 26 KB for 3390 devices). The block size must be at least as large as the largest incoming IMS log record. The range of valid values is 4096 to 32760.

MODE=
Specifies whether SINGLE or DUAL RLDS logging is required. The default is SINGLE.

RSR=
Specifies whether the active IMS subsystem is to be enabled for RSR processing. The default is YES.

With RSR(YES), you must also specify a GSG name and a TMI name, either in the DFSRSRxx member or in the IMSCTRL macro

Important: If the RSR feature (RLT, DLT, or both) is installed and the GSGNAME keyword is specified on the IMSCTRL macro, but you do not want RSR enabled for this particular IMS, you **must** specify RSR(NO). See the keyword descriptions under "IMSCTRL macro" on page 340 for information about the GSGNAME keyword.

SLDSDEF=
Defines the shadow system log data set (SLDS) characteristics. This parameter is required for all tracking subsystems. No default for SLDS definition exists.

DSN=
Specifies the initial qualifiers of the data set name to be used for SLDS data sets. No default data set name exists.

The character string specified must satisfy data set naming conventions and be less than or equal to 27 characters in length. The actual data set name generated looks like: `highqual.SLDSc.Nnnnnnnn`, where `c` is `1` for the primary SLDS data set and `2` for the secondary SLDS, and `nnnnnnn` is a number generated to uniquely identify the data set name.

UNIT=
Specifies the primary and secondary unit types to be used when creating or reading SLDS data sets.

PRIUNIT=
Specifies the unit type to be used for the data set (if `MODE(SINGLE)` is specified) or for the primary data set (if `MODE(DUAL)` is specified). Example unit types are: SYSDA, TAPE, 3380, 3390.

No default unit type exists. You cannot specify different device types for the two units used in dual logging of a tracker log data set type.

PRICOUNT=
Specifies the number of devices to be allocated for the data set or primary data set. This is most useful for tape devices because multiple tapes can be mounted, thus eliminating the need to wait for tape rewind and new tape mount.

The default for PRICOUNT is 0 (zero). Specifying 0 results in data sets consisting of up to five volumes for tape data sets. Any other value for PRICOUNT results in a maximum of 255 volumes per data set.

SECUNIT=
Specifies the unit type to be used for the secondary data set. Example unit types are: SYSDA, TAPE, 3380, 3390.

No default unit type exists. You cannot specify different device types for the two units used in dual logging of a tracker log data set type.

SECCOUNT=
Specifies the number of devices to be allocated for the secondary data set. This is most useful for tape devices, because multiple tapes can be mounted, thus eliminating the need to wait for tape rewind and new tape mount.

The default for SECCOUNT is 0 (zero). Specifying 0 results in data sets consisting of up to five volumes for tape data sets. Any other value for SECCOUNT results in a maximum of 255 volumes per data set.

Recommendation: Specify `PRICOUNT(0)` and `SECCOUNT(0)` to limit the number of volumes to five or less when allocating archived SLDS

during restart. Ensure the amount of data being archived fits into five volumes if tape data sets are being used.

If you are using database readiness level on your tracking subsystem, the recommendation is that the UNIT be a direct access device, because the volume mounting characteristics of tape devices can result in unsatisfactory operation.

SPACE=
Specifies the space allocation, in megabytes, for each SLDS data set.

The range of valid values is 1 to 4095. No default space allocation exists.

BLOCK=
Specifies the block size to be used for all SLDS data sets. No default block size exists.

This value should be a fairly large number to ensure good RSR performance. A block size of 32 KB is recommended for tape archiving and half-track for DASD (22 KB for 3380 and 26 KB for 3390 devices). The block size must be at least as large as the largest incoming IMS log record. The range of valid values is 4096 to 32767.

MODE=
Specifies whether SINGLE or DUAL SLDS logging is required. The default is SINGLE.

TMINAME=
Specifies the one- to four-character TMS instance name that is used to identify a particular TMS in the RSR complex.

If specified, TMINAME overrides the TMINAME= parameter in the IMSCTRL macro.

If no *tminame* is specified, IMS uses a series of blanks as the default *tminame*, and associates IMS with the TMS that uses a default instance name.

TRKMODS=
Specifies the tracking subsystem's MODSTAT DFSMDA member names for tracking MODSTAT status of the active subsystem.

This parameter is only valid for active IMS subsystems, and use of the parameter prohibits sharing of DFSRSRxx members between active subsystems.

The DD names specified must be included in the dynamic allocation list at the tracking site, and the data sets defined by these DD names need to be available to the tracking subsystem.

A tracking subsystem that was previously an active subsystem is not required to use the same DD name as was specified for it when it was an active subsystem.

MODSTAT=
Specifies the DFSMDA member name used to track the MODSTAT data set.

This parameter is only valid for IMS active subsystems.

No default member name exists. If MODSTAT is not specified, no MODSTAT tracking is done.

MODSTAT2=
Specifies the DFSMDA member name used to track the MODSTAT2 data set.

This parameter is only valid for XRF alternate subsystems.

No default member name exists. If MODSTAT2 is not specified, no MODSTAT2 tracking occurs.

DFSSPMxx member of the IMS PROCLIB data set

Use the DFSSPMxx member of the IMS PROCLIB data set to override the default buffer definitions for the storage pools managed by the DFSPOOL storage manager.

The storage pools that are managed by the DFSPOOL storage manager include:

AOIP
CESS
CIOP
CMDP
DYNP
EMHB
FPWP
HIOP
LUMC
LUMP

xx is the 2-character suffix specified in the SPM= startup parameter.

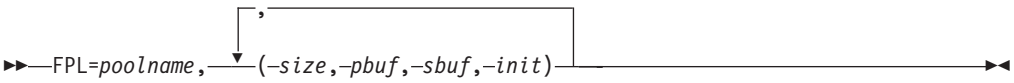
Default storage pool definitions are generated by IMS. Each definition contains information about the pool including the default buffer definitions.

Table 79. Storage pools and their locations

Storage pool	Where the pool is allocated
AOIP	Extended private
CESS	Subpool 231 ECSA
CIOP	24-bit private
CMDP	Extended private
DYNP	Extended private
EMHB	Subpool 231 ECSA
EPCB	Subpool 231 ECSA
FPWP	Extended private
HIOP	Extended private
LUMC	Subpool 231 ECSA
LUMP	Extended private

The buffer size definitions contain the buffer sizes that the storage manager is allowed to choose from when allocating a buffer from the pool. For each buffer size, the definition specifies how many buffers to obtain in the primary block, how many buffers to obtain in secondary blocks, and whether to obtain the primary block when the pool is allocated. If the primary block is obtained during initialization, it is not released during compression.

Syntax



Restrictions

If you establish an FDBR region, all pool names that you specify are ignored. The FDBR region internally specifies the following values for all pools regardless of whether they are explicitly specified by the control statements or defined as default in the system definition process:

2 Number of buffers in the primary storage allocation

2 Number of buffers in secondary storage allocation

N Primary storage allocation during IMS initialization =

Usage

The buffer definitions shown in Table 80 are the existing default definitions.

For all pools, 16 bytes are added to the buffer sizes for prefix and suffix information. For AOIP, CIOP, CMDP, HIOP, EMHB, LUMP, LUMC, and SPAP pools, an additional 8 bytes are added for the overlay detection constant.

IMS Fast Path (IFP) dependent regions also use EMHB pools. IFP dependent regions use a value equal to the largest EMHB as the value for the EPSTESRT buffer. This amount of storage is used in addition to the number of buffers in the primary or secondary storage allocation that you define for EMHB pools in member DFSSPMxx. For example, if you had 100 terminals using 1 buffer of primary storage each, then you would specify 100 as the number of buffers in the primary storage allocation. But if you also had 50 IFP dependent regions using EMHB, the actual number of buffers used in the primary storage would be 150.

For EMHB, the buffer size includes the prefix and data portion. For information about the prefix length, see the EMHBHL field of the expedited message handler block control block that is mapped by the DBFEMHB macro.

Table 80. Default buffer definitions for storage pools managed by the DFSPOOL storage manager

AOIP								
	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Buffer 5	Buffer 6	Buffer 7	Buffer 8
Size	48	132	256	576	1048	2096	4192	32 768
Primary Allocation	50	500	100	32	16	8	4	4
Secondary Allocation	50	1500	100	32	8	4	2	2
Obtain primary allocation?	Yes	No	No	No	No	No	No	No

CIOP								
	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Buffer 5	Buffer 6	Buffer 7	Buffer 8
Size	256	512	1024	2048	4096	8192	16 384	32 768
Primary Allocation	64	32	32	32	16	8	4	4
Secondary Allocation	32	16	16	16	8	4	2	2
Obtain primary allocation?	Yes	No	No	No	No	No	No	No

CMDP, HIOP, EMHB								
	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Buffer 5	Buffer 6	Buffer 7	Buffer 8
Size	256	512	1024	2048	4096	8192	16 384	32 768
Primary Allocation	64	64	32	32	16	8	4	4
Secondary Allocation	32	32	16	16	8	4	2	2
Obtain primary allocation?	No	No	No	No	No	No	No	No

CESS								
	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Buffer 5	Buffer 6	Buffer 7	Buffer 8
Size	128	256	512	1024	2048	4096	8192	16 384
Primary Allocation	32	32	32	32	32	16	8	4
Secondary Allocation	16	16	16	16	16	8	4	2
Obtain primary allocation?	Yes	No	No	No	No	No	No	No

DYNP									
	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Buffer 5	Buffer 6	Buffer 7	Buffer 8	Buffer 9
Size	256	512	768	1024	2048	3072	4096	8192	32768
Primary Allocation	32	32	32	32	16	12	8	4	4
Secondary Allocation	16	16	16	16	8	12	8	2	2
Obtain primary alloca-tion?	No	No	No	No	No	No	No	No	No

FPWP								
	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Buffer 5	Buffer 6	Buffer 7	Buffer 8

FPWP								
Size	25	51	102	2048	4096	8192	16 384	32 768
Primary Allocation	6	6	3	32	16	8	4	4
Secondary Allocation	3	3	1	16	8	4	2	2
Obtain primary allocation?					No	No	No	No

LUMP, LUMC								
	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Buffer 5	Buffer 6	Buffer 7	Buffer 8
Size	128	256	512	1024	2048	3064	4096	33 024
Primary Allocation	32	32	32	32	16	12	8	4
Secondary Allocation	32	16	16	16	8	12	8	2
Obtain primary allocation?	No	No	No	No	No	No	No	No

You can define a maximum of 32 different buffer sizes that can be used to satisfy requests for storage. The size of the buffer that is used to satisfy a request is determined on a "best-fit" basis.

The FPL= statement is used to specify buffer definition overrides. Multiple buffer definitions are allowed on a single statement. You can have one or more FPL= statements within the DFSSPMxx member. If the member contains more than 32 buffer definitions for a single pool, only the first 32 are used. Remaining definitions are ignored.

DFSSPMxx is not required to initialize IMS. It is intended only to override existing IMS storage pool definitions; therefore, no default member is supplied.

If an FPL= is incorrectly specified, message DFS0639W is issued. One message is issued for each occurrence of an invalid parameter.

Parameters

FPL=
Is the required keyword for IMS storage pool buffer override definitions. This keyword must be in position 1. The FPL= statement cannot be continued; however, more than one statement is allowed per pool.

poolname
Is the four-character name of the IMS storage pool. The valid names are:

AOIP
CESS
CIOP
CMDP
DYNP

EMHB
FPWP
HIOP
LUMC
LUMP

size
Specifies the buffer size, in bytes. The value must be a one- to five-digit number between 8 and 65536. The buffer size is rounded up to the next multiple of 8. If the rounded value is not unique, it is discarded. You can specify values in 1K increments; that is, 1K, 2K, 3K, up to 64K.

pbuf
Specifies the number of buffers in the primary storage allocation. The value must be a one- to five-digit number between 2 and 65535. If the size of the primary allocation exceeds the upper expansion limit, the default is 2.

sbuf
Specifies the number of buffers in secondary storage allocations. The value must be a one- to five-digit number between 2 and 65535. If the size of the primary allocation exceeds the upper expansion limit, the default is 2.

init
Specifies whether (Y) or not (N) the primary storage allocation is obtained during IMS initialization. If the primary storage allocation is not obtained during initialization, it is not obtained until it is necessary to satisfy a buffer request. If an upper expansion limit is specified for the pool, and the size of either the primary or secondary storage allocation exceeds the upper expansion limit, the upper expansion limit is set to the default value of 2G-1.

Examples

The following are examples of how the DFSSPMxx member of the IMS PROCLIB data set can be used to override the default buffer definitions for the CIOP pool.

```
FPL=CIOP,(248,20,15,Y),(500,20,10,N),(1016,15,8,Y)
FPL=CIOP,(2040,15,8,N),(4088,10,5,N),(8184,10,5,N)
FPL=CIOP,(16K,8,4,N),(32K,4,2,N)
```

The buffer sizes being defined range from 248 to 32 768 (32K) bytes. Each buffer size is rounded up to the next multiple of 8, and 8 bytes are added for internal processing. Table 81 shows the values used by the Storage Manager when allocating CIOP storage for the definition in the example.

Table 81. Primary and secondary buffer allocations

Buffer size	Primary buffer allocation	Secondary buffer allocation	Obtain primary allocation?
256	20	15	Y
512	20	10	N
1024	15	8	Y
2048	15	8	N
4096	10	5	N
8192	10	5	N
16392	8	4	N
32776	4	2	N

DFSSQxxx member of the IMS PROCLIB data set

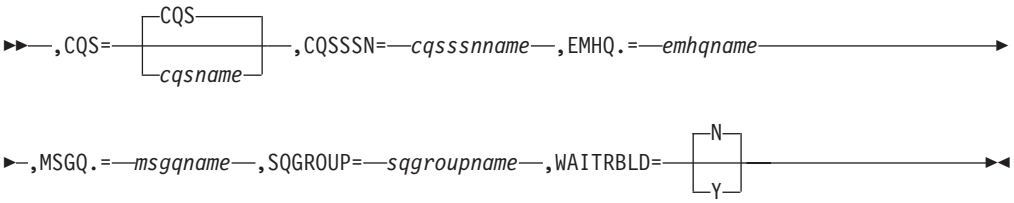
Use the DFSSQxxx member of the IMS PROCLIB data set to specify parameters related to the shared message queues and the CQS address space in DB/DC and DCCTL environments.

You can find an example of the DFSSQxxx member of the IMS PROCLIB data set, DFSSQ999, in library SDFSSLIB.

The parameters for DFSSQxxx can also be specified on the DFSDFxxx member of the IMS PROCLIB data set. See "DFSDFxxx member of the IMS PROCLIB data set" on page 672 for more information.

Syntax

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.



Usage

A DFSSQxxx member consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a "*" or "#" in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between " /* "and "*/", for example, /* PROCLIB comments */. Values coded in this member of the IMS PROCLIB data set are case sensitive. In general, you should use uppercase for all parameters.

Errors encountered during DFSSQxxx member processing cause IMS initialization to terminate.

Parameters

CQS=

Specifies the one- to eight-character name of the member of the IMS PROCLIB data set that contains one of the following:

- The procedure for the CQS address space.

  When shared message queues are requested the IMS control region automatically starts the CQS procedure during IMS initialization. The default procedure name is CQS.
- The START command.

  The START command begins in column 1 with the characters START. The START command and its parameters must not extend beyond column 71.

When shared message queues are requested, the IMS control region issues the user-specified START command to start the CQS address space.

CQSSSN=

Specifies the name for the CQS address space. The value must be 1 - 4 alphanumeric characters. IMS uses this name to connect to the proper CQS address space. When connecting IMS to CQS, you must specify the same value on CQSSSN= and on the SSN= parameter of the CQSIPxxx member of the IMS PROCLIB data set for the target CQS. The parameter is required and no default exists.

As many as 32 different IMS control regions can specify the same CQSSSN= parameter.

EMHQ=

Specifies the one- to sixteen-character name of the primary structure that contains the shared expedited message handler queues (EMHQs). In a shared-queues environment when Fast Path is installed, the presence of this statement indicates that the EMHQ structure, along with its associated log structure, checkpoint data set, and structure recovery data set are required. To disable shared expedited message handler processing, you can remove the EMHQ statement from this member, and the EMHQ structure, along with its associated log structure, checkpoint, and structure recovery data sets, are not required.

If you do not want to utilize an EMHQ structure for an IMS system, you need to delete the STRUCTURE statement for the EMHQ from the CQSSLxxx member of the IMS PROCLIB data set. If you do not want to utilize an EMHQ structure for all the IMS systems in a sysplex, you also need to delete the STRUCTURE statement for the EMHQ from the CQSSGxxx member of the IMS PROCLIB data set.

When you do not use an EMHQ structure, you should also do the following:

- In the CRFM policy, delete the STRUCTURE definitions for the EMHQ structure and its CQS log.
- In the z/OS Logger Structure Definition policy, delete the LOGSTREAM definition for CQS log for an EMHQ structure.
- If you are using an IMS automation program for any command that deals with an EMHQ structure, delete the command in the automation.

If you specify an EMHQ= structure, you must

- specify shared message queues on the MSGQ= parameter.

  IMS requires the message queues to process system messages that result from Fast Path activity. If the EMH queues are built in a coupling facility structure, the message queues must also be in a structure.
- Specify FP=Y as a control region execution parameter.

The name you specify for EMHQ= must not be the same as the name you specify for MSGQ=. The name must also be specified to CQS on the STRNAME= parameter in the CQSSLxxx and CQSSGxxx members of the IMS PROCLIB data set.

This parameter is valid only when Fast Path is installed.

MSGQ=

Specifies the one- to sixteen-character name of the primary structure that contains the shared message queues. If the message queues are built in a coupling facility structure, the EMH queues, if used, must also be in a structure.

The name you specify for MSGQ= must not be the same as the name you specify for EMHQ= The name must also be specified to CQS on the STRNAME= parameter in the CQSSLxxx and CQSSGxxx members of the IMS PROCLIB data set.

SQGROUP=
Specifies a one- to five- character identifier. IMS concatenates this identifier to DFS to create the z/OS cross-system coupling facility IMS shared queues group name. You must specify the same identifiers for all IMS subsystems that share the same set of structures. You can use the same identifier for the CQSGROUP= parameter in the CQSIPxxx member of the IMS PROCLIB data set. This parameter is required and there is no default.

WAITRBLD=Y|N
Specifies whether activity against the EMHQ structure waits while the structure is being rebuilt. If you specify Y (yes), all activity against the EMHQ structure waits until the structure rebuild completes. If you specify N (no), activity against the EMHQ structure continues while CQS rebuilds the structure.

The value specified for this keyword remains in effect for the life of the structure and cannot be changed.

Restriction: This parameter does not apply to the MSGQ. The MSGQ must wait for rebuild to complete before activity can resume.

If you change the WAITRBLD= parameter, you must first deallocate the fast path structure.

Related concepts:

Chapter 14, "IMS Syntax Checker," on page 283

DFSVSMxx member of the IMS PROCLIB data set

Use the control statements in the DFSVSMxx member of the IMS PROCLIB data set to define settings for buffer pools, trace options, DASD logging, coupling facility structures, IRLM lock timeout, and transactions in a HALDB partition.

In an IMS batch system, these control statements are put in a data set with ddname DFSVSAMP. The data set must contain blocked or unblocked 80-character records.

Recommendation: Do not place DFSVSAMP in the same PDS as your user application files. This causes batch jobs to abend with ABEND0C4 in DFSRTM00.

In an IMS online system, these control statements are put in the IMS.PROCLIB data set in member DFSVSMxx. The VSPEC symbolic parameter in the IMS, DBC, and DCC procedures is used to specify the suffix (xx) of the DFSVSMxx member to be used. During stage 2, IMS creates a default member with suffix 00. If a suffix in not specified on the startup procedure, the default member is used. The default values are also used if the control statements in the startup procedure are missing or invalid.

The control statements in this member are processed during IMS initialization. The control statements allow you to perform various tasks.

Related concepts:

"VSAM subpool definition" on page 180

"OSAM subpool definition" on page 180

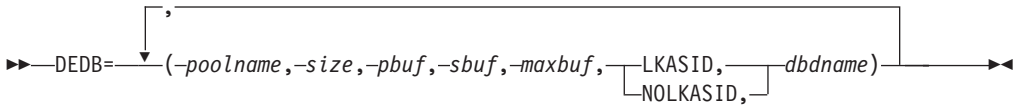
"Specifying VSAM and OSAM subpools" on page 182

Defining Fast Path DEDB buffer pools for single-area structures

This section describes control statements in the DFSVSMxx member that you use to define Fast Path DEDB buffer pools for single-area structures.

Control statements

The format of the control statements for defining a Fast Path buffer pool for a single-area structure is:



Parameter descriptions

DEDB=
Is the required keyword for defining Fast Path buffer pools. It must be in the first position of the control statement.

poolname
The one- to eight-character name for the pool. The poolname is used as identification on display terminals and reports.

size
A three- to five-digit number specifying the size of the pool. All the standard DEDB supported buffer sizes are supported. You can express sizes numerically, or as standard CI sizes, 1K, 2K, 4K.

pbuf
The primary buffer allocation. Specify a value from 1 to 32766.

sbuf
The secondary buffer allocation. Specify a value from 1 to 9999. This secondary allocation is used when the primary allocation runs low.

maxbuf
The maximum number of buffers allowed for this pool. It is a combination of PBUF plus some iteration of SBUF. The maximum value allowed is 32767. If maxbuf is specified greater than 32767, it defaults to 32767.

LKASID | NOLKASID
Specifies whether buffer lookaside is to be performed on read requests for this area.

For VSO DEDB areas that use a single-area structure, this parameter is optional. The LKASID value that is defined when using the DBRC batch command INIT.DBDS or CHANGE.DBDS is the default value. The DBRC LKASID value takes precedence over the DFSVSMxx PROCLIB member's DEDB LKASID value. If you do not specify a value using one of the DBRC batch commands or in the DFSVSMxx PROCLIB member's DEDB LKASID parameter, NOLKASID is used.

For VSO DEDB areas that use a multi-area structure, LKASID or NOLKASID must be specified by using the DFSVSMxx PROCLIB member. Specifications using the DBRC batch command INIT.DBDS or CHANGE.DBDS are ignored.

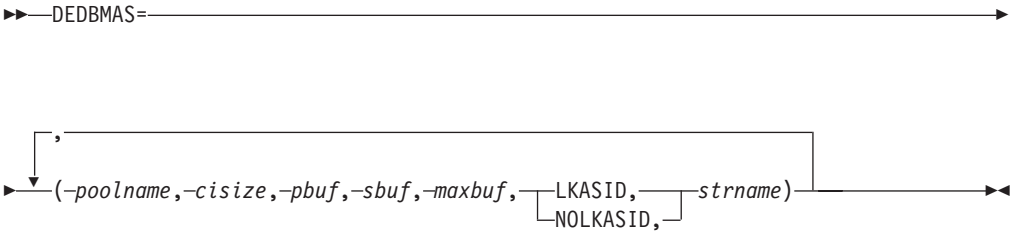
dbdname
Associates a pool to a specific area or DBD. If dbdname is an area name, then the pool is used only by that area. If dbdname is a DBD name, then the pool is used by all the areas within that DBD.

Defining Fast Path DEDB buffer pools for multi-area structures

You can enable all DEDB areas that share a single coupling facility structure to use the same buffer pool. This section describes control statements in the DFSVSMxx member that you use to define Fast Path DEDB buffer pools for multi-area structures.

Control statements

The format of the control statements for defining a Fast Path buffer pool for a multi-area structure is:



Parameter descriptions

DEDBMAS=
Defines a Fast Path buffer pool that is to be used by a multi-area structure.

poolname
The 1- to 8-character name for the pool. The *poolname* is used as identification on display terminals and reports.

cisize
The control interval size of the area. All areas that share a multi-area structure must have the same control interval size. If there is a discrepancy between the control interval size of the area used in creating the structure and the control interval size of the area attempting to share the structure, the open process for the area attempting to share the structure fails.

pbuf
The primary buffer allocation. Specify a value from 1 to 32,766.

sbuf
The secondary buffer allocation. Specify a value from 1 to 9999. This secondary allocation is used when the primary allocation runs low.

maxbuf
The maximum number of buffers allowed for this pool. It is a combination of PBUF plus some iteration of SBUF. The maximum value allowed is 32,767. If *maxbuf* is specified greater than 32,767, it defaults to 32,767.

LKASID | NOLKASID
Indicates whether this pool is to be used as a local cache with buffer lookaside capability.

strname
The required 1- to 16-character name of the primary coupling facility structure.

The installation must have defined the structure in the coupling facility resource management (CFRM) administrative policy. The structure name must follow the naming rules of the CFRM. If the name has fewer than 16 characters, the system pads the name with blanks. The valid characters are A-Z, 0-9, and the following special characters:

```
$ @ # _
```

Names must be uppercase and start with an alphabetic character.

Restriction: Avoid using names that IBM uses for its structures. Do not begin structure names with the letters A-I, or the character string SYS.

Related reading: For more information about shared VSO and multi-area structures, refer to *IMS Version 11 Database Administration*.

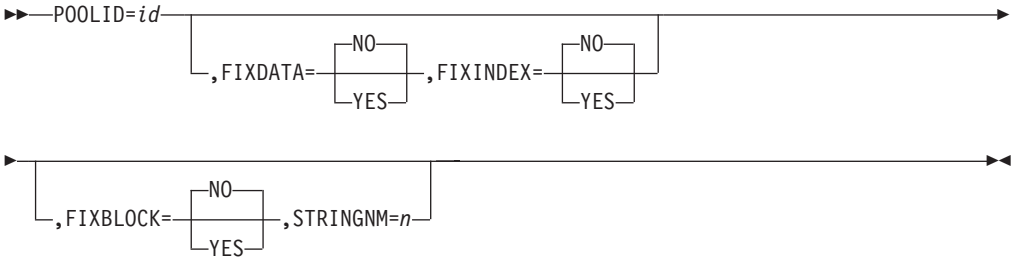
Defining VSAM buffer pools

This section describes control statements in the DFSVSMxx member that you use to define multiple VSAM local shared resource pools.

Related reading: For overview information, refer to "IMS buffer pools" on page 179.

One or more VSAM local shared resource pools can be defined through the subpool definition statement POOLID. One POOLID statement is specified for each shared resource pool. The POOLID statement must be followed by one or more VSRBF subpool definition statements, which define the subpools within that shared pool.

POOLID control statements



The buffer fix and string number parameters must be separated by commas and coded without intervening blanks. They can be coded in any order. If any parameter is invalid, the remainder of the statement is ignored and defaults apply for the remaining parameters.

The POOLID buffer fix and string number values override the VSAMFIX and STRINGMX parameters on the OPTIONS statement for that particular shared pool.

POOLID=

Required keyword for VSAM shared resource pool definition. It must begin in the first position of the control statement. Only one set of subparameters can be entered for each control statement. Each statement defines one shared resource pool. The POOLID statement is used with the DBD statement to direct a given data set to a specific shared resource pool.

Only the first VSAM local shared resource pool defined with the POOLID statement can contain VSAM subpool definition statements (VSRBF) for subpools not dedicated to specific data sets. All VSAM local shared resource pools defined after the first pool must be dedicated to specific data sets. Add a DBD statement that defines which data sets use the shared resource pool. Therefore, defining a VSAM local shared resource pool other than the first pool defined, without a DBD statement referencing that pool, results in a failure for the POOLID statement.

POOLIDs 0 and 1-254 can be used. The total number of POOLIDs must not exceed 255. Also see the paragraph under *RESVPOOL=*.

id A one- to four-character alphanumeric field that specifies a user identifier assigned to a shared resource pool. This parameter is required. It is used with the DBD statement to direct a given data set to a specific shared pool.

FIXDATA=YES|NO

Specifies the data shared resource pool long-term-page-fixing option. If YES is specified, all buffers in the data shared resource pool are long-term-page-fixed at initialization of the shared resource pool. If NO is specified, no buffers in the data shared resource pool are long-term-page-fixed. If this parameter is omitted, the default is NO.

FIXINDEX=YES|NO

Specifies the index shared resource pool long-term-page-fixing option. If YES is specified, all buffers in the index shared resource pool are long-term-page-fixed at initialization of the shared resource pool. If NO is specified, no buffers in the index shared resource pool are long-term-page-fixed. If this parameter is omitted, the default is NO.

FIXBLOCK=YES|NO

Specifies the I/O-related control blocks long-term-page-fixing option. If YES is specified, all I/O-related control blocks are long-term-page-fixed at initialization of the shared resource pool. If NO is specified, no I/O-related control blocks are long-term-page-fixed. If this parameter is omitted, the default is NO.

STRINGNM=n

Specifies the maximum number of VSAM I/O requests that can be concurrently active. It can be used to override the MAXREGN parameter specified at system definition time and the PST value on the EXEC statement. The specified value must be a decimal number 1 - 255. The default is 255. Specify a number as close as possible to the maximum number of regions expected to be running concurrently, including those regions that can be dynamically started. If the value is omitted or invalid, the default is the MAXREGN value or the PST value, which, if present, overrides the MAXREGN value.

RESVPOOL control statements

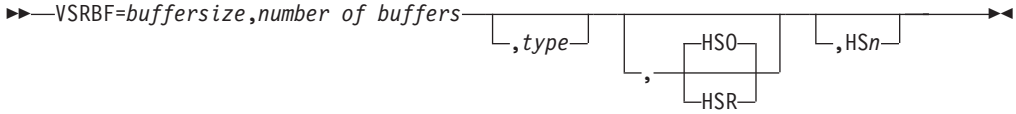
RESVPOOL=shared-pool-number
The *shared-pool-number* variable represents the VSAM local shared resource pool number of the shared pool that IMS cannot use. One or more numbers 1 - 254 can be specified in any order and must be separated by commas, but without any blanks. Shared resource pool 0 is reserved for IMS and you cannot use it. A blank must follow the last number specified, and the remaining portion of the statement is not examined.

The RESVPOOL statement is only applicable if you have a product other than IMS using VSAM local shared resource pools that IMS cannot use.

Defining VSAM subpools

This section describes control statements in the DFSVSMxx member that you use to define VSAM subpools.

VSRBF control statements



VSRBF=
The keyword for VSAM subpool definition. It must begin in the first position of the control statement. Only one set of subparameters can appear on each control statement. Each statement defines one subpool.

buffersize
A three- to five-digit number specifying the buffer size for this subpool. Acceptable values are 512, 1024, 2048, 4096, 8192, 12288, 16384, 20480, 24576, 28672, and 32768.

number-of-buffers
A one- to five-digit number (3 to 32767) specifying the number of buffers in this subpool. If you specify a number of buffers less than the minimum number required, IMS increases the number to the minimum and issues a warning message.

type
A one-character field that specifies whether the subpool in the shared resource pool is an index subpool (`I`) or a data subpool (`D`). This parameter is optional. If this parameter is invalid or not specified, data subpool (`D`) is assumed.

If you do not specify a value for type in any VSRBF statements, both data and index occupy the same buffer subpool. When you do specify `D` or `I` in any VSRBF statement, you must also specify a type of `I` for each size that might be required to handle index buffering requests. An error with the message `DFS0730I RC 0,DC` might result if a subpool of sufficient size is not found.

The following example provides buffers for data (2K) and index (4K) if these are the only VSRBF statements in DFSVSMxx.

```
VSRBF=2048,20
VSRBF=4096,40
```

In the next example, ten index buffers (2K) are provided in the first line, and ten data buffers (2K) are provided in the second line. The third line provides 20 data buffers (4K). No index buffers of 4K are provided in this example.

```
VSRBF=2048,10,I
VSRBF=2048,10,D
VSRBF=4096,20
```

Although the index and data components of a VSAM data set must reside in the same shared resource pool, they can be assigned different subpools. You can define subpools that accommodate each component of the data set. Note that an index subpool cannot exist without a data subpool (within a given shared resource pool), but a data subpool can exist without an index subpool. If any index subpool exists in a local shared resource pool, all index components of VSAM data sets in that shared pool must share the index subpool. The CONTROLINTERVALSIZE parameter in the listing of the VSAM catalog that shows the control interval sizes assigned to the components can help you determine the buffer sizes needed for both index and data components.

HSO|HSR
Optionally specifies the kind of action IMS should take if Hiperspace™ (extended storage on z/OS) buffering for this subpool is not available. If this parameter is invalid or not coded when HSn is given, the HSO option is assumed.

HSO
Indicates that Hiperspace buffering is optional and IMS can continue initialization without Hiperspace buffering.

HSR
Indicates that Hiperspace buffering is required and IMS must terminate if Hiperspace buffering is not available.

HSO and HSR are the minimum truncations of which up to the full terms (HSOptional and HSRequired) are recognized.

HSn
Is an optional one- to eight- digit number *n* ranging from (3 to 16777215) that specifies the number of Hiperspace buffers to build for this subpool.

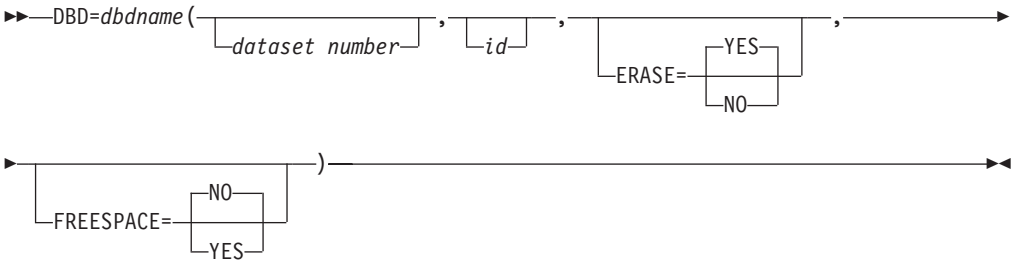
HSO|HSR and HSn are valid only on 4K and larger boundaries. See the following example.

```
VSRBF=4096,10,HS40,HSR
VSRBF=8192,10,HS50,HSRE
VSRBF=12288,4,HS12
VSRBF=16384,4,HS20,HSO
VSRBF=20480,4,HSOP,HS8
VSRBF=24576,4,HSOPTIONAL,HS8
VSRBF=28672,1,HS9,HSREQUIRED
VSRBF=32768,1,HSREQ,HS9
```

Only the buffer size and number-of-buffers parameters are positional. The type and HS parameters can be coded in any order but must be separated by commas without any intervening blanks. If any parameter is invalid, the remainder of the statement is ignored and defaults apply for the remaining parameters.

Using the VSRBF statement without any POOLID statement can still define one, and only one, VSAM local shared resource pool. However, even with one VSAM shared resource pool, you can now use the TYPE parameter to define either an index or data subpool within the shared resource pool.

DBD control statement



DBD=

If coded, specifies that the data set having a matching ID parameter, as defined on the POOLID shared pool definition statement, is to be directed to the indicated shared resource pool.

A shared resource pool is assigned to a data set based upon the shared resource pool identifier specified on the DBD statement for that data set. Then a subpool within the assigned shared resource pool is assigned to the data set based upon buffer length.

If no subpool in the shared resource pool is big enough, an attempt to open the data set fails. An open is then retried in the default shared pool, which is the first shared resource pool you define. To avoid another open failure, subpools of the largest sizes that might be encountered in the system should be defined in the default pool.

If no DBD statement is coded for a data set, the default shared resource pool is assigned.

dbdname

Specifies the name on the DBD macro statement NAME= keyword. The DBD name can be for a non-partitioned database, a HALDB partition, or a HALDB master. For a HALDB partition, specify the name of the partition as the dbdname. If you specify a HALDB master name, all the HALDB partitions associated with the HALDB are assigned to the same shared resource pool.

Related reading: See Database Description (DBD) Generation utility (System Utilities) in *IMS Version 11 System Utilities* for information about coding the DBD macro statement.

data set number

Identifies the specific data set of a data set group within a database (identified by the dbdname parameter) that requests assignment of a specific shared pool. The number is an IMS internally assigned value between 1 and 10.

For data organizations such as primary index, unique secondary index, and HISAM without dependent segments, the primary data set of the data set group is assigned data set number 1. No secondary data set of the data set group exists for these data organizations.

For data organizations such as non-unique secondary index and HISAM with dependent segments, the primary data set of the data set group is assigned data set number 1, and the secondary data set of the data set group is assigned data set number 2.

For hierarchic direct data organization, the data set group always consists of a single data set. The data set of the first data set group is assigned data set number 1, and data set numbers for subsequent data set groups are sequentially assigned numbers 2,3,...10. The maximum number of data set groups for a database is 10.

Requirement: For High Availability Large Databases (HALDBs), specify the data set number as an alphabetic character. The valid data set numbers defined for HALDB partition data sets are A through J, L, and X. (When HALDB Online Reorganization is used, data sets M through V and Y are automatically directed to the same shared resource pools for data sets A through J and X. The specification of M through V and Y are not valid.) The data set of the first data set group must be assigned the letter A (for both data sets A and M when HALDB Online Reorganization is used). The data set numbers for subsequent data set groups (B through J and N through V when HALDB Online Reorganization is used) must be sequentially assigned the other valid letters (B,C, ..., J). Use the letter L to specify the indirect list data set. For PHIDAM databases, use the letter X (for both data sets X and Y when HALDB Online Reorganization is used) to specify the primary index data set.

For PSINDEX databases, you cannot code an alphanumeric to designate the data set number. You must code a numeric value.

id Specifies the user-defined identifier that is assigned to a specific shared resource pool. The ID is a one- to four-character alphanumeric field and must be equal to the ID assigned to the specific shared resource pool on the POOLID statement.

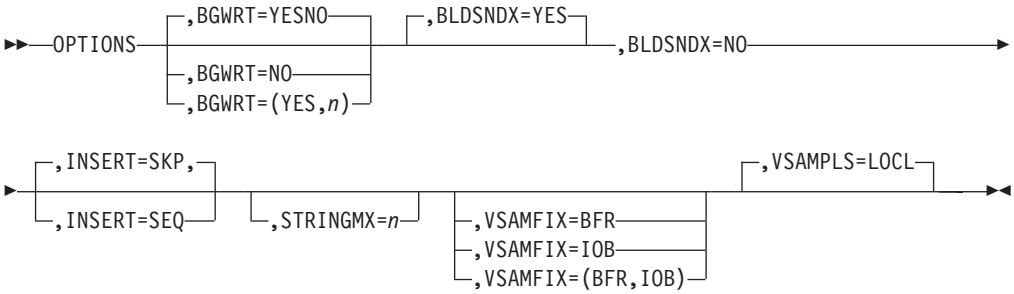
ERASE=YES|NO
Indicates the treatment of logical records that are deleted. YES indicates that the deleted record should be erased. NO indicates that the deleted record should not be erased but should be marked as a deleted record. The default is YES.

FREESPACE=YES|NO
Indicates the treatment of the defined free space percent in the KSDS. If YES is specified, the defined free space should be preserved. If NO is specified, the defined free space should not be preserved. The default is NO.

Defining VSAM performance options

This section describes the OPTIONS control statement parameters in the DFSVSMxx member that can be used to influence the performance of the IMS DL/I buffering services.

The word OPTIONS starting in position one identifies the OPTIONS statement. You can specify the parameters in any order, but you must separate them with commas. A blank must follow the last parameter. The remaining portion of the statement is not examined. You cannot continue an OPTIONS statement on a subsequent line, but you can provide several statements. If an OPTIONS parameter appears more than once, its setting is determined by the last occurrence. The OPTIONS statement and all its parameters are optional.



BGWRT=

Specifies whether (YES) or not (NO) the background write (BGWRT) function of the buffer handler is to be active. You can also activate background write by coding BGWRT=(YES,*n*) or omitting the parameter. *n:* is a two-digit number from 10 to 99 specifying the percentage of buffers in each subpool to be considered as candidates for writing by the background write function; the default is 34.

Related reading: For an explanation of background write, see *IMS Version 11 Database Administration*.

BLDSNDX=

Provides an option to control the building of secondary indexes during initial load of an HALDB.

If BLDSNDX=YES, or the parameter is omitted, then the creation of secondary indexes during initial load is allowed.

IF BLDSNDX=NO, then the creation of secondary indexes during initial load is suppressed.

INSERT=

Specifies the insert mode that the buffer handler uses when inserting new KSDS logical records into any of the following databases: HISAM, Single segment HISAM, and INDEX.

If a program inserts many new root segments in sequence by key, specifying INSERT=SEQ causes the buffer handler to use VSAM sequential mode PUTs. VSAM leaves free space (if you specify it in the DEFINE) for VSAM data sets in CIs created for the new records that are inserted using VSAM's mass insert. Specifying INSERT=SKP, or omitting the parameter, causes the buffer handler to use VSAM skip sequential mode PUTs if the database is in sequential mode, or to use direct mode PUTs if the database is not in sequential mode. For more information about sequential mode databases, see "VSAM subpool definition" on page 180. In either mode, VSAM does not leave free space, even if it is specified in the DEFINE regardless of whether DL/I inserts in load or insert mode.

The default is SKP, except in utilities, where it is SEQ.

You can override INSERT=SEQ with the FREESPACE parameter of the DBD control statement. FREESPACE=YES causes the buffer handler to user VSAM sequential mode PUTs. Specifying FREESPACE=NO, or omitting the parameter, causes the buffer handler to use VSAM skip sequential mode PUTs if the database is in sequential mode, or to use direct mode PUTs if the database is not in sequential mode.

STRINGMX=
Specifies the maximum number of VSAM requests that can be concurrently active. *n* is any number between 1 and 255. If you omit this keyword, the default value is the value specified on either the MAXREGN= parameter on the IMSCTRL macro or the PST= parameter on the IMS procedure EXEC statement (which overrides the MAXREGN= value). The STRINGMX= keyword overrides either of these two values.

VSAMFIX=
Causes VSAM resource pools to be page fixed in main storage.

BFR
Specifies that the buffer subpools and control blocks should be fixed. If you define a large number of buffers and then use VSAMFIX=BFR the system might abend.

IOB
Specifies that the input/output related blocks should be fixed.

Either or both operands can be specified. The syntax for specifying both operands is (BFR,IOB) or (IOB,BFR).

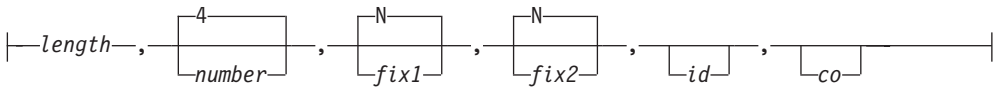
VSAMPLS=LOCL
Specifies that the VSAM shared resource pools are to be built. The only valid value is the default, LOCL.

Defining OSAM subpools

This section describes control statements in the DFSVSMxx member that you use to define OSAM subpools.

IOBF control statements

A:



IOBF=
Is the required keyword for OSAM subpool definition. It must begin in the first position of the control statement. Only one set of subparameters can appear on each control statement.

length
Specifies the length of the buffers in the subpool. Specify the value in bytes, between 512 and 32000. Depending on the value that you specify, IMS rounds it up to 512, 1024, 2048, and thereafter to multiples of 2048. You can code specifications of 1024 and above as 1K, 2K, 4K, and thereafter round them up to multiples of 2K to a maximum of 32K. If you enter an invalid value, IMS ignores the entire IOBF statement. This parameter is required.

number
Specifies the number of buffers in the subpool. If specified, you must choose a value between 4 and 32767. If you choose a number greater than

32767, the default value of 255 is used. If not specified, the default value is 4. If this parameter is invalid, the remainder of the entry is ignored, and defaults apply for all remaining parameters. This parameter is optional.

fix1
Specifies the buffer long-term page-fixing option. If you specify Y, all buffers and buffer prefixes associated with this subpool are long-term page-fixed at initialization of the subpool. If you specify N, no buffers associated with this subpool are long-term page-fixed at initialization of the subpool. The default is N. This parameter is optional.

fix2
Specifies the buffer prefix long-term-page-fixing option. If you specify Y, all buffer prefixes associated with this subpool and the subpool header are long-term-page-fixed at initialization of the subpool. If you specify N, the subpool header and all buffer prefixes associated with this subpool are not long-term-page-fixed at initialization of the subpool. The default is N. This parameter is optional.

id Specifies a user-defined identifier to be assigned to a subpool. This ID is a one- to four-character alphanumeric field and is used with the DBD statement to assign a specific subpool to a given data set. If this parameter is not coded, IMS assigns a null ID to the subpool.

If two or more subpool definition statements specify the same buffer size (without the subpool ID), the number of buffers from the statements are summed. If the total does not exceed 32767, IMS builds a single subpool with the total number of buffers; however, if it does exceed 32767, a single subpool with 255 buffers is built.

co Specifies the subpool caching option. For more information about OSAM data caching, see "OSAM subpool definition" on page 180.

You can specify the caching option in one of the following ways:

omitted
No data caching.

Caching is not active for the subpool.

N No data caching.

Caching is not active for the subpool.

A Cache all data.

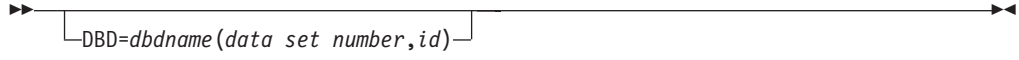
Write all data read from DASD and all changed data to the coupling facility

C Cache only changed data.

Write all changed data written to DASD to the coupling facility.

If you specify any value other than A, C, or N, no data caching for the subpool occurs. The default is N.

DBD control statements



DBD=
If coded, specifies that the indicated subpool is to be assigned to the data set

having a matching ID parameter as defined on the IOBF= subpool definition statement. For OSAM, ERASE and FREESPACE are invalid.

dbdname
Specifies the name on the DBD macro statement NAME= keyword. The DBD name can be for a non-partitioned database, a HALDB partition, or a HALDB master. For a HALDB partition, specify the name of the partition as the dbdname. If you specify a HALDB master name, all the HALDB partitions associated with the HALDB are assigned to the same subpool.

Related reading: For information about coding the DBD macro statement, see Database Description (DBD) Generation utility (System Utilities).

data set number
Identifies the specific data set of a data set group within a database (identified by the dbdname parameter) that requests assignment of a specific shared pool. The number is an IMS internally assigned value between 1 and 10.

For data organizations such as primary index, unique secondary index, and HISAM without dependent segments, the primary data set of the data set group is assigned data set number 1. No secondary data set of the data set group exists for these data organizations.

For data organizations such as non-unique secondary index, and HISAM with dependent segments, the primary data set of the data set group is assigned data set number 1, and the secondary data set of the data set group is assigned data set number 2.

For hierarchic direct data organization, the data set group always consists of a single data set. The data set of the first data set group is assigned data set number 1, and data set numbers for subsequent data set groups are sequentially assigned numbers 2,3, ...10. The maximum number of data set groups for a database is 10.

Requirement: For High Availability Large Databases (HALDBs), specify the data set number as an alphabetic character. The valid data set numbers defined for HALDB partition data sets are A through J, L, and X. (When HALDB Online Reorganization is used, data sets M through V and Y are automatically directed to the same shared resource pools for data sets A through J and X. The specification of M through V and Y are not valid.) The data set of the first data set group must be assigned the letter A (for both data sets A and M when HALDB Online Reorganization is used). The data set numbers for subsequent data set groups (B through J and N through V when HALDB Online Reorganization is used) must be sequentially assigned the other valid letters (B,C, ..., J). Use the letter L to specify the indirect list data set. For PHIDAM databases, use the letter X (for both data sets X and Y when HALDB Online Reorganization is used) to specify the primary index data set.

id Specifies the user-defined identifier to be assigned to a specific subpool. The ID is a one- to four-character alphanumeric field and must be equal to the ID assigned to the specific subpool on the IOBF statement.

Subpools are assigned to data sets based upon buffer length. First, a subpool having buffers equal to or greater than the buffer length required for the data set is located. Then, if a specific subpool was requested, it is assigned, if its length is not less than the required length. In this case, the first subpool that meets the length criterion is assigned. The subpool might not have an ID assigned to it.

OSAMOP control statements

The OSAMOP statement allows options specific to the OSAM access method.

IOSCB=NO
Indicates that no OSAM resources are to be allocated to the batch applications that do not require OSAM services.

OSAMGTF=YES
Enables OSAM to generate GTF trace records while in batch.

Requesting that z/OS dynamic allocation use extended storage

IMS uses z/OS dynamic allocation extensively to provide access to full-function and Fast Path databases.

The blocks that z/OS provides (DSABs and TIOTs) are put into low private storage, unless otherwise requested. Use the DBALLABOVE statement to change the default and put them into extended private storage. This option is applicable only in batch regions.

DBALLABOVE
This statement changes the location of the DSAB and TIOT blocks that z/OS builds and dynamically allocates for IMS database data sets from low private storage to extended private storage.

Specifying sequential buffering for an online system

Use the SBONLINE control statement in the DFSVSMxx member to request sequential buffering for an IMS or IBM CICS Transaction Server for z/OS system.

This statement applies to the IMS DB/DC or DBCTL environments. SBONLINE requests loading of SB modules and specifies the maximum amount of buffer space that SB can use.

The format of the SBONLINE control statement is as follows:

Control statements

Parameter descriptions

SBONLINE =
Specifies that sequential buffering modules are to be loaded in the IMS DB/DC or DBCTL environments, thereby allowing subsequent use of sequential buffering.

By default, IMS does not load the SB modules. This prevents an increase in virtual storage requirements.

MAXSB=nnnn
Specifies the maximum amount of space (in kilobytes) that can be allocated to SB buffers. This number represents the total amount of space to be allocated for **all** concurrently executing programs, not for just one individual program. Setting a limit on the amount of SB space protects you from allocating excessive virtual storage. The default for MAXSB is unlimited space.

When the MAXSB limit is reached, IMS stops allocating SB buffers until terminating programs release buffer storage.

Defining serviceability and trace options

This section describes the OPTIONS control statement parameters in the DFSVSMxx member, which can be used to influence the serviceability of IMS.

For example, the OPTIONS statement allows you to specify which IMS traces should be activated automatically by the system during IMS initialization.

The word OPTIONS starting in position 1 identifies the OPTIONS statement. You can specify the parameters in any order, but you must separate them with commas. A blank must follow the last parameter. The remaining portion of the statement is not examined. You cannot continue an OPTIONS statement on a subsequent statement, but you can provide several statements. If an OPTIONS parameter appears more than once, its setting is determined by the last occurrence. The OPTIONS statement and all its parameters are optional.

Recommendation: Select ON for the following parameters:

- DL/I=
- LOCK=

For an online system, select ON for these additional parameters:

- DISP=
- SCHD=

Activating the trace tables specified by these parameters does not cause a noticeable performance impact and each of the trace tables can be helpful to you in diagnosing various problems that might occur in your environment. For certain types of problems, the IMS Support Center needs the trace output. If the trace output is not available, you might need to re-create the problem, which takes time and delays the resolution of the problem.

If you specify OUTMED for DL/I and lock traces, IMS replaces this with a specification of OUTHIGH. If turning on external tracing for DL/I and lock, you should use the highest level of tracing.

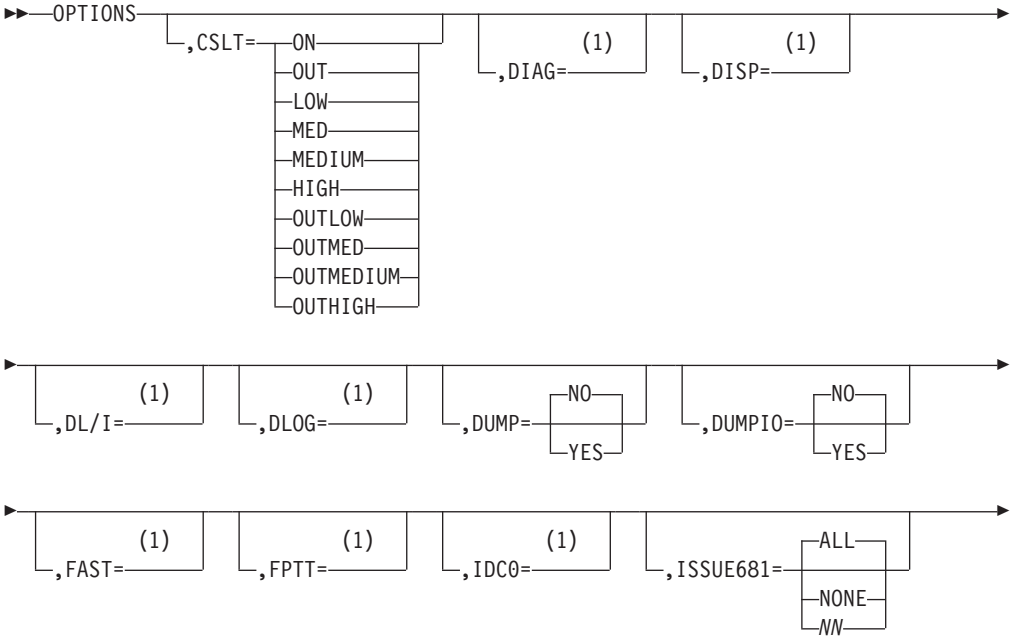
Related reading: For more information about all these traces, see *IMS Version 11 Diagnosis*.

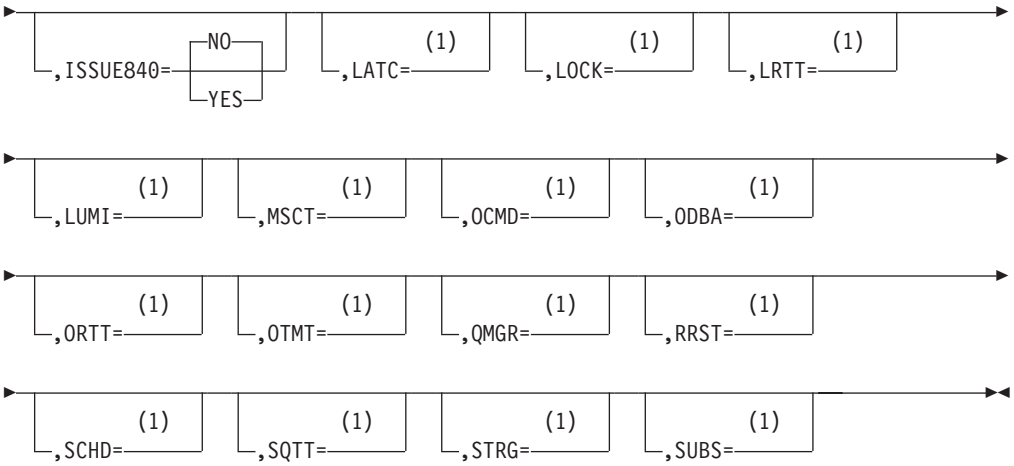
Valid environments

The following table lists serviceability and trace options that can be activated for particular environments. If you attempt to activate a trace in an environment that does not support that type of trace, IMS ignores your request.

Environment	Valid serviceability and trace options		
Batch	• DIAG= • DL/I= • DLOG=	• DUMP= • DUMPIO= • ISSUE681=	• ISSUE840= • LOCK=
DB/DC	All trace types, except for LRTT=, are valid in a DB/DC environment.		
DBCTL	• CSLT • DIAG= • DISP= • DL/I= • DLOG= • DUMP= • DUMPIO=	• FPTT= • ISSUE681= • ISSUE840= • LATC= • LOCK= • OCMD • ODBA=	• ORTT= • QMGR= • SCHD= • SQTT= • STRG= • SUBS=
DCCTL	• DIAG= • DISP= • DL/I= • DLOG= • DUMP= • DUMPIO= • FAST=	• FPTT= • IDC0= • ISSUE681= • ISSUE840= • LATC= • LUMI • MSCT=	• OTMT= • QMGR= • RRST= • SCHD= • SQTT= • STRG= • SUBS=
RSR Tracker	• DIAG= • DISP= • DL/I= • DLOG= • DUMP= • DUMPIO=	• FAST= • FPTT= • IDC0= • ISSUE681= • ISSUE840= • LATC=	• LOCK= • LRTT= • QMGR= • SQTT= • STRG=

Control statements





Notes:

1 This parameter accepts values from the common options listed below.

Common options

ON Turns on the trace table.

OUT Initializes and turns on the trace table for output to the log. Log records 4096 bytes long are written to the log when trace records are written.

LOW Turns on the trace for all low volume events.

MED Turns on the trace for all low and medium volume events.

MEDIUM
Turns on the trace for all low and medium volume events.

HIGH Turns on the trace for all low, medium, and high volume events.

OUTLOW
Initializes the trace table for output to the log and turns it on for tracing all low volume events.

OUTMED
Initializes the trace table for output to the log and turns it on for tracing all low and medium volume events.

OUTMEDIUM
Initializes the trace table for output to the log and turns it on for tracing all low and medium volume events.

OUTHIGH
Initializes the trace table for output to the log and turns it on for tracing all low, medium, and high volume events.

Related Reading: See *IMS Version 11 Diagnosis* for information about when and why trace tables are used, and defining and setting up trace facilities.

Parameter descriptions

CSLT=
Activates an IMS trace that traces activity related to IMS's interaction with the Common Service Layer. This includes IMS's interaction with OM, RM, and SCI.

No default value exists. If this parameter is omitted, the trace is not activated by the OPTIONS statement. It can be activated and deactivated by the online /TRACE command.

DIAG=

Activates the /DIAGNOSE command trace tables. If this parameter is omitted, the trace is not activated by the OPTIONS statement. It can be activated and deactivated by the online /TRACE command.

Related reading: For more details on the /DIAGNOSE and /TRACE commands, see *IMS Version 11 Commands, Volume 1: IMS Commands A-M*.

DISP=

Activates an IMS trace and traces the calls of the IMS dispatcher. No default value exists. If this parameter is omitted, the trace table is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: DISP= is applicable to the online system only. The DISP trace is not allowed in a batch environment. If this trace is requested in a batch environment, the request is ignored.

DL/I=

Activates an IMS trace that traces the calls of certain DL/I modules and the functions invoked by these DL/I modules. The default value is ON. The only exception to this is for batch regions. For batch regions, the default is that the DL/I trace is off. To deactivate the DL/I trace table, issue the /TRACE command in an online environment.

The DL/I trace, the program isolation trace, and the lock trace share the same trace table.

This keyword can also be used to turn on the DL/I retrieve trace table, but it does not control the number of table entries, which is fixed at 255.

DLOG=

Activates an IMS trace. The DASD log trace traces the activity of the IMS DASD logger. No default value exists. If this parameter is omitted, the trace table is not activated by the OPTIONS statement. It can be activated deactivated by using the /TRACE command.

DUMP=

Provides a serviceability aid. If you specify NO or omit the parameter and an abnormal condition is encountered, the buffer handler issues a pseudoabend. If you specify YES and an abnormal condition is encountered, the buffer handler issues a standard abend, which results in the control region abnormally terminating also. If DUMP=YES and any abnormal termination occurs, the modified or newly created buffers in the pool at that time are not written to the data set.

For batch environments, if you specify NO or omit the parameter, a dump is taken and all modified buffers in all subpools are written to the data set. This cleans up any outstanding I/Os before purging the pools. If you specify YES, only a dump is taken. The modified buffers are not written to the data set.

DUMPIO=

Provides a serviceability aid for analyzing OSAM I/O errors. If you specify YES and an OSAM I/O error occurs, the IMS region abnormally terminates with a U0764. If the DUMPIO option is requested, the DFS0762I error message is not displayed.

FAST=
Activates an IMS trace that traces DBF trace entries from various Fast Path modules. No default exists. If this parameter is omitted, the trace is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Recommendation: Run the FPTRACE in a test environment only. FPTRACE output is very large and can impact performance.

Restriction: The FAST trace is not allowed in a batch environment. If this trace is requested in a batch environment, the request is ignored.

Related reading: For more details on the FPTRACE, see *IMS Version 11 Diagnosis*.

FPTT=
Activates a Fast Path trace that traces certain Fast Path modules and the functions invoked by these Fast Path modules. The default value is OFF. To activate the Fast Path trace table, issue the /TRACE command in an online environment.

The Fast Path trace resides in its own trace table.

Related reading: For more details on the Fast Path trace, see *IMS Version 11 Diagnosis*.

IDC0=
Activates the tracing of errors in modules DFSCNXA0 and DFSIDC00. No default value exists. If this parameter is omitted, the trace table is not activated by the OPTIONS statement. It can be activated or deactivated by using the online /TRACE command.

Restriction: The IDC0 trace is not allowed in a batch or DBCTL environment. If this trace is requested in either a batch or DBCTL environment, the request is ignored.

ISSUE681=
Specifies the number of DFS681I messages to be issued per second within a batch or BMP region.

ALL
Allows all DFS681I messages generated to be issued. All is the default.

NONE
Allows no DFS681I messages to be issued.

NN Is the number (1 - 99) of DFS681I messages that you allow to be issued. If some DFS681I messages are not issued, DFS683I is issued and states the number of DFS681I messages omitted.

ISSUE840=
Specifies whether message DFS0840I is issued if there is a duplicate segment in a unique secondary index, even if it is possible to back out prior changes for the call.

NO Specifies that DFS0840I message is not issued for this condition. NO is the default.

YES Specifies that DFS0840I message is issued for this condition.

LATC=
Activates an IMS trace. LATCH traces IMS latch activity. No default value

exists. If this parameter is omitted, the trace table is not activated by the OPTIONS statement. It can be activated and deactivated by using by the /TRACE command.

Restriction: The LATC trace is not allowed in a batch environment. If this trace is requested in a batch environment, the request is ignored.

LOCK=

Activates an IMS trace that traces the lock activity of certain DL/I modules. The default value is ON. The only exception to this is for batch regions. For batch regions, the default is that the lock trace is off. To deactivate the LOCK trace table, issue the /TRACE command in an online environment.

The lock trace, the program isolation trace, and the DL/I trace share the same trace table.

Restriction: The LOCK trace is not allowed in a DCCTL environment. If this trace is requested in a DCCTL environment, the request is ignored.

LRTT=

Activates an IMS trace that traces the activity of the log router component of the RSR tracking system. No default exists. If this parameter is omitted, the trace is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: The LRTT trace is **only** allowed in an RSR Tracker environment. If this trace is requested in any other environment, the request is ignored.

LUMI=

Activates an IMS trace that traces LUM code that supports an LU 6.2 device. It contains the following information:

- Record written on entry and exit to selected LU 6.2 device modules
- Information from APPC/MVS verb with acceptable or unacceptable return code

If this parameter is omitted, the trace table is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: The LUMI trace is not allowed in a batch, RSR Tracker, or DBCTL environment. If this trace is requested in one of these environments, the request is ignored.

MSCT=

Activates an IMS trace for Multiple Systems Coupling (MSC).

OCMD=

Activates an IMS trace that traces IMSplex command activity. No default value exists. If this parameter is omitted, the trace is not activated by the OPTIONS statement. It can be activated and deactivated by the online /TRACE command.

ODBA=

Activates the ODBA trace table. If this parameter is omitted, the trace is not activated by the OPTIONS statement.

Restriction: The ODBA trace is not allowed in batch, DCCTL or RSR tracker environments. If this trace is requested in any of these environments, the request is ignored.

ORTT=

Activates an IMS trace that traces the IMS Database Recovery Facility activity in the IMS control region. If this parameter is omitted, the IMS Database

Recovery Facility trace is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: The ORTT trace is not allowed in a batch or DC environment. If this trace is requested in this environment, the request is ignored.

OTMT=

Activates an IMS trace. OTMT traces the flow of control through IMS OTMA. No default exists. If this parameter is omitted, the trace is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: The OTMT trace is not allowed in a batch, RSR Tracker, or DBCTL environment. If this trace is requested in any of these environments, the request is ignored.

QMGR=

Activates an IMS trace for an online system. QMGR traces calls made to the IMS queue manager. No default value exists. If you omit this parameter, the trace table is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: The QMGR trace is not allowed in a batch environment. If this trace is requested in a batch environment, the request is ignored.

RRST=

Activates an IMS trace for z/OS Resource Recovery Services (RRS). No default value exists. If this parameter is omitted, the trace table is not activated by the OPTIONS statement. The trace can be activated and deactivated by using the online /TRACE command.

Restriction: The RRST trace is not allowed in a batch environment. If this trace is requested in a batch environment, the request is ignored.

SCHD=

Activates an IMS trace. SCHD traces the calls of the IMS scheduler. No default value exists. If this parameter is omitted, the trace table is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: The SCHD trace is not allowed in batch or RSR Tracker environments. If this trace is requested in either of these environments, the request is ignored.

SQTT=

Activates an IMS trace for an online system. SQTT traces the shared queues interface including:

- Initialization of the connection to CQS
- CQS requests
- Notification of work available from CQS
- Termination of the connection to CQS

No default value exists. If you omit this parameter, the trace table is not activated by the OPTIONS statement, but can be activated and deactivated by using means of the online /TRACE command.

Restriction: The SQTT trace is not allowed in a batch environment. If this trace is requested in a batch environment, the request is ignored.

STRG=

Activates an IMS trace. STRG traces all calls to the IMS storage manager that require modifications to one of the following pools:

AOIP
CESS
CIOP
CMDP
DYNP
EMHB
FPWP
HIOP
LUMC
LUMP
SPAP

If this parameter is omitted, the trace table is not activated by the OPTIONS statement; instead, it can be activated and deactivated by using the online /TRACE command.

STRG= is applicable only to the online system.

Restriction: The STRG trace is not allowed in a batch environment. If this trace is requested in a batch environment, the request is ignored.

SUBS=

Activates an IMS trace. SUBS traces the connection and disconnection of DB2 for z/OS subsystems to the IMS control region. No default value exists. If this parameter is omitted, the trace is not activated by the OPTIONS statement. It can be activated and deactivated by using the online /TRACE command.

Restriction: The SUBS trace is not allowed in batch or RSR Tracker environments. If this trace is requested in either of these environments, the request is ignored.

Table 82 is a summary of the traces that can be activated using the OPTIONS parameter. It shows the number of trace entries and the storage requirements for the various traces.

Table 82. Trace entries and storage requirements

Type	Number of trace tables	Entry size	Entries per table	Storage bytes required
Common Service Layer	8	20 Hex	126	32 KB
/DIAGNOSE command	64	20 Hex	126	256 KB
Dispatcher	10	20 Hex	126	40 KB
DLI/Loc KB	18	20 Hex	126	72 KB
DLog	6	20 Hex	126	24 KB
Fast Path	2	20 Hex	126	8 KB
Intercommunications	18	20 Hex	126	72 KB
Latch	12	20 Hex	126	48 KB
Log Router	8	X '20'	126	32 KB
LU 6.2	8	X '20'	126	32 KB

Table 82. Trace entries and storage requirements (continued)

Type	Number of trace tables	Entry size	Entries per table	Storage bytes required
Multiple Systems Coupling	10	X '20'	126	40 KB
OCMD	8	X '20'	126	32 KB
Online Recovery	8	X '20'	126	32 KB
OTMA	8	X '20'	126	32 KB
Queue Manager	8	X '20'	126	32 KB
Scheduler	5	X '20'	252	20 KB
Shared Queues	8	X '20'	126	32 KB
Storage Manager	10	X '20'	126	40 KB
SubSystem	8	X '40'	63	32 KB
z/OS Resource Recovery Services	12	X '20'	126	48 KB

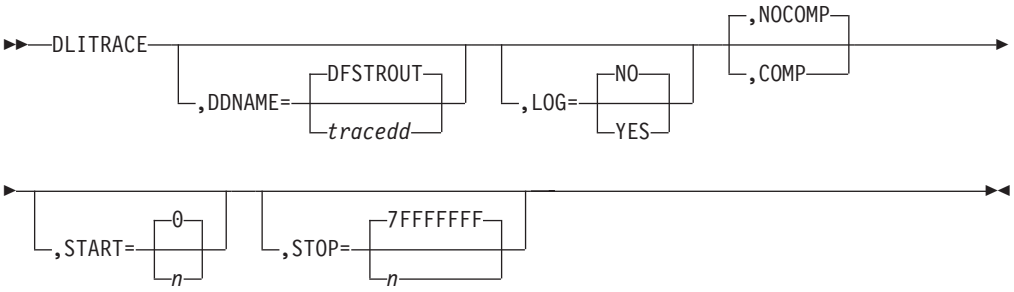
Note: Only DLog storage is below the 16-MB line. For all other types listed here, storage is above the 16-MB line.

Related reading: For additional information about the traces that you can define using these keywords, see *IMS Version 11 System Utilities*.

Defining DL/I call image trace

This section describes control statements in the DFSVSMxx member that you use to specify a DL/I call image trace.

Control statements



The DLITRACE statement invokes the tracing of DL/I calls in a batch environment. Specify the word DLITRACE in the first position to identify this statement. You can specify the parameters in any sequence, but you must separate them with commas; each parameter can be specified only once. If you do not specify any parameters, IMS uses the default values.

A DLITRACE statement cannot be continued on a subsequent statement. The first statement encountered in the DFSVSAMP data set establishes the options. Subsequent statements are bypassed, and message DFS2471 is issued. In cases where a parameter is repeated, the options remain set as specified by the first occurrence, and message DFS2471 is issued.

Related reading: For details on the use of DL/I trace, see *IMS Version 11 Diagnosis*.

Parameter descriptions

DDNAME=
Indicates that sequential data set output is requested and gives the DD statement name (tracedd) to be present in the JCL. If you omit this parameter and do not supply output DD, DDNAME=DFSTROUT is assumed, and an attempt is made to route output to the sequential data set defined by DFSTROUT DD. However, if you omit this parameter and request log output, no attempt is made to open such a sequential data set.

LOG=
Specifies whether (YES) or not (NO) DL/I call image capture output is to be routed to the IMS system log. The default is NO, and a sequential output data set contains the trace output.

NOCOMP | COMP
Specifies whether (COMP) or not (NOCOMP) DL/I Test Program COMPARE statements are generated for both PCB comparisons and data comparisons. The default is NOCOMP.

Related reading: For more information, see *IMS Version 11 Application Programming*.

START=
Specifies a one- to eight-character hexadecimal value representing a count of DL/I calls issued by an application program against a specific PSB. This value indicates at which point in the program processing tracing should begin. The default value of 0 begins the tracing at the first DL/I call.

STOP=
Specifies a one- to eight-character hexadecimal value representing a count of DL/I calls. This value indicates the point at which DL/I image capture tracing against a specific PSB should stop. The default value of X'7FFFFFFF' indicates that all the DL/I calls for the batch program are traced.

Defining DASD logging initialization parameters

In a DB/DC environment, use the OLDSDEF and WADSDEF control statements to specify the log data sets and parameters to be established during DLOG initialization. These control statements are mandatory in the DFSVSMxx member of the IMS PROCLIB data set. Use the ARCHDEF control statement to modify the GENJCL.ARCHIVE command generated by IMS when it switches OLDSs.

Related reading: For additional information about archiving OLDSs and the GENJCL.ARCHIVE command, see "ARC=" in "Parameter descriptions for IMS procedures" on page 448. When you specify ARCHDEF in the DFSVSMxx member of the IMS PROCLIB data set, it applies only to the active system. ARCHDEF specified in the DFSRSRxx member of the IMS PROCLIB data set is meaningful only to a remote system.

ARCHDEF statement

When IMS automatically archives OLDSs in response to the ARC= parameter of the IMS procedure, it does so by internally issuing a GENJCL.ARCHIVE command to DBRC. The default command is:

```
GENJCL.ARCHIVE ALL SSID (xxxx)
```

Use the ARCHDEF control statement to change this command to:

```
GENJCL.ARCHIVE aaaa SSID (xxxx)
```

In this statement, *aaaa* represents any valid GENJCL parameter up to 63 characters in length.

LOGEDIT statement

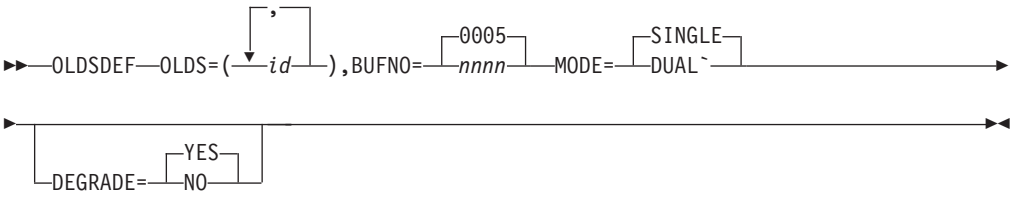
The LOGEDIT statement identifies the IMS log record types that are passed to the Log Edit exit routine (DFSFLGE0). Each instance of the parameter can be either:

record_type
: Supported values are 01, 03, 4002, 5901, and 5903. You can specify multiple record types.

MSGLOG
: Specify MSGLOG to pass all supported record types to the exit.

Any other value is ignored. If more than one LOGEDIT statement is supplied, the last one is used.

OLDSDEF statement



The OLDSDEF statement defines the OLDS data sets to be established during initialization. It consists of one or more 80-byte records placed within the DFSVSMxx member of IMS PROCLIB. If multiple OLDSDEF statements exist, all statements are processed. The OLDSDEF syntax is free-form, and continuation statements are indicated by a character other than a blank (X'40') in column 72. Keywords are delimited from associated parameters by an equal sign (=), and keyword/parameter pairs are delimited by a comma (,).

OLDSDEF
: Identifies this statement as an OLDSDEF control statement. This keyword must be specified first.

OLDS=
: Identifies between 3 and 100 OLDS to be allocated at DLOG initialization, where *id* indicates a list of two-digit OLDS IDs, 00 - 99. You must specify at least three OLDSs.

BUFNO=
Identifies the number of log buffers allocated for OLDS read and write operations, where *nnnn* is a one- to four-digit numeric integer, 0002 - 9999. The default is 0005.

If 64-bit virtual storage is specified in the BUFSTOR parameter, BUFNO is increased so that the amount of buffer storage obtained is as close as possible to the next 1 MB boundary.

MODE=
Identifies whether the logger operates in SINGLE or DUAL logging mode. The default is SINGLE.

DEGRADE=
Identifies whether to degrade to single-logging mode (DEGRADE=YES), or to terminate IMS (DEGRADE=NO) after an OLDS write error occurs and there are less than three pairs of available OLDSs. Also, if one of the OLDSs in the pair to be used next had encountered an error during preopen and DEGRADE=YES was specified, that pair will be used in degraded mode. However, after it is switched, the pair will be stopped. If there are fewer than three good pairs of OLDSs, the pair will not be stopped and will continue to be used in degraded mode. The default is DEGRADE=YES.

Provide dynamic allocation members (through the DFSMDA macro) for all OLDS data sets. All (primary and secondary) OLDS must be preallocated. You must specify the OLDS block size when the data set is allocated.

WADSDEF statement

The WADSDEF statement defines the WADS data sets to be established during initialization. It consists of one or more 80-byte records placed within the DFSVSMxx member of IMS PROCLIB. If multiple WADSDEF statements exist, only the first encountered statement is processed. The WADSDEF syntax is free-form, and continuation statements are indicated by a character other than a blank (X'40') in column 72. Keywords are delimited from associated parameters by an equal sign (=), and keyword/parameter pairs are delimited by a comma (,).

WADSDEF
Identifies this statement as a WADSDEF control statement. This keyword must be specified first.

WADS=
Identifies a list of WADS to be allocated at DLOG initialization, where *n* indicates a list of one-digit WADS IDs, 0 - 9. You can specify 1 - 10 WADS IDs. A minimum of one WADS ID is required.

Provide dynamic allocation members (using the DFSMDA macro) for all WADS data sets. All (primary and secondary) WADS must be preallocated. The WADS block size is extracted from the DSCB. The specification of WADS duplexing is provided by the WADS= EXEC parameter specified in the control region JCL (WADS=S for single WADS, WADS=D for dual WADS).

Related concepts:

Allocating log data sets (System Definition)

Exit routines (Exit Routines)

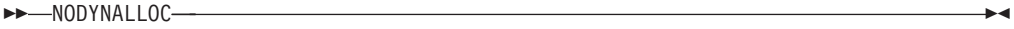
Related reference:

Log edit exit routine (DFSFLGE0) (Exit Routines)

Defining IMS batch without dynamic allocation

This section describes control statements in the DFSVSMxx member that you use to disable batch dynamic allocation.

Control statements



Parameter descriptions

NODYNALLOC
Disables batch dynamic allocation. Message DFS2480 appears at initialization time to indicate that batch dynamic allocation is not to occur. This statement disables the default setting, which is batch dynamic allocation enabled.

Disabling the re-opening of databases after an IMS restart

After an IMS warm or emergency restart, IMS re-opens all databases that were open when IMS was brought down. To disable this default behavior, specify the NOPDBO option.

Control statements

Parameter descriptions

NOPDBO
Disables the automatic re-opening after an IMS warm or emergency restart of databases previously open and authorized. The keyword NOPDBO must start in column 1. If this option is not specified, the databases are reopened.

Defining coupling facility structure names for sysplex data sharing

If you use sysplex data sharing, three coupling facility structure names must be passed to IMS: IRLM, OSAM, and VSAM.

These names are passed to IMS in the CFNAMES control statement in the DFSVSMxx member. The purpose of this statement is to let you select which of the structures, previously defined in the z/OS policy, is to be used at run time.

You can specify one or more CFNAMES control statements. Each statement can contain one or more keyword parameters. Each keyword parameter must be specified in its entirety on a single CFNAMES statement. A specific keyword parameter can not be continued from one CFNAMES statement to another. If keyword parameters are duplicated, the first parameter encountered is used and the others are registered with an error message.

The structure names that you specify in the CFNAMES control statement can be 1 to 16 characters long. All keywords in the statement must be coded, but null values for the CFOSAM and CFVSAM keywords are allowed.

IRLM

If you do not specify structure names in this PROCLIB member or specify only the IRLM structure name, the default is for the environment defaults to use two-way data sharing using the notify protocol.

If the IRLM and OSAM or VSAM structure names do not match the structure names known to the coupling facility at the time of the IDENTIFY, the IDENTIFY is rejected and IMS initialization fails. The first IMS in a data sharing group that identifies itself to the IRLM, sets the data sharing environment for all other IMS subsystems connecting to the same IRLM structure. The first IMS, in other words, sets the structure names for the coupling facility. If, for example, the first IMS does not specify structure names in this PROCLIB member, no other IMS subsystems identifying to this IRLM can specify structure names (except for an IRLM structure name matching the one set by the IRLM); otherwise, the IDENTIFY is rejected and IMS initialization fails. The same thing happens if the first IMS specifies all three structure names and, later, another IMS subsystem tries to identify using one or more different structure names.

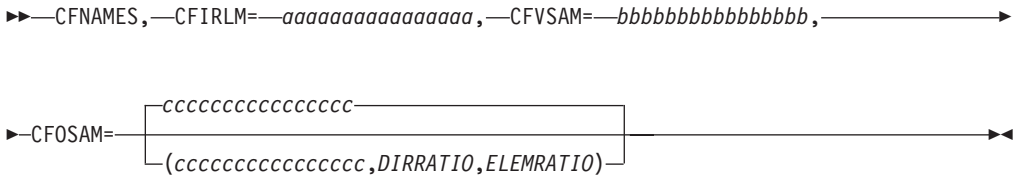
OSAM

If you are using OSAM sequential buffering, the OSAM structure registers SB buffers in the coupling facility's cross-system buffer invalidation process.

Control statements

CFNAMES must appear in column 1. No blanks can appear between the keywords.

Syntax



Parameter descriptions

CFNAMES=
Specifies the coupling facility structure names that must be passed to IMS.

CFIRLM=aaaaaaaaaaaaaaaa
Specifies the coupling facility lock table structure name for the IRLM.

CFVSAM=bbbbbbbbbbbbbbbb
Specifies the coupling facility XI structure name for VSAM.

CFOSAM=

You can specify either the coupling facility name only, or the coupling facility name and directory-to-element ratios.

ccccccccccccccc
Specifies the coupling facility structure name for OSAM.

(ccccccccccccccc,DIRRATIO,ELEMRATIO)
Specifies the coupling facility structure name for OSAM and the directory-to-element ratio that the CF uses to configure the structure for data caching.

The DIRRATIO and ELEMRATIO subparameters are optional. The installation uses the CFRM policy to specify the size of a cached structure. When the structure is allocated, its storage is subdivided to reserve space for directory entries and data elements. The directory-to-element ratio determines the proportion of the cache used for directories and data elements. The ratio is expressed as a pair of whole numbers and is used when a connection is made to the structure. The directory-to-element ratio should reflect the average number of data elements per cache entry. The size of each data element is 2 KB. OSAM data is contained in the coupling facility as multiples of 2 KB data elements. If the ratio is incorrect for an installation's use of the structure, frequent rejections of cache requests can occur because either the cache or the cache structure is full.

The result of dividing the ELEMRATIO value by the DIRRATIO value must not exceed 16, the maximum number of elements supported by an OSAM structure. If the result is greater than 16, the specified ratio is rejected and IMS supplies a ratio.

The DIRRATIO value represents the directory part of the ratio and ELEMRATIO represents the element part of the ratio. Both subparameters are coded as one to three digits. If you choose to specify one of the subparameters, you must specify both.

If you omit the DIRRATIO or specify a value of zero, IMS provides a default DIRRATIO of 1:0. If DIRRATIO specifies a non-zero value and ELEMRATIO specifies zero, IMS supplies a ratio of 1:0. This ratio causes the coupling facility to configure the OSAM structure to preclude data caching. If data caching is precluded, the OSAM structure only supports buffer invalidation processing and does not support cache data.

If you omit the DIRRATIO and ELEMRATIO subparameters, or if they fail validation of the maximum elements number, IMS provides a default ratio.

IMS supplies a default ratio of 999:1. This ratio causes the coupling facility to configure the data caching OSAM structure to optimize buffer invalidation and not data caching. While data caching might be possible, the structure provides limited space for data elements.

Related reading: For additional information about sysplex data sharing, see *IMS Version 11 System Administration*.

Specifies the coupling facility XI structure name for OSAM.

Using the coupling facility for OSAM data caching

The OSAM database coupling facility caching function allows you to optionally specify the caching of OSAM database buffers.

For more information, see the description of the caching option (co) parameter under “Defining OSAM subpools” on page 748.

When you specify the “cache only changed data” option and an application program modifies data in a subpool, changed data is written first to DASD then to the coupling facility. If you select the “cache all data” option and an application program requests data that is not already in a subpool or in the coupling facility, that data is either read from DASD or copied from the SB buffer into the subpool and then written to the coupling facility.

Performance varies, depending upon:

- The cache option you choose.
- The number of databases utilized.
- The database blocksizes.
- The number of sharing IMS/z/OS images.

The “cache only changed data” option probably provides greater performance improvement than the “cache all data” option. If you specify the “cache only changed data” option, each data block from the selected database, once read from DASD, must be written to system memory. This action requires additional path length and cycles. The specific use of the database by your application program dictates whether this option is beneficial.

Data caching initialization processing is dependent upon the proper environment and the specification of the caching option in the subpool definition statement. The following actions are performed:

- Establish an element size of 2 KB for the coupling facility.
- Allocate a buffer prefix extension for caching subpools. The extension resides as a non-contiguous area and is anchored from the prefix. It consists of two parts:
  - A fixed-length section for asynchronous processing.
  - A variable length section for data transfer buffer lists. The number of buffer lists is dependent upon the subpool buffer size.

  Example: A subpool buffer size of 6 KB requires three buffer lists and a subpool size of 0.5 KB, or 1 KB requires one buffer list. A single buffer list accommodates 2 KB of subpool buffer.
- Establish miscellaneous values for coupling facility data transfer parameters including:
  - Element and directory ratios
  - Buffer increment number (number of 256 byte segments)
  - Number of buffers in the buffer list.

The write data option is supported at the subpool level and permits the definition of each subpool with a write cache option.

Restriction: When using sequential buffering and the coupling facility for OSAM data caching, the OSAM database block size must be defined in multiples of 256 bytes (decimal). Failure to define the block size accordingly can result in ABENDS0DB PIC15 from the coupling facility. This condition exists even if the IMS system using sequential buffering is accessing the database in read-only mode.

Enabling the long busy handling function

RAMAC disk arrays sometimes cause a problem to online programs when the DASD subsystem takes a very long time to do error recovery.

This condition (known as *long busy*) might range in time from a couple of seconds to many minutes. You can avoid these excessive long wait times by defining the Multiple Area Data Sets I/O Timing (MADSIOT) keyword on the DFSVSMxx member of the IMS PROCLIB data set, thus enabling IMS system's long busy handling function.

For more information about the long busy function, see Managing I/O errors and long wait times (Database Administration) in *IMS Version 11 Database Administration*.

structurename
Specifies the name of the coupling facility list structure that IMS uses for the long busy handling function.

iotime Specifies the maximum time (in seconds) before the long busy handling function is activated. The allowable values are 0 to 255. This value is passed to the I/O subsystem for all I/O requests to MADS. If the time interval specified expires before the request completes, the request is terminated. This explicit request timing value takes precedence over an I/O timing value specified at the volume level.

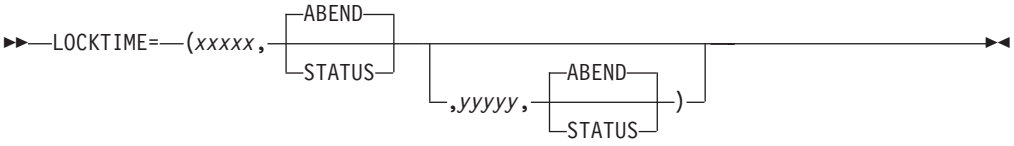
Enabling the IRLM lock timeout function

The IRLM Lock Timeout function enables you to interrupt processes that are waiting for locks for longer than a specified number of seconds.

To use this function, specify a positive decimal integer value from 1 to 32767 for the LOCKTIME parameter in the DFSVSMxx member of the IMS PROCLIB data set (or DFSVSAMP DD statement for IMS batch procedures). The LOCKTIME parameter specifies the number of seconds that IRLM waits before rejecting lock requests that have not been granted. For example, when LOCKTIME=10 is specified, IRLM waits ten seconds before rejecting lock request that have not been granted.

When no value is specified for the LOCKTIME parameter, IRLM issues message DXR162I to indicate that a task has held (or has been waiting for) a lock. The first message is issued after five minutes; thereafter, messages are issued at intervals of one minute. RMF records (type 79.15) can be formatted for more information about the task that is holding (or waiting for) the lock.

When a value is specified for the LOCKTIME parameter, IRLM interrupts all the dependent regions that have been waiting for a lock for longer than the number of seconds specified by the LOCKTIME parameter. An ABENDU3310 is issued for every dependent region that waits for longer than the number of seconds specified by the LOCKTIME parameter. The number of seconds can be changed after IMS initialization by issuing the command `MODIFY,irlmproc,SET,TIMEOUT=nn`.



ABEND
Specifies that an abend occurs when the value specified for LOCKTIME is exceeded. ABEND is the default.

STATUS
Specifies that a status code of BD be returned when the value specified for LOCKTIME is exceeded.

xxxxx When LOCKTIME=(*xxxxx*), *xxxxx* specifies the online timeout value in seconds. Valid values are from 1 to 32767. Optionally, you can specify ABEND or STATUS after *xxxxx* to determine whether an abend occurs or a status code is issued when the *xxxxx* value is reached. If a second timeout value is not specified, a value of *xxxxx* is also used for batch or a BMP. Unless *xxxxx* is specified as a multiple of the local IRLM deadlock timer value, the timeout occurs at the next deadlock timer interval.

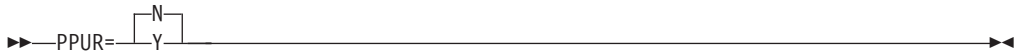
yyyyy If specified in the DFSVSMxx PROCLIB member after *xxxxx*, defines the timeout value, in seconds, for BMPs or JBPs. Valid values are from 1 to 32767. Optionally, you can specify ABEND or STATUS after *yyyyy* to determine whether an abend occurs or a status code is issued when the *yyyyy* value is reached.

If specified in the DFSVSAMP PROCLIB member after *xxxxx*, defines the timeout value, in seconds, for batch. Valid values are from 1 to 32767. Optionally, you can specify ABEND or STATUS after *yyyyy* to determine whether an abend occurs or a status code is issued when the *yyyyy* value is reached.

Unless *xxxxx* and *yyyyy* are specified as a multiple of the local IRLM deadlock timer value, the timeout occurs at the next deadlock timer interval. For example, if *xxxxx* is 5 seconds and *yyyyy* is not specified, and the deadlock timer value is (3,1), the timeout occurs at 6 seconds because that is a multiple of 3.

Preventing transactions from being terminated when HALDB partitions are unavailable

Use the PPUR= control statement on the DFSVSMxx member to prevent transactions with a processing option of PROCOPT=GOx from being terminated if a HALDB partition is unavailable due to a database command in progress. This condition would cause either STATUSGG or STATUSBA to be issued.



N Do not prevent STATUSGG or STATUSBA from being issued if a HALDB partition is unavailable because a database command is in progress.

Y If a HALDB partition is being accessed by a transaction with a processing option of PROCOPT=GOx, and the partition is unavailable due to a database command in progress, terminate the transaction and reschedule it.

Preventing DBRC calls for HALDB version verification

Use the PSELNODBRC control statement in the DFSVSMxx member of the IMS PROCLIB data set to prevent the Database Recovery Control facility (DBRC) from being called when partition selection (either with key range selection or with a partition selection exit) finds that a key requested by an application call is not in the range of any partitions.

Calling DBRC in this case can cause performance degradation especially when many similar application calls are made or when many other applications or utilities are using the RECON data set.

PSELNODBRC
Specify this keyword if you want to prevent DBRC from being called when partition selection (either with key range selection or with a partition selection exit) finds that a key requested by an application call is not in the range of any partitions.

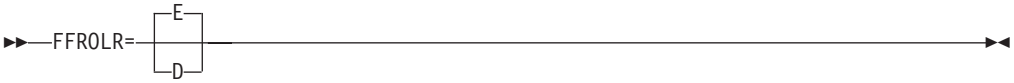
Otherwise, omit this keyword.

Related concepts:

Overview of DBRC (System Administration)

Resuming an online reorganization for HALDBs during IMS warm or emergency restart

To resume the online reorganization of a high-availability large database (HALDB) during an IMS warm or emergency restart, use the FFROLR= control statement in the DFSVSMxx member of the IMS PROCLIB data set.



E Specifies that after an IMS warm or emergency restart, IMS should attempt to automatically resume the OLREORG process that it owned before the restart. This is the default.

D Specifies that after an IMS warm or emergency restart, IMS does not automatically resume the OLREORG process that it owned before the restart. Instead, the RECON is updated to release the ownership of the OLREORG process when IMS restarts.

Discarding preallocated SDEP control intervals

When the SDEP scan and delete utilities are invoked, they write out all preallocated control intervals (CIs). These CIs contain no user data. When the CIs are written to disk, the SDEP utilities can either maintain the preallocated CIs for subsequent SDEP inserts or discard them.

If you want the SDEP utilities to discard the preallocated CIs, use the SDEPQCI keyword of the DFSVSMxx member of the IMS PROCLIB data set. By using SDEPQCI, you do not have to include the QUITCI utility control statement in each SYSIN member. SDEPQCI makes QUITCI the default for the SDEP utilities.

SDEPQCI=
Specifies that QUITCI is the default option for the SDEP scan and delete utilities. If you do not specify SDEPQCI, the SYSIN of the utility JCL determines how the preallocatted CIs are treated.

utilityoption
One-charater alphabetic field that identifies which SDEP utilities use the QUITCI function. There is no default.

S The SDEP scan utility uses the QUITCI function.

D The SDEP delete utility uses the QUITCI function.

B Both the scan and delete SDEP utilities use the QUITCI function.

If you specify a value other than S, D, or B, message DFS2835I is issued.

Preventing a /DBRECOVERY command for a database that has INDOUBT EEQEs

When the NODBR keyword is specified, a /DBRECOVERY command is not processed for any database that has an INDOUBT Extended Error Queue Element (EEQE), and message DFS0488I RC=43 is issued.

This statement causes a /DBRECOVERY command to fail if the command is issued against a database that has INDOUBT EEQEs. NODBR does not apply to Fast Path databases.

Related reference:

/DBRECOVERY command (Commands)

Related information:

DFS0488I (Messages and Codes)

DFSYDTx member of the IMS PROCLIB data set

Use the DFSYDTx member of the IMS PROCLIB data set to specify the OTMA client descriptors and the OTMA destination descriptors that are built during IMS initialization.

The *x* on DFSYDTx is the IMS nucleus suffix.

OTMA client descriptor syntax and parameters

Use the OTMA client descriptor to specify attributes, limits, and types of support for an individual OTMA client.

To specify global attributes that apply to all OTMA clients, use the DFSOTMA global client descriptor.

The values that you can specify for an individual OTMA client include:

- The use of a DRU exit
- Message flood protection
- CM1/CM0 ACK timeout value

An OTMA client descriptor can also be used to set the global tpipe limit for all the OTMA members or clients.

OTMA client descriptors are optional.

You can specify a maximum of 254 OTMA client descriptors in the DFSYDTx member of the IMS PROCLIB data set. If you do not use the DFSOTMA descriptor, you can specify 255 OTMA client descriptors.

Client descriptors are always loaded from the DFSYDTx PROCLIB member regardless of the IMS restart type. There are no checkpoint log records for client descriptors.

Format

Up to 50 lines can be used in the specification of a descriptor. Columns 1 - 18 must be the same for each line. All parameters are delimited with a blank.

The following table includes information about the format of an OTMA client descriptor:

Table 83. Format of an OTMA client descriptor

Column	Contents	Description
1	M	Identifies this descriptor as an OTMA client descriptor.
2	Blank	This field is left blank.
3-18	A 1- to 16-character OTMA client name.	1- to 16-character OTMA client name, left-aligned, and padded with blanks if necessary. This parameter is required and positional. OTMA client names must be unique. For IMS Connect, this name must match the value of the MEMBER parameter on a DATASTORE configuration statement. The OTMA client name must follow the resource naming conventions: • Can use only characters A - Z, 0 - 9, @, $ • Cannot contain embedded blanks • Cannot be a reserved word (for example, “TO” or “SECURITY”) • Cannot begin with “DFS” or “DBCDM” • Cannot be an IMS keyword (for example, “LINE” or “NODE”)
19	Blank	This field is left blank.
20-72	OTMA client descriptor parameters.	Enter the parameters in any order. Separate parameters by using a blank space.
73–80	Sequence numbers.	These columns are ignored by IMS.

Parameters

You can specify the following parameters on the OTMA client descriptor:

ALTPCBE=NO | YES
Specifies whether the OTMA shared-queues ALTPCB output, which is originated from an OTMA front-end IMS and generated at the back-end IMS, must be delivered to the front-end IMS for the OTMA client. This parameter is optional and defaults to ALTPCBE=NO.

If the super member function is activated for the ALTPCB output at the shared-queues back-end IMS, the super member takes precedence.

DRU=*exit_name*
8-character OTMA Destination Resolution exit routine name. Duplicate exit routine names are allowed.

INPT=
A 1- to 4-digit decimal integer that indicates the maximum number of input messages from this member that can be waiting at the same time to be processed.

Valid values are 200 - 9999 or 0. If the value is 1 - 0200, it is treated as 0200. If the value is over 9999, a value of 9999 is used.

If you specify a value of 0, OTMA deactivates the input message flood detection for this client; however, OTMA still includes the input messages from this client in the calculation global message counts.

Internally, each OTMA input message is represented by a transaction instance block (TIB). Therefore, within the context of OTMA, the terms TIB and input message are generally synonymous.

When message flood control is active for the OTMA client and the number of input messages from this client reaches 80% of the INPT= value, OTMA issues the DFS1988W warning message to the MTO and system console, and issues a protocol messages to the OTMA client. Thereafter, OTMA reissues DFS1988W at every 5% increase in the message count, until the number of messages reaches the maximum.

When the message flood limit is reached, OTMA rejects all the new input transactions from the member. OTMA issues DFS1989E error message to the console and MTO and a server state protocol command to the OTMA client. The OTMA client can then choose to take corrective action, such as rerouting new transaction requests to a different IMS system. The /DISPLAY TMEMBER command shows FLOOD.

When the input messages in the system are processed and the number of total TIBs in the system is 50% or less of the global limit, the flood status is relieved. A DFS0767I message is sent to the MTO and system console, and a server state protocol message is sent to the OTMA client.

MAXTP=
A 1- to 5-digit decimal integer from 0 - 99999 that defines the maximum number of tpipes that this OTMA client member can have in an IMS system.

If 0 is specified, OTMA does not monitor or limit the number of tpipes that this OTMA client can have in the IMS system. If a value of 1 to 200 is specified, it is treated as 200. Any number over 99999 is rejected.

Setting a maximum number of tpipes for the OTMA client activates the monitoring of the number of tpipes that the OTMA client has in the IMS system.

When tpipe monitoring is active, if the total number of tpipes reaches 80% of the value that is specified on the MAXTP parameter, IMS issues warning message DFS4382W to the system console and the master terminal operator (MTO).

If the total number of tpipes from this client reaches the maximum number, OTMA rejects all new TPIPE creation requests from the OTMA member with a NAK with sense code X'29'. IMS sends DFS4383E error message to the system

console and MTO and notifies the OTMA client with an OTMA protocol message. The /DISPLAY TMEMBER command shows "MAX TPIPE" in the user status of the client member.

After the total number of the tpipes for the client drops 50% of the maximum allowable number, IMS issues message DFS4384I to indicate that the number of tpipes returned to normal.

If MAXTP is not specified in the DFSOTMA descriptor, the OTMA client descriptor with the highest MAXTP value defines a global warning threshold for the total number of tpipes in the system for all OTMA clients. When this global warning threshold is reached, IMS issues message DFS4385W to the MTO and system console and OTMA issues a server state protocol command to all OTMA clients. IMS does not prevent the creation of new TPIPEs.

When the global TPIPE count falls to 80% of the global TPIPE warning threshold or below, IMS issues message DFS4386I to the MTO and system console and OTMA notifies all OTMA clients with another protocol message.

When MAXTP is specified in the DFSOTMA descriptor, IMS uses the DFSOTMA MAXTP value as the global warning limit instead of the highest MAXTP value among the OTMA client descriptors and the warning thresholds are different. For more information about the effect of the DFSOTMA MAXTP parameter, see DFSOTMA descriptor syntax and parameters (System Definition).

The /DISPLAY OTMA and /DISPLAY TMEMBER commands result in displays of the current numbers of TPIPEs and MAXTP value for members and OTMA server using the function.

MAXTPBE=YES | NO
Specifies whether (YES) or not (NO) the MAXTP must be monitored when this IMS is a shared queues back-end system processing a front-end initiated OTMA input transaction at the application GU time. This is designed for member when MAXTP is specified. It is ignored if the member does not have MAXTP specified.

A MAXTPBE value specified on an OTMA client descriptor overrides the MAXTPBE value specified on the DFSOTMA descriptor, if any.

If the MAXTPBE parameter is specified in the DFSOTMA descriptor, the DFSOTMA specification determines the default value of the MAXTPBE parameter for OTMA clients. If the MAXTPBE parameter is not specified on the DFSOTMA descriptor or no DFSOTMA descriptor exists, MAXTPBE=YES is the default for an OTMA client.

MAXTPRL=
Specifies the relief level for the MAXTP threshold as a percentage of the maximum number of tpipes that this client can have. Valid values are 50 - 70. If a value less than 50 is specified, it is set to 50. Values over 70 are set to 70.

A MAXTPRL value specified on an OTMA client descriptor overrides the MAXTPRL value specified on the DFSOTMA descriptor, if any.

If the MAXTPRL parameter is specified in the DFSOTMA descriptor, the DFSOTMA specification determines the default value of the MAXTPRL parameter for OTMA clients. If the MAXTPRL parameter is not specified on the DFSOTMA descriptor or no DFSOTMA descriptor exists, MAXTPRL=50 is the default for an OTMA client.

TODUMP=NO | YES
Specifies whether the symptom dump and the DFS554A message of U0243

pseudoabend for OTMA transaction expiration must be issued. This parameter is optional and defaults to TODUMP=NO.

T/O=

A 1- to 3-digit decimal integer 0 - 255. It indicates the timeout value, in seconds, for OTMA to wait for acknowledgments. If the value is over 255, OTMA uses 255. If the value is 0, OTMA deactivates the timeout function. The default value is 120.

ACK timeout intervals can be specified for the following types of OTMA output messages:

- Some send-then-commit (CM1) response messages
- Commit-then-send (CM0) response messages

You can override the descriptor timeout value by taking one of the following actions:

- Issuing the /START TMEMBER TIMEOUT command to reset the timeout value
- Specifying a lower timeout value by using the client bid of an OTMA client, such as IMS Connect.

Example

The following example shows an OTMA client descriptor. The descriptor, HWSICON1, specifies the DFSYDRU0 exit for TMEMBER HWSICON1.

```
M HWSICON1         DRU=DFSYDRU0
```

OTMA destination descriptor syntax and parameters

Use OTMA destination descriptors to define message switch destinations from the ALT IOPCB for OTMA clients, such as IMS Connect, or non-OTMA clients, such as SNA terminals or printers. The OTMA destination descriptors can also be used to define destinations for synchronous callout messages and messages that are destined for remote IMS systems on a TCP/IP connection.Use OTMA destination descriptors to define message switch destinations from the ALT IOPCB for OTMA clients, such as IMS Connect, or non-OTMA clients, such as SNA terminals or printers. The OTMA destination descriptors can also be used to define destinations for synchronous callout messages.

Subsections:

- "Creating and modifying destination descriptors"
- "Destination descriptor format" on page 776
- "Common parameters" on page 776
- "IMS Connect parameters" on page 778
- "Example" on page 778

Creating and modifying destination descriptors

The OTMA destination descriptors are specified in the OTMA DFSYDTx member of the IMS.PROCLIB data set. You can use them to externalize message switch definitions that would otherwise be coded in the OTMA routing exit routines DFSYPRX0 and DFSYDRU0. The OTMA destination descriptors can be used instead of the DFSYPRX0 and DFSYDRU0 exit routines or in addition to them.

Destination descriptors are loaded from the DFSYDTx member of the IMS.PROCLIB data set during IMS cold start or /ERESTART COLDCOMM processing, or from the X'4035' checkpoint log records during IMS restart by the /NRESTART or /ERESTART command.

Alternatively, you can dynamically add, update, or delete OTMA destination descriptors with the following type-2 commands:

- CREATE OTMADESC
- UPDATE OTMADESC
- DELETE OTMADESC

A restart of IMS is not required; in addition, any changes that you make with these type-2 commands persist across warm and emergency restarts, when the changes are read from the IMS logs.

Use the QUERY OTMADESC command to display the characteristics of a specific destination routing descriptor.

For callout requests to IMS Connect clients, you can use the OTMA destination descriptors to specify adapter and converter names for XML message conversion.

A descriptor that matches the destination sets the default routing as either IMS Connect, WebSphere MQ, or non-OTMA, as coded on the descriptor. OTMA routing exits are not called if they exist. The OTMA routing descriptors override the exits unless EXIT=Y is specified for the descriptor.

You can specify a maximum of 255 OTMA destination descriptors in the DFSYDTx member of the IMS PROCLIB data set.

Destination descriptor format

Up to 50 lines can be used in the specification of a descriptor. Columns 1 - 10 must be the same for each line.

Table 84. Format of an OTMA destination descriptor

Column	Contents	Description
1	D	Identifies this descriptor as an OTMA destination descriptor.
2	Blank	This field is left blank.
3-10	A 1- to 8- character destination name	The destination name must be left-aligned and padded with blanks if necessary. The destination name can be masked by ending it with an asterisk (*). OTMA destination names must be unique. This is a required, positional parameter.
11	Blank	This field is left blank.
12-72	Descriptor parameters	Enter the parameters in any order. Separate parameters by using a blank space.
73–80	Sequence numbers	These columns are ignored by IMS.

Common parameters

You can specify the following parameters on the OTMA destination descriptor:

SMEM=NO | YES
Specifies whether this destination is a supermember. This parameter is optional and the default is SMEM=NO. When SMEM=YES, TMEMBER is the 1- to 4-character supermember name.

If you specify this parameter when TYPE=NONOTMA, it is ignored.

SYNTIMER=*value*
An optional parameter to specify the time to wait in 100th of a second for the synchronous callout process to finish.

This parameter is ignored when TYPE=NONOTMA or when the descriptor is used to route a non-synchronous callout message.

The valid range is 0 - 999999. If SYNTIMER is set to 0, a system default of 10 seconds is used instead. If a value larger than 999999 (more than 6 characters) is specified, a DFS2385E error message is generated and no default value is set.

The AIBRSFLD parameter in the DL/I ICAL SENDRECV call that is used to issue a synchronous callout request can override this value. If the AIBRSFLD parameter is not set or set to 0, the timeout value that is specified in the OTMA destination descriptor is used. If no timer is specified in either the AIBRSFLD parameter of the ICAL SENDRECV call or the OTMA destination descriptor, the system default is 10 seconds. If the timeout value is specified both on the OTMA destination descriptor and on the ICAL call, OTMA uses the smaller value of the two.

- If AIBRSFLD is set to 0, the SYNTIMER timeout value from the OTMA destination descriptor is used.
- If values are specified for both AIBRSFLD and SYNTIMER, the smaller value is used.
- If AIBRSFLD is blank and the OTMA destination descriptor does not have a value for SYNTIMER, the timeout value is set to 10 seconds.

When the timeout value is reached, the IMS application that issues the synchronous callout request receives a return code of X'100' and a reason code of X'104'. The message is discarded.

TMEMBER=*name*
A 1- to 16-character OTMA TMEMBER name. When SMEM=YES, TMEMBER is the 1- to 4-character supermember name.

This parameter is required for TYPE=IMSCON and ignored for TYPE=NONOTMA.

TPIPE=*name*
A 1- to 8-character TPIPE name. The default is the destination name.

This parameter is required for TYPE=IMSCON and ignored for TYPE=NONOTMA.

TYPE=*type*
Specifies what type of destination the descriptor is for. This parameter is required. The valid values are:

IMSCON
IMS Connect client

NONOTMA
Non-OTMA destination

IMS Connect parameters

The TMEMBER, TPIPE,SMEM, SYNTIMER, EXIT, and USERID parameters are valid for IMS Connect descriptors (TYPE=IMSCON), as well as the following additional parameters:

ADAPTER=
A 1- to 8-character name of the adapter that identifies the IMS Connect adapter to be used on these messages. For example, you can specify an adapter for XML transformation. This parameter is optional. If you specify this parameter, CONVRTR= is also required.

CONVRTR=
A 1- to 8-character name of the converter to be used by the adapter specified on ADAPTER=. This parameter is required when TYPE=IMSCON if the ADAPTER parameter is specified.

Example

The following example shows several OTMA destination descriptors.

```
D OTMACL99 TYPE=IMSCON TMEMBER=HWS1 TPIPE=HWS1TP01
D OTMACL*  TYPE=IMSCON TMEMBER=HWS2
D PRNTR3A  TYPE=NONOTMA
D SOAPGWAY TYPE=IMSCON TMEMBER=HWS2 TPIPE=HWS2SOAP SYNTIMER=20
D SOAPGWAY ADAPTER=XMLADPTR CONVRTR=XMLCNVTR
```

The first descriptor is a destination descriptor for destination OTMACL99 to route messages to TMEMBER HWS1 and TPIPE HWS1TP01.

The second descriptor is for destinations that match the mask OTMACL*. Messages are routed to IMS Connect TMEMBER HWS2, with a TPIPE of the destination that matches the mask (for example, OTMACL04).

The third descriptor is another destination descriptor for destination PRNTR3A that is routed to IMS.

The last descriptor is a destination descriptor for IMS Enterprise Suite SOAP Gateway, with a destination name of SOAPGWAY. Messages are routed to IMS Connect TMEMBER HWS2 with TPIPE HWS2SOAP, with a timeout value of 0.2 seconds for synchronous callout processing, and then are processed by the specified XML adapter.

Related reference:

CREATE OTMADESC command (Commands)

DELETE OTMADESC command (Commands)

QUERY OTMADESC command (Commands)

UPDATE OTMADESC command (Commands)

DFSOTMA descriptor syntax and parameters

Use the optional DFSOTMA descriptor to define global and default attributes, limits, and types of support that apply to all OTMA clients.

The DFSOTMA descriptor is an OTMA type-M descriptor that sets system defaults and global parameters for all OTMA clients that connect to this IMS system.

The DFSOTMA descriptor does not support all of the parameters that you can specify on an OTMA client descriptor parameters. Parameters that are not supported, such as DRU=, T/O=, and INPT=, are ignored if they are specified in the DFSOTMA client descriptor.

The global values that you can specify for all OTMA clients include:

- Message flood protection
- CM1/CM0 ACK timeout value
- Global tpipe limit for all the OTMA members or clients

If DFSOTMA is used, IMS builds a type-M descriptor block to store the global information specified. The block counts toward the 255 maximum number of type-M descriptors, so when the DFSOTMA descriptor is used, the maximum number of OTMA client descriptors is 254.

The DFSOTMA descriptor is always loaded from the DFSYDT*x* PROCLIB member regardless of the IMS restart type. There are no checkpoint log records for the DFSOTMA descriptor.

Format

Up to 50 lines can be used in the specification of a descriptor. Columns 1 - 18 must be the same for each line. All parameters are delimited with a blank.

The following table shows the format of the DFSOTMA descriptor:

Table 85. Format of the DFSOTMA descriptor

Column	Contents	Description
1	M	The DFSOTMA descriptor uses the same type designator as an OTMA client descriptor.
2	Blank	This field is left blank.
3-18	DFSOTMA	The name of the descriptor. The name DFSOTMA defines this descriptor as the OTMA descriptor that sets system defaults and global attributes for all OTMA clients. The name DFSOTMA is required and positional.
19	Blank	This field is left blank.
20-72	DFSOTMA descriptor parameters.	Enter the parameters in any order. Separate parameters by using a blank space.
73–80	Sequence numbers.	These columns are ignored by IMS.

Parameters

You can specify the following parameters on the DFSOTMA descriptor:

MAXTP=
A 1- to 5-digit decimal integer from 0 - 99999 that defines the maximum total number of tpipes in an IMS system for all OTMA clients.

If a value of 0 is specified, which is the default, OTMA stops monitoring the creation of the TPIPE. If the value is between 1 and 200, it is treated as 200. Any number over 99999 is rejected.

If the total number of tpipes in the IMS system reaches the number that is specified on the DFSOTMA MAXTP parameter, OTMA rejects all new TPIPE creation requests from all OTMA members with a NAK with sense code X'29'. IMS sends DFS4516E error message to the system console and MTO and notifies all OTMA clients with an OTMA protocol message. The /DISPLAY OTMA command shows "MAX TPIPE" in the user status of the OTMA server.

Before the DFSOTMA MAXTP value is reached, IMS issues message DFS4515W as a warning when the number of tpipes in the IMS system reaches 80% of DFSOTMA MAXTP value. OTMA also issues a protocol message to all OTMA clients. When the number of TPIPEs for all members falls to 50% or a user-specified percentage of the global TPIPE warning threshold, IMS issues message DFS4517I to the IMS system console and MTO and OTMA notifies all OTMA clients of the improved status with a protocol message.

If MAXTP is not specified in the DFSOTMA descriptor and one or more OTMA members define a MAXTP value, the highest MAXTP value that is specified among the OTMA clients defines only a global warning threshold for the total number of tpipes in the system for all OTMA clients. When this global warning threshold is reached, IMS does not prevent the creation of new TPIPEs, but merely issues message DFS4385W to the MTO and system console and OTMA warns all OTMA clients with a server state protocol message. When the global TPIPE count falls to 80% of the global TPIPE warning threshold or below, IMS issues message DFS4386I to the MTO and system console and OTMA notifies all OTMA clients with another protocol message.

The /DISPLAY OTMA and /DISPLAY TMEMBER commands can display the current numbers of TPIPEs and the DFSOTMA MAXTP value.

MAXTPBE=YES | NO
Defines the IMS system default for MAXTPBE. The DFSOTMA MAXTPBE value is used by OTMA clients that do not specify MAXTPBE themselves.

MAXTPBE determines whether the number of tpipes in the IMS system must be monitored when this IMS is a shared queues back-end system that processes front-end initiated OTMA input transactions at the application GU time. MAXTPBE applies only when MAXTP is specified and is ignored if MAXTP value is not specified. The default is YES.

MAXTPRL=
Specifies the relief level for tpipe flood conditions as a percentage of the DFSOTMA MAXTP value. The DFSOTMA MAXTPRL value sets the default for OTMA clients that do not specify a MAXTPRL value themselves.

Valid values are from 50 to 70. Values specified that are below 50 are set to 50. Values over 70 are set to 70.

The default relief level is 50% of the TPIPE MAXTP threshold.

Example

The following example shows a DFSOTMA descriptor.

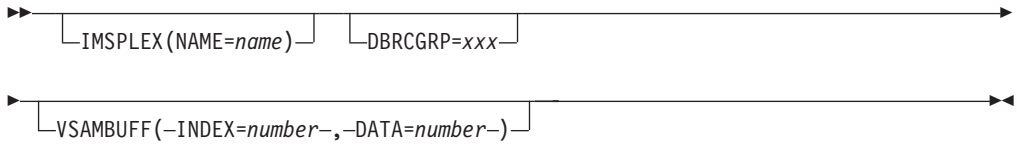
```
M DFSOTMA          MAXTP=5000 MAXTPRL=60
```

DSPBIxxx member of the IMS PROCLIB data set

Use the DBRC initialization member of the IMS PROCLIB data set (DSPBIxxx) to specify parameters that initialize the DBRC address space. Some parameters in this member of the IMS PROCLIB data set can be overridden with DBRC execution parameters.

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.

Syntax



Usage

A DSPBIxxx member consists of one or more fixed-length character records (the configuration data set can be of any LRECL greater than eight, but it must be fixed record format). The rightmost-eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords. Statements that begin with a "*" or "#" in column 1 are comment lines and are ignored. Additionally, comments can be included anywhere within a statement by enclosing them between " /* " and "*/", for example, `/* PROCLIB comments */`. Values coded in this member of the IMS PROCLIB data set are case sensitive. In general, you should use uppercase for all parameters.

BPE considerations

Use the BPE user exit list member of the IMS PROCLIB data set to define DBRC user exits to BPE. The member is specified on the EXITMBR= parameter in the BPE configuration parameter member of the IMS PROCLIB data set.

A sample DBRC BPE user exit list member of the IMS PROCLIB data set is shown in the following example.

```
/*****************************************************************/
/* DBRC USER EXIT LIST PROCLIB MEMBER                            */
/*****************************************************************/
#---------------------------------------------------------------#
# DEFINE 1 RECON I/O EXIT: ZDBRCIO0                             #
# WITH AN ABEND LIMIT OF 8.                                     #
#---------------------------------------------------------------#
EXITDEF(TYPE=RECONIO,EXITS=(ZDBRCIO0),ABLIM=8,COMP=DBRC)
#---------------------------------------------------------------#
# DEFINE 1 DBRC SECURITY EXIT: ZDBRCSE0                         #
#---------------------------------------------------------------#
EXITDEF(TYPE=SECURITY,EXITS=(ZDBRCSE0),COMP=DBRC)
```

Parameters

IMSPLEX()
Specifies the IMSplex name used by DBRC for SCI registration. The IMSplex name is passed to the DBRC SCI registration exit routine, DSPSCIX0, if it exists. The sample version of DSPSCIX0 that ships with IMS returns the value that you supply to DBRC as the IMSplex name. This parameter is optional and can be overridden by the IMSPLEX execution parameter. Use DBRC SCI registration exit, DSPSCIX0, to determine the IMSplex name instead of the IMSPLEX parameter. Only one IMSPLEX keyword can be specified. The IMSPLEX keyword must precede the left parenthesis. The IMSPLEX definition parameters follow:

NAME=
A 1- to 5-character user-specified identifier that is concatenated to 'CSL' to create the cross-system coupling facility (z/OS cross-system coupling facility) CSL IMSPLEX group name. The value specified here must match the IMSPLEX NAME= value specified in the SCI startup procedure. All DBRC instances in the same IMSplex group that are sharing either databases or message queues must specify the same identifier.

DBRCGRP=
Specifies the three-character DBRC group ID, left-justified and padded with blanks, if necessary. The DBRC group ID is passed to the DBRC SCI registration exit routine, DSPSCIX0 if it exists. The sample version of DSPSCIX0 that ships with IMS returns the value that you supply to DBRC as the DBRC group ID. This parameter is optional and can be overridden by the DBRCGRP execution parameter.

Recommendation: Use the DBRC SCI registration exit, DSPSCIX0, to determine the DBRC group ID instead of the DBRCGRP parameter.

VSAMBUFF()
Specifies the maximum number of index and data buffers to be assigned to the VSAM local shared resource (LSR) pool. Only one VSAMBUFF keyword can be specified. The VSAMBUFF keyword must precede the left parenthesis. The VSAMBUFF definition parameters include:

INDEX=
Specifies the number of index buffers to be used. The valid range is 4 to 32 767. The default is 60.

DATA=
Specifies the number of data buffers to be used. The valid range is 4 to 32 767. The default is 120.

Examples

A sample DBRC initialization member of the IMS PROCLIB data set is shown in the following example. The sample, called DSPBI000, is provided as part of the IMS PROCLIB data set.

```
/************************************************************/
/* DBRC INITIALIZATION PROCLIB MEMBER                       */
/************************************************************/
VSAMBUFF(INDEX=60,DATA=120)  /* Set number of #VSAM LSR buffs*/
```

Related concepts:

Chapter 14, “IMS Syntax Checker,” on page 283

HWSCFGxx member of the IMS PROCLIB data set

Use the IMS Connect configuration member of the IMS PROCLIB data set, HWSCFGxx, to specify environmental settings for IMS Connect.

IMS Connect uses the information it retrieves from this member of the IMS PROCLIB data set to establish communication with IMS and TCP/IP. You can define several configuration members in the partitioned data set to select from during IMS Connect startup. Specify the member name to use in the HWSCFG= parameter of the IMS Connect startup JCL.

Use uppercase characters to specify the statement and parameter keywords in the IMS Connect configuration member.

You can use the IMS Syntax Checker to modify this member of the IMS PROCLIB data set.

Environments

The HWSCFGxx member of the IMS PROCLIB data set applies to IMS Connect, which runs in its own z/OS address space.

IMS Connect can interface with IMS DB systems that run in the DB/DC and DBCTL environments and with IMS TM systems that run in DB/DC and DCCTL environments.

Usage

The IMS Connect configuration member defines how IMS Connect communicates with TCP/IP, Open Transaction Manager (OTMA), IMSplexes, Open Database Manager (ODBM), and security software.

A HWSCFGxx member consists of one or more fixed-length character records. The configuration member must be in a data set whose format is fixed block, 80-byte record length. The right-most eight columns are ignored but can be used for sequence numbers or any other notation. Keyword parameters can be coded in the remaining columns in free format, and can contain leading and trailing blanks. You can specify multiple keywords in each record; use commas or spaces to delimit keywords.

Statements that begin with an asterisk (*) or number sign (#) in column 1 are comment lines, which are ignored.

Related concepts:

Chapter 14, “IMS Syntax Checker,” on page 283

Related tasks:

“Configuring IMS Connect” on page 210

Related reference:

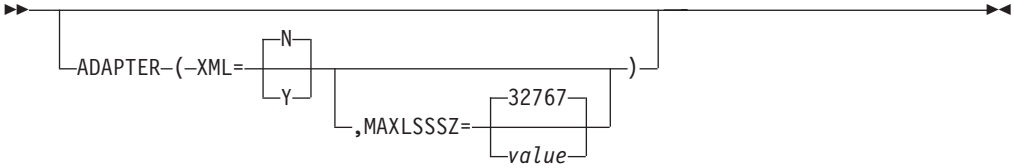
Format of user portion of IRM for HWSSMPL0, HWSSMPL1, and user-written message exit routines (Communications and Connections)

SSL initialization (Communications and Connections)

ADAPTER statement

Use this statement to define the characteristics of adapters that are used to convert XML input messages to other application programming languages, such as COBOL or PL/I.

ADAPTER statement syntax



ADAPTER statement parameters

XML

N XML adapter support should not be enabled. This is the default.

Y XML adapter support should be enabled. The XML adapter is used to convert the user data in the response message into XML. IMS Connect then sends the output message to the IMS Enterprise Suite SOAP Gateway.

MAXLSSSZ
Specifies the maximum language structure segment size. This value is passed to the XML converter when it is called. Valid values are from 5 to 32767. This parameter is optional and defaults to 32767.

Related tasks:

"Configuring XML conversion support for IMS Connect clients" on page 215

DATASTORE statement

The DATASTORE statement defines a connection to the OTMA component of an IMS data store. You can specify multiple DATASTORE statements or omit the DATASTORE statement if this IMS Connect instance does not communicate with OTMA.

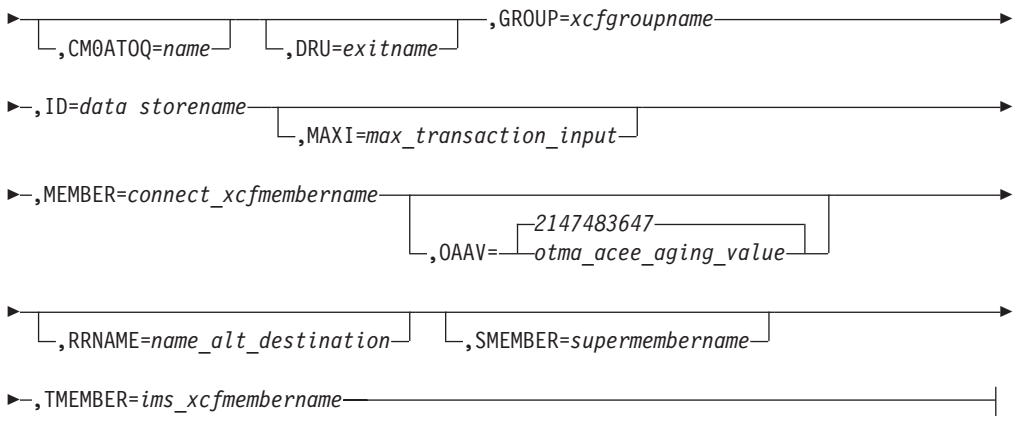
IMS Connect does not enforce a limit on the number of DATASTORE statements that you can define. Resource supply and consumption, which vary depending on the configuration of your subsystems, your operating environments, and your workloads, could possibly limit the number of active data store connections that you can use at any one time; however, you are unlikely to hit that limit because IMS Connect control blocks are relatively small and can extend into extended private storage (EPVT). A degradation in performance is likely to occur before any resource-imposed limit is reached.

Another potential limit could be imposed by your z/OS cross-system coupling facility (XCF) configuration. An XCF group has a limit to the number of XCF members that it can support. Each data store connection is treated by XCF as a separate member and an XCF group also contains non-IMS Connect members, such as IMS or WebSphere MQ. Therefore, the number of XCF members and the number of XCF groups that you define, as well as your XCF workload, could limit the number of data store connections that you can define.

For more information about XCF groups, see *z/OS MVS Setting Up a Sysplex*.

DATASTORE statement syntax

A:



DATASTORE statement parameters

ACKTO=

Specifies a timeout interval for acknowledgements to OTMA for CM0 and CM1 output messages.

The timeout value can be from 0 to 255 seconds. If the timeout value is 0 or is not specified, the OTMA ACK timeout default value of 120 seconds is set.

APPL=

A 1- to 8-character alphanumeric TCP/IP APPL name defined to RACF in the PTKTDATA statement. This parameter is optional and defaults to blanks. If you are using PassTicket and user message exits, you must specify the APPL on the DATASTORE statement.

The APPL name is passed to RACF whether or not a PassTicket is actually used, and therefore can be used to verify a user's authority to access the datastore.

CM0ATOQ

1- to 8-character name of the OTMA CM0 ACK timeout queue. The value specified here overrides both the OTMA default value of DFS$$TOQ and any value set on the HWS statement.

DRU=

A 1- to 8-character alphanumeric DRU keyword. The DRU keyword enables you to specify your own OTMA destination resolution user exit name that is to be passed to OTMA. The DRU exit is required to support asynchronous output to IMS Connect clients. The default is DFSYDRU0, but you can write your own exit.

GROUP=

The z/OS cross-system coupling facility group name for the IMS OTMA. IMS Connect uses this value to join one or more XCF groups. Because IMS Connect and IMS must be in the same XCF group in order to communicate, this group name must match the XCF group name that you define to IMS (*GRNAME*) in the IMS startup JCL (for example, "OTMA=Y,***GRNAME***=&***GROUP***,USERVAR=&MEMBER",...). Each IMS Connect can join any number of groups.

ID=

The data store name, which:

- Consists of alphanumeric character data
- Begins with an alphabetic character
- Has a length between 1 byte and 8 bytes

- Is unique within the IMS Connect configuration member

This ID must match the data store ID that is supplied by the client. For IMS TM Resource Adapter clients, this ID must match the name that is specified in the IMS Interaction Spec for IMS TM Resource Adapter. For non-IMS TM Resource Adapter clients, the ID must match the data store ID that is placed in the IMS Request Message (IRM) that is sent to IMS Connect.

Restriction: The name specified on the ID parameter cannot be the same as a name specified on the TMEMBER parameter of any IMSPLEX statement or substatement in the HWSCFGxx member of the IMS PROCLIB data set.

MAXI=
Specifies the OTMA input message flood control value in the IMS Connect configuration file. The valid range is from 0 to 9999. If you do not specify a value, or specify a value of 0, the OTMA default value of 5000 is used. If you specify a value between 1 and 200, the OTMA minimum value of 200 is used. If you specify a value greater than 9999, the OTMA maximum value of 9999 is used. If you specify a value less than zero or greater than 65 535, abend U3401 is issued.

MEMBER=
The XCF member name that identifies IMS Connect in the XCF group specified by the GROUP parameter. This name is the XCF name that IMS uses to communicate with IMS Connect in that XCF group. This XCF member name for IMS Connect must be unique in the data store definitions for all data stores that are members of the same XCF group.

OAAV=
A decimal integer that defines the OTMA accessor environment element (ACEE) aging value, in seconds, for the data store that is specified by the ID keyword. When the OTMA ACEE aging value is reached, OTMA refreshes the ACEE before it processes the next input message received from IMS Connect. Valid values are from 0 to 2147483647. The default is 2147483647 seconds. If you specify 0, OTMA uses the default value. If you specify a value from 1 to 300, OTMA uses a value of 300 seconds. OTMA requires a value of at least 300 to enable ACEE refreshes.

RRNAME=
The name of an alternate destination specified in a client reroute request. If this string is not provided, IMS Connect uses HWS$DEF as the default name. The string is terminated by any blank or invalid character. The reroute name is truncated at any invalid character. This value must be a string of 1- to 8-uppercase alphanumeric (A - Z, 0 - 9) or special characters (@, #, $), left-aligned, and padded with blanks. IMS Connect translates lowercase characters to uppercase characters.

SMEMBER=
1 - 4 alphanumeric character field that specifies the name of the OTMA super member to which this DATASTORE belongs. If specified on the DATASTORE statement, this value overrides the value of the SMEMBER parameter specified on the HWS statement. To disable the value of SMEMBER= specified on DATASTORE, use SMEMBER=####.

TMEMBER=
The XCF member name for IMS that IMS Connect uses in order to communicate with an IMS in its XCF group. This target member name must match the member name IMS uses when it joins the XCF group. The XCF

member name for IMS is specified in the IMS startup JCL (for example, "...,OTMA=Y,GRNAME=&GROUP, ***OTMANM***=&***TMEMBER***,...").

Related reference:

Format of user portion of IRM for HWSSMPL0, HWSSMPL1, and user-written message exit routines (Communications and Connections)

IMS Connect sample OTMA User Data Formatting exit routine (HWSYDRU0) (Exit Routines)

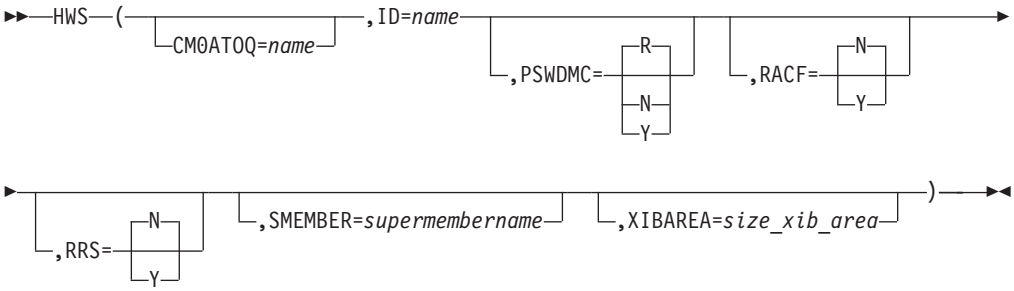
OTMA User Data Formatting exit routine (DFSYDRU0) (Exit Routines)

HWS statement

The HWS statement defines characteristics specific to an instance of IMS Connect.

Specify only one HWS statement for an IMS Connect instance.

HWS statement syntax



HWS statement parameters

The HWS statement includes the following keyword parameters:

CM0ATOQ=
1- to 8-character name of the OTMA CM0 ACK timeout queues. The value specified here overrides the OTMA default value of DFS$$TOQ and is used for all data stores except those that have their own CM0ATOQ value (that is, if you specify a value for CM0ATOQ on the DATASTORE statement, it overrides a value for CM0ATOQ specified on the HWS statement).

ID=
The IMS Connect name, which:

- Consists of alphanumeric character data
- Begins with an alphabetic character
- Has a length between 1 and 8 characters

PSWDMC=
Specifies whether IMS Connect supports mixed-case passwords. Before you enable support for mixed-case passwords, you must configure RACF to support mixed-case passwords.

R IMS Connect uses whatever is defined for mixed-case passwords in RACF. If mixed-case passwords are active in RACF (which is done through the SETROPTS command) then IMS uses it. If mixed-case passwords are not

active in RACF, then IMS Connect uses uppercase passwords. IMS Connect adjusts whenever there are changes to the mixed-case password definition in RACF. R is the default.

N Disables IMS Connect support for mixed-case passwords. IMS Connect converts any lowercase characters in passwords to uppercase characters.

Y Enables IMS Connect support for mixed-case passwords.

RACF=
Enables security support by RACF or a similar security product. When RACF=Y, IMS Connect issues a RACROUTE REQUEST=VERIFY command to authenticate the user associated with incoming messages. RACF=N is the default.

For messages from a client application, IMS Connect can use passwords and user IDs for authentication with RACF or RACF PassTickets. The user IDs, passwords, or PassTicket application names can be provided by either the client application or a user exit routine.

The RACF= setting can also be changed using the IMS Connect `SETRACF` command.

RRS=
Specifies whether z/OS Resource Recovery Services (RRS) communication is to be enabled or disabled. Set RRS= to yes (Y) or no (N). No is the default.

SMEMBER=
1- to 4-character field that specifies the name of the OTMA super member to which this instance of IMS Connect belongs. The super member can also be specified on the DATASTORE statement. If a value is specified on both the HWS statement and the DATASTORE statement, the value on the DATASTORE statement takes precedence.

XIBAREA=
Specifies the number of fullwords allocated for the XIB user area. Both the user initialization exit routine and the user message exit routines can access and modify the XIB user area. The default value is 20; the maximum value is 500. If you do not specify a value for this parameter, or you specify a value outside of the 20 to 500 range, the system uses the default value of 20.

IMSPLEX statement

The optional IMSPLEX statement registers IMS Connect as a member of an IMSplex and enables both IMS type-2 command support for IMS Connect and communications between IMS Connect and other members of the IMSplex.

An IMSplex is built on the IMS Common Service Layer (CSL). Communications in an IMSplex are managed by the CSL Structured Call Interface (SCI). IMS type-2 commands are processed by the CSL Operations Manager (OM). After an IMS Connect instance registers with SCI in an IMSplex, the IMS Connect instance supports IMS type-2 commands and can communicate with other members of the IMSplex.

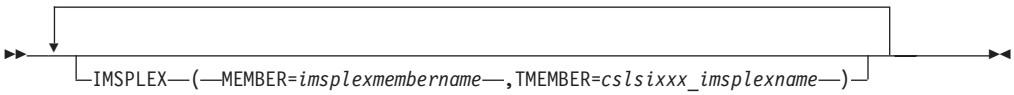
IMS Connect must be registered with SCI in the IMSplex to provide the following support:

- TCP/IP access to IMS databases through the CSL Open Database Manager (ODBM)
- IMS type-2 command support

You can join an IMS Connect instance to more than one IMSplex by specifying multiple IMSPLEX statements.

Note: An IMS Connect instance can register only a single name in an IMSplex. If the IMSPLEX statement and the IMSPLEX parameter on another configuration statement in the HWSCFGxx member of the IMS PROCLIB data set both specify the same IMSplex name on TMEMBER parameter, they must also specify the same IMS Connect name on the MEMBER parameter.

IMSPLEX statement syntax



IMSPLEX statement parameters

MEMBER=
A 1- to 8-character alphanumeric name that identifies IMS Connect in the IMSplex that is specified on the TMEMBER parameter of this IMSPLEX statement. The name must start with an alphabetic character.IMS Connect registers this name with SCI. SCI uses the name to manage communications between IMS Connect and the other members of the IMSplex.

TMEMBER=
The name of the IMSplex that IMS Connect is joining, as specified on the IMSPLEX(NAME=) statement of the CSLSIxxx member of the IMS PROCLIB data set of the SCI instance that is managing communications between IMS Connect and the IMSplex.

ODACCESS statement

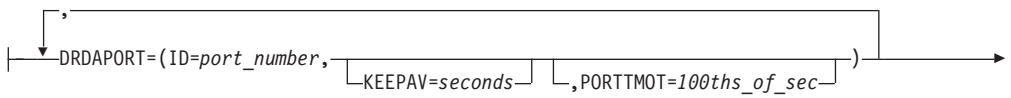
The ODACCESS statement defines connection attributes for IMS DB communications.

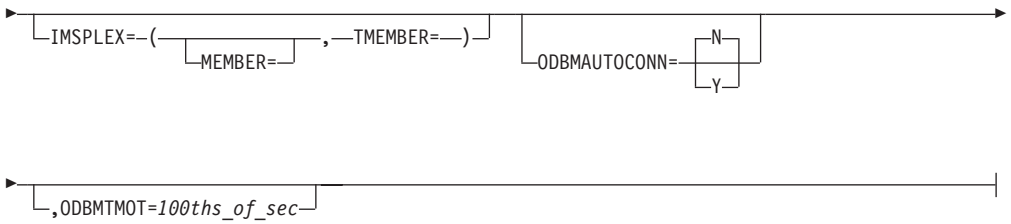
On the client side, the ODACCESS statement defines connection attributes between IMS Connect and the IMS Universal drivers and other DRDA clients. On the server side, the ODACCESS statement defines attributes for connections between IMS Connect and the CSL Open Database Manager (ODBM). IMS Connect must register with ODBM to enable access to IMS DB for clients that use the IMS Open Database architecture.

Specify only one ODACCESS statement.

ODACCESS statement syntax

B:





ODACCESS statement parameters

DRDAPORT=

Defines the port numbers, the TCP/IP keep alive value, and the timeout values for the ports that IMS Connect uses to provide access to IMS for client applications of the Open Database APIs and user-written DRDA client applications.

You can define up to 50 TCP/IP ports for an instance of IMS Connect. The total combined number of ports defined on all parameters in an IMS Connect configuration member cannot exceed 50 ports.

If you use SSL with DRDA ports, implement SSL using the z/OS Application Transparent Transport Layer Security (AT-TLS), which performs SSL processing at the z/OS TCP/IP layer.

Subparameters of DRDAPORT:

ID=

The port number of the port being defined by the DRDAPORT parameter. Port numbers are specified by a 1- to 5-character numeric value. Valid port numbers are from 1 to 65535. Port numbers must be unique among all ports used by IMS Connect within the TCP/IP domain, including the basic TCP/IP ports specified on the PORT or PORTID parameter and any Secured Socket Layer (SSL) port specified on the SSLPORT parameter.

KEEPAV=

A 1- to 8-character decimal field that sets the interval after which the keep alive mechanism of the z/OS TCP/IP layer sends a packet on idle connections on this port to maintain the connections. TCP/IP accepts a range of 1 seconds to 2,147,460 seconds. Specify 0 to use the Keepalive value defined in the z/OS TCP/IP stack. The default for KEEPAV is 0.

PORTTMOT=

Defines the amount of time that IMS Connect waits for the next input message from a client application that is connected on a DRDA port before IMS Connect disconnects the client.

The timeout interval is specified as a decimal integer in hundredths of a second. Valid values for PORTTMOT are from 0 to 2,147,483,647 (X'7FFFFFFF'). The default is 6,000 (1 minute). A 0 disables the timeout function.

Specifying a timeout value can avoid hang conditions when a client stops sending messages as expected due to, for example:

- A looping client application
- A terminated client application

For client connections, ODBMTMOT differs from PORTTMOT in that ODBMTMOT applies only to the first input message after a socket connection is established and PORTTMOT applies only to input messages that follow a previous input message.

The following code provides an example of DRDAPORT parameter specifications:

```
DRDAPORT=(ID=1111,KEEPAV=5,PORTTMOT=50),
DRDAPORT=(ID=2222,KEEPAV=10,PORTTMOT=500)
```

IMSPLEX= ()

Enables IMSplex communications between IMS Connect and ODBM. IMSplex communications are managed by the Structured Call Interface (SCI) component of the IMS Common Service Layer (CSL).

If the IMSPLEX statement in this member of the IMS PROCLIB data set already joins IMS Connect to the IMSplex in which ODBM is a member, then this IMSPLEX substatement is optional; however, at least one IMSPLEX statement must be specified in the HWSCFGxx member of the IMS PROCLIB data set to enable IMSplex communications between this IMS Connect instance and ODBM.

If an IMSPLEX substatement is not included on the ODACCESS statement, and more than one IMSPLEX statement is specified elsewhere in this HWSCFGxx member of the IMS PROCLIB data set, IMS Connect uses the first IMSPLEX statement in the HWSCFGxx member of the IMS PROCLIB data set for communications between IMS Connect and ODBM.

The IMSPLEX substatement keyword parameters include:

MEMBER=

A 1- to 8-character name that identifies IMS Connect within the IMSplex. SCI uses this name when managing communications between IMS Connect and ODBM. The name must start with an alphabetic character.

TMEMBER=

The name of the IMSplex that manages communications between IMS Connect and ODBM. This name must match the IMSplex name that is defined in the NAME parameter of the IMSPLEX keyword of the SCI initialization member of the IMS PROCLIB data set (CSLSIxxx).

Restriction: A name specified on the TMEMBER parameter cannot be the same as a name specified on the ID parameter of a DATASTORE statement.

ODBMAUTOCONN=

Specifies whether IMS Connect automatically connects to new and existing instances of ODBM within an IMSplex.

When ODBMAUTOCONN=N is specified (or left blank), IMS Connect does not automatically register with instances of ODBM during IMS Connect initialization or when new instances of ODBM enter the IMSplex while IMS Connect is already running.

When ODBMAUTOCONN=Y is specified, IMS Connect automatically registers with all instances of ODBM currently in the IMSplex when IMS Connect initializes and with any instances of ODBM that join the IMSplex while IMS Connect is already running.

You can enable or disable automatic connection with ODBM by using the IMS Connect command SETOAUTO.

If ODBMAUTOCONN=N, you can start a connection with ODBM by using the IMS Connect command STARTOD.

ODBMTMOT=

Defines the amount of time that IMS Connect waits for both:

- A response message on connections with ODBM.
- An initial input message after a socket connection is established on connections with a client application

The timeout interval is specified as a decimal integer in hundredths of a second. Valid values for ODBMTMOT are from 0 to 2 147 483 647 (X'7FFFFFFF'). The default value is 18 000 (3 minutes). A value of 0 disables the timeout function.

For connections with ODBM, specifying a timeout value can avoid hang conditions when an ODBM instance stops responding.

For connections to a client application, specifying a timeout value terminates a socket connection if a client does not send data after obtaining the socket connection before the ODBMTMOT value expires.

For example, if you specify a value of 1000, if a client application establishes a socket connection with IMS Connect and then does not send an input message within 10 seconds, IMS Connect terminates the socket connection; however, if the client application establishes a socket connection and sends an input message within 10 seconds, but after IMS Connect sends the input message to ODBM, IMS Connect does not receive a response from ODBM within 10 seconds, IMS Connect returns message HWSJ2530W to the client application and retains the socket connection.

For client connections, ODBMTMOT differs from PORTTMOT in that ODBMTMOT applies only to the first input message after a socket connection is established and PORTTMOT applies only to input messages that follow a previous input message.

Related tasks:

"Configuring IMS support for the IMS Universal drivers" on page 32

"Setting up AT-TLS SSL for IMS Connect" on page 213

RUNOPTS statement

The RUNOPTS statement defines the IBM Language Environment for z/OS runtime options to be used to override the IMS Connect default runtime options in support of SSL.

RUNOPTS statement syntax

RUNOPTS statement parameters

The RUNOPTS statement consists of only one parameter: RUNOPTS=.

RUNOPTS=

A 1- to 255-character string field that specifies the Language Environment for z/OS (LE) runtime options to be used to override the IMS Connect default runtime options in support of SSL. This parameter is optional and applies only to the LE environment for SSL support.

If they are not overridden by the RUNOPTS statement, IMS Connect uses the following default runtime options:

```
POSIX(ON),ALL31(ON),ANYHEAP(1M,1M,ANY,KEEP),
BELOWHEAP(256K,512K,FREE),HEAP(1M,1M,ANY,FREE),
LIBSTACK(9K,9K,FREE),STACK(256K,128K,ANY,FREE),
ENVAR("_CEE_ENVFILE=/SYSTEM/etc/profile"),
TRAP(ON,NOSPIE),TERMTHDACT(UAIMM)
```

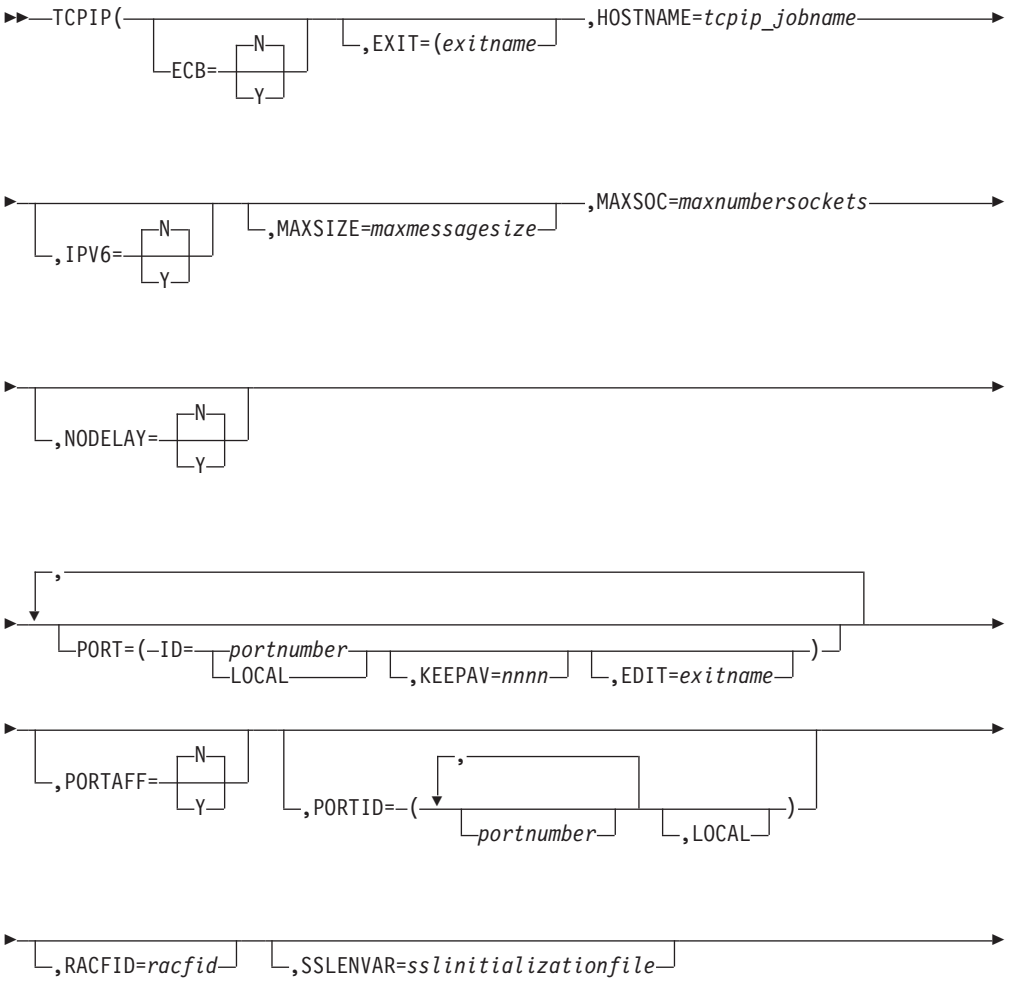
For more information about the LE runtime options, see *z/OS Language Environment Programming Reference*.

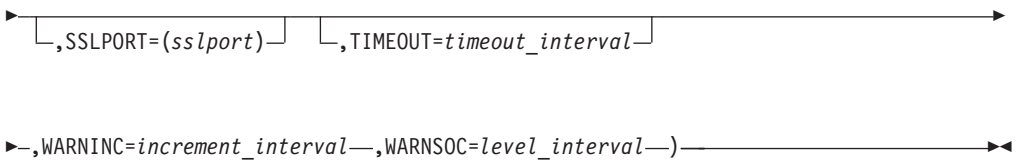
TCPIP statement

The TCPIP statement defines characteristics of the communication between TCP/IP and IMS Connect.

Specify only one TCPIP statement.

TCPIP statement syntax





TCPIP statement parameters

The TCPIP statement keyword parameters are as follows:

ECB=
Specifies whether TCP/IP exit or ECB (Event Control Block) processing is to be used. ECB processing enhances IMS Connect performance by increasing throughput. Set ECB to yes (Y) or no (N).

Y When ECB=Y is specified, IMS Connect executes with TCP/IP driving IMS Connect with the posting of an ECB.

N When ECB= N is specified (or left blank), IMS Connect executes with TCP/IP driving an IMS Connect exit. No is the default.

EXIT=
Specifies the 1- to 8-alphanumeric character names of one or more IMS Connect user message exit routines that receive control when messages are received from and sent to TCP/IP clients. For example, EXIT=(EZAEXIT,EZBEXIT,EZCEXIT,HWSCSLO0,HWSCSLO1). You can specify a maximum of 254 user message exit routines.

The exit routines specified on the EXIT parameter support OTMA linkage through IMS Connect to IMS, as well as OM command clients for IMSplexes. To use OM command clients, such as IBM Management Console for IMS and DB2 for z/OS, with IMS Connect, you must specify HWSCSLO0 and HWSCSLO1 on the EXIT parameter.

Do not specify either HWSJAVA0 or HWSUINIT in the EXIT= parameter.

The HWSJAVA0 exit routine is automatically loaded by IMS Connect after you install it into the ADFSLOAD library. The HWSJAVA0 exit routine supports the IMS TM resource adapter (previously know as IMS Connector for Java). The sample source for HWSJAVA0 is shipped with IMS in the SDFSSRC library.

The HWSUINIT exit routine causes IMS Connect to abend if it is specified on the EXIT parameter. The HWSUINIT exit routine is called only during IMS Connect startup and termination and is not a user message exit routine.

HOSTNAME=
A 1- to 8-alphanumeric character field set to the TCP/IP JOBNAME.

IPV6=
At IMS Connect startup time, determines whether Internet Protocol Version 6 (IPv6) is enabled. Set this parameter to yes (Y) or no (N). If you leave this field blank, the default value is used.

Y IPv6 is used.

N IPv4 is used. This is the default.

MAXSIZE=
Specifies the maximum message size that is allowed in the 4-byte length field

that precedes the IMS request message (IRM). Use the MAXSIZE= keyword to override the internal default of 10 000 000 bytes.

MAXSOC=

A decimal value between 50 and 65 535 that sets the maximum total number of sockets that this instance of IMS Connect can open. The maximum number of physical connections that can be made is the MAXSOC= value less the number of ports, because IMS Connect uses one socket on each port for listening. For example, if you specify MAXSOC=80 and have five ports, 75 physical connections can be made. The default value is 50.

When IMS Connect starts, each regular port has a socket count of 1, which reflects the port listen socket. The SSL port has a socket count of 2, which reflects one port listen socket and one SSL-related file descriptor. When the SSL port receives a connection for the first time, one additional SSL-related file descriptor is created and is counted towards the socket count.

When the number of sockets reaches the MAXSOC limit, IMS Connect refuses any new connections and issues message HWSS0771W. After the number of connections falls below the MAXSOC value, IMS Connect resumes accepting connections.

Important: The MAXSOC parameter is related to the z/OS UNIX System Services parameter MAXFILEPROC. The values of MAXSOC and MAXFILEPROC must be compatible. If the values of each parameter are not compatible, IMS Connect cannot open any ports. The values are compatible when the value for MAXFILEPROC is equal to or greater than the value of MAXSOC.

You can ensure compatibility between MAXSOC and MAXFILEPROC by granting IMS Connect UNIX System Services *superuser* privileges, which allows IMS Connect to change the value of the MAXFILEPROC parameter automatically. You can grant UNIX System Services superuser privileges to IMS Connect by using the RACF command `ALTERUSER` to assign an OMVS segment with a UID of 0 to the user ID of the IMS Connect started task. Alternatively, your UNIX System Services administrator can adjust the value of MAXFILEPROC directly in the BPXPRMxx member of the z/OS SYS1.PARMLIB data set.

If IMS Connect does not have superuser privileges, and the MAXSOC value is greater than the MAXFILEPROC value, IMS Connect issues the message "HWSP1415E TCP/IP SOCKET FUNCTION CALL FAILED; F=SETRLIMI, R=-1, E=139, M=SDOT" and does not open any ports.

You can check the value of MAXFILEPROC for IMS Connect by issuing the UNIX command `D OMVS,L,PID=`, where PID is the process ID for IMS Connect. You can determine the PID for IMS Connect by issuing the UNIX command: `D OMVS,V`.

Both MAXSOC and MAXFILEPROC affect the number of sockets that IMS Connect can open, but with an important difference: the MAXFILEPROC parameter limits the number of sockets for each port, whereas MAXSOC limits the total number of sockets for IMS Connect. For example, if the value of both parameters is 100, and IMS Connect has two ports, the MAXSOC limit is reached if there are 55 sockets on one port and 45 on the other. The MAXFILEPROC limit is reached only if the number of sockets on one of the ports reaches 100.

As the number of sockets on an IMS Connect port approaches the MAXFILEPROC value, UNIX System Services issues message BPXI040I. For example, "BPXI040I PROCESS LIMIT MAXFILEPROC HAS REACHED 85% OF ITS CURRENT 404". The BPXI040I message is displayed by UNIX System Services only if LIMMSG is set to SYSTEM or ALL in SYS1.PARMLIB(BPXPRMxx) or using the SETOMVS command.

When the MAXFILEPROC value is reached, IMS Connect issues the following messages:

- HWSP1415E TCP/IP SOCKET FUNCTION CALL FAILED; F=ACCEPT4 , R=-1, E=124, M=SDCO
- HWSS0771W LISTENING ON PORT=*portid* FAILED; R=*rc*, S=*sc*, M=*mc*

Note that when the MAXSOC limit is reached, IMS Connect issues only the HWSS0771W message. When the MAXFILEPROC limit is reached, IMS Connect issues both the HWSP1415E and the HWSS0771W messages.

NODELAY=
Specifies whether (Y) or not (N) the TCP/IP protocol option TCP_NODELAY is to be used. The default is no.

If you specify NODELAY=Y, IMS Connect sets the TCP_NODELAY option to the IPPROTO_TCP level for all UNIX System Services API interface calls, which disables the Nagle algorithm. This parameter is valid only when sending packets.

Using the NODELAY option can enhance IMS Connect performance and increase throughput because it forces a socket to send the data in its buffer without having to wait for an ACK from the client's TCP/IP. However, using the NODELAY option can cause network traffic to increase.

PORT=
Specifies a TCP/IP port or enables local option connections.

You can define up to 50 TCP/IP ports for an instance of IMS Connect. The total combined number of ports defined on all parameters in an IMS Connect configuration member cannot exceed 50 ports.

ID=
A 1- to 5-character field that defines a TCP/IP port. Valid values are a decimal number from 1 to 65535 or the alphabetic value LOCAL, which enables the local option connection.

Port numbers must be unique for a specific instance of IMS Connect and must not conflict with other ports selected in the TCP/IP domain.

KEEPAV=
Specifies a 1- to 8-character decimal field of the number of seconds for the TCP/IP KeepAlive interval for sockets on this port. TCP/IP accepts a range from 1 to 2 147 460 seconds. If you specify zero, the KeepAlive interval value is bypassed and the setting for the TCP/IP stack is used (this is the default).

The TCP/IP KeepAlive function enables you to detect error situations on inactive sockets. KEEPAV enables you to override the default TCP/IP KeepAlive interval value. This interval can be specified for each TCP/IP port that IMS Connect uses, and it is set for all the sockets for that port.

EDIT=
A 1- to 8-character name of a Port Message Edit exit routine that can modify messages that do not conform to IMS Connect's standard message

formats. The exit routine must be accessible to IMS Connect by either JOBLIB, STEPLIB, or LinkList. This field is optional. The default is no exit.

PORTAFF=

Specifies that commit-then-send (CM0) output messages that IMS sends to this IMS Connect have affinity to the port on which IMS Connect received the original input message. When PORTAFF=Y, IMS returns all CM0 output for this IMS Connect to the same port on which it received the original input message. IMS Connect identifies the port by using the port ID number as specified in the PORTID parameter of the TCP/IP configuration file. When PORTAFF=N, IMS Connect attempts to return the CM0 output to the first port on which it finds the client ID of this IMS Connect. PORTAFF=N is the default.

PORTID=

A 1- to 5-character field that defines the TCP/IP port. Valid values are decimal numbers from 1 to 65535 and the alphabetic value LOCAL, which enables the local option connection. Port numbers must be unique for a specific instance of IMS Connect, and must not conflict with other ports selected in the TCP/IP domain.

For TCP/IP port communications, specify the port number or numbers that bind to the socket. Port numbers must not conflict with other ports selected in the TCP/IP domain.

You can use either the PORTID keyword or the PORT keyword of the TCP/IP statement to define a TCP/IP port. Use the PORT keyword to specify a TCP/IP KeepAlive value or a Port Message Edit exit routine for a port.

You can define up to 50 TCP/IP ports for an instance of IMS Connect. The total combined number of ports defined on all parameters in an IMS Connect configuration member cannot exceed 50 ports.

Some PORTID configuration examples include:

```
PORTID=(9999,8888,7777)
PORTID=(LOCAL)
PORTID=(6666,5555,4444,3333,LOCAL)
```

RACFID=

A 1- to 8-alphanumeric character field set to the default RACF ID for exits to pass to OTMA for security checking if the RACF ID has not explicitly been set in the incoming message or by the user exit.

SSLENVAR=

The member name of the SSL initialization file.

SSLPORT=

A 1- to 5-numeric character decimal field to define a Secure Sockets Layer (SSL) port. Valid values are from 1 to 65535.

For SSL port communication, specify the port number that binds to the socket. This port must not conflict with any other ports selected in the TCP/IP domain or those selected under the PORT or PORTID parameter as basic TCP/IP ports.

You can define only one SSL port. If you define more than one SSL port IMS Connect abends during startup. If you want to define and use more than one SSL port for IMS Connect, you can define additional SSL ports using the z/OS 1.7 Communications Server. IMS considers these ports as additional PORTIDs on the IMS Connect TPCIP statement. For more information, see *z/OS Communications Server: IP Configuration Reference*.

You can define up to 50 TCP/IP ports for an instance of IMS Connect. The total combined number of ports defined on all parameters in an IMS Connect configuration member cannot exceed 50 ports.

An example of an SSLPORT configuration is:

```
SSLPORT=(7776)
```

TIMEOUT=

A decimal integer field to disconnect the client. The timeout interval is in hundredths of seconds. The maximum value of timeout is 2147483647 (X'7FFFFFFF') and the default is 0 (which means no timeout). The range is from 0 to 2147483647.

IMS Connect uses the timeout value to determine the amount of time to wait for a response from IMS that is being sent to the client. This timeout value is used to prevent the client from appearing to be "hung." A hang condition occurs when the IMS host application is not responding, because either:

- The IMS program for this transaction code is stopped
- The dependent region that would run the transaction is not active
- The IMS host application is looping

The client sets a second timeout value in the IRM (IMS Request Message) header field IRM_TIMER for use with a READ to OTMA following a RESUME TPIPE call (see *IMS Version 11 Communications and Connections* for more information), and the ACK following the READ(s) for a RESUME TPIPE.

This timeout value is also used to disconnect a client and not send data following a client socket connection. If the timeout value is set to 10 seconds, and the client application performs a socket connection, then the client application has 10 seconds in which to send the transaction code and data. If the socket connection is made and the client application delays for more than 10 seconds, the socket connection terminates. This timeout value on an IMS Connect read of the client only applies to the wait time between the socket connection and the first input from the client application. The timeout function is not activated between reads but only between the connection and the first IMS Connect read of the client application input.

WARNINC=

Specifies a warning level incremental percentage. After the WARNSOC value is reached, every time the number of sockets increases by the percentage value specified on WARNINC, IMS Connect reissues warning message HWSS0772W. You can specify a value between 1 and 50. The default value is 5 if not specified otherwise. If a value of less than 1 is specified, IMS Connect sets WARNINC=1. If a value greater than 49 is specified, then IMS Connect sets WARNINC=49.

WARNSOC=

Specifies a warning level when the number of sockets increases to a certain percentage of the MAXSOC limit. When the number of sockets that are currently supported reaches this warning level, IMS Connect issues a warning message, HWSS0772W. For example, if MAXSOC=2000 and you set WARNSOC to 75, message HWSS0772W is issued when the number of sockets reaches 1500. You can specify a value between 50 and 99. The default value is 80 if not specified otherwise. If a value less than 50 is specified, IMS Connect sets WARNSOC=50. If a value greater than 99 is specified, IMS Connect sets WARNSOC=99.

When the number of sockets decreases to the reset percentage, IMS Connect issues a HWSS0773I message to indicate that the number of sockets is no

longer reaching the maximum sockets limit. The reset percentage is two times the WARNINC value, below the WARNSOC value, or 5 percent below the WARNSOC value, whichever is lowest.

Here is an example of what happens when using the default WARNSOC=80 and WARNINC=5 values: When the number of sockets increases to 80% of the MAXSOC value, IMS Connect issues the first HWSS0772W message. When the number of sockets increases to 85%, IMS Connect issues the second HWSS0772W message. When the number of sockets increases to 90%, IMS Connect issues the third HWSS0772W message. When the number of sockets decreases to 89% and then increases to 90% again, IMS Connect does not issue a HWSS0772W message. This prevents IMS Connect from flooding the console with messages. When the number of sockets decreases to the reset percentage of 70%, IMS Connect issues a HWSS0773I message and resets the warning limits. When the number of sockets increases to 80%, 85%, and so on again, IMS Connect issues the HWSS0772W warning messages again.

Related reference:

Format of user portion of IRM for HWSSMPL0, HWSSMPL1, and user-written message exit routines (Communications and Connections)

SSL initialization (Communications and Connections)

IMS Connect configuration examples

The following examples of IMS Connect configuration statements show how to enable various IMS Connect functions and support during IMS Connect system definition.

You can also find a sample IMS Connect configuration member, HWSCFG00, in the ADFSSMPL data set.

Important: In all the example configurations shown, you can continue parameters beyond an 80-column line by using any combination of the following techniques:

- Inserting a comma followed by three blanks, then continuing the parameter on the next line. An example of this technique is shown in the `IMS Connect Example 1 Configuration File`.
- Using all 80 columns of a line, then continuing in the next statement. You do not need to use a continuation indicator (such as an "x" in column 72).

Example of a simple IMS Connect configuration for IMS TM support

The following figure is an example of a simple IMS Connect system configuration for IMS TM support.

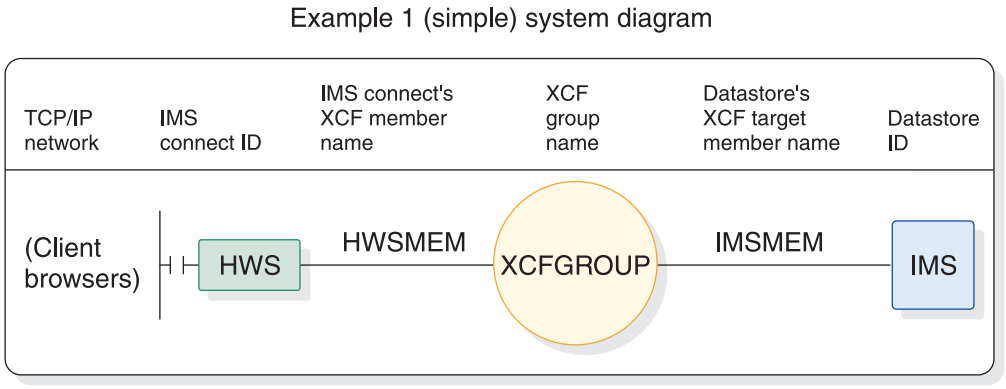


Figure 43. Simple IMS Connect system configuration for IMS TM support

In the following example of an IMS Connect configuration member:

- The IMS Connect ID is defined as `HWS`. This IMS Connect is configured to include the ports defined for TCP/IP communications and the IMS OTMA group and member names for communication with IMS.
- The TCP/IP configuration defines the HOSTNAME as `MVSTCPIP`, the RACFID as `RACFID`, the PORTID as `9999`, and the EXIT as `HWSSMPL0`.
- The data store configuration defines the ID as `IMS`, the GROUP as `XCFGROUP`, the MEMBER as `HWSMEM`, and the TMEMBER as `IMSMEM`.

```
************************************************
* IMS Connect example configuration file
************************************************
HWS (ID=HWS,RACF=N,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,PORTID=(9999),MAXSOC=2000,TIMEOUT=8888,
EXIT=(HWSSMPL0))
DATASTORE  (ID=IMS,GROUP=XCFGROUP,MEMBER=HWSMEM,TMEMBER=IMSMEM,DRU=HWSYDRU0)
```

Example of a complex IMS Connect configuration for IMS TM support

The following figure shows an example of a more complex IMS Connect system configuration for TM support.

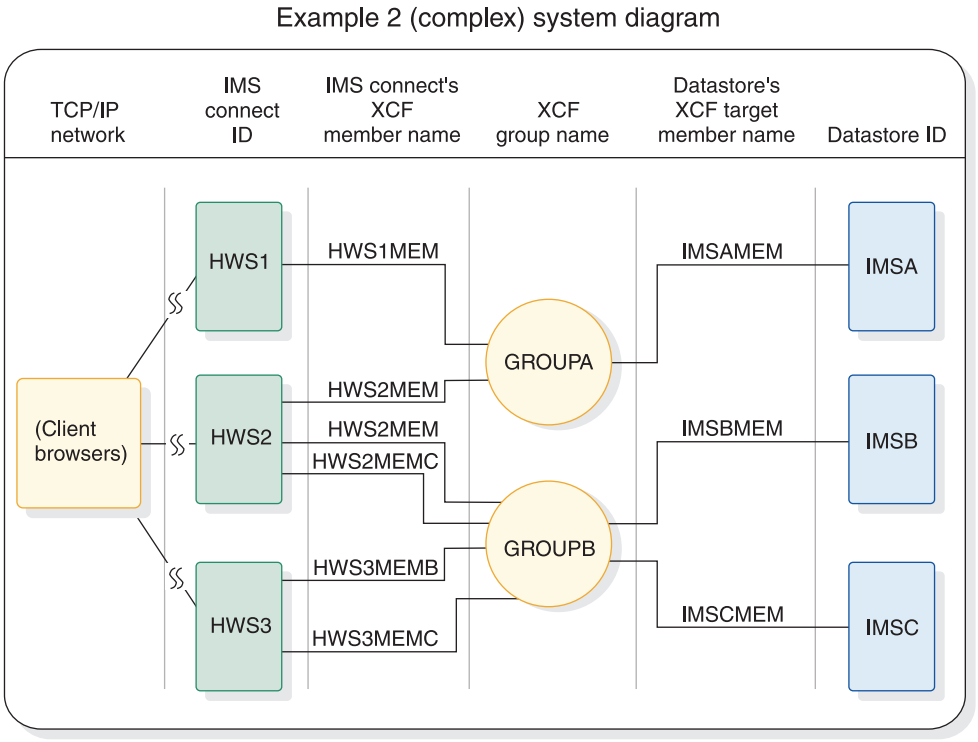


Figure 44. Complex IMS Connect system configuration for IMS TM support

In following example of an IMS Connect configuration member:

- Three IMS Connects are configured. Each IMS Connect has its own configuration member.
- Each IMS Connect uses a different port number for TCP/IP communications and can belong to multiple z/OS cross-system coupling facility (XCF) groups.
- One or more IMS systems can belong to each XCF group.
- When defining multiple data stores that belong to the same XCF group in a single IMS Connect configuration member, the XCF member name for that IMS Connect must be unique in each DATASTORE statement. However, if the data stores are members of different XCF groups, the XCF member names can be the same for different data stores within a single IMS Connect configuration member.

For example, observe that the XCF member name for IMS Connect in the IMSA and IMSB DATASTORE statements in the HWS2 configuration member in the configuration example, HWSMEM2, is the same for both DATASTORE statements. The IMSA and IMSB data stores are members of different XCF groups—GROUPA and GROUPB—so the XCF member names can be identical. These member names could have been made unique, for example, HWS2MEMA and HWS2MEMB, but it is not necessary to do so. However, the XCF member names for IMS Connect in the IMSB and IMSC DATASTORE statements in the HWS2 configuration member are different because the IMSB and IMSC data stores are members of the same XCF group, GROUPB.

```
************************************************
* IMS Connect example configuration member for HWS1
************************************************
HWS (ID=HWS1,RACF=N,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,PORTID=(9999),MAXSOC=2000,TIMEOUT=8888,
EXIT=(HWSSMPL0))
DATASTORE (ID=IMSA,GROUP=GROUPA,MEMBER=HWS1MEM,TMEMBER=IMSAMEM,DRU=HWSYDRU0)
```

```
************************************************

************************************************
* IMS Connect example configuration member for HWS2
************************************************
HWS (ID=HWS2,RACF=N,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,PORTID=(9998),MAXSOC=2000,TIMEOUT=8888,
EXIT=(HWSSMPL0))
DATASTORE (ID=IMSA,GROUP=GROUPA,MEMBER=HWS2MEM,TMEMBER=IMSAMEM,DRU=HWSYDRU0)
DATASTORE (ID=IMSB,GROUP=GROUPB,MEMBER=HWS2MEM,TMEMBER=IMSBMEM,DRU=HWSYDRU0)
DATASTORE (ID=IMSC,GROUP=GROUPB,MEMBER=HWS2MEMC,TMEMBER=IMSCMEM,DRU=HWSYDRU0)
************************************************

************************************************
* IMS Connect example configuration member for HWS3
************************************************
HWS (ID=HWS3,RACF=Y,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,PORTID=(9997),MAXSOC=2000,TIMEOUT=8888,
EXIT=(HWSSMPL0))
DATASTORE (ID=IMSB,GROUP=GROUPB,MEMBER=HWS3MEMB,TMEMBER=IMSBMEM,DRU=HWSYDRU0)
DATASTORE (ID=IMSC,GROUP=GROUPB,MEMBER=HWS3MEMC,TMEMBER=IMSCMEM,DRU=HWSYDRU0)
************************************************
```

IMS Connect configuration statement for IMS DB support

The following example shows a simple configuration member for an IMS Connect instance that supports clients that connect to IMS DB in a DBCTL system.

When configuring IMS Connect to support only a DBCTL system, a DATASTORE statement is not required. Instead, the ODACCESS statement defines the connection to IMS via the Open Database Manager (ODBM) of the Common Service Layer (CSL). For IMS Connect to communicate with ODBM, you must code an IMSPLEX substatement in the IMS Connect configuration member. You do not identify ODBM or IMS in the configuration member. The communication and connection between IMS Connect and ODBM is managed automatically by IMS Connect and ODBM through the Structured Call Interface (SCI) component of CSL.

When ODBMAUTOCONN=Y is specified, as shown in the example, IMS Connect automatically connects to all active ODBM instances in the IMSplex that is specified on the TMEMBER parameter of the IMSPLEX substatement. Communications with the IMS systems that are defined to the ODBM instances is also automatically enabled.

If ODBMAUTOCONN=N, IMS Connect does not connect to ODBM during startup. Communication can then be enabled for ODBM instances and IMS systems individually by issuing the STARTIA command.

Although the values of ID on the HWS statement and MEMBER on the IMSPLEX substatement are the same in the example, the values do not need to match.

```
************************************************
* IMS Connect example for IMS Universal drivers and DRDA client support
************************************************
HWS (ID=HWS1,RACF=N,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,MAXSOC=2000)
ODACCESS (DRDAPORT=(ID=1111,KEEPAV=5,PORTTMOT=50),
IMSPLEX=(MEMBER=HWS1,TMEMBER=PLEX1),
ODBMAUTOCONN=Y)
************************************************
```

The following figure illustrates the configuration defined in the preceding example.

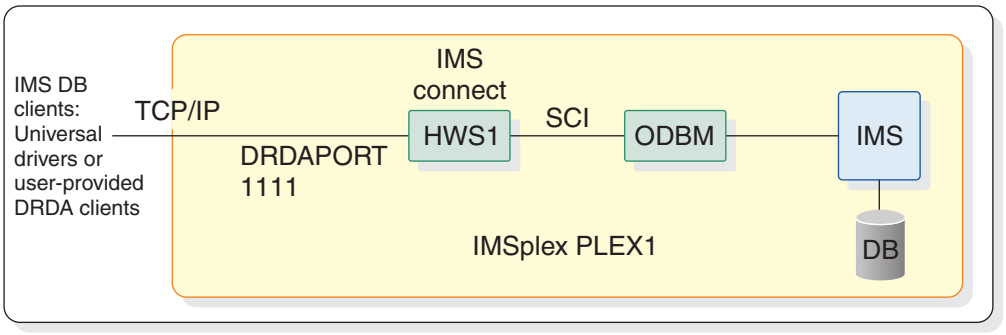


Figure 45. Simple IMS Connect system configuration for IMS DB support

Additional IMS Connect configuration statement examples

The following series of IMS Connect configuration statements are examples of how to configure IMS Connect to support various functions, including:

- OM command clients, such as IBM Management Console for IMS and DB2 for z/OS
- Internet Protocol version 6 (IPV6)
- RACF PassTicket
- Secure Sockets Layer (SSL)
- XML conversion with the XML adapter
- IMS Universal drivers and DRDA clients

```
************************************************
* IMS Connect example of including the support for Control Center
************************************************
HWS (ID=HWS4,RACF=Y,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,PORTID=(9999,LOCAL),MAXSOC=2000,
TIMEOUT=8800,
EXIT=(HWSCSLO0,HWSCSLO1,HWSSMPL1))
IMSPLEX (MEMBER=HWS4,TMEMBER=PLEX1)
************************************************

************************************************
* IMS Connect example of including the support for Control Center and IPV6
************************************************
HWS (ID=HWS4,RACF=Y,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,PORTID=(9999),MAXSOC=2000,TIMEOUT=8800,
EXIT=(HWSCSLO0,HWSCSLO1,HWSSMPL1),IPV6=Y)
IMSPLEX (MEMBER=HWS4,TMEMBER=PLEX1)
************************************************

************************************************
* IMS Connect example of including the APPL name for passticket support
************************************************
HWS (ID=HWS5,RACF=Y,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,PORTID=(9999),MAXSOC=2000,TIMEOUT=8800,
EXIT=(HWSSMPL0)
DATASTORE (ID=IMS,GROUP=XCFGROUP,MEMBER=HWSMEM,TMEMBER=IMSMEM,DRU=HWSYDRU0,
APPL=APPLID1)
************************************************

************************************************
* IMS Connect example of including the support for SSL
************************************************
HWS (ID=HWSG7,RACF=N,XIBAREA=20))
TCPIP (HOSTNAME=TCIPI,PORTID=(9998),SSLPORT=(9999),SSLENVAR=SSLENVAR,
EXIT=(HWSSMPLO))
DATASTORE (ID=SOCKEYE,MEMBER=COHO,TMEMBER=CHINOOK,GROUP=SALMON)
```

```
************************************************

************************************************
* IMS Connect example of including XML adapter support
************************************************
HWS (ID=HWS8,RACF=Y,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,
PORTID=(9999,LOCAL,MAXSOC=2000,TIMEOUT=8800,
EXIT=(HWSSMPL1,HWSSOAP1))
ADAPTER (XML=Y,MAXLSSSZ=30000)
************************************************
```

Related tasks:

"Configuring XML conversion support for IMS Connect clients" on page 215

"Configuring IMS Connect" on page 210

Related reference:

Format of user portion of IRM for HWSSMPL0, HWSSMPL1, and user-written message exit routines (Communications and Connections)

Chapter 18. Other control statements used in IMS environments

These topics include control statements that can be used in an IMS environment.

Sequential buffering control statements

Use sequential buffering control statements to specify I/O operations that require sequential buffering (SB), capture internal calls to the SB buffer handler, take a snapshot of SB control blocks, and perform an SB buffer handler self-check.

This section describes the control statements you can use to modify sequential buffering definitions.

Syntax of SB control statements

SB control statements are contained in 80-character records. Positions 72-80 are ignored.

Each SB control statement must begin on a new record. The first record must contain at least the statement name. IMS considers the first non-blank character the start of the statement. The first word on the statement must be the statement name, such as SBPARM and SBIC, followed by one or more blanks.

Depending upon the type of SB control statement, one or more keyword parameters follow the statement name. Keywords must be followed immediately by an equal (=) sign and a value. If you specify multiple keywords, separate them with commas. If you specify multiple values, enclose them in parentheses. Individual keyword parameters must be contained on a single record.

A blank or a /* signals the end of a statement. You can continue SB control statements on multiple records. Use a comma at the end of a line to indicate continuation.

An asterisk (*) in the first position on a record indicates a comment. Elsewhere on a record, comments begin with /*.

Two examples of proper syntax follow:

```
SBPARM ACTIV=COND,DB=SKILLDB,BUFSETS=6   /*Comment

SBPARM ACTIV=COND,   /*This is a comment
       DB=SKILLDB,   /*This is a comment
       BUFSETS=6     /*This is a comment
```

SB parameters (SBPARM) control statement

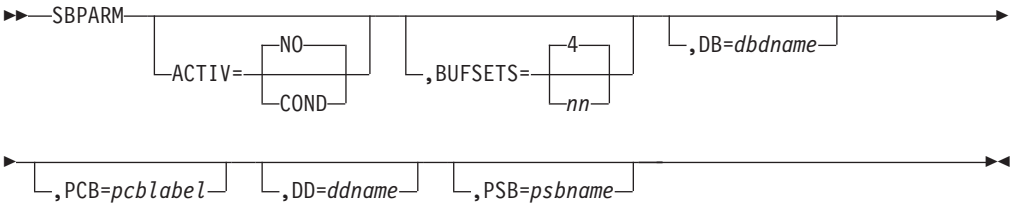
The SBPARM control statement is used to:

- Specify which I/O operations require sequential buffering (I/O operations can be specified by database, PCB, ddname, and PSB)
- Override SB default parameters

All SBPARM parameters are optional.

© Copyright IBM Corp. 1974, 2015

The format of the SBPARM control statement is:



Default values are underlined. The default value, however, might have been overridden by other specifications. If you provide the same SB option or parameter in more than one place, the following priority list applies (item 1 has the highest priority):

1. SB control statement specifications (the *n*th control statement overrides the *m*th control statement, where *n*>*m*)
2. PSB specifications
3. Defaults changed by the SB Initialization exit routine

If you specify multiple SBPARM control statements applying to the same I/O operation, the *n*th control statement overrides the *m*th one (where *n*>*m*). This lets you specify defaults on the first statement which can be overridden by subsequent statements.

The SBPARM control statement has two types of keyword parameters:

1. Option parameters, such as ACTIV= and BUFSETS=, which set and override an SB option or parameter, respectively
2. Qualification parameters, such as DB, PCB, DD, and PSB, which specify the I/O operations to which the SBPARM statement applies

Option parameters

Option parameters set or override an SB option or parameter. If an option parameter is not specified, the previously established value of the parameter is in effect. The previously established value might have been set by an IMS default, the SB Initialization exit routine, PSBGEN, or preceding SBPARM statements.

ACTIV=
Specifies if and when SB should be activated.

NO Specifies that SB is not activated. NO is the default; however, the default might have been changed by what is specified in the SB Initialization exit routine, PSBGEN, or a preceding SBPARM control statement.

COND
Specifies that IMS monitors the I/O reference patterns of PCBs and eventually activates SB if analysis shows the I/O reference pattern is sequential, and the rate of I/O activity is high enough.

Recommendation: Specify COND for batch and BMP programs that sometimes or always process a database sequentially. Do not use SB for short-running MPP, Fast Path, or IBM CICS Transaction Server for z/OS programs because of the overhead associated with initializing SB for each program execution.

If ACTIV= is not specified on an SBPARM control statement, IMS uses the previously established value of the ACTIV parameter. Therefore, omitting ACTIV lets you override the BUFSETS parameter without inadvertently changing SB PSBGEN specifications.

BUFSETS=
Specifies how many buffer sets are to be allocated for a given SB buffer pool. You can specify a value from 1 through 25. The default is 4. (The default can be changed by the SB Initialization exit routine.)

Each SB buffer pool consists of *n* buffer sets. Each buffer set has 10 buffers. Each buffer is large enough to hold one OSAM data set block.

You must specify a value greater than 1 if SB is to use overlapped I/O.

A well-organized database might only require a BUFSETS value of 2 for efficient sequential buffering. Less well-organized databases might require a BUFSETS value of 6 or more. One indicator of database organization can be found in the optional //DFSSTAT reports. If, for a sequential program, these reports show a low percentage of random I/O operations, this can indicate a well-organized database. Conversely, a high percentage of random I/O can indicate a poorly organized database. In either case, the //DFSSTAT reports can help you adjust the BUFSETS value to improve buffering performance.

Qualification parameters

Qualification parameters specify the I/O operations to which the SBPARM statement applies. You can associate SBPARM statements with a database, PCB, ddname, or PSB.

You can specify any combination of qualification parameters. If you specify more than one qualification parameter, the SBPARM statement applies only to I/O operations that satisfy all qualification parameters. If you do not specify any qualification parameters, the SBPARM statement applies to all I/O operations.

DB=
Specifies the DBD name (as specified in the PCB macro of PSBGEN) of those PCBs to which the SBPARM statement applies.

If multiple PCBs have the same DBD name, and you want different specifications for different PCBs, use the PCB= keyword. If the PCB refers to multiple database data sets and you want different specifications for these data sets, use the DD= keyword.

If no DB= keyword is provided, the SBPARM statement applies to all I/O operations that satisfy the other qualification keywords.

PCB=
Specifies the label coded on the database PCB during PSBGEN. Use this keyword to distinguish between multiple PCBs with the same DBD name.

If no PCB= keyword is provided, the SBPARM statement applies to all I/O operations that satisfy the other qualification keywords.

DD=
Specifies the ddname of the database data set to which the SBPARM statement applies. Use this keyword to distinguish between multiple data sets to which the PCB refers.

If no DD= keyword is provided, the SBPARM statement applies to all I/O operations that satisfy the other qualification keywords.

PSB=

Specifies the name of the PSB to which the SBPARM statement applies.

If no PSB=keyword is provided, the SBPARM statement applies to all I/O operations that satisfy the other qualification keywords.

Use of the PSB= keyword is recommended for message regions. Normally, SB should not be activated for all programs running in the message region; rather, SB should be restricted to a small number (if any) of these programs. If SB is to be used for more than one program, you need to provide one SBPARM statement for each program and code the PSBname of the program on the PSB= keyword.

PSB= is also useful if you decide to centralize all SB control statements in one common SB control statement file. In this case, PSB= identifies for each SBPARM statement, the PSB or program to which it applies.

SBPARM examples	Comments
`SBPARM ACTIV=COND`	SBPARM requests conditional activation of SB for all I/O operations. No default values for SB parameters are overridden.
`SBPARM ACTIV=COND,DB=SKILLDB`	SBPARM requests conditional activation of SB for I/O operations associated with all PCBs coded with DBDNAME=SKILLDB during PSBGEN. No default values for SB parameters are overridden.
`SBPARM ACTIV=COND,DB=SKILLDB,PCB=LABEL1`	SBPARM requests conditional activation of SB for all PCBs that satisfy both of the following conditions: • DBDNAME=SKILLDB coded during PSBGEN • The PCB label coded as LABEL1 during PSBGEN
`SBPARM ACTIV=COND,DB=SKILLDB,BUFSETS=6`	SBPARM requests conditional activation of SB for all PCBs coded with DBDNAME=SKILLDB during PSBGEN. The BUFSETS parameter is set to 6 (instead of 4, which is the default). This increase in BUFSETS value can improve buffering efficiency if the database is poorly organized.
`SBPARM BUFSETS=6`	SBPARM requests that the default values of BUFSETS be overridden. Increasing the value of BUFSETS to 6 (rather than using the default, which is 4) can improve buffering efficiency if the database is poorly organized. In this example, ACTIV has not been coded. As a result, the SBPARM statement does not change the previously established set of I/O operations being buffered by SB.
`SBPARM ACTIV=NO`	SBPARM requests that SB not be used. The request applies to all I/O operations.

SB Image Capture (SBIC) control statement

The SBIC control statement captures all internal calls to the SB buffer handler. The captured information is put in the X'5E' log record.

The X'5E' log records can be used as input to the SB test program (DFSSBHD0).

Related reading: For detailed information about the SB test program (DFSSBHD0), see *IMS Version 11 Database Utilities*.

The format of the SBIC control statement is as follows:

SB Compare Option (SBCO) control statement

The SBCO control statement asks the SB buffer handler to perform a self check to determine if SB is putting incorrect block images in the OSAM buffers, and if so, to provide problem determination information.

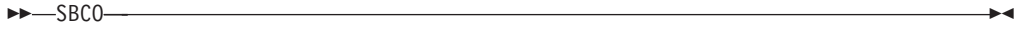
Problem determination information is provided in the form of a snap. In the IMS online environment, this snap is written to the IMS log. In the IMS batch environment, you can request that the snap is written to any of the following:

- The data set specified in the DFSDDLT0 //PRINTDD DD statement
- A user-specified data set
- The IMS log

See "Snap Destination (SNAPDEST) control statement" on page 811 for a description of how to request these snap destinations. In the batch environment, the default snap destination is the data set specified in the //DFSSNAP DD statement.

Related reading: For detailed information about the SBCO control statement, see *IMS Version 11 Diagnosis*.

The format of the SBCO control statement is as follows:



SB Snap (SBSNAP) control statement

The SBSNAP control statement snaps SB control blocks after each internal call of the OSAM buffer handler to the SB buffer handler. As an option, you can request that, for each internal SB call, the following additional information be snapped:

- The OSAM database block returned by the SB buffer handler to the OSAM buffer handler
- The SB buffers for the current data set group

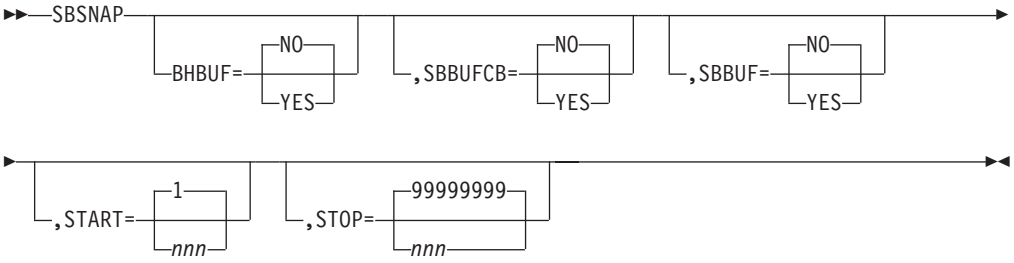
In IMS online environment, this snap is written to the IMS log. In the IMS batch environment, you can request that the snap is written to any of the following:

- The data set specified in the DFSDDLT0 //PRINTDD DD statement
- A user-specified data set
- The IMS log

See "Snap Destination (SNAPDEST) control statement" on page 811 for a description of how to request these snap destinations. In the batch environment, the default snap destination is the data set specified in the //DFSSNAP DD statement.

All SBSNAP parameters are optional.

The format of the SBSNAP control statement is as follows:



If you specify multiple SBSNAP control statements, the *n*-th control statement overrides the *m*th control statement (where *n*>*m*).

Related reading: For additional information about the SBSNAP control statement, see *IMS Version 11 Diagnosis*.

BHBUF=
Specifies whether (YES) or not (NO) the OSAM buffer handler buffer is to be included in the snap produced at the end of each internal call to the SB buffer handler. The default is NO.

SBBUFCB=
Specifies whether (YES) or not (NO) the buffer control blocks of the SB buffer handler are to be included in the snap produced at the end of each internal call to the SB buffer handler. The default is NO.

SBBUF=
Specifies whether (YES) or not (NO) the buffers of the SB buffer handler are to be included in the snap produced at the end of each internal call to the SB buffer handler. The default is NO.

START=
Specifies that the snap is to start on the *n*th call to the SB buffer handler (instead of the first call). The default is 1.

The START= and the STOP= parameters are useful if you want to limit snapping to a specific period during which an application is running. The SBSNAP option often creates a large amount of output.

STOP=
Specifies that the snap is to be stopped on the *m*th call to the SB buffer handler (instead of the last call).

The default is that the snap is stopped after the last call.

SB Evaluation Snap (SBESNAP) control statement

The SBESNAP control statement snaps SB control blocks at the end of each periodic evaluation of the buffering process. The information in this snap can help you understand why IMS did or did not use SB.

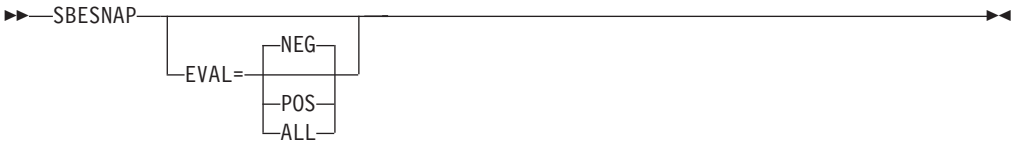
In the IMS online environment, the snap is written to the IMS log. In the IMS batch environment, you can request that the snap is written to any of the following:

- The data set specified in the DFSDDLT0 //PRINTDD DD statement
- A user-specified data set
- The IMS log

See “Snap Destination (SNAPDEST) control statement” for a description of how to request these snap destinations. In the batch environment, the default snap destination is the data set specified in the //DFSSNAP DD statement.

All SBESNAP parameters are optional.

The format of the SBESNAP control statement is:



If you specify multiple SBESNAP control statements, the *n*th control statement overrides the *m*th control statement (where *n*>*m*).

Related reading: For additional information about the SBESNAP control statement, see *IMS Version 11 Diagnosis*.

EVAL=

NEG requests a snap only when a periodical evaluation results in a decision not to use SB.

POS requests a snap only when a periodical evaluation results in a decision to use SB.

ALL requests a snap whenever a periodical evaluation of a data set group is made. The default is NEG.

Snap Destination (SNAPDEST) control statement

The SNAPDEST control statement is applicable in batch environments only. SNAPDEST specifies the destination of the snap created by the SBSNAP, SBESNAP, or SBCO options. In the IMS online environment, SNAPDEST is ignored, and snap information is written to the IMS log.

The default destination for snap information in the batch environment is the data set specified in the //DFSSNAP DD statement.

If you specify an invalid destination in the SNAPDEST control statement, IMS tries to write the snap to the IMS log.

The format of the SNAPDEST control statement is as follows:

The destination specified in the SNAPDEST control statement can be any one of the following:

DFSSNAP

Specifies that snap information is to be put in the data set associated with the //DFSSNAP DD statement.

PRINTDD
Specifies that snap information is to be put in the data set associated with the //PRINTDD DD statement of the DL/I test program DFSDDLT0. This data set also contains information created by DFSDDLT0.

ddname
Specifies that snap information is to be put in the data set with the specified ddname. The DD statement for this data set must conform to z/OS requirements for snap.

When the snap destination is ddname, DCB attributes are provided by IMS. You should not code them on the DD statement.

For this type of destination, each snap opens and closes the data set. If there are many snaps, this can result in significant overhead.

LOG
Specifies that snap information is to be put in the IMS log.

When snaps are written to the log, the log record codes are as follows:

X'67ED'- snap created by the SBESNAP option

X'67EE'- snap created by the SBSNAP option

X'67EF'- snap created by the SBCO option

You can select and print these log records using the File Select and Formatting Print utility (DFSERA10) and the Log Type X'67' Record Format and Print Module (DFSERA30).

Related reading: Both of these programs are described in *IMS Version 11 System Utilities*.

High-speed sequential processing control statements

Use HSSP control statements to set up the environment in which you process a selected PCB with HSSP, create an image copy of a designated DEDB area, and restrict access of a specific HSSP or non-HSSP application program to only designated DEDB areas.

This section describes the syntax, keywords, and values of the two high-speed sequential processing (HSSP) control statements: SETO and SETR.

HSSO, HSSR, and HSSD control blocks are built from SETO and SETR statements. These control blocks—specifically those that represent image copy data sets and those that are used for UOW locking—are formatted for offline dumps. These control statements are in the DFSCTL data set.

Syntax of HSSP control statements

Each HSSP control statement must begin on a new line. IMS considers the first nonblank character the start of the statement. You must leave one or more blanks following the statement name. Keywords follow the statement name. Keywords must be followed immediately by an equal sign (=) and a value. If you specify multiple keywords, separate them with commas. If you specify multiple values, enclose them in parentheses.

A blank or a closing parenthesis signals the end of a statement. You can continue HSSP control statements on multiple records. Use a comma at the end of a line to indicate continuation.

An asterisk (*) in the first position on a record indicates a comment. Elsewhere on a record, comments begin with a slash and asterisk (/*). You can include comments on each line with at least one blank between the keyword, value, or comma and the comment.

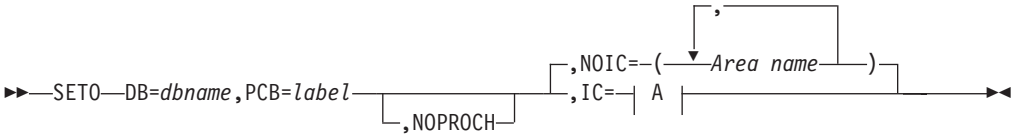
Set Options (SETO) control statement

The SETO control statement allows you to specify the options in processing a PCB with HSSP. With SETO, you can:

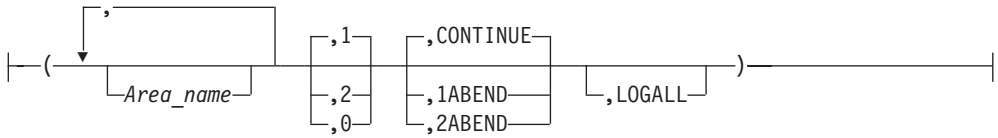
- Deactivate the HSSP option for a particular PCB
- Make an image copy of an updated area
- Specify the processing options in case of image copy failure

IMS interprets SETO when a region is started. After the region is active, you cannot alter the options.

Syntax



A:



Keyword parameters

DB=
Specifies the name of the database to which SETO applies. Because HSSP is only applicable to DEDBs, only the names of these DEDBs are valid.

PCB=
Specifies a one- to eight- character label that identifies the PCB to which SETO applies. It must be identical to the label on the associated DEDB PCB. If you use the same PCB label more than once for the same database name, only the first SETO statement with that label is used.

NOPROCH
Specify NOPROCH if you want to deactivate the HSSP option that was initialized in PSBGEN (PROCOPT=H). If you specify this keyword, PROCOPT=P is set, and the IC option is ignored. You can only specify one HSSP PCB per database per PSB. If more than one HSSP PCB points to the same database, you should use the NOPROCH keyword to deactivate additional PCBs before the SETO statements are processed. If more than one HSSP PCB exists after the SETO statements have been processed, the region is not scheduled.

NOIC=
Specifies that no image copy should be made for all or some of the areas listed

in DBDGEN. If you specify NOIC= without any area names, no image copy is made for any area in the database. You can use the option NOIC= with area names several times per SETO statement; however, you can specify a particular area only once. NOIC is the default if you do not specify IC=.

IC=

Specifies that an image copy should be made of all or some of the areas listed in DBDGEN. Allocation information about the image copy data sets is obtained from DBRC.

Values: The following values control the use of HSSP:

Area name

Lists the names of the areas to be image copied. If you need the areas to be processed in a specific order (other than the order in the DBDGEN), you must list them in the desired order and separate them with a comma. If the sequence of areas required is the same sequence as that in the DBDGEN, you only need to list the first and the last area names of the series and separate them with a hyphen. You can specify an area name only once. For example, *A4,A1-A3,A5-A8* is valid, but *A4,A1-A8* is invalid, because *A4* is repeated.

If you specify IC= without any area names, an image copy of all areas in the database occurs. You can specify IC= without area names only once per SETO statement. You can use the option IC= with area names several times per SETO statement; however, you can specify an area only once.

Commas and parentheses are optional if you specify only one parameter on the IC= parameter. You must register the area in DBRC in order for image copying to take place.

1 Specifies that one image copy data set should be made. The default is one.

2 Specifies that two image copy data sets should be made. If you specify IC=2, two image copy data sets must be registered with DBRC for an area; otherwise, no image copy is done for that area.

0 Specifies that no image copy data sets should be made. IC=0 is the same as specifying NOIC.

CONTINUE | 1ABEND | 2ABEND

These keywords describe the action a program should take in the event of an image copy failure.

CONTINUE specifies that the program continues processing if the image copy option cannot be performed. Under this circumstance, messages are sent to the MTO console and the job log (WTP). CONTINUE can be abbreviated as C.

1ABEND specifies that if image copying cannot be completed for one data set, the program must abend. 1ABEND can be abbreviated as 1A.

2ABEND specifies that if image copying cannot be completed for two data sets, the program must abend. If you request IC=2 and one of the data sets is still usable, HSSP continues to write to it. 2ABEND can be abbreviated as 2A.

LOGALL

Specifies that X'5950' log records are to be logged for DEDB updates in an HSSP environment. If LOGALL is not specified, only the X'5947' log record is logged.

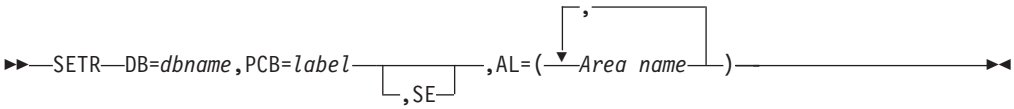
In an RSR environment, only X'5950' log records are logged, regardless of the specification of the LOGALL parameter.

If you specify only one parameter on the IC= keyword, you do not need to include the parentheses or the commas. Trailing commas for those parameters you do not include are also not required. For example, `IC=1` is equivalent to `IC=(,1,,)`, and `IC=1,LOGALL` is equivalent to `IC=(,1,,LOGALL)`.

Set Range (SETR) control statement

The SETR control statement specifies the processing range of PCBs to a database during scheduling of an application program. If a PSB has several PCBs pointing to the same database, you can restrict the access of each PCB to that database. Each program can only access data in the DEDB within the range defined.

Syntax



Keyword parameters

DB=
Specifies the name of the database to which the SETR statement applies.

PCB=
Specifies a one- to eight-character label that identifies the PCB to which the SETR statement applies. This label must be identical to the label on the associated DEDB PCB. If you use the same PCB label more than once for the same database name, only the first SETR statement with that label is used.

SE An optional keyword that instructs IMS to skip processing empty areas when SETR is in effect and the area is within the specified range.

AL=
Specifies the areas to which the PCB has access. The SETR area list does not have to match the SETO area list. (The SETR area list determines which areas can be processed. The SETO area list indicates which of the processed areas must have an image copy.) If you specify the SETR statement without a matching SETO statement (that is, the same DB= and PCB= values), the SETR statement processes the specified area list (AL=) values according to the PROCOPT option in PSBGEN.

HSSP control statement examples

The following are examples of SETO and SETR statements in the DFSCTL data set with their associated PCBs.

Sample PSBGEN:

```
L1 PCB TYPE-DB,DBDNAME=DEDB1,PROCOPT=A
L2 PCB TYPE-DB,DBDNAME=DEDB2,PROCOPT=HI
L3 PCB TYPE=DB,DBDNAME=DEDB3,PROCOPT=HRD
L4 PCB TYPE=DB,DBDNAME=DEDB4,PROCOPT=HA
L5 PCB TYPE=DB,DBDNAME=DEDB5,PROCOPT=HG
```

Sample of DFSCTL set option statements:

```
SETO DB=DEDB1,PCB=L1,IC=1               /* Example 1
SETO DB=DEDB2,PCB=L2,IC=(1,CONTINUE)    /* Example 2
SETO DB=DEDB3,PCB=L3,IC=(A1,A3-A6,2,1A) /* Example 3
SETR DB=DEDB3,PCB=L3,AL=(A1-A5)
SETO DB=DEDB4,PCB=L4,IC=(2,2ABEND)      /* Example 4
SETO DB=DEDB5,PCB=L5,NOIC=(A4-A5)       /* Example 5
SETO DB=DEDB5,PCB=L5,IC=1,LOGALL
```

Example 1

This SETO statement matches the PCB with label L1. However, the PROCOPT=A defines it as a non-HSSP PCB. The SETO statement is ignored.

Example 2

This SETO statement matches the PCB with label L2. The PROCOPT=HI defines it as an HSSP PCB.

IC= Indicates a list of values for the image copy option.

1 Requests one image copy of all referenced areas. A referenced area is an area for which a DL/I call has been issued.

CONTINUE
Indicates that processing should continue if image copying cannot be performed.

Example 3

This SETO statement matches the PCB with label L3.

A1,A3–A6
Requests an image copy of each indicated area when referenced by the application.

2 Requests two image copies of all referenced areas.

1ABEND
Requests that the program must abend if one image copy fails.

This SETR statement matches the PCB with label L3. The AL=(A1=A5) sets the range of areas to which PCB L3 has access.

Example 4

This SETO statement matches the PCB with label L4.

2 Requests two image copies of all referenced areas

2ABEND
Requests that the program must abend if both image copies fail. If one image copy data set is usable, the program continues to write on it.

Example 5

This SETO statement matches the PCB with label L5.

NOIC=
Excludes the areas for which HSSP image copying is to be done.

A4–A5
Indicates the areas for which no image copies are requested.

IC= Indicates a list of values for the image copy option.

1 Requests one image copy of all referenced areas. A referenced area is an area for which a DL/I call has been issued.

LOGALL
Indicates that updates are logged in X'5950' log records.

Set Index Maintenance Off (SETI) control statement

The Set Index Maintenance Off (SETI) control statement enables you to suppress index maintenance for any DEDB database where the secondary index is defined for the BMP application and where the PSB is set to *psbname*.

When a DEDB database has secondary index defined, IMS automatically performs index maintenance when the source statement is inserted, updated, or deleted. The index suppression option provides the capability for updating a DEDB database with one or more secondary index defined without index maintenance. If an application has many updates to the primary DEDB database that would result in significant index maintenance to its associated secondary index database, you can suppress the index maintenance for the application. Then synchronize your primary DEDB database and its secondary index databases at a later time using an in-house application or vendor tool product.

To suppress index maintenance, specify the //DFSCTL DD statement in the JCL of the IMS BMP region.

```
 //DFSCTL  DD  *
SETI PSB=psbname
```

The SETI PSB=psbname parameter suppresses index maintenance for any DEDB database with secondary index defined for the BMP application for PSB of psbname.

If *psbname* in the PSB=*psbname* parameter in the SETI statement does not match the PSB name for the BMP application, or the PSB= parameter is not specified in the SETI statement, message DFS0510E is issued and the application is terminated with an ABENDU1060. You will need to correct the SETI statement and rerun the BMP application.

Database resource adapter startup table for CCTL regions

Use the database resource adapter (DRA) startup table to provide definitional parameters for the coordinator control (CCTL) region. The DRA startup table establishes the subsystem name of the IMS that is being connected to, and defines the characteristics of the DRA, such as the number of threads.

This topic presents sample source code for DFSPZP00 and describes the parameters that are specified through the DFSPRP macro in DFSPZP00.

Sample DFSPZP00 source code

The DRA startup table, DFSPZPxx, is created by assembling the DFSPZPxx module. This code sample is the actual source code for DFSPZP00. To define other variations of DFSPZPxx, modify this code through the DFSPRP macro. Specify the DRA parameters as keywords on the DFSPRP macro. The keywords and their descriptions are described in “DFSPRP macro keywords in DFSPZP00” on page 819.

```
*********************************************************************
*                                                                   *
*        MODULE NAME: DFSPZP00                                      *
*                                                                   *
*        DESCRIPTIVE NAME: DATABASE RESOURCE ADAPTER (DRA)          *
*                  STARTUP PARAMETER TABLE.                         *
*                                                                   *
*************************************************************@SCPYRT*
*                                                                   *
*  Licensed Materials - Property of IBM                             *
*                                                                   *
*  5635-A01                                                         *
*                                                                   *
*  (C) Copyright IBM Corp. 1989,2009 All Rights Reserved.           *
*                                                                   *
*  US Government Users Restricted Rights - Use, duplication or      *
*  disclosure restricted by GSA ADP Schedule Contract with          *
*  IBM Corp.                                                        *
*************************************************************@ECPYRT*
*                                                                   *
*        FUNCTION: TO PROVIDE THE VARIOUS DEFINITIONAL PARAMETERS   *
*                  FOR THE COORDINATOR CONTROL REGION. THIS         *
*                  MODULE MAY BE ASSEMBLE BY A USER SPECIFYING      *
*                  THEIR PARTICULAR NAMES, ETC. AND LINKEDITED      *
*                  INTO THE USER RESLIB AS DFSPZPXX . WHERE XX      *
*                  IS EITHER 00 FOR THE DEFAULT, OR ANY OTHER ALPHA-*
*                  NUMERIC CHARACTERS.                              *
*                                                                   *
*        KEYWORDS FOR THE DFSPRP MACRO:                             *
*              DSECT=NO-A DSECT STATEMENT FOR PRP WILL NOT BE       *
*                       GENERATED (LABEL DFSPRP WILL BE ON DS 0D).  *
*                                                                   *
*              FUNCLV=  DEFAULT (2). ADAPTER FUNCTIONAL LEVEL.      *
*                                                                   *
*              DDNAME=  1 TO 8 CHARACTER DD NAME TO BE USED WITH    *
*                       DYNAMIC ALLOCATION OF THE DBCTL RESLIB.     *
*                       DEFAULT (CCTLDD).                           *
*                                                                   *
*              DSNAME=  1 TO 44 CHARACTER DATASET NAME OF THE       *
*                       DBCTL RESLIB.                               *
*                       DEFAULT (IMS.SDFSRESL)                      *
*                                                                   *
*              DBCTLID=XXXX-NAME OF THE DBCTL REGION                *
*                           DEFAULT = SYS1                          *
*                                                                   *
*              USERID=XXXXXXXX-NAME OF THE USER REGION              *
*                                                                   *
*              MINTHRD=XXX -MINIMUM NUMBER OF THREADS TO BE         *
*                          AVAILABLE (MAXIMUM NUMBER IS 999 )       *
*                          DEFAULT = 1                              *
*                                                                   *
*              MAXTHRD=XXX -MAXIMUM NUMBER OF THREADS TO BE         *
*                          AVAILABLE (MAXIMUM NUMBER IS 999 )       *
*                          DEFAULT = 1                              *
*                                                                   *
*              TIMER=XX-IDENTIFY TIMER VALUE IN SECONDS (DEFAULT 60)*
*                                                                   *
*              IDRETRY=XXX-Retry IDENTIFY attempt count             *
*                          before DFS690A message will be           *
*                          reissued.                                *
*                          (Default = 0, Min = 0, Max =255)         *
*                                                                   *
*              FPBUF=XXX-NUMBER OF FAST PATH BUFFERS TO BE ALLOCATED*
*                        AND FIXED PER THREAD (DEFAULT 00)          *
*                                                                   *
*              FPBOF=XXX-NUMBER OF FAST PATH OVERFLOW BUFFERS TO BE *
*                        ALLOCATED PER THREAD (DEFAULT 00)          *
```

```
*                                                                     *
*             SOD=X-OUTPUT CLASS TO BE USED FOR SNAP DUMP OF          *
*                   ABNORMAL THREAD TERMINATIONS (DEFAULT A)          *
*                                                                     *

*                                                                     *
*             TIMEOUT=XXX-DRATERM TIMEOUT VALUE IN SECONDS(DEFLT 60)* 
*                                                                     *
*             CNBA=XXXX TOTAL FP NBA BUFFERS FOR CCTL                 *
*                                                                     *
*                                                                     *
*             AGN=XXXXXXXX-1 to 8 CHARS (THIS PARM IS NO              *
*                          LONGER SUPPORTED, BUT IS ALLOWED           *
*                          FOR COMPATIBILITY REASONS.)                *
*                                                                     *
*             PCBLOC=24|31 Specifies the storage location             *
*                          of the application PCB list.               *
*                    24 -  DIRCA allocated in 24 bit private          *
*                    31 -  DIRCA allocated in 31 bit private          *
*                                                                     *

*                                                                     *
*        LOCATION: PRIVATE STORAGE, USER KEY                          *
*                                                                     *
*        THIS MODULE CONTAINS NO EXECUTABLE CODE.                     *
*                                                                     *
***********************************************************************
```

DFSPRP macro keywords in DFSPZP00

The following information describes DRA table parameters specified through the DFSPRP macro in DFSPZP00.

CNBA=
Is the total number of Fast Path NBA buffers for the CCTL's use.

DBCTLID=
Is the four-character name of the DBCTL region. This is the same as the IMSID parameter in the DBC procedure. The default name is SYS1. For more information about the DBC procedure, see "DBC procedure" on page 523.

DSECT=
Specifies whether a DSECT statement for PRP is generated.

YES
A DSECT statement is generated. The existing CSECT will be reestablished. The default is YES.

NO A DSECT statement is not generated. Label DFSPRP will be on DS 0D.

DDNAME=
Is a one- to eight-character ddname to be used with the dynamic allocation of the DBCTL execution library. The default ddname is CCTLDD. This library must contain the DRA modules.

DSNAME=
Is a one- to forty-four-character data set name of the DBCTL execution library. This library must contain the DRA modules and must be z/OS authorized. The default dsname is IMS.SDFSRESL.

FPBOF=
Is the number of Fast Path DEDB overflow buffers to be allocated per thread. The default is 00.

FPBUF=
Is the number of Fast Path DEDB buffers to be allocated and fixed per thread. The default is 00.

FUNCLV=
Specifies the level of the DRA that the CCTL supports:

1 Base CCTL function support.

2 ODBM Callable Interface support.

The default is 2. You should accept the default value since no other value is valid. The higher FUNCLV value includes lower values, and is required to support certain new functions.

PCBLOC=
Identifies where the application PCB list is built. It can be built either below the 16M line (PCBLOC=24), which is the default, or in extended private storage (PCBLOC=31), above the 16M line.

SOD= Is the output class to be used for a SNAP DUMP of abnormal thread terminations. The default is A.

TIMEOUT=
Is the amount of time (in seconds) a CCTL should wait for the successful completion of a DRA TERM request. Specify this value only if the CCTL is coded to use it. This value is returned to the CCTL upon completion of an INIT request. The default is 60 seconds.

TIMER=
Is the amount of time (in seconds) between attempts of the DRA to identify itself to DBCTL during an INIT request. The default is 60 seconds.

USERID=
Is an eight-character name of the CCTL region.

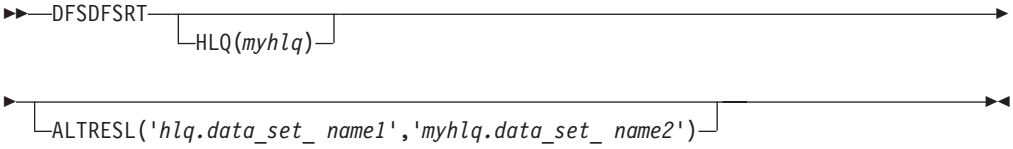
DFSDFSRT

DFSDFSRT is a REXX program that starts the z/OS Interactive Problem Control System (IPCS) with the IMS Dump Formatter.

Prerequisite: To provide access to the z/OS Interactive Problem Control System (IPCS) with the IMS Dump Formatter, include the IMS.SDFSEXEC data set in the SYSPROC DD concatenation.

Attention: Ensure that the IPCS is started before the IMS Application Menu is started. Otherwise, message `DFSIX103` is displayed. See *IMS Version 11 Installation* for more information about the IMS Application Menu.

DFSDFSRT syntax



DFSDFSRT keyword parameters

DFSDFSRT
Command to start the IPCS with the IMS Dump Formatter.

HLQ
Keyword that enables you to specify the high-level qualifier of the IMS distribution data sets.

The **HLQ** parameter is required the first time you use the DFSDFSRT statement. If you do not specify a high-level qualifier, DFSDFSRT uses the most recently specified high-level qualifier. After the first required use, this parameter is optional.

myhlq
High-level qualifier of the IMS distribution data sets.

ALTRESL
Keyword that enables you to specify a list of data set names that contain load modules.

If you specify the **ALTRESL** parameter, you should include SDFSRESL in the list of data set names. If you do not specify the **ALTRESL** parameter, *myhlq*.SDFSRESL is used as the IPCS TASKLIB data set.

myhlq.data_set_ name1
Fully-qualified name of a data set that contains load modules.

Starting the z/OS IPCS with the IMS Dump Formatter

To start IPCS with the IMS Dump Formatter, use either of the following commands:

- TSO %DFSDFSRT HLQ(myhlq)
- EXEC 'IMS.SDFSEXEC(DFSDFSRT)' 'HLQ(myhlq)'

IRLM use of SDUMP

The IRLM uses the z/OS SDUMP program for dumping whenever its ESTAE or FRR routines are entered. SDUMP dumps are directed to the SYS1.DUMPxx data sets and printed using the IPCS service aid. SDUMP has the same advantages over the SYSABEND dump program as the spinoff dump program. The operator is told which data set contains the SDUMP by message `IEA911E COMPLETE DUMP ON SYS1.DUMPxx`.

Here are sample control statements for executing the IPCS program.

```
//SYSIN    DD    *
          NEWDUMP DD=INPUT
          FORMAT
          LOGDATA
          VTAMMAP
          IRLM  irlm.subsystem.name
          PRINT JOBNAME=(irlm.job.name)
          END
```

Related reading: For a description of the IPCS utility, see *MVS/ESA Interactive Problem Control System (IPCS) Users Guide*.

Notices

This information was developed for products and services offered in the U.S.A.

This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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, MD-NC119
Armonk, NY 10504-1785
United States of America

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

© Copyright IBM Corp. 1974, 2015

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
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 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. The sample programs are provided "AS IS," without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

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.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Programming interface information

This information documents Product-sensitive Programming Interface and Associated Guidance Information provided by IMS.

Product-sensitive Programming Interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of this software product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive Programming Interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service. Product-sensitive Programming Interface and Associated Guidance Information is

identified where it occurs, either by an introductory statement to a section or topic, or by a Product-sensitive programming interface label. IBM requires that the preceding statement, and any statement in this information that refers to the preceding statement, be included in any whole or partial copy made of the information described by such a statement.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Bibliography

This bibliography lists all of the publications in the IMS Version 11 library, supplemental publications, publication collections, and accessibility titles cited in the IMS Version 11 library.

IMS Version 11 library

Title	Acronym	Order number
IMS Version 11 Application Programming	APG	SC19-2428
IMS Version 11 Application Programming APIs	APR	SC19-2429
IMS Version 11 Commands, Volume 1: IMS Commands A-M	CR1	SC19-2430
IMS Version 11 Commands, Volume 2: IMS Commands N-V	CR2	SC19-2431
IMS Version 11 Commands, Volume 3: IMS Component and z/OS Commands	CR3	SC19-2432
IMS Version 11 Communications and Connections	CCG	SC19-2433
IMS Version 11 Database Administration	DAG	SC19-2434
IMS Version 11 Database Utilities	DUR	SC19-2435
IMS Version 11 Diagnosis	DGR	GC19-2436
IMS Version 11 Exit Routines	ERR	SC19-2437
IMS Version 11 Installation	INS	GC19-2438
IMS Version 11 Licensed Program Specifications	LPS	GC19-2439
IMS Version 11 Master Index and Glossary	MIG	SC19-2440
IMS Version 11 Messages and Codes, Volume 1: DFS Messages	MC1	GC19-4232
IMS Version 11 Messages and Codes, Volume 2: Non-DFS Messages	MC2	GC19-4233
IMS Version 11 Messages and Codes, Volume 3: IMS Abend Codes	MC3	GC19-4234
IMS Version 11 Messages and Codes, Volume 4: IMS Component Codes	MC4	GC19-4235
IMS Version 11 Operations and Automation	OAG	SC19-2441
IMS Version 11 Release Planning	RPG	GC19-2442
IMS Version 11 System Administration	SAG	SC19-2443
IMS Version 11 System Definition	SDG	GC19-2444
IMS Version 11 System Programming APIs	SPR	SC19-2445
IMS Version 11 System Utilities	SUR	SC19-2446

Supplementary publications

Title	Order number
IMS and SOA Executive Overview	GC19-2516
Program Directory for Information Management System Transaction and Database Servers V11.0	GI10-8788
IMS Version 11 Fact Sheet	GC19-2451
IRLM Messages and Codes	GC19-2666

© Copyright IBM Corp. 1974, 2015

Publication collections

Title	Format	Order number
IMS Version 11 Product Kit	CD	SK5T-7375

Accessibility titles cited in the IMS Version 11 library

Title	Order number
z/OS TSO/E Primer	SA22-7787
z/OS TSO/E User's Guide	SA22-7794
z/OS ISPF User's Guide Volume 1	SC34-4822

Index

Special characters

/DBRECOVERY command
  preventing for a database that has INDOUBT EEQEs 771
/DBRECOVERY command, preventing 771
/DISPLAY POOL PSBP 71
/START OLDS command 120
/TRACE
  IMS Database Recovery Facility, tracing 757
* (asterisk)
  BPE trace table type 587
  CQS trace table type 587
  OM trace table type 587
  RM trace table type 587
  SCI trace table type 587

Numerics

2305 device 339, 369
2740 terminal
  fast path terminal 391
  LINEGRP macro statement, specifying 365
  TERMINAL macro statement, specifying 391
2741 terminal
  LINEGRP macro statement, specifying 365
  TERMINAL macro statement, specifying 391
2780 terminal
  LINEGRP macro statement, specifying 365
  TERMINAL macro statement, specifying 391
3270 terminal
  LINEGRP macro statement, specifying 365
  master terminal devices, choosing 98
  TERMINAL macro statement, specifying 391
3275 terminal
  LINEGRP macro statement, specifying 365
3340 device 367
3350 device 367
3375 device 367
3380 device 367
3390 device 367
3600 work station 391
64-bit storage pool 183

A

abend search and notification
  customizing 248
  DFSDFxxx 680
abends 541
abends *(continued)*
  U3303
    stopping transactions and PSBs after ten 89
ABND= parameter 448
ACB generation utility, PSB pools and 71
ACBGEN utility, PSB pools and 313
ACBLIB
  64-bit storage pool 183
  adding data sets to concatenation 116
  allocating using DFSMDA 116
  allocating using JCL 116
  changing data sets 116
  defining with DFSMDA member 118
  defining with IMS procedure 118
  dynamically allocated 116
  member online change
    defining with DFSMDA member 118
    defining with IMS procedure 118
  resizing inactive data set 116
ACCESS= parameter 324
accessibility
  features xi
  keyboard shortcuts xi
ACF/VTAM terminals
  IMS Extended Terminal Option Support 186
ACTIV= parameter 805
adapter
  names for XML message conversion 775
ADAPTER
  parameters 783
  statement 783
  syntax 783
ADDR= parameter 363, 376
address spaces
  CSL
    sequence for starting 208
AL= parameter 815
ALARM= parameter 710
ALL type system definition 2
allocating communication (MFS) pool space 74
allocating IMS system data sets
  large system definition environment 24
  message queue
    data set 126
    secondary allocation 126
  RECON data set for DBRC 136
  restart data set 125
allocation
  for HWSRCDR data set 220
  for required libraries 220
allocation of data sets
  considerations for
    global resource serialization 111
allocation of data sets *(continued)*
  considerations for *(continued)*
    JES 111
    XRF 137
  direct output data sets 135
  extended address volumes 130
  global resource serialization considerations 111
  JES considerations 111
  log data set. 119
  OLDS 119
  OSAM data sets 128
  resource definition data set (RDDS) 130
  SLDS 119
  spool SYSOUT 133
  VSAM data sets 129
  WADS 119, 123
  XRF data set considerations
    dynamic allocation considerations 137
    mandatory replication data sets 137
    mandatory shared data sets 137
    optional replication data sets 137
    other data sets impacted by XRF 137
    requirements for placing IMS data sets 137
ALOT= parameter 448, 692
ALTID= parameter 449
ALTRESL parameter 821
AOEXIT= parameter 316
AOI= parameter 420
AOI1= parameter 449
AOIP= parameter 449
AOIS= parameter 450
AOS= parameter 657
APAR= parameter 450
APARM= parameter 450
APF (authorized program facility)
  specifying in batch procedures 445
APNDG= parameter 338
APPC= parameter 450
APPCASY= parameter 657
APPCIOT= parameter 657
APPCRCV= parameter 657
APPCSE= parameter 450
APPLCTN macro
  CCTL, use with 79
  dynamic reassignment 307
  MSNAME macro, and 310
  ODBA, use with 79
  parameters
    FPATH= 308
    GPSB= 308
    LANG= 308
    PGMTYPE= 309
    PSB= 310
    SCHDTYP= 311
    SYSID= 310

© Copyright IBM Corp. 1974, 2015

APPLCTN macro *(continued)*
TRANSACT macro, and 309, 310
APPLCTN macro statement
for Fast Path 25
APPLFE= parameter 451
application programs
associated transactions 420
deleting 57
deleting dynamically 57
macros 254
online applications, defining 79
scheduling in parallel 92
APPLID= parameter 317
APPLID1= parameter 452
APPLID2= parameter 452
APPLID3= parameter 452
ARC= parameter 159, 452
ARCHDEF statement 761
ARCHDEF= parameter 723
ARCHIVE= parameter 724
ARMRST= parameter 452
ASM= parameter 351
ASMPRT= parameter 351
ASOT= parameter 452, 693, 698
ASR option 372
changing 376
displaying 376
ASSNCHANGE= parameter 657
asynchronous work element (AWE) 587
AUTHLOG= parameter 657
authorized program facility (APF)
specifying in batch procedures 445
AUTLCHANGE= parameter 657
AUTLDESC= parameter 698
AUTLGN= parameter 698
AUTLID= parameter 698
AUTLMOD= parameter 698
AUTO= parameter 453, 710
automatic dump data set allocation 236
automatic import 61
AUTOSCH option 133
AWE (asynchronous work element) 587

B

BACKUP= parameter 363, 365, 372, 376, 391, 441, 693
Base Primitive Environment (BPE)
associating exit types with exit routines 605
configuration member of the IMS PROCLIB data set 205
defining 202
entries for z/OS PPT 28
procedures for CSL 205
sample configurations 587
sharing configuration parameters 587
specify language 587
specify trace level 587
traces
writing to external data sets 587
tracing processing 587
user exit member of the IMS PROCLIB data set 205
batch dynamic allocation, disabling 764
batch initialization
modules that must be loaded 445
BATCH type system definition 2
batching messages 86
BHBUF= parameter 805
BKO= parameter 453
BLKSIZE= parameter
DFSMDA TYPE=DFSDCMON statement 329
BLOCK= parameter 724, 728, 730
BMP
declaring 76
EXEC parameters 165
BMP regions
choosing number 69
BMPUSID= parameter 657
BPE (Base Primitive Environment)
associating exit types with exit routines 605
configuration member of the IMS PROCLIB data set 205
configuring 218
defining 202
procedures for CSL 205
sample configurations 587
sharing configuration parameters 587
specify language 587
specify trace level 587
trace table types 587
traces
writing to external data sets 587
tracing processing 587
user exit member of the IMS PROCLIB data set 205
BPE configuration member of the IMS PROCLIB data set
keywords 218, 587
recommendations 587
specify 587
specifying 218
BPE exit routine member of the IMS PROCLIB data set
EXITMBR parameter 587
BPE trace table types
* (asterisk) 587
AWE (asynchronous work element) 587
CBS (control block service) 587
CMD (command trace table) 587
DISP (dispatcher trace table) 587
ERR (error trace table) 587
HASH (hash trace table) 587
LATC (latch trace table) 587
SSRV (system services trace table) 587
STG (storage service trace table) 587
USRX (user exit routine trace table) 587
BPECFG= parameter 453
BPEINI00
IMS Connect
authorized supervisor state 209
BSIZ= parameter 454
BUF= parameter 454
buffer pool
defining OSAM subpools 748
defining VSAM subpools 743
description of 179
Fast Path 153
buffer pool *(continued)*
MFS 75
OSAM 179, 181
OSAM buffer pool compatibility 181
OSAM SB 179
sizes 150
specifying 179
VSAM 179, 180
buffers
declaring availability 155
OSAM
adjusting 179
specifying
in DFSVSMxx member control statement 738
VSAM
adjusting 179
BUFFERS= parameter 367
BUFNO= parameter 763
DFSMDA TYPE=DFSDCMON statement 329
BUFPOOLS macro
description 69
MFS considerations 142
parameters
DMB= 312
EPCB= 312
FORMAT= 312
FRE= 313
PSB= 313
PSBW= 313
SASPSB= 313
BUFSETS= parameter 805
BUFSIZE= parameter 363, 376, 391

C

CBS (control block service) 587
CCTCVCAN= parameter 454
CCTL
DBCTL databases, and 185
preparing 186
starting 817
CFIRLM= parameter 765
CFNAMES= parameter 765
CFOSAM= parameter 765
CFRM (coupling facility resource management)
policy, defining 202
CFSizer 197
CFVSAM= parameter 765
changing block size
resource definition data set (RDDS) 132
channel-end appendages 101
checkpoint
data sets 508
frequency of, setting 69
CHECKPOINT 69
restrictions 380
CHNGDUMP MAXSPACE
recommended setting 236
CIOP= parameter 454
CKPTID= parameter 454
CL1=,CL2=CL3=,CL4= parameters 455
CLASS= parameter 455
client descriptor, OTMA 771

CMD (command trace table) 587
CMDMCS= parameter 455
CMDP= parameter 456
CMDSEC= parameter 649
CNBA= parameter 817
COBOL
  data conversion
    from XML 216
  XML-to-COBOL conversion support
    configuring 216
    example 216
    prerequisites 216
    restrictions 216
CODE= parameter 384, 420
cold start
  changing RECLNG 367
  impact on resource definitions 41
  result of not specifying
    MODBLKS 340, 350
COMM macro
  IMSGEN macro, and 386
  MFS 142
  parameters 316
    APPLID= 317
    COPYLOG= 317
    EDTNAME= 318
    FESEXIT= 318
    MFSEXIT= 318
    no longer supported 322
    OPTIONS= 319
    PASSWD= 320
    RECANY= 320
    SECCNT= 322
    SIMEXIT= 322
  relationship to IMSGEN and
    SECURITY 386
  SECURITY macro, and 386
  syntax diagram 315
  TERMINAL macro, and 322
  VTAM terminals, and 317
COMM macro statement 259
command trace table (CMD) 587
commands
  /CHECKPOINT 69
  /DISPLAY 71
  /START OLDS 120
  /TRACE 224
    DASD log activity, tracing 755
    DB2 for z/OS subsystem
      connection, tracing 759
    DBF entries from FP, tracing 756
    dispatcher activity, tracing 755
    DL/I activity, tracing 755
    Fast Path activity, tracing 756
    IMS Database Recovery Facility,
      tracing 757
    latch activity, tracing 756
    lock activity, tracing 757
    options 754
    OTMA control, tracing 758
    queue manager, tracing 758
    RSR log router, tracing 757
    scheduler, tracing 758
    shared queues interface,
      tracing 758
    storage manager calls, tracing 759
  CREATE TRAN 81, 83
commands *(continued)*
  DELETE.LOG DBRC 120
  DFSDFSRT 821
  IMPORT
    backing out resources created
      using IMPORT command 63, 64
Common Queue Server (CQS)
  entries for z/OS PPT 28
Common Service Layer (CSL)
  CSLDCxxx IMS PROCLIB data set
    member 205
  CSLDIxxx IMS PROCLIB data set
    member 205
  DFSCGxxx IMS PROCLIB data set
    member 205
  DFSDFxxx 673
  DFSDFxxx IMS PROCLIB data set
    member 205
  DFSVSMxx
    CSLT= 754
  entries for z/OS PPT 28
  IMS PROCLIB data set members 205
  Member DFSCGxxx 649
  OM trace table types 587
  Resource Manager
    startup procedure 518
  RM trace table types 587
  SCI trace table types 587
  Structured Call Interface
    startup procedure 519
  system definition and tailoring 205
Common Storage Tracker 235
communication devices
  defining
    non-VTAM devices 98
    VTAM terminals 95
COMPT= parameter 380, 391
COMPT1= parameter 391
COMPT2= parameter 391
COMPT3= parameter 391
COMPT4= parameter 391
COMPTn= parameter 693
configuration
  alternative IMS online systems 23
  port id examples 782
configuration members
  DATASTORE 782
  HWS 782
  IMS Connect
    ODACCESS statement 782
  IMSplex 782
control block service (CBS) 587
control intervals 473
control region
  execution parameters 174
control statements 805
converter
  names for XML message
    conversion 775
CONVTYPE= parameter 648
COPYLOG= parameter 317
CORE= parameter 338, 457
coupling facility
  enabling multiple DEDB areas to
    share same CF structure 740
  OSAM data caching 766
coupling facility *(continued)*
  structure names for Sysplex data
    sharing 764
coupling facility resource management
  (CFRM)
  policy, defining 202
CPLOG= parameter 338, 457
CPUTIME= parameter 457
CQS (Common Queue Server) 197
  address space 736
  customizing 202
  data sets 510
  defining 197, 202
  execution data sets 510
    structure recovery data set 510
    system checkpoint data set 510
  execution parameters
    specifying 510
  global structure definition member of
    the IMS PROCLIB data set
    keywords 619
  initialization parameters member of
    the IMS PROCLIB data set
    specifying 614
  local structure definition PROCLIB
    member
    specifying 617
  LOGR policy 202
  MAXBUFSIZE 202
  monitoring 202
  multiple clients 197
  parameters
    CQS PROCLIB 614
    execution 510
  resource structure
    calculating storage for 198
  set up tracing 244
  shared queues, placing messages
    on 494
  starting 202
  tailoring 197
CQS exit routine member of the IMS
  PROCLIB data set
  EXITMBR parameter 587
CQS setup recommendations 239
  trace environment
    conservative 239
    more aggressive 239
CQS trace table types
  * (asterisk) 587
  CQS (common queue server trace
    table) 587
  ERR (error trace table) 587
  INTF (interface trace table) 587
  STR (structure trace table) 587
CQS= parameter 736
CQSIPxxx
  defining CSL 205
  format rules 615
  overview 614
  sample member of the IMS PROCLIB
    data set 617
CQSSGxxx
  ELEMENT value 198
  ENTRY value 198
  formatting rules 621
  overview 619

CQSSGxxx *(continued)*
sample member of the IMS PROCLIB data set 627
CQSSLxxx
formatting rules 617
overview 617
sample member of the IMS PROCLIB data set 619
CQSSSN= parameter 737
CRC (command recognition character) 340
CRC= parameter 457
CREATE TRAN command 81, 83
creating output data sets 243
CSAPSB= parameter 458
CSL (Common Service Layer)
address spaces
sequence for starting 208
DFSCGxxx IMS PROCLIB data set member 205
DFSDFxxx IMS PROCLIB data set member 205
IMS PROCLIB data set members 205
OM trace table types 587
RM trace table types 587
SCI trace table types 587
system definition and tailoring 205
CSLDCxxx
syntax 629
CSLDCxxx IMS PROCLIB data set member 205
CSLDCxxx, ODBM configuration member of the IMS PROCLIB data set 628
CSLDIxxx 633
IMS PROCLIB data set member 205
sample member of the IMS PROCLIB data set 635
CSLG= parameter 459
CSLIPxxx
validating with Syntax Checker 283
CSLODBM procedure
parameters
ODBMCFG= 481
ODBMINIT= 481
ODBMNAME= 481
CSLOIxxx 636
sample member of the IMS PROCLIB data set 637
validating with Syntax Checker 283
CSLRIxxx 641
validating with Syntax Checker 283
CSLRM procedure 518
CSLSGxxx
validating with Syntax Checker 283
CSLSIxxx 205, 644
validating with Syntax Checker 283
CSLSLxxx
validating with Syntax Checker 283
CSLT= parameter 754
CTLBLKS type system definition 2

D

DASD
log tracing 755
logging 502, 503, 504, 761
DASD logging
OLDS 120
SLDS 125
WADS 123
data communication
EXEC parameters 162, 165
line groups 253
macro statements 258
master terminal, specifying 98
terminals 253
terminals, defining
non-VTAM terminals 98
VTAM terminals 95
data entry databases (DEDBs)
LKASID value 739
data set allocation
considerations for
global resource serialization 111
JES 111
XRF 137
direct output data sets 135
global resource serialization considerations 111
JES considerations 111
log data sets
OLDS 120
SLDS 125
WADS 123
OLDS 119
OSAM data sets 128
SLDS 119
spool SYSOUT data sets
defining spool line groups 133
XRF considerations for spool line groups 133
VSAM data sets 129
WADS 119, 123
XRF data set considerations
discussion of 137
dynamic allocation considerations 137
mandatory replication data sets 137
mandatory shared data sets 137
optional replication data sets 137
other data sets impacted by XRF 137
data sets
ACBLIB
allocating 116
allocating 243
extended address volumes 130
allocation 126
CQS execution 510
direct output 135
entry-sequenced 510
IMS Connect 220
initializing IMS system data sets 111, 114
large sequential data sets
fallback from 142
message queue
restrictions in XRF 126
OLDS 119
online 115
OSAM 128
output 243
data sets *(continued)*
resource definition 35, 41
SLDS 119
structure recovery 510
SYSOUT 133
system
online change function 111
system checkpoint 510
WADS 119
XRF requirements 137
data sharing
DATABASE macro statement 79
database-level example 275, 277
IMSCTRL macro statement 79
inter-CPC block-level example 279
intra-CPC block-level example 278
system configuration example 275
data store
APPL parameter 782
DRU parameter 782
GROUP parameter 782
ID parameter 782
keyword parameters 782
MEMBER parameter 782
RRNAME parameter 782
TMEMBER parameter 782
database
buffers, Fast Path 153
deleting dynamically 55
deleting with DRD 55, 57
Database Control (DBCTL)
IMS traces
activating 242
DL/I 242
Fast Path 242
DATABASE macro
description 323
online applications, defining 79
parameters 324
ACCESS= 324
DBD= 325
RESIDENT= 324
syntax diagram 324
databases
DBRC, registering with 461
deleting 55
DFSDFxxx
processing options 679
example 253
Fast Path 459, 467
macros 254
main storage (MSDBs) 646
managing 323
online, declaring 76
DATASTORE
parameters 784
statement 784
syntax 784
DB/DC
enabling subsystem for FDBR 106
DB= parameter 805, 813, 815
DB2
subsystem connection, tracing 759
subsystem identification parameters 147
DB2 for z/OS
defining to IMS 221

DB2 for z/OS *(continued)*
external subsystem module table 227
preparing 227
specifying groups for IMS online regions 224
DB2 for z/OS access
FSDB2AF DD statement 228
JBP region, from a configuring 228
JMP region, from a configuring 228
RRSAF 228, 229
RRSAF (Recoverable Resource Manager Services attachment facility) 229
DB2 Recoverable Resource Manager Services attachment facility 228
DBBBATCH procedure 520
description 520
parameters
APARM= 450
BKO= 453
BUF= 454
DBRC= 460
EXCPVR= 468
FMTO= 468
GSGNAME= 471
IMSID= 472
IMSPLEX= 473
IOB= 473
IRLM= 473
IRLMNM= 474
LOCKMAX= 476
LOGA= 476
LOGT= 476
MBR= 479
MON= 480
PRLD= 487
PSB= 487
RGN= 492
RGSUF= 492
RST= 492
SOUT= 495
SPIE= 495
SRCH= 496
SSM= 496
SWAP= 498
SYS= 499
SYS2= 499
TEST= 499
TMINAME= 499
DBBBATCH procedures
parameters
CKPTID= 454
DBBF= parameter 459
DBC (DBCTL online environment)
procedure for DBCTL 101
DBC procedure
description 523
dynamic resource definition 523
parameters
AOIP= 449
ARC= 452
ARMRST= 452
AUTO= 453
BSIZ= 454
CCTCVCAN= 454
DBC procedure *(continued)*
parameters *(continued)*
CIOP= 454
CMDMCS= 455
CORE= 457
CRC= 457
CSAPSB= 458
DBBF= 459
DBFP= 459
DBFX= 460
DBRCNM= 461
DBRSE= 461
DBWP= 461
DESC= 464
DFSDF= 464
DLINM= 464
DLIPSB= 464
DMB= 465
DMHVF= 465
EPCB= 467
FDRMBR= 468
FIX= 468
FMTO= 468
FP= 468
FPWP= 470
IMSID= 472
IOB= 473
IRLM= 473
IRLMNM= 474
ISIS= 474
LGNR= 476
MAXPST= 479
MCS= 479
ORSMBR= 482
OTHR= 482
PIINCR= 485
PIMAX= 485
PRDR= 486
PREMSG= 486
PRLD= 487
PSBW= 487
PST= 488
PSWDC= 488
RDMNM= 491
READNUM= 491
RES= 492
RGSUF= 492
RSRMBR= 492
SPM= 496
SRCH= 496
SSM= 496
SUF= 497
SVC2= 498
TRACK= 500
UHASH= 501
USERVAR= 501
VSPEC= 501
WADS= 502
WKAP= 502
YEAR4= 502
storing 445
DBCTL
CCTL, and 185
enabling subsystem for FDBR 106
environment, example 280
IVP base environment 101
DBCTL *(continued)*
maximum number of BMP and DRA threads 69
DBCTL (Database Control)
IMS traces
activating 242
DL/I 242
Fast Path 242
DBCTLID= parameter 817
DBD= parameter 325, 459, 647, 749
DBFMSDBx
description 646
Main Storage Databases, specifying 646
parameters 647
DBFP= parameter 459
DBFX= parameter 460
DBLDL= parameter 460
DBNAME= parameter
control statements
DFSMDA TYPE=DATABASE 325
DFSMDA TYPE=FPDEDB 325
DFSMDA TYPE=IMSACB 325
DBRC (Database Recovery Control)
DATABASE macro, ACCESS parameter 324
initialization parameters member of the IMS PROCLIB data set 781
initializing the RECON data set 136
procedure
storing 445
user exit list member of the IMS PROCLIB data set 781
DBRC procedure
automatic initiation 529
choosing DBRC parallel processing 529
manual initiation 529
parameters 529
DBRCNM= 461
DD statements 529
DPRTY= 465
IMSID= 472
IMSPLEX=, description of 473
IMSPLEX=, restriction on using 473
RGN= 492
SOUT= 495
SYS2= 499
PDS member name 529
DBRCNM= 529
START commands
example 529
starting DBRC in a BPE 529
DBRC= parameter 340, 460
DBRCGRP= parameter 461
DBRCNM= parameter 159, 340, 461
DBRSE= parameter 461
DBWP= parameter 461
DC= parameter 462
DCC parameters
parameters
ARMRST= 452
DCC procedure
description 532
dynamic resource definition 532

DCC procedure *(continued)*
parameters
ALOT= 448
AOIP= 449
AOIS= 450
APPC= 450
APPCSE= 450
APPLID1= 452
APPLID2= 452
APPLID3= 452
ARC= 452
ARMRST= 452
ASOT= 452
AUTO= 453
CIOP= 454
CMDMCS= 455
CRC= 457
DC= 462
DESC= 464
DFSDF= 464
DLQT= 465
DPRTY= 465
DSCT= 465
DYNP= 465
EMHB= 466
EMHL= 466
EPCB= 467
ETO= 467
EXVR= 468
FBP= 468
FESTIM= 468
FIX= 468
FMTO= 468
FRE= 470
GRNAME= 470
GRSNAME= 471
HIOP= 471
HSBID= 471
HSBMBR= 471
IMSID= 472
ISIS= 474
LGMSGSZ= 475
LHTS= 476
LOGT= 476
LTERM= 478
LUMC= 478
LUMP= 478
MAXPST= 479
MCS= 479
NHTS= 480
NLXB= 480
ORSMBR= 482
OTMA= 482
OTMANM= 483
OTMASP= 484
PASSWD= 484
PASSWD1= 484
PRDR= 486
PRLD= 487
PSB= 487
PSBW= 487
PST= 488
PSWDC= 488
QBUF= 489
QBUFMAX= 489
QBUFSZ= 489
QTL= 490
DCC procedure *(continued)*
parameters *(continued)*
QTU= 490
RCF= 490
RECA= 491
RECASZ= 491
RES= 492
RGN= 492
RGSUF= 492
RSRMBR= 492
RVFY= 493
SAV= 493
SGN= 494
SHAREDQ= 494
SHMSGSZ= 495
SOD= 495
SOUT= 495
SPM= 496
SRCH= 496
SSM= 496
SUF= 497
SVC2= 498
SYS= 499
SYS1= 499
SYS2= 499
TRACK= 500
TRN= 500
TSR= 500
UHTS= 501
USERVAR= 501
VAUT= 501
VSPEC= 501
WADS= 502
WKAP= 502
YEAR4= 502
storing 445
DCCTL
including Fast Path 25
installing with ETO Feature 101
DCLWA= parameter 340, 420
DD statements
DFSDB2AF 228
INPARMS 506
MODBLKS 43
DD= parameter 805
DDNAME= keyword
control statements
DFSMDA TYPE=DATASET 325
DFSMDA
TYPE=DFSDCMON 325
DDNAME= parameter 365, 376, 761, 817
control statements
DFSMDA
TYPE=DFSDCMON 329
DEDB (data entry database) 174
LKASID value 739
online utility region parameters 158
DEDBMAS= parameter 740
DEFERFIX= parameter 710
defining
BPE 202
CQS 202
defining external subsystems to IMS 221
defining language interface module 223
defining large sequential data sets 141
definition process, system
ALL 2
definition process, system *(continued)*
BATCH 2
CTLBLKS 2
JCLIN process 28
MODBLKS 2
MSVERIFY 2
NUCLEUS 2
ON-LINE 2
overview of 1
SMP/E maintenance 28
stage 1 21
stage 2 27
types of 2
DEGRADE= parameter 763
DELETE.LOG DBRC command 120
dependent address space
procedures 101
dependent regions
Fast Path parameters 156
DESC= parameter 340, 464
descriptor definitions
exporting 59
exporting to RDDS 60
importing 61, 62
descriptors
client descriptor, OTMA 771
DFSOTMA descriptor 778
DFSUSER 190
ETO
rules for defining 187
MFS device 189
MSC 189
OTMA
client descriptor 771
DFSOTMA descriptor 778
OTMA destination descriptor 775
user 190
design limit 299
destination descriptor, OTMA 775
devices
2305 339
3340 339
3350 339
3375 339
3380 339
3390 339
defining non-VTAM 98
LGDK 339
DFS62DTx
description 647
parameters 648
DFSCGxxx 205
DFSCMC10 module
installing MSC CTC links 338
DFSDB2AF DD statement 228
DFSDCxxx 205
data communications options,
specifying 657
description 657
validating with Syntax Checker 283
DFSDF= parameter 464
DFSDFSRT 820
parameters 821
syntax 820
DFSDFSRT command 821
DFSDFxxx 205
Common Service Layer 673

DFSDFxxx *(continued)*
databases 679
diagnostics and statistics 680
dynamic resource definition (DRD) 40
dynamic resource definitions 149, 682
examples 672
exit routines 687
IMS abend search and notification 680
overview 672
parameters 672
sections 672
shared queues 686
syntax 672
validating with Syntax Checker 283
DFSDRFnn
description 688
parameters 689
DFSDSCMx 689
ETO descriptors
common format 690
DFSDSCTy 701
DFSFDRxx
description 702
parameters 703
DFSFIXnn
description 704
DFSHALDB ddname
selective partition processing 192
DFSHSBxx
description 709, 714
parameters 710
preinitialization routines, specifying dependent region 714
XRF options 709
DFSIASN0 541
DFSIASN0 procedure
parameters
ABND= 448
APAR= 450
FMID= 468
GEN= 470
IMS= 472
MOD= 480
MSG= 480
RC= 490
SYSID= 499
T= 499
DFSIDEF macro 159
DFSIDEF0 macro 159
DFSINTxx
description 714
parameters 714
DFSJBP procedure
description 543
parameters
ENVIRON= 466
JVMPOPMAS= 475
DFSJMP procedure
description 545
parameters
ENVIRON= 466
JVMOPWKR= 475
DFSJVMAP 714
DFSJVMEV member
DB2 for z/OS JDBC driver 228
DFSJVMMS 715
DB2 for z/OS JDBC driver 228
DFSLI macro 223
DFSLI000 (language interface module) 223
DFSMDA (Dynamic Allocation macro)
allocating ACBLIB data sets 116
allocating the system log data sets 125
definitions 101
dynamically allocated ACB staging library 118
dynamically allocated ACBLIB data sets 118
examples 334
Fast Path DEDBs 325
IMSDALOC procedure 577
invoking the procedure 577
JCL requirements 577
logical relationships 336
monitor data set 325
multiple DEDBs 325
OLDS 325
overview 325
restrictions 336
SLDS 325
statement types
DATABASE 325
DATASET 325
DFSDCMON 325
FINAL 325
FPDEDB 325
IMSACB 325
INITIAL 325
OLCSTAT 325
OLDS 325
RECON 325
SLDS 325
WADS 325
DFSMPLxx
description 716
high-use program modules, making resident 716
parameters 717
DFSMPR procedure
description 547
IMSMSG job 579
parameters
ALTID= 449
APARM= 450
APPLFE= 451
CL1=,CL2=CL3=,CL4= 455
DBLDL= 460
IMSID= 472
LOCKMAX= 476
NBA= 480
OBA= 480
OPT= 481
OVLA= 484
PCB= 485
PREINIT= 486
PRLD= 487
PWFI= 488
RGN= 492
SOD= 495
DFSMPR procedure *(continued)*
parameters *(continued)*
SOUT= 495
SPIE= 495
SSM= 496
STIMER=, message-driven programs 497
SYS2= 499
TLIM= 499
VALCK= 501
VFREE= 501
VSFX= 501
DFSMRC20 103
DFSMRCL0 103
DFSORSxx member 718
DFSOTMA descriptor 778
DFSPBDBC
description 719
parameters 719
DFSPBDCC
description 719
parameters 719
DFSPBIMS
description 720
parameters 720
DFSPBxxx 174, 205
defining 174
description 718
IMS Syntax Checker 174
sample 174
updating 174
validating with Syntax Checker 283
DFSPRP 817
DFSPZP00 817
DFSRESLB DD statement
IMS procedures 445
DFSRSRxx
description 720
parameters 723
DFSSPMxx
description 731
examples 735
parameters 734
DFSSQxxx
description 736
parameters 736
validating with Syntax Checker 283
DFSSTAT report 504
DFSURDD0 65
DFSUSER 190
DFSVNUCx module 101
DFSVSMxx
/DBRECOVERY command, preventing 771
control statements, types of 738
description 738
discarding preallocated SDEP CIs 770
Fast Path DEDB buffer pools, defining 739, 740
long busy handling function, enabling 768
OSAM buffer pools, defining 748
OSAM subpools, defining 748
parallel database open, disabling 764
PPUR= control statement 769
PSELNODBRC control statement 770

DFSVSMxx *(continued)*
resuming an online reorganization for HALDBs 770
sequential buffering, specifying 751
serviceability and trace options, defining 752
VSAM buffer pools, defining 741
VSAM performance options 746
VSAM subpools, defining 743
DFSVSMxxx 205
DFSYDTx 771
DIAG= parameter 755
diagnosis 541
IMS abend search and notification 245
send email about abends 245
z/OS trace tables
setting the size 235
diagnostics
automatic dump data set allocation 236
CHNGDUMP MAXSPACE 236
Common Storage Tracker 235
IMS Control Region EXEC 237
setting up IMS for 235
diagnostics and statistics
DFSDFxxx 680
DIRCA= parameter 464
direct output data sets 135
discarding preallocated SDEP CIs 770
DISP (dispatcher trace table) 587
DISP= keyword
DFSMDA TYPE=DATASET control statement 325
DISP= parameter 755
dispatcher trace table (DISP) 587
DISPLAY command 71
DL/I
accounting procedures 71
call image trace 760
DLISAS procedure, modifying 73
execution parameters, system 147
exits, modifying 73
lock activity, tracing 757
security considerations 71
selecting 71
starting 73
storage considerations 71
tuning considerations 71
DL/I application programs
example macro statements 255
DL/I= parameter 755
DLIBATCH procedure
description 548
parameters
APARM= 450
BKO= 453
BUF= 454
CKPTID= 454
DBRC= 460
EXCPVR= 468
FMTO= 468
GSGNAME= 471
IMSID= 472
IMSPLEX= 473
IOB= 473
IRLM= 473
DLIBATCH procedure *(continued)*
parameters *(continued)*
IRLMNM= 474
LOCKMAX= 476
LOGA= 476
LOGT= 476
MBR= 479
MON= 480
PRLD= 487
RGN= 492
RGSUF= 492
RST= 492
SOUT= 495
SPIE= 495
SRCH= 496
SSM= 496
SWAP= 498
SYS2= 499
TEST= 499
TMINAME= 499
DLIDSIZE= parameter 725
DLINM= parameter 340, 464
DLIPSB= parameter 464
DLISAS
exit routines in 73
starting 73
using 71
DLISAS procedure
description 551
parameters
DPRTY= 465
IMSID= 472
RGN= 492
SOUT= 495
SYS2= 499
storing 445
DLOG= parameter 755
DLQT= parameter 465
DMB= parameter 312, 465
DMHVF= parameter 465
DOPT online change 306
DPRTY= parameter 465
DRA startup table 817
DRD (dynamic resource definition)
backing out
resources created with IMPORT command 63, 64
commands that support DRD 38
commands to manage resource and descriptor definitions 36
creating
application program descriptor definitions 48
application program resource definitions 48
descriptor definitions 47
Fast Path routing code definitions 48
Fast Path routing code descriptor definitions 48
resource descriptor definitions 46
resource descriptor definitions in IMSplex 50, 54
runtime database resources 47
runtime resource definitions 46
runtime resource definitions in IMSplex 50, 54, 59
DRD (dynamic resource definition) *(continued)*
creating *(continued)*
transaction descriptor definitions 49
transaction resources 49
default resource descriptors 36
deleting
application programs 57
database 55
resource descriptor definitions 54
resource descriptor definitions in IMSplex 59
routing codes 58
runtime resource definitions 54
transactions 58
DFSDFxxx 40
disabling for MODBLKS resources 66
enabling 44
exporting
descriptor definitions 59
resource definitions 59
importing
automatic import function 61
descriptor definitions 61
resource definitions 61
using IMPORT command 61
ISPF panels for managing resources 36
main storage databases 47
maintaining DRD environment 64
manage resources from ISPF panels 36
MODBLKS
enabling 45
preparing to enable 45
overview 35
RDDS
enabling 45
requirements 41
MODBLKS 41
resource definition data set (RDDS)
allocating 130
restrictions 42
updating
application program descriptor definitions 51
application program resources 51
database resources 51
descriptor definitions 51
Fast Path routing code descriptor definitions 52
Fast Path routing code resource definitions 52
resource descriptor definitions 50
runtime resource definitions 50
transaction descriptor definitions 53
transaction resource definitions 53
DREF storage, defining 688
DSCT= parameter 465
DSETS= parameter 367
DSN= parameter 727, 729
DSNAME= data set
control statements
DFSMDA TYPE=DATASET 325

DSNAME= data set *(continued)*
control statements *(continued)*
DFSMDA
TYPE=DFSDCMON 329
DSNAME= parameter 817
DSPBIxxx 781
sample member of the IMS PROCLIB data set 781
DSSIZE= parameter 725
dump
system execution parameters 147
DUMP= parameter 755
DUMPIO= parameter 755
DXRJPROC procedure
description 554
parameters
DEADLOK= 462
dynamic allocation
ACBLIB data sets 118
terminals 103
using extended storage 751
Dynamic Allocation macro (DFSMDA)
examples 334
Fast Path DEDBs 325
IMSDALOC procedure 577
invoking the procedure 577
JCL requirements 577
logical relationships 336
monitor data set 325
multiple DEDBs 325
OLDS 325
restrictions 336
SLDS 325
statement types
DATABASE 325
DATASET 325
DFSDCMON 325
FINAL 325
FPDEDB 325
IMSACB 325
INITIAL 325
OLCSTAT 325
OLDS 325
RECON 325
SLDS 325
WADS 325
dynamic database buffer pool definition
OSAM 182
VSAM 180, 182
dynamic PSB 76
dynamic reassignment and APPLCTN macro 307
dynamic resource definition (DRD)
backing out
resources created with IMPORT command 63, 64
commands that support DRD 38
commands to manage resource and descriptor definitions 36
creating
application program descriptor definitions 48
application program resource definitions 48
descriptor definitions 47
Fast Path routing code definitions 48
dynamic resource definition (DRD) *(continued)*
creating *(continued)*
Fast Path routing code descriptor definitions 48
resource descriptor definitions 46
resource descriptor definitions in IMSplex 50
runtime database resources 47
runtime resource definitions 46
runtime resource definitions in IMSplex 50, 54
transaction descriptor definitions 49
transaction resources 49
DBC procedure 523
DCC procedure 532
default resource descriptors 36
deleting
application programs 57
database 55
resource descriptor definitions 54
resource descriptor definitions in IMSplex 59
routing codes 58
runtime resource definitions 54
runtime resource definitions in IMSplex 59
transactions 58
DFSDFxxx 40, 149, 682
disabling 66
enable automatic export 64
enable automatic import 64
enabling 44
exporting
descriptor definitions 59
resource definitions 59
importing
automatic import function 61
descriptor definitions 61
resource definitions 61
using IMPORT command 61
ISPF panels for managing resources 36
main storage databases 47
maintaining DRD environment 64
manage resources from ISPF panels 36
MODBLKS
enabling 45
preparing to enable 45
overview 35
RDDS
enabling 45
remove IMS.MODBLKS 64
requirements 41
resource definition data set (RDDS)
allocating 130
restrictions 42
synchronize MODBLKS and RDDS 64
updating
application program descriptor definitions 51
application program resources 51
database resources 51
descriptor definitions 51
dynamic resource definition (DRD) *(continued)*
updating *(continued)*
Fast Path routing code descriptor definitions 52
Fast Path routing code resource definitions 52
resource descriptor definitions 50
resource descriptor definitions in IMSplex 54
runtime resource definitions 50
transaction descriptor definitions 53
transaction resource definitions 53
DYNP= parameter 465

E

ECB (Even Control Block) 782
ECSA (extended common storage area) 153
EDIT= parameter 365, 380, 391, 420, 441, 693
EDTNAME= parameter 318
emergency restart
resuming an online reorganization for HALDBs 770
EMHB= parameter 466
EMHL= parameter 466
EMHQ (EMH queue)
disabling 620
EMHQ (EMH queue) structures
disabling 618
EMHQ= parameter 737
enqueue/dequeue tables
allocation, changing 108
specifying 108
system definition, in 108
entry sequenced data set (ESDS) 510
ENVIRON= parameter 466
EPCB= parameter 312, 467
ERPL (error parameter list trace table) 587
ERR (error trace table)
BPE trace table type 587
CQS trace table type 587
OM trace table type 587
RM trace table type 587
SCI trace table type 587
error parameter list trace table (ERPL) 587
error trace table (ERR)
BPE trace table type 587
CQS trace table type 587
OM trace table type 587
RM trace table type 587
SCI trace table type 587
ESDS (entry sequenced data set) 510
ETO
descriptors 689, 701
building in large systems 25
format 186
logon 188
MFS device 189
MSC 189
overrides 188

ETO *(continued)*
descriptors *(continued)*
statements, common format 690
user 190
DFSDSCMx PROCLIB member 689
DFSDSCTy PROCLIB member 701
IMSCTRL macro 186
IMSCTRL macro, ETOFEAT 340
including in IMS 186
keyword rules 187
PROCLIB member DFSDSCMx 689
PROCLIB member DFSDSCTy 701
terminal naming rules 299
ETO (Extended Terminal Option
descriptors
MFS 696
MFS devices
descriptors 696
ETO (Extended Terminal Option)
descriptors
MSC 696
logon 691
descriptor format 691
MSC (Multiple Systems Coupling)
descriptor format 696
user descriptors
format 697
ETO= parameter 467
ETOFEAT= parameter 340
EVAL= parameter 805
examples
coding conventions 299
CQSIPxxx member of the IMS PROCLIB data set 617
CQSSGxxx member of the IMS PROCLIB data set 627
CQSSLxxx sample member of the IMS PROCLIB data set 619
data sharing 275, 277, 278, 279
data sharing system configuration 275
DL/I application program macro statements 255
Dynamic Allocation macro 334
IMS DB/DC system definition 253
IMS DBCTL environment 280
MSC 271
OBJAVGSZ calculation 623
RSRCSTRUCTURE= parameter 626
SSN= parameter 616
startup procedures 510
structure recovery data set 510
system checkpoint data set 510
system definition 253
transaction grouping 84
exclusive access to databases 324
exclusive intent
with scheduling algorithm 94
EXCPVR= parameter 468
EXEC statement parameters
Fast Path 153
executing online utilities 174
execution parameters
categories and purpose 165
data communication 162, 165
database 150
database performance options 151
execution parameters *(continued)*
db buffer sizes 150
performance-related 162
PSB-related 165
PSBs 150
recovery-related 159, 162, 165
region control 162, 165
security-related 161, 162, 165
specifying 147
system
active regions 147
DL/I address space 147
IRLM options 147
nucleus identifier 147
performance options 147
z/OS options 147
TM buffer sizes 151
TM performance options 151
execution procedures, tailoring
for Fast Path 174
exit routines
DFSDFxxx 687
Large System Definition Sort/Split Input exit routine (DFSSS050) 24
preprocessor 19
running in DLISAS 73
EXITDEF statement
BPE types 605
CQS types 605
IMS Connect types 605
keywords 605
OM types 605
RM types 605
SCI types 605
EXITMBR parameter
BPE exit routine member of the IMS PROCLIB data set 587
CQS exit routine member of the IMS PROCLIB data set 587
HWS exit member of the IMS PROCLIB data set 587
ims_component 587
member_name 587
OM exit routine member of the IMS PROCLIB data set 587
RM exit routine member of the IMS PROCLIB data set 587
SCI exit routine member of the IMS PROCLIB data set 587
extended address volumes
allocating data sets 130
extended common storage area (ECSA) 153
extended storage
requesting that z/OS use 751
Extended Terminal Option (ETO)
descriptors
MFS device 696
MSC 696
logon
descriptor format 691
MFS devices
descriptors 696
MSC (Multiple Systems Coupling)
descriptor format 696
user descriptors
format 697
external subsystem
module placement 227
table placement 227
external subsystem attach facility
adding an SSM member to the IMS PROCLIB data set 221
defining external subsystems to IMS 221
defining language interface module 223
DFSLI macro 223
IMS.PROCLIB, description 221
language interface token 223
OPTIONS statement 224
specifying DB2 for z/OS groups 224
specifying trace options 224
SSM= EXEC parameter 221
SUBS= parameter 224
external subsystem module table 227
external trace environment 241
EXVR= parameter 468

F

F= parameter 647
fallback
large sequential data sets 142
Fast Path 158
and DBCTL system definition 27
BMP region parameters 157
buffer manager 25
buffers 27, 153
CCTL region parameters 157
control region EXEC statement parameters 153, 155
database buffers 153
DBF trace entries 756
declaring application programs 80
declaring program processing 80
DEDB buffers pools, defining 739, 740
defining a buffer manager 25
defining the system 25
defining transaction characteristics 83
dependent regions parameters 156
execution procedures 174
IMSFP procedure 578
including in DCCTL 25
macro statements
APPLCTN 308, 309
DATABASE 323
FPCTRL 337
required macros 25
routing codes, adding 80
system definition and 25
terminals
2740 terminal 391
traces 242
FAST= parameter 756
FBP= parameter 468
FDBR (Fast Database Recovery)
DB/DC 106
DBCTL 106
defining 104
enabling 106

FDBR (Fast Database Recovery) *(continued)*
enabling DB/DC subsystem for 106
enabling DBCTL subsystem for 106
IMS PROCLIB data set member, create 104
MODBLKS DD statement 43
regions
configuration 105
requirements
IMSplex 107
support for RDDS 41
FDR procedure
description 555
parameters
ARC= 452
ARMRST= 452, 454
BSIZ= 454
CSAPSB= 458
CSLG= 459
DBBF= 459
DBWP= 461
DESC= 464
DFRMBR= 468
DLIPSB= 464
DMB= 465
DPRTY= 465
FMTO= 468
IMSID= 472
IMSPLEX= 473
IRLMNM= 474
LGNR= 476
MCS= 479
PSB= 487
PSBW= 487
RGN= 492
RGSUF= 492
SOUT= 495
SPM= 496
SUF= 497
SVC2= 498
SYS1= 499
SYS2= 499
UHASH= 501
VSPEC= 501
WADS= 502
WKAP= 502
FDRMBR= parameter 468
FEAT= parameter 365, 391, 693
FES exit routine 318
FESEXIT= parameter 318
FESTIM= parameter 468
Field Edit exit routine 318
finance communication system 263
Finance work station 391
FIX= parameter 468
FIXBLOCK= parameter 742
FIXDATA= parameter 742
FIXINDEX= parameter 742
FMID= parameter 468
FMTO option
specify FMTO option 237
FMTO= parameter 468
FMTO=D parameter value 237
FORMAT= parameter 312
forms control
vertical 391
FP= parameter 468
FPATH keyword on APPLCTN macro 80
FPATH= parameter 308, 420
FPBOF= parameter 817
FPBUF= parameter 391, 817
FPCTRL
description 337
FPDSSIZE= parameter 725
FPOPN= parameter 469
FPRLM= parameter 469
FPTT= parameter 756
FPUTIL procedure 174
description 559
parameters
ALTID= 449
DBD= 459
DIRCA= 464
IMSID= 472
PRLD= 487
REST= 492
RGN= 492
SOUT= 495
SSM= 496
SYS2 499
FPWP= parameter 470
FRE= parameter 313, 470
FREs
controlling 74
estimating 74
MFS, and 75
parameter, BUFPOOLS macro 313
FUNCLV= parameter 817

G

GEN= parameter 470
GENSNAP= parameter 817
GPSB= parameter 308
GRAFFIN= parameter 657
GRESTAE= parameter 657
GRMESTAE= parameter 657
GRNAME= parameter 470
GRSNAME= parameter 471
GSGNAME= parameter 340, 471, 726

H

HALDB
DFSHALDB ddname 192
HALDB control statement 192
partitions
selective partition processing 192
selective partition processing 192
HALDB control statement 192
HALDB partition
system definition types
ALL 5
MODBLKS 5
HALDBs
master database
deleting dynamically 55
resuming an online reorganization during emergency restart 770
resuming an online reorganization during warm start 770
hang condition 782
HASH (hash trace table) 587
hash table slots, specifying 191
hash trace table (HASH) 587
HIDAM databases
declaring 76
macro instructions 323
High Availability Large Database (HALDB) partition
system definition types
ALL 5
MODBLKS 5
high-speed sequential control statements, specifying
examples 815
SETO 813
SETR 815
syntax 812
high-speed sequential processing
control statements 185
high-use program modules, making resident with DFSMPLxx 716
HIOP= parameter 471
HLQ parameter 821
HSB= parameter 340
HSBID= parameter 471
HSBMBR= parameter 471
HSn= parameter 744
HSSP
control statements 185
HWS
CM0ATOQ parameter 782
KEEPAV parameter 782
keyword parameters 782
parameters 787
statement 787
syntax 787
WARNINC parameter 782
WARNSOC parameter 782
HWS exit routine member of the IMS PROCLIB data set
EXITMBR parameter 587
HWSCFGxx
configuration
examples 799
examples 799
IMS DB support 802
Open Database 802
Universal driver support 802
HWSSOAP1 216
HWSUINIT
and user message exits 782
restrictions for EXIT= parameter 782

I

IC= parameter 814, 816
ICOMPT= parameter 380
ICON_NAME
and security configuration 212
IDC0= parameter 756
IDLE command 380
IEBGENER 242
IEBGENER utility 133
IEFBR14 utility 128
ILDS (Indirect List Data Set) 140
ILE (Indirect List Entry) 140

ILTMODE= parameter 726
IMPORT
  backing out resources created using IMPORT command 63, 64
IMPORT command 61, 62
IMS
  DB/DC 101
  designing 69
  procedure for IMS 101
  solutions for Java
    IMS PROCLIB data set members 715
    PROCLIB data set members 714
IMS abend search and notification
  setting up 245
IMS Connect
  authorized supervisor state 209
  BPEINI00 209
  configuration
    examples 799
    IMS DB support example 802
    Open Database example 802
    Universal driver support example 802
  configuration members 782
  configuration statement parameters 782
  configuration statement syntax 782
  configuration statements
    ADAPTER 783
    DATASTORE 784
    HWS 787
    IMSPLEX 788
    ODACCESS 789
    RUNOPTS 792
    TCPIP 793
  definition and tailoring 209
  DRDAPORT parameter 782
  entries for z/OS PPT 28
  environment 209
  examples 799
  in IMSplex environment 209
  invoking 209
  ODBM access configuration parameters 782
  ODBMAUTOCONN parameter 782
  Open Database
    configuration example 802
  sample configurations 782
  sample user exit list member of the IMS PROCLIB data set 605
  security for 212
  set up tracing 244
  SSL 213
    IMS Connect 213
  trace tables
    CMDT 587
    ENVT 587
    ERRV 587
    HWSI 587
    HWSN 587
    HWSO 587
    HWSW 587
    OMDR 587
    OTMA 587
    PCDR 587
    RCTR 587
IMS Connect *(continued)*
  trace tables *(continued)*
    TCPI 587
  Universal drivers
    configuration example 802
  XML conversion support 216
  XML-to-COBOL conversion support
    configuring 216
    example 216
    prerequisites 216
    restrictions 216
  XML-to-PL/I conversion support
    configuring 216
    example 216
    prerequisites 216
    restrictions 216
IMS control region
  entries for z/OS PPT 28
IMS Database Recovery Facility
  activating trace for 757
  tracing activity 757
IMS dump formatter 820
IMS Dump Formatter
  DD concatenations to update 240
  installing 240
IMS Extended Terminal Option Support
  ACF/VTAM terminals 186
  including in IMS 103
IMS procedure
  description 560
  parameters
    ALOT= 448
    AOI1= 449
    AOIS= 450
    APPC= 450
    APPCSE= 450
    APPLID1= 452
    APPLID2= 452
    APPLID3= 452
    ARC= 452
    ARMRST= 452
    ASOT= 452
    AUTO= 453
    BSIZ= 454
    CCTCVCAN= 454
    CIOP= 454
    CMDMCS= 455
    CPLOG= 457
    CRC= 457
    CSAPSB= 458
    DBBF= 459
    DBFP= 459
    DBFX= 460
    DBRCGRP= 461
    DBRCNM= 461
    DBWP= 461
    DC= 462
    DESC= 464
    DFSDF= 464
    DLINM= 464
    DLIPSB= 464
    DLQT= 465
    DMB= 465
    DMHVF= 465
    DPRTY= 465
    DSCT= 465
    DYNP= 465
IMS procedure *(continued)*
  parameters *(continued)*
    EMHB= 466
    EMHL= 466
    EPCB= 467
    ETO= 467
    EXVR= 468
    FBP= 468
    FDRMBR= 468
    FESTIM= 468
    FIX= 468
    FMTO= 468
    FP= 468
    FPOPN= 469
    FPRLM= 469
    FPWP= 470
    FRE= 470
    GRNAME= 470
    GRSNAME= 471
    HIOP= 471
    HSBID= 471
    HSBMBR= 471
    IMSGROUP= 472
    IMSID= 472
    IOVFI= 473
    IRLM= 473
    IRLMNM= 474
    ISIS= 474
    LGMSGSZ= 475
    LGNR= 476
    LHTS= 476
    LOGT= 476
    LSO= 477
    LTERM= 478
    LUMC= 478
    LUMP= 478
    MCS= 479
    MNPS= 479
    MNPSPW= 479
    PRDR= 486
    PRLD= 487
    PSB= 487
    PSBW= 487
    PST= 488
    PSWDC= 488
    QBUF= 489
    QBUFMAX= 489
    QBUFSZ= 489
    QTL= 490
    QTU= 490
    RCF= 490
    RCFTCB= 491
    RCLASS= 491
    RDMNM= 491
    READNUM= 491
    RECA= 491
    RECASZ= 491
    RES= 492
    RGN= 492
    RGSUF= 492
    RSRMBR= 492
    RVFY= 493
    SAV= 493
    SGN= 494
    SHAREDQ= 494
    SHMSGSZ= 495
    SOD= 495

IMS procedure *(continued)*
parameters *(continued)*
SOUT= 495
SPM= 496
SRCH= 496
SSM= 496
SUF= 497
SVC2= 498
SVSODR= 498
SYS= 499
SYS1= 499
SYS2= 499
TCORACF= 499
TRACK= 500
TRN= 500
TSR= 500
UHASH= 501
UHTS= 501
USERVAR= 501
VAUT= 501
VSPEC= 501
WADS= 502
WKAP= 502
YEAR4= 502
sample 560
storing 445
z/OS, and 183
IMS PROCLIB data set
assigning procedure names 174
controlling 174
controlling modifications 174
DFSPBxxx 174
initializing 174
tailoring 174
IMS PROCLIB data set members
alphabetic listing 587
DBFMSDBx 646
DFS62DTx 647
DFSDFRxx 702
DFSFIXnn 704
DFSINTxx 714
DFSORSxx 718
DFSPBDBC 719
DFSPBDCC 719
DFSPBIMS 720
DFSSQxxx 736
generated 171
Syntax Checker 291
tailoring 171
IMS Release and Control Region panel
panel field descriptions 286
IMS Release panel
panel field descriptions 287
IMS setup recommendations
FMTO option 237
SYSDUMP DD 237
IMS Syntax Checker 174
IMS transactions
defining 81
TRANSACT macro, using 81
IMS.ACBLIB as a staging library 111
IMS.ADFSRESL
XRF impact to 137
IMS.ADFSTLIB
XRF impact to 137
IMS.FORMAT
XRF impact to 137
IMS.FORMAT as a staging library 111
IMS.JOBS data set 174
IMS.LGMSG data set 367
IMS.MODBLKS
XRF impact to 137
IMS.MODSTAT, used for active library status 111
IMS.OBJDSET
XRF impact to 137
IMS.OPTIONS
XRF impact to 137
IMS.PROCLIB
CSL manager 205
DFSCGxxx member 205
DFSDFxxx member 205
DFSPBxxx IMS PROCLIB data set member 205
DFSVSMxxx IMS PROCLIB data set member 205
members that support a CSL 205
XRF impact to 137
IMS.PROCLIB members
DFSDCxxx 657
DFSHSBxx 709
DFSPBxxx 718
DFSRSRxx 720
DFSSPMxx 731
DFSVSMxx 738
IMS.QBLKS data set 367
IMS.REFERAL
XRF impact to 137
IMS.SDFSMAC
XRF impact to 137
IMS.SDFSRESL 23
concatenating for interactive dump 240
IMS.SHMSG data set 367
IMS.TFORMAT
XRF impact to 137
IMS= parameter 472
IMSBATCH procedure 520, 547, 548, 571, 581, 584
parameters
ALTID= 449
APARM= 450
CKPTID= 454
CPUTIME= 457
DIRCA= 464
IMSID= 472
IMSPLEX= 473
IN= 473
LOCKMAX= 476
MBR= 479
NBA= 480
OBA= 480
OPT= 481
PAGES= 484
PARDLI= 484
PREINIT= 486
PRLD= 487
PSB= 487
RGN= 492
SOUT= 495
SPIE= 495
SSM= 496
STIMER=, message-driven programs 497
IMSBATCH procedure *(continued)*
parameters *(continued)*
SYS2= 499
TEST= 499
IMSCOBGO procedure
description 572
parameters
BKO= 453
BUF= 454
CKPTID= 454
DBRC= 460
EXCPVR= 468
FMTO= 468
IMSID= 472
IMSPLEX= 473
IOB= 473
IRLM= 473
IRLMNM= 474
LOGA= 476
LOGT= 476
MBR= 479
MON= 480
PRLD= 487
PSB= 487
RGN= 492
RST= 492
SOUT= 495
SPIE= 495
SRCH= 496
SWAP= 498
SYS2= 499
TEST= 499
IMSCOBOL procedure
parameters
MBR= 479
SOUT= 495
SYS2= 499
IMSCTF macro
description 69, 337
parameters 338
APNDG= 338
CORE= 338
CPLOG= 338
LOG= 338
PRDR= 339
RDS= 339
SVCNO= 339
setting checkpoints 69
syntax diagram 337
IMSCTRL macro 133
defining BMP regions 69
defining message regions 69
defining regions
BMP 69
message 69
description 69, 340
parameters 341
CMDCHAR= 340
DBRC= 159, 340
DBRCNM= 340
DCLWA= 340
DESC= 340
DLINM= 340
ETOFEAT= 340
GSGNAME= 340
HSB= 340
IMSID= 340

IMSCTRL macro *(continued)*
parameters *(continued)*
IRLM= 340
IRLMNM= 340
MAXCLAS= 340
MAXIO= 340
MAXREGN= 340
MCS= 340
MODBLKS= 340
MSVERIFY 340
MSVID= 340
NAMECHK= 340
NUCLEUS 340
ON-LINE 340
RSRFEAT= 340
SYSTEM= 340
selecting a lock manager 70
syntax diagram 340
use without specifying MODBLKS 340
IMSDALOC procedure, process 577
IMSFP procedure 174
description 578
parameters
ALOT= 448
ALTID= 449
APARM= 450
CPUTIME= 457
DBLDL= 460
DIRCA= 464
IMSID= 472
LOCKMAX= 476
MBR= 479
NBA= 480
OPT= 481
PREINIT= 486
PRLD= 487
PSB= 487
RGN= 492
SOD= 495
SOUT= 495
SSM= 496
STIMER=, message-driven programs 497
SYS2= 499
TLIM= 499
IMSGEN macro 101
assembler and binder options 355
COMM macro, and 386
communication options parameters
MFSTEST= 351
SYSMSG= 351
data set option parameters
MACLIB= 351
MACSYS= 351
MODGEN= 351
NODE= 351
OBJDSET= 351
PROCLIB= 351
SCEERUN= 351
UMAC0= 351
UMACx= 351
USERLIB= 351
general communication options 351
IMS data set options 351
JCL statement parameters
JCL= 351
IMSGEN macro *(continued)*
JCL statement parameters *(continued)*
JOBCTL= 351
MFSDFMT= 351
ONEJOB= 351
PRTY= 351
SCL= 351
UJCLx= 351
JCL statements 351
MFS 142
parameters 351
ASM= 351
ASMPRT= 351
LKPRT= 351
LKRGN= 351
LKSIZE= 351
SUFFIX= 351
TERM= 351
UPDTPRT= 351
relationship to COMM and SECURITY 386
sample IMSGEN macro statement 351
SECURITY macro, and 386
security options 351
security options parameters
SECCNT= 351
syntax diagram 351
IMSGROUP= parameter 472
IMSID 101
IMSID= parameter 340, 472
IMSMSG job 579
IMSplex
FDBR
requirements 107
keyword parameters 782
managing serialized programs 626
MEMBER parameter 782
shared queues
managing serialized programs 626
TMEMBER parameter 782
IMSPLEX
parameters 788
statement 788
syntax 788
IMSPLEX macro
data set option parameters 351
IMSPLEX= parameter 473, 649
IMSPLI procedure
DD statements 580
parameters 580
MBR= 479
SPIE= 495
SYS2= 499
IMSPLIGO procedure
DD statements 581
description 581
parameters 581
BKO= 453
BUF= 454
CKPTID= 454
DBRC= 460
EXCPVR= 468
FMTO= 468
IMSID= 472
IMSPLEX= 473
IMSPLIGO procedure *(continued)*
parameters *(continued)*
IOB= 473
IRLM= 473
IRLMNM= 474
LOGA= 476
LOGT= 476
MBR= 479
MON= 480
PRLD= 487
PSB= 487
RGN= 492
RST= 492
SOUT= 495
SPIE= 495
SRCH= 496
SWAP= 498
SYS2= 499
TEST= 499
IMSRDR procedure
DD statement 584
description 584
parameters 584
CLASS= 455
MBR= 479
SYS2= 499
storing 445
z/OS, and 183
IMSWT= parameter 657
IN= parameter 473
incore trace tables
formatting 218
Indirect List Data Set (ILDS)
HALDB
partitions 140
pointers
direct 140
indirect 140
Indirect List Entry (ILE) 140
INDOUBT EEQEs
/DBRECOVERY command, preventing for 771
initializing data sets 136
initializing IMS system data sets 111
INQ= parameter 420
INQUIRY= parameter 384, 420
installation
multiple copies of IMS
different release levels 103
same release level and type 101
steps for a full installation 1
integrity
lock manager 70
Interactive Dump Formatter
making IMS.SDFSRESL available 240
interactive problem control system 820
Interactive Problem Control System (IPCS)
sample control statements 821
starting with IMS Dump Formatter 821
interface parameter trace table (INTP) 587
interface trace table (INTF)
CQS trace table type 587
SCI trace table type 587

INTF (interface trace table)
CQS trace table type 587
SCI trace table type 587
INTP (interface parameter trace table) 587
invoking
IMS Connect 209
IOB= parameter 473
IOVF control intervals 473
IOVFI= parameter 473
IPCS (Interactive Problem Control System)
sample control statements 821
starting with IMS Dump Formatter 821
IQMRH0 126
IRLM
coupling facility structure names 765
DEADLOK parameter 462
entries for z/OS PPT 28
execution parameters 147
how it uses SDUMP 821
IMSCTRL macro
IRLM parameter 340
IRLMNM parameter 340
LOCKTIME parameter 768
performance and DEADLOK parameter 463
selecting a lock manager 70
startup procedure 554
IRLM= parameter 340, 473
IRLMGRP= parameter 474
IRLMID= parameter 474
IRLMNM= parameter 340, 474
ISC surveillance link 391, 410
ISIS= parameter 474
ISPF
Syntax Checker panels 284
ISSUE681= parameter 756
ISSUE840= parameter 756

J

Java batch processing (JBP) regions
DB2 for z/OS access
configuring 228
Java message processing (JMP) regions
DB2 for z/OS access
configuring 228
JBP (Java batch processing) regions
DB2 for z/OS access
configuring 228
JCL
to print RECORDER output 220
JCL values 209
BPECFG 209
HWSCFG 209
RGN 209
SOUT 209
JCL= parameter 351
JCLIN process of system definition 28
JDBC drivers
DB2 for z/OS JDBC/SQLJ 1.2 driver 228
DB2 for z/OS JDBC/SQLJ 2.0 driver 228
JDBC drivers *(continued)*
DB2 for z/OS Universal JDBC driver 228
JMP (Java message processing) regions
DB2 for z/OS access
configuring 228
job accounting 351
JOBCTL= parameter 351
JVM options 714, 715
JVMOPMAS=
parameter description 475
JVMOPWKR=
parameter description 475
JVMOPWKR= parameter 475
JVMPOPMAS= parameter 475

K

KeepAlive function of TCP/IP 782
keyboard shortcuts xi
KEYEVENT= parameter 710
Keyword Display panel
action pull-down options 292
display options 288
inserting keywords 294
interrupting processing 294
panel field descriptions 288, 289
saving processed members 295
keywords
MAXREGN 133

L

LANG parameter
member of the IMS PROCLIB data set 587
LANG= parameter 308
language interface module 223
large sequential data sets
defining 141
fallback from 142
large system definition
additional data sets 24
configuration 17
description 24
ETO descriptors, building 25
LGEN parameter 24
LGENIN data set 24
LGENOUT data set 24
online change 24
preprocessor 18
stage 1 processing 24
storage requirements 19, 24
system definition 19
Large System Definition Sort/Split Input exit routine (DFSSS050) 24
large system generation 24
LATC (latch trace table) 587
LATC= parameter 756
latch trace table (LATC) 587
LBUFMAX= parameter 726
legal notices
notices 823
trademarks 823
LGDK device 339
LGEN subparameter
checking keywords 15
LGEN/MODBLKS system definition
online change 24
LGENIN data set 24
LGENOUT data set 24
LGMSGSZ= parameter 475
LGNR= parameter 476
LHTS= parameter 476
LIBATCH procedure
parameters
PSB= 487
libraries
active 111
inactive 111
maintenance 111
staging 111
limit count scheduling 86
limit priority scheduling 86
LINE macro
parameters 363
ADDR= 363
BACKUP= 363
BUFSIZE= 363
syntax diagram 363
SYSIN Local Card Reader 363
LINEGRP macro
description 365
parameters 365
BACKUP= 365
DDNAME= 365
EDIT= 365
FEAT= 365
UNITYPE= 365
syntax 365
LINEGRP macro statement 133
LINKLIST 227
LIT (Language Interface Token) 223
LIT= parameter 223
LKASID 739
LKPRT= parameter 351
LKRGN= parameter 351
LKSIZE= parameter 351
LMODE= parameter 726
LNK= parameter 711
local option client communications
configuring security 212
local SYSOUT line groups
macro statements 258
lock manager
selecting 70
lock timeout function, IRLM 768
LOCK= parameter 757
LOCKMAX= parameter 476
LOCKSEC= parameter 657
LOCKTAB= parameter 476
Log Archive utility 125
LOG= parameter 338, 711, 761
LOGA= parameter 476
LOGCOUNT= parameter 726
LOGEDIT statement 762
logging
integrity 125
logical record length 615
logical term name 380
logon descriptors
format 689, 701

logon descriptors *(continued)*
parameters 689, 701
syntax 689, 701
LOGR policy
CQS
MAXBUFSIZE specification 202
MAXBUFSIZE specification for CQS 202
LOGT= parameter 476
long busy handling function
enabling 768
LRECL 615
LRTT= parameter 757
LSO (local storage option)
DL/I address space, specifying 71
PSB pools, defining 71
storage considerations 71
LSO= parameter 477
LTE= parameter 477
LTERM
specifying 380
LTERM= parameter 391, 478
LU 6 terminal
TERMINAL macro statement, specifying 391
VTAM devices 390
LUMC= parameter 478
LUMI= parameter 757
LUMP= parameter 478
LUNAME= parameter 648

M

MACLIB= parameter 351
macro keywords
EDIT on TRANSACT macro 83
INQUIRY on TRANSACT macro 83
MSGTYPE on TRANSACT macro 83
PROCLIM on TRANSACT macro 83
PRTY on TRANSACT macro 86
macro statements
alphabetic listing 299
APPLCTN
online programs, declaring 76
application program 254
BUFPOOLS
MFS pool size. controlling 74
checking 23
coding conventions 299
COMM 259
data communication 258
database 254
DATABASE
data sharing 79
online databases, declaring 76
finance communication systems 263
FPCTRL 337
IMSCTF 337
IMSCTRL
checkpoint frequency, setting 69
data sharing 79
description 340
LINEGRP 365
local SYSOUT line groups 258
maximum occurrences 299
MSGQUEUE 367
MSLINK 372
macro statements *(continued)*
MSNAME 375
MSPLINK 376
NAME 380
NTO devices 266
relationships between COMM, IMSGEN, and SECURITY 386
resource naming rules 299
RTCODE 384
SECURITY 386
SUBPOOL 390
system configuration 270
system configuration macros 254
BUFPOOLS 69
IMSCTF 69
IMSCTRL 69
system configuration macros, use of 69
TERMINAL
2740 terminal 391
2741 terminal 391
2780 terminal 391
3270 terminal 391
3600 work station 391
Finance work station 391
LU 6 391
SLU 1 391
SLU 2 391
SLU P 391
SPOOL 391
System/3 terminal 391
System/7 terminal 391
TRANSACT
description 420
IMS transactions, defining 81
TYPE 441
VTAM 259, 261, 265, 267, 268
VTAMPOOL 443
macros
DFSMDA 125
TERMINAL 133
type of system definition required for change 5
MACSYS= parameter 351
maintenance
SMP/E 28
with online change 111
mandatory data sets
replication 137
shared 137
master terminal
configuring 98
devices, choosing 98
secondary logging 98
secondary, commands that can be copied to 318
specifying 98, 380
master trace table Size, z/OS 237
MAXBUFSIZE parameter
CQS recommendation 202
LOGR policy for CQS 202
MAXCLAS= parameter 340
MAXCSA= parameter 478
MAXFILEPROC parameter, UNIX System Services 782
MAXIO= parameter 340
MAXPST= parameter 479
MAXREGN keyword 133
MAXREGN= parameter 340
MAXRGN= parameter 420
MAXSB= parameter 752
MAXTHRD= parameter 817
MAXUSRS= parameter 479
MBR= parameter 479
MCS= parameter 340, 479
members of the IMS PROCLIB data set
Base Primitive Environment (BPE)
trace table types 587
LANG parameter 587
sharing configurations 587
TRCLEV 587
message
batching 86
CQS0268W 624
migrating 126
queue
data set 126
testing users 126
scheduling 83
message classes
assigning 85
priorities 86
message format pool
size, estimating 74
size, specifying 151
Message Format Service (MFS)
pool space
BUFPOOLS macro 74
message prefix
sizes
Version 10 367
Version 8 367
Version 9 367
Message Processing Region
data communication EXEC parameters 162, 165
parameters, categories and purposes 162
performance-related parameters 162
PSB-related EXEC parameters 162, 165
recovery-related parameters 159, 162, 165
region-control EXEC parameters 162, 165
security-related EXEC parameters 161, 162, 165
message queue
data set allocation 367
data sets
IMS.LGMSG 367
IMS.QBLKS 367
IMS.SHMSG 367
message queue data sets
restrictions in XRF 126
message regions
choosing number 69
messages
processing limits 87
MFS
buffer pool use 75
BUFPOOLS macro 142
COMM macro 142
concatenating format libraries 145

MFS *(continued)*
device descriptor format 696
device descriptors 189
displaying buffer pool counters 145
FREs 75
IMSGEN macro 142
INDEX directory 145
MSGQUEUE macro 142
pool space, allocating 74
pool space, defining 74
system definition and programming considerations 142
TERMINAL 142
TYPE macro 142
MFS device descriptors
format 689, 701
parameters 689, 701
syntax 689, 701
MFSDFMT= parameter 351
MFSEXIT= parameter 318
MFSTEST (Message Format Service test facility) 319, 361
MFSTEST= parameter 351
MILEINTV= parameter 726
MINTHRD= parameter 817
MNPS= parameter 479, 712
MNPSPW= parameter 479, 712
MOD= parameter 480
MODBLKS 43
in a DRD environment 41
use IMSCTRL without specifying 340, 350
MODBLKS resources
dynamic definition
enabling 45
preparing to enable 45
MODBLKS type system definition 2
MODBLKS= parameter 340
MODE= parameter 420, 648, 724, 728, 730, 763
MODEL= parameter 391
MODETBL= parameter 372, 391
description 376
displaying 376
displaying name 372
MODGEN= parameter 351
MODSTAT= parameter 730
MODSTAT2= parameter 730
module preload
description 716
modules
DFSMRC20 (resource cleanup) 103
DFSMRCL0 (resource cleanup) 103
DFSVC000 101
DFSVNUCx 101
suffix rules 101
MON= parameter 480
monitoring
message queue users 126
MRQPSBN= parameter 367
MSC
descriptors 189
example 271
macro statements, preparing 101
macros
MSLINK 101
MSNAME 101
MSC *(continued)*
macros *(continued)*
MSPLINK 101
NAME 101
MSC (multiple systems coupling)
network 101
MSC (Multiple Systems Coupling)
descriptor 696
specifying different buffer size 376
MSC descriptors
format 689, 701
parameters 689, 701
syntax 689, 701
MSCSEC= parameter 657
MSCT= parameter 757
MSCVGR= parameter 657
MSDB
DBFMSDBx member 646
deleting dynamically 55
MSDB (main storage database)
dynamic resource definition 47
loading 153
MSDB= parameter 480
MSDBABND= parameter 647
MSG= parameter 480
MSGDEL= parameter 390, 391
MSGQ= parameter 737
MSGQUEUE macro
description 367
MFS 142
parameters 368
BUFFERS= 367
DSET= 367
MRQPSBN= 367
RECLNG= 367
SHUTDWN= 367
syntax diagram 367
MSGTYPE= parameter 420
MSLINK macro
description 372
parameters 372
BACKUP= 372
MODETBL= 372
MSPLINK= 372
OPTIONS= 372
PARTNER= 372
syntax diagram 372
MSNAME macro
description 375
label field 375
parameters 375
SYSID= 375
syntax diagram 375
MSPLINK macro
description 376
parameters 376
ADDR= 376
BACKUP= 376
BUFSIZE= 376
DDNAME= 376
MODETBL= 376
NAME= 376
OPTIONS= 376
SESSION= 376
TYPE= 376
syntax diagram 376
MSPLINK= parameter
assigning logical links 372
MSVERIFY type system definition 2
MSVERIFY= parameter 340
MSVID= parameter 340
MTOID= parameter 726
MTOUSID= parameter 657
multi-area structures
defining DEDB buffer pools 740
multiple copies of IMS
running on one operating system 103
multiple IMS systems, module suffix rules 101
multiple message queues
long 126
short 126
Multiple Systems Coupling (MSC)
descriptor 696
specifying different buffer size 376

N

NAME macro
description 380
parameters 380
COMPT= 380
EDIT= 380
ICOMPT= 380
lterm 380
OUTPUT= 380
remote Item name 380
syntax diagram 380
NAME= parameter 390, 391
changing VTAM node name 376
description 376
NAMECHK= parameter 340
NBA= parameter 480
NBRSEGS= parameter 647
NHTS= parameter 480
NLXB= parameter 480
NODE= keyword 101
NODE= parameter 351
NOIC= parameter 813
NOLKASID 739
nonswitched communication lines
described by LINE macro 363
NOPROCH= parameter 813
normal priority for scheduling 86
NTO device
macro statements 266
TERMINAL macro statement 391
NUC= parameter 708
nucleus 23
when generated 4
nucleus definition 349
NUCLEUS type system definition 2
NUCLEUS= parameter 340

O

OBA= parameter 480
OBJDSET= parameter 351
OCMD= parameter 757
ODACCESS
parameters 789

ODACCESS *(continued)*
statement 789
syntax 789
ODBA (Open Database Access)
defining PSB names 79
ODBA= parameter 757
ODBASE= parameter 481
ODBM
user exit list member of the IMS PROCLIB data set 605
ODBM (Open Database Manager)
configuration member of the IMS PROCLIB data set 205
configuring, CSLDCxxx member of the IMS PROCLIB data set 628
CSLDCxxx member of the IMS PROCLIB data set 628
execution parameters 515
initialization member of the IMS PROCLIB data set 205
initialization parameters member of the IMS PROCLIB data set 633
PROCLIB members
CSLDCxxx 628
sample startup procedure 515
user exit list member of the IMS PROCLIB data set 634
ODBM user exit list member of the IMS PROCLIB data set 605
ODBMCFG= parameter 481
ODBMINIT= parameter 481
ODBMNAME= parameter 481
offloading trace data set 242
OLC= parameter 649
OLCSTAT= parameter 649
OLDS (online log data set)
allocating 119
block sizes 120
ddnames requirements 120
dynamic allocation 120
formatting 122
OLDSDEF control statement 120
OLDS= parameter 762
OLDSDEF control statement 120
OLDSDEF statement 762
OM exit routine member of the IMS PROCLIB data set
EXITMBR parameter 587
OM trace table types
* (asterisk) 587
CSL 587
ERR 587
OM 587
PLEX 587
recommendations 587
OMPROC= parameter 649
ON-LINE type system definition 2
ON-LINE= parameter 340
ONEJOB= parameter 351
online change
ACBLIB members
defining with DFSMDA member 118
defining with IMS procedure 118
large system definition 24
system data sets 111
Online Change data sets 111
online databases, declaring 76
online ddnames 115
online DEDB utility region parameters 158
online programs
characteristics 76
declaring 76
online system data sets
dependencies 115
list of 115
online systems
configuring alternative IMS online systems 23
SUFFIX= keyword to specify alternate configurations 23
Open Database Access (ODBA)
defining PSB names 79
Open Database Manager
CSLDCxxx syntax 629
Open Database Manager (ODBM)
configuration member of the IMS PROCLIB data set 205
configuring, CSLDCxxx member of the IMS PROCLIB data set 628
CSLDCxxx member of the IMS PROCLIB data set 628
execution parameters 515
initialization member of the IMS PROCLIB data set 205
initialization parameters member of the IMS PROCLIB data set 633
PROCLIB members
CSLDCxxx 628
sample startup procedure 515
user exit list member of the IMS PROCLIB data set 634
Open Transaction Manager Access (OTMA)
client attributes, global 778
client descriptor 771
clients
defaults 778
default client parameters 778
descriptors
client descriptor 771
DFSOTMA descriptor 778
message flood protection
client descriptor 771
wait-syncpoint timeout values
client descriptor 771
Operations Manager
set up tracing 244
Operations Manager (OM)
execution parameters 517
initialization parameters member of the IMS PROCLIB data set 636
Resource Manager (RM)
sample user exit list member of the IMS PROCLIB data set 205
sample startup procedure 517
user exit list member of the IMS PROCLIB data set 205, 637
OPT= parameter 481
optional replication data sets 137
options
AUTOSCH 133
OPTIONS statement 224
OPTIONS= parameter 319, 391, 441, 694, 699
changing ASR option 372
description 372, 376
displaying 376
displaying ASR option 372
overriding 375
ORS= parameter 481
ORSMBR= parameter 482
ORTT= parameter 757
OSAM
buffer pools, defining 179, 181, 748
dynamically 182
buffers
adjusting 179
coupling facility 765, 766
I/O errors, tracing 755
sequential buffering 182
subpools
defining 182
subpools, defining 748
OSAM (overflow sequential access method)
allocating data sets 128
reallocating data sets 128
sample OSAM data set allocation JCL 128
OTHR= parameter 482
OTMA
client descriptors 771
control flow, tracing 758
DCC procedure 532
destination descriptor 775
destination descriptors 771
DFSYDTx PROCLIB member 771
generic resource (GRNAME) parameter 470
message prefix sizes 367
PROCLIB member DFSYDTx 771
OTMA (Open Transaction Manager Access)
client attributes, global 778
client descriptor 771
clients
defaults 778
default client parameters 778
descriptors
client descriptor 771
DFSOTMA descriptor 778
DFSOTMA descriptor 778
message flood protection
client descriptor 771
wait-syncpoint timeout values
client descriptor 771
OTMA destination descriptor 775
OTMA= parameter 482
OTMANM= parameter 483
OTMASE= parameter 483
OTMASP= parameter 484
OTMT= parameter 758
OUT= parameter 484
OUTBND= parameter 649, 657
OUTBUF= parameter 391, 694
output data sets, creating 243
OUTPUT= parameter 380
overflow sequential access method. 128
OVLA= parameter 484

P

PAGES= parameter 484
parallel database open, disabling 764
parallel session support 443
parameter
  performance-related 151
Parameter Syntax Checker panel
  panel field descriptions 297
parameters 221
  ABND= 448
  ACBIN64= 183
  ACCESS= 324
  ACTIV= 805
  ADDR= 363, 376
  AL= 815
  ALARM= 710
  ALOT= 448, 692
  ALTID= 449
  ALTRESL 821
  AOEXIT= 316
  AOI= 420
  AOI1= 449
  AOIP= 449
  AOIS= 450
  AOS= 657
  APAR= 450
  APARM= 450
  APNDG= 338
  APPC= 450
  APPCASY= 657
  APPCIOT= 657
  APPCRCV= 657
  APPCSE= 450
  APPLFE= 451
  APPLID= 317
  APPLID1= 452
  APPLID2= 452
  APPLID3= 452
  ARC= 159, 452
  ARCHDEF= 723
  ARCHIVE= 724
  ARMRST= 452
  ASM= 351
  ASMPRT= 351
  ASOT= 452, 693, 698
  ASSNCHANGE= 657
  AUTHLOG= 657
  AUTLCHANGE= 657
  AUTLDESC= 698
  AUTLGN= 698
  AUTLID= 698
  AUTLMOD= 698
  AUTO= 453, 710
  BACKUP= 363, 365, 372, 376, 391, 441, 693
  BHBUF= 805
  BKO= 453
  BLOCK= 724, 728, 730
  BMPUSID= 657
  BPECFG= 453
  BSIZ= 454
  BUF= 454
  BUFFERS= 367
  BUFNO= 763
  BUFSETS= 805
  BUFSIZE= 363, 376, 391
  CCTCVCAN= 454
parameters *(continued)*
  CFIRLM= 765
  CFNAMES= 765
  CFOSAM= 765
  CFVSAM= 765
  CIOP= 454
  CKPTID= 454
  CL1=,CL2=CL3=,CL4= 455
  CLASS= 455
  CM0ATOQ 782
  CMDMCS= 455
  CMDP= 456
  CMDSEC= 649
  CNBA= 817
  CODE= 384, 420
  COMPT= 380, 391
  COMPT1= 391
  COMPT2= 391
  COMPT3= 391
  COMPT4= 391
  COMPTn= 693
  CONVTYPE= 648
  COPYLOG= 317
  CORE= 338, 457
  CPLOG= 338, 457
  CPUTIME= 457
  CQS= 736
  CQSSSN= 737
  CRC= 457
  CSAPSB= 458
  CSLG= 459
  CSLT= 754
  DB= 805, 813, 815
  DBBF= 459
  DBCTLID= 817
  DBD= 325, 459, 647, 749
  DBFP= 459
  DBFX= 460
  DBLDL= 460
  DBRC= 340, 460
  DBRCGRP= 461
  DBRCNM= 159, 340, 461
  DBRSE= 461
  DBWP= 461
  DC= 462
  DCLWA= 340, 420
  DD= 805
  DDNAME= 365, 376, 761, 817
  DEDBMAS= 740
  DEFERFIX= 710
  DEGRADE= 763
  DESC= 340, 464
  DIAG= 755
  DIRCA= 464
  DISP= 755
  DL/I= 755
  DLIDSIZE= 725
  DLINM= 340, 464
  DLIPSB= 464
  DLOG= 755
  DLQT= 465
  DMB= 312, 465
  DMHVF= 465
  DPRTY= 465
  DSCT= 465
  DSETS= 367
  DSN= 727, 729
parameters *(continued)*
  DSNAME= 817
  DSSIZE= 725
  DUMP= 755
  DUMPIO= 755
  DYNP= 465
  EDIT= 365, 380, 391, 420, 441, 693
  EDTNAME= 318
  EMHB= 466
  EMHL= 466
  EMHQ= 737
  ENVIRON= 466
  EPCB= 312, 467
  ETO= 467
  ETOFEAT= 340
  EVAL= 805
  EXCPVR= 468
  EXIT= 782
  EXVR= 468
  F= 647
  FAST= 756
  FBP= 468
  FDRMBR= 468
  FEAT= 365, 391, 693
  FESEXIT= 318
  FESTIM= 468
  FIX= 468
  FIXBLOCK= 742
  FIXDATA= 742
  FIXINDEX= 742
  FMID= 468
  FMTO= 468
  FORMAT= 312
  FP= 468
  FPATH= 308, 420
  FPBOF= 817
  FPBUF= 391, 817
  FPDSSIZE= 725
  FPOPN= 469
  FPRLM= 469
  FPTT= 756
  FPWP= 470
  FRE= 313, 470
  FUNCLV= 817
  GEN= 470
  GENSNAP= 817
  GPSB= 308
  GRAFFIN= 657
  GRESTAE= 657
  GRMESTAE= 657
  GRNAME= 470
  GRSNAME= 471
  GSGNAME= 340, 471, 726
  HIOP= 471
  HLQ 821
  HSB= 340
  HSBID= 471
  HSBMBR= 471
  HSn= 744
  IC= 814, 816
  ICOMPT= 380
  IDC0= 756
  ILTMODE= 726
  IMS= 472
  IMSGROUP= 472
  IMSID= 340, 472
  IMSPLEX= 351, 473, 649

parameters *(continued)*
IMSWT= 657
IN= 473
INQ= 420
INQUIRY= 384, 420
IOB= 473
IOVFI= 473
IRLM= 340, 473
IRLMGRP= 474
IRLMID= 474
IRLMNM= 340, 474
ISIS= 474
ISSUE681= 756
ISSUE840= 756
JCL= 351
JOBCTL= 351
JVMOPWKR= 475
JVMPOPMAS= 475
KEEPAV 782
KEYEVENT= 710
LANG= 308
LATC= 756
LBUFMAX= 726
LGMSGSZ= 475
LGNR= 476
LHTS= 476
LIT= 223
LKPRT= 351
LKRGN= 351
LKSIZE= 351
LMODE= 726
LNK= 711
LOCK= 757
LOCKMAX= 476
LOCKSEC= 657
LOCKTAB= 476
LOG= 338, 711, 761
LOGA= 476
LOGCOUNT= 726
LOGT= 476
LRTT= 757
LSO= 477
LTE= 477
LTERM= 391, 478
LUMC= 478
LUMI= 757
LUMP= 478
LUNAME= 648
MACLIB=
IMSPLEX= parameter 351
MACSYS= 351
MAXCLAS= 340
MAXCSA= 478
MAXIO= 340
MAXPST= 479
MAXREGN= 340
MAXRGN= 420
MAXSB= 752
MAXTHRD= 817
MAXUSRS= 479
MBR= 479
MCS= 340, 479
MFSDFMT= 351
MFSEXIT= 318
MFSTEST= 351
MILEINTV= 726
MINTHRD= 817

parameters *(continued)*
MNPS= 479, 712
MNPSPW= 479, 712
MOD= 480
MODBLKS= 340
MODE= 420, 648, 724, 728, 730, 763
MODEL= 391
MODETBL= 372, 376, 391
MODGEN= 351
MODSTAT= 730
MODSTAT2= 730
MON= 480
MRQPSBN= 367
MSCSEC= 657
MSCT= 757
MSCVGR= 657
MSDB= 480
MSDBABND= 647
MSG= 480
MSGDEL= 390, 391
MSGQ= 737
MSGTYPE= 420
MSPLINK= 372
MSVERIFY= 340
MSVID= 340
MTOID= 726
MTOUSID= 657
NAME= 376, 390, 391
NAMECHK= 340
NBA= 480
NBRSEGS= 647
NHTS= 480
NLXB= 480
no longer supported
AOEXIT= 316
NODE= 351
NOIC= 813
NOPROCH= 813
NUC= 708
NUCLEUS= 340
OBA= 480
OBJDSET= 351
obsolete
PASSWD 484
OCMD= 757
ODBA= 757
ODBASE= 481
ODBMCFG= 481
ODBMINIT= 481
ODBMNAME= 481
OLC= 649
OLCSTAT= 649
OLDS= 762
OMPROC= 649
ON-LINE= 340
ONEJOB= 351
OPT= 481
OPTIONS= 319, 372, 376, 391, 441, 694, 699
ORS= 481
ORSMBR= 482
ORTT= 757
OTHR= 482
OTMA= 482
OTMANM= 483
OTMASE= 483
OTMASP= 484

parameters *(continued)*
OTMT= 758
OUT= 484
OUTBND= 649, 657
OUTBUF= 391, 694
OUTPUT= 380
OVLA= 484
PAGES= 484
PARDLI= 484
PARLIM= 420
PARTNER= 372
PASSWD= 320, 484
PASSWD1= 484
PCB= 485, 805, 813, 815
PGMTYPE= 309
PGPROT= 485
PIINCR= 485
PIMAX= 485
PMTO= 657
PMTO1-8= 657
PMTOG= 657
POOLID= 742
PRDR= 339, 486
PREINIT= 486
PREMSG= 486
PRICOUNT= 723, 728, 729
PRIUNIT= 723, 727, 729
PRLD= 487
PROCLIB= 351
PROCLIM= 420
PRTY= 351, 420
PSB= 310, 313, 487, 805
PSBW= 313, 487
PSDEPAB= 712
PST= 488
PSTIMER= 657
PSWDC= 488
PTRSIZE= 391, 694
PU= 391
PWFI= 488
QBUF= 489
QBUFHITH= 489
QBUFLWTH= 489
QBUFMAX= 489
QBUFSZ= 489
QMGR= 758
QTL= 159, 490
QTU= 159, 490
RC= 490
RCF= 490
RCFTCB= 491
RCLASS= 386, 491, 657
RCVYCONV= 657, 699
RCVYFP= 666, 699
RCVYRESP= 657
RCVYSTSN= 657, 700
RDMNM= 491
RDS= 339, 712
READNUM= 491
RECA= 491
RECANY= 320
RECASZ= 491
RECLNG= 367
REDO= 725
RES= 492
RESIDENT= 324
REST= 492

parameters *(continued)*
RETPD= 727
RGN= 492
RGSUF= 492
RLDSDEF= 727
RMENV= 649
RNR= 657
ROUTING= 420
RRST= 758
RSENAME= 712
RSR= 728
RSRFEAT= 340
RSRMBR= 492
RST= 492
RVFY= 493
SAPPLID= 657
SASPSB= 313
SAV= 493
SBBUF= 805
SBBUFCB= 805
SBONLINE= 752
SCEERUN= 351
SCHD= 420, 758
SCHDTYP= 311
SCIINIT= 493
SCIPROC= 649
SCL= 351
SCOPE= 493
SECCNT= 322, 351, 386, 657
SECCOUNT= 724, 728, 729
SECLVL= 386
SECUNIT= 724, 728, 729
SEGNO= 420
SEGSIZE= 391, 420, 695
SERIAL= 420
SESSION= 376, 391
SGN= 494
SHAREDQ= 494
SHMSGSZ= 495
SHUTDWN= 367
SIDE= 649
SIGNON= 657
SIMEXIT= 322
SIZE= 391
SLDSDEF= 729
SLU2= 657
SMTO= 657
SMTO1-8= 657
SMTOG= 657
SMTOUSID= 657
SOD= 495, 817
SOUT= 495
SPA= 420
SPACE= 724, 728, 730
SPAP= 495
SPIE= 495
SPM= 496
SQGROUP= 738
SQTT= 758
SRCH= 496
SRMDEF= 657, 700
SSM= 496
START= 761, 805
STIMER=
message-driven programs 497
STM= 657
STOP= 761, 805
parameters *(continued)*
STRG= 759
STRINGNM= 742
SUBS= 759
SUF= 497
SUFFIX= 351
SURV= 713
SVC2= 498
SVCNO= 339
SVSODR= 498
SWAP= 498
SWITCH= 713
SYNCLEVEL= 649
SYS= 499
SYS1= 499
SYS2= 499
SYSID= 310, 375, 420, 499
SYSMSG= 351
SYSTEM= 340
T= 499
TCORACF= 499
TCPIP 782
TERM= 351
TEST= 499
TIME= 726
TIMEOUT= 817
TIMER= 817
TLIM= 499
TMINAME= 340, 499, 730
TPNAME= 649
TRACE= 500
TRACK= 500
TRKMODS= 730
TRN= 500
TRUNC= 657
TSR= 500
TYPE= 223, 376, 380, 386, 391
UHASH= 501
UHTS= 501
UJCL1= 351
UJCL2= 351
UJCL3= 351
UJCL4= 351
UJCL5= 351
UJCLx= 351
UMAC0= 351
UMAC1= 351
UMAC2= 351
UMAC3= 351
UNIT= 391, 695, 723, 727, 729
UNITYPE= 365, 441, 695
UOM= 649
UPDTPRT= 351
USERID= 817
USERLIB= 351
USERVAR= 501, 713
VACBOPN= 657
VALCK= 501
VAUT= 501
VFREE= 501
VSFX= 501
VSPEC= 501
WADS= 159, 502, 763
WADSDEF 763
WARNINC 782
WARNSOC 782
WFI= 420
parameters *(continued)*
WKAP= 502
WTORUSID= 657
XPLINK= 502
YEAR4= 502
parameters specifying IMS 511
PARDLI= parameter 484
PARLIM= parameter 420
PARM1= and PARM2= on EXEC statement 153
partitioned data set. 101
partitions
DFSHALDB ddname 192
HALDB control statement 192
selective partition processing 192
PARTNER= parameter 372
passtickets, using instead of passwords 657
PASSWD= parameter 320, 484
PASSWD1= parameter 484
PCB= parameter 485, 805, 813, 815
PDS (partitioned data set) 101
used online 115
PDSE resource restrictions 111, 133
performance
parameters for 151
performance-related EXEC parameters 162
PGMTYPE= parameter 309
PGPROT= parameter 485
PI (program isolation)
with PROCOPT=GO option 94
PIINCR= parameter 485
PIMAX= parameter 485
PL/I
data conversion
from XML 216
XML-to-PL/I conversion support
configuring 216
example 216
prerequisites 216
restrictions 216
PL/I modules
organizing 184
PL/I Optimizer 184
planning
scheduling algorithm 83
PLEX
OM trace table type 587
RM trace table type 587
SCI trace table type 587
PMTO= parameter 657
PMTO1-8= parameter 657
PMTOG= parameter 657
POOLID= parameter 742
port
configuration examples 782
positional
supported in SSM member of the IMS PROCLIB data set 221
PPT
updating 28
PPUR= control statement 769
PRDR= parameter 339, 486
PREINIT= parameter 486
preinitialization routines, specifying in DFSHSBxx 714

PREMSG= parameter 486
preprocessor
executing 20
exit routines 19
JCL 20
LGEN output 17
resource names
verified 16
storage requirements, estimating 18
preprocessor for system definition 15
PRICOUNT= parameter 723, 728, 729
PRIUNIT= parameter 723, 727, 729
PRLD= parameter 487
procedure
IMSDALOC 577
procedures
alphabetic listing 445
cataloged 445
DBBBATCH 520
DBC 523
DCC 532
DD statements 502
DFSDB2AF 503
DFSESL 503
DFSHALDB 503
DFSOLPnn 503
DFSOLSnn 503
DFSRESLB 503
DFSSTAT 504
DFSTCF 504
DFSVSAMP 504
DFSWADSn 504
FORMATA 504
FORMATB 504
FPTRACE 505
IEFRDER 505
IEFRDER2 505
IMS 505
IMSACB 505
IMSACBA 505
IMSACBB 505
IMSIRD 506
IMSLOGR 506
IMSMON 506
IMSRDS 506
IMSRDS2 506
IMSTFMTA 506
IMSTFMTB 506
INPARMS 506
JAVAERR 506
JAVAIN 506
JAVAOUT 506
JCLOUT 507
JCLPDS 507
LGMSG 507
LGMSGL 507
MODBLKSA 507
MODBLKSB 507
MODSTAT 507
MODSTAT2 507
MSDBCP1 508
MSDBCP2 508
MSDBCP3 508
MSDBCP4 508
MSDBDUMP 508
MSDBINIT 508
OLCSTAT 508
procedures *(continued)*
DD statements *(continued)*
PRINTDD 509
PROCLIB 509
QBLKS 509
QBLKSL 509
RECONn 509
SHMSG 509
SHMSGL 509
STEPLIB 509
SYSABEND 509
SYSHALDB 509
SYSIN 510
SYSLIB 510
SYSLIN 510
SYSLMOD 509
SYSPRINT 509
SYSTSPRT 509
SYSUDUMP 510
SYSUTn 510
DFSIASN0 541
DFSJBP 543
DFSMPR 547
DLIBATCH 548
DLISAS 551
FDR 555
FPUTIL 559
IMS 560
IMSBATCH 571
IMSCOBGO 572
IMSCOBOL 575
IMSCOBOL procedure
description 575
IMSFP 578
IMSPLI 580
IMSPLIGO 581
IMSRDR 584
parameters 448
ABND= through AUTO= 448
BKO= through CSLG= 453
DBBF= through DIRCA= 459
DLINM= through EXVR= 464
FBP= through HSBMBR= 468
IMS= through LHTS= 472
LOCKMAX= through
MNPSPW= 476
MOD= through ORSMBR= 480
OTHR= through PIMAX= 482
PRDR= through QTU= 486
RC= through RVFY= 490
SAV= through SUF= 493
SVC2= through TMINAME= 498
TRACE= through YEAR4= 500
RDIBATCH 584
storing 445
processing options, as part of
scheduling 84
PROCLIB
CQSIPxxx 205
DFSCDxxx 205
DFSCGxxx 205
EXITMBR parameter 587
external subsystem member 221
members 205
BPE 205
positional parameters supported 221
STATINTV parameter 587
PROCLIB members
DFSDSCMx 689
DFSDSCTy 701
DFSYDTx (OTMA) 771
ETO
DFSDSCMx 689
DFSDSCTy 701
OTMA DFSYDTx 771
PROCLIB= parameter 351
PROCLIM= parameter 420
program modules, making high-use
resident 716
Program Properties Table (PPT) for
z/OS 28
entries for Base Primitive
Environment 28
entries for Common Queue
Server 28
entries for Common Service Layer 28
entries for IMS Connect 28
entries for IMS control region 28
entries for IRLM 28
program scheduling
for BMPs 94
into message classes 85
message priorities 86
program specification block (PSB)
abend U3303 89
stopped due to 10 U3303 abends 89
programs
application, characteristics of 76
online, declaring 76
scheduling CPI-communications-
driven 94
PRTY= parameter 351, 420
PSB
calculating EPCB storage 314
EPCB storage
calculating 314
pools
defining 71
stopping 85
PSB (program specification block)
abend U3303 89
stopped due to 10 U3303 abends 89
PSB performance options
dynamic 76
PSB pools, defining
/DISPLAY POOL PSBP command 71
ACBGEN utility 71
PSB-related EXEC parameters 162, 165
PSB= parameter 310, 313, 487, 805
PSBW= parameter 313, 487
PSDEPAB= parameter 712
PSELNODBRC control statement 770
pseudo wait-for-input (PWFI) 88
PST= parameter 488
PSTIMER= parameter 657
PSWDC= parameter 488
PTRSIZE= parameter 391, 694
PU= parameter 391
publications
changes xiii
organization xiii
titles xiii
PWFI (pseudo wait-for-input),
description 88

PWFI= parameter 488

Q

QBUF= parameter 489
QBUFHITH= parameter 489
QBUFLWTH= parameter 489
QBUFMAX= parameter 489
QBUFSZ= parameter 489
QCF (Queue Control Facility)
  IQMRH0 126
  migrating messages 126
  monitoring users 126
  User Queue Space Notification Exit routine 126
QMGR= parameter 758
QTL= parameter 159, 490
QTU= parameter 159, 490
Queue Control Facility 126
queue manager
  concurrent I/O 126
quick reschedule
  description 88
  specifying 88

R

RACF
  for local option security 212
  passtickets, using 657
RC= parameter 490
RCF= parameter 490
RCFTCB= parameter 491
RCLASS= parameter 386, 491, 657
RCVYCONV= parameter 657, 699
RCVYFP= parameter 666, 699
RCVYRESP= parameter 657
RCVYSTSN= parameter 657, 700
RDDS
  exporting
    descriptor definitions 60
    resource definitions 60
  extraction utility 65
  importing
    descriptor definitions 62
    resource definitions 62
    using IMPORT command 62
  in FDBR region 41
  on IMS RSR tracking system 41
RDDS (resource definition data set)
  allocating 130
RDIBATCH procedure
  DD statements 584
  description 584
  parameters 584
    APARM= 450
    BKO= 453
    BUF= 454
    CKPTID= 454
    DBRC= 460
    EXCPVR= 468
    FMTO= 468
    IMSID= 472
    IMSPLEX= 473
    IOB= 473
    IRLM= 473
RDIBATCH procedure (continued)
  parameters (continued)
    IRLMNM= 474
    LOCKMAX= 476
    LOGA= 476
    MBR= 479
    MON= 480
    PRLD= 487
    PSB= 487
    RGN= 492
    RST= 492
    SOUT= 495
    SPIE= 495
    SRCH= 496
    SSM= 496
    SWAP= 498
    SYS= 499
    SYS2= 499
    TEST= 499
RDMNM= parameter 491
RDS= parameter 339, 712
read access to databases 325
read-only access to databases 325
READNUM= parameter 491
RECA= parameter 491
RECANY= parameter 320
RECASZ= parameter 491
RECLNG= parameter 367
recommendations
  checkpoint frequency
    take frequent checkpoints to minimize restart time 69
  deadlock frequency
    choose value high enough to resolve deadlocks and low enough not to impact performance 463
    choose value low enough not to impact performance 463
  deadlocks
    monitor number of deadlocks and adjust local deadlock value accordingly 463
  exit routines
    lower-numbered exit routines should be field exit routines 319
    lower-numbered exit routines should be segment exit routines 319
  Fast Path (FPCTRL macro)
    specify BLOCKS=FP to avoid performance problems and system abends 705
  IMS batch 738
  IRLM startup procedures
    specify same value for DEADLOK parameters to avoid last IRLM value becoming the value for all 463
  JAVAERR DD statements
    specify to take advantage of function 506
  JAVAOUT DD statements
    specify to take advantage of function 506
  language interface module
    CSECT name 223
recommendations (continued)
  LINEGRP macro
    allocate at least two data sets 365
  RSR
    If RSR is not installed or enabled, do not specify a value for GSGNAME 344
  security
    protect IMS data sets from unauthorized access by using RACF or an equivalent product 283
    use OM (Operations Manager) command security rather than IMS security 649
  serviceability
    specify ON for the DL/I=, LOCK=, DISP=, and SCHD=parameters 752
  system log data set (SLDS)
    specify PRICOUNT(0) and SECCOUNT(0) to limit to 5 volumes or less during restart 724, 728, 729
  tracing
    run the FPTRACE in a test environment only 756
  transactions
    use upper-case characters for TPNAMES 649
RECORDER
  JCL to print output 220
Recoverable Resource Manager Services attachment facility (RRSAF) 228
recovery service
  IMS PROCLIB data set member (DFSORSxx) 718
recovery-related EXEC parameters 159, 162, 165
REDO= parameter 725
region-control EXEC parameters 162, 165
regions
  FDBR
    configuration 105
regions, dependent
  number of, choosing the 69, 340, 488
  preinitialization routines 714
replication data sets, optional 137
RES= parameter 492
RESIDENT online change 306
RESIDENT= parameter 324
resource definition data set
  in FDBR region 41
  on IMS RSR tracking system 41
resource definition data set (RDDS)
  allocating 130
  changing block size 132
  exporting
    descriptor definitions 60
    resource definitions 60
  importing
    descriptor definitions 62
    resource definitions 62
    using IMPORT command 62
resource definitions
  and IMS cold start 41

resource definitions *(continued)*
exporting to RDDS 60
exporting with DRD 59
extraction utility 65
importing with DRD 61, 62
recoverability 41
resource descriptor definitions
creating 46
creating in IMSplex 50
deleting 54
deleting in IMSplex 59
updating 50
updating in IMSplex 54
Resource Manager
set up tracing 244
Resource Manager (RM)
initialization parameters 205, 640
sample CSLRIxxx member 643
sample user exit list member of the IMS PROCLIB data set 643
startup procedure 518
user exit list member of the IMS PROCLIB data set 205, 641
resource name checks 23
resource name list. 111
Resource Name table, storage 18
resource naming
rules
ETO terminals 299
macros 299
node names 299
subpool names 299
resource naming rules
ETO terminals 299
macros 299
resource structure
calculating storage for 198
CQS support 205
resources that are stored 198
resources
in large system definition processing 24
resources created with IMPORT command
backing out 63, 64
REST= parameter 492
restarting IMS
checkpoints, using 69
restrictions
defining MSDBs dynamically 47
RETPD= parameter 727
RGN= parameter 492
RGSUF= parameter 492
RLDSDEF= parameter 727
RM exit routine member of the IMS PROCLIB data set
EXITMBR parameter 587
RM trace table types
* (asterisk) 587
CSL 587
ERR 587
PLEX 587
recommendations 587
RM 587
RMENV= parameter 649
RMF
accounting procedures 71
RNL (resource name list) 111
RNR= parameter 657
routing codes 80
deleting 58
deleting dynamically 58
deleting with DRD 58
ROUTING= parameter 420
RRSAF (Recoverable Resource Manager Services attachment facility) 228
RRST= parameter 758
RSENAME= parameter 712
RSR
COMM macro
APPLID parameter 317
PSWD parameter 320
DFSRSRxx member 720
DL/I batch jobs 520
enabling 340
IMSCTRL macro, RSRFEAT parameter
GSGNAME parameter 340
RSRFEAT parameter 340
TMINAME parameter 340
including in IMS 108
log router activity, tracing 757
options, specifying 720
support for RDDS 41
RSR= parameter 728
RSRFEAT= parameter 340
RSRMBR= parameter 492
RST= parameter 492
RTCODE macro 80
description 384
parameters 384
CODE= 384
INQUIRY= 384
syntax diagram 384
RTCODE macro statement
for Fast Path 25
RUNOPTS
parameters 792
statement 792
syntax 792
runtime resource definitions
creating with DRD 46
creating with DRD in IMSplex 50
deleting with DRD 54
deleting with DRD in IMSplex 59
updating with DRD 50
updating with DRD in IMSplex 54
RVFY= parameter 493

S

Sample
BPE configuration files 587
BPE user exit list member of the IMS PROCLIB data set 605
combined user exit list member of the IMS PROCLIB data set 605
CQS user exit list member of the IMS PROCLIB data set 605
IMS Connect user exit list member of the IMS PROCLIB data set 605
OM user exit list member of the IMS PROCLIB data set 605
RM user exit list member of the IMS PROCLIB data set 605
Sample *(continued)*
SCI user exit list member of the IMS PROCLIB data set 605
SAPPLID= parameter 657
SASPSB= parameter 313
SAV, online execution parameter 151
SAV= parameter 493
SAVE AS prompt panel
panel field descriptions 296
Save Member prompt panel
panel field descriptions 295
SBBUF= parameter 805
SBBUFCB= parameter 805
SBCO 805
SBESNAP 805
SBIC 805
SBONLINE control statement 751
SBONLINE= parameter 752
SBPARM 805
SBSNAP 805
SCEERUN= parameter 351
SCHD= parameter 420, 758
SCHDTYP= parameter 311
scheduling
CPI-communications-driven programs 94
parallel 92
setting processing limits 87
scheduling algorithm
developing 83
SCI exit routine member of the IMS PROCLIB data set
EXITMBR parameter 587
SCI trace table types
* (asterisk) 587
CSL 587
ERPL 587
ERR 587
INTF 587
INTP 587
PLEX 587
recommendations 587
SCIPROC= parameter 649
SCL= parameter 351
SCOPE= parameter 493
screen sizes, relating device names to 419
SDEP utilities
discarding preallocated SDEP CIs 770
SDUMP
how IRLM uses it 821
searching for online 541
SECCNT= parameter 322, 351, 386, 657
SECCOUNT= parameter 724, 728, 729
SECLVL= parameter 386
secondary allocation 243
secondary indexing
suppressing index entries 817
secondary master terminal 98
commands that can be copied to 318
SECUNIT= parameter 724, 728, 729
Secure Sockets Layer (SSL)
libraries required for 209
security
EXEC parameters 161, 162, 165
for local option 212

security *(continued)*
maintenance blocks 101
specifying options 108
SECURITY macro
COMM macro, and 386
description 386
IMSGEN macro, and 386
parameters 386
RCLASS= 386
SECCNT= 386
SECLVL= 386
TYPE= 386
relationship to COMM and IMSGEN 386
syntax diagram 386
security support 212
IMS Connect 213
SEGNO= parameter 420
SEGSIZE= parameter 391, 420, 695
selective partition processing
DFSHALDB ddname 192
HALDB control statement 192
separate address space 71
sequential buffering
control statements, specifying 184
SBCO 805
SBESNAP 805
SBIC 805
SBPARM 805
SBSNAP 805
SNAPDEST 805
online system 751
OSAM 179, 182
SERIAL= parameter 420
serialization
programs in an IMSplex 626
serviceability and trace options, defining 752
SESSION= parameter 376, 391
SETI 817
SETO 813
SETR 815
SETRLIMI 782
setting up IMS for diagnostics 235
SGN= parameter 494
shared queues
DFSDFxxx 686
interface, tracing 758
managing serialized programs 626
SHAREDQ= parameter 494
Sharing BPE configuration parameters 587
SHMSGSZ= parameter 495
SHUTDWN= parameter 367
SIDE= parameter 649
SIGNON= parameter 657
SIMEXIT= parameter 322
SIZE= parameter 391
SLDS (system log data set)
archiving OLDS 119, 120
creating 125
SLDSDEF= parameter 729
SLU 1
TERMINAL macro statement, specifying 391
SLU 2
TERMINAL macro statement, specifying 391
SLU P
TERMINAL macro statement, specifying 391
SLU2= parameter 657
SMF
accounting procedures 71
SMP/E
used for maintenance 28
SMTO= parameter 657
SMTO1-8= parameter 657
SMTOG= parameter 657
SMTOUSID= parameter 657
SNAPDEST 805
SOD= parameter 495, 817
Sort/Split utility
large system definition Stage 1 processing 17
SOUT= parameter 495
procedures
IMSDALOC 577
SPA= parameter 420
space requirements, data sets
direct output 135
SYSOUT 133
SPACE= parameter 724, 728, 730
SPAP= parameter 495
specify exit list member name 587
specify language 587
specify SYSMDUMP statement 237
specify SYSUDUMP statement 237
specify time intervals 587
specify trace level 587
specifying desired trace options 224
specifying security options 108
SPIE= parameter 495
SPM= parameter 496
spool
estimating SYSOUT requirements 133
Spool
TERMINAL macro statement, specifying 391
spool line group
logical record length 133
specifying LINEGRP macro 133
spool SYSOUT data sets 133
SQGROUP= parameter 738
SQTT= parameter 758
SRCH= parameter 496
SRMDEF= parameter 657, 700
SSM=
DB2 for z/OS
connection parameters 226
IMS batch regions 226
SSM= parameter 496
SSRV (system services trace table) 587
stage 1
IMS PROCLIB data set updates 171
input, sequencing 2
stage 1 processing
large system definition environment 24
stage 1, IMS system definition 21
stage 2, IMS system definition 27
staging libraries 111
START command
database access, defining 324
DATABASE macro, ACCESS parameter 324
dependent regions, starting 69
START= parameter 761, 805
starting CQS 202
startup CQS 510
startup procedure
Structured Call Interface 519
static definition
terminals 103
when required 103
STG (storage service trace table) 587
STIMER= parameter
message-driven programs 497
STM= parameter 657
STOP= parameter 761, 805
storage
CQS
resource structure 198
extended
requesting that z/OS use 751
storage estimates
extended private storage 18, 19
storage pool
64-bit 183
storage pool definitions
default 731
DFSSPMxx, overriding defaults 731
storage service trace table (STG) 587
STR (structure trace table) 587
STRG= parameter 759
STRINGNM= parameter 742
structure
EMHQ (EMH queue)
CQSSGxxx 620
CQSSLxxx 618
disabling 618, 620
recovery data set, example 510
size 197
structure trace table (STR) 587
Structured Call Interface
set up tracing 244
user exits
initialization parameters 205, 644
Structured Call Interface (SCI)
sample startup procedure 519
startup procedure 519
user exits
List member of the IMS PROCLIB data set 205, 644
SUBPOOL macro
description 390
LU 6.1 VTAM Devices 390
parameters 390
MSGDEL= 390
NAME= 390
syntax diagram 390
SUBS= parameter 224, 759
subsystem identification parameters 147
SUF= parameter 497
SUFFIX= keyword
specifying alternate configurations of online system 23
SUFFIX= parameter 351

supporting multiple clients 197
suppressing index entries 817
SURV= parameter 713
suspend queue
for scheduling transactions 90
SVC2= parameter 498
SVCNO= parameter 339
SVSODR= parameter 498
SWAP= parameter 498
SWITCH= parameter 713
switched communication lines
described by LINE macro 363
SYNCLEVEL= parameter 649
Syntax Checker 174
action pull-down options 292
checking parameter values 283
display options 288
error checking 290
function keys 284
help information 284
IMS PROCLIB data set members 291
IMS Release and Control Region panel 286
IMS Release panel 287
inserting keywords 294
interrupting processing 294
Keyword Display panel 288, 289, 292, 294, 295
online help 285
overview 445
panel field descriptions 286, 287, 288, 289, 295, 296, 297
Parameter Syntax Checker panel 297
SAVE AS prompt panel 296
Save Member prompt panel 295
saving processed members 295
starting 285
using 285
syntax diagram
how to read x
SYS= parameter 499
SYS1= parameter 499
SYS2= keyword
procedures
IMSDALOC 577
SYS2= parameter 499
SYSID= parameter 310, 420, 499
changing 375
description 375
SYSMDUMP statement
specify 237
SYSMSG= parameter 351
SYSOUT
estimating spooled requirements 133
SYSOUT data sets
allocation of data sets 133
BSAM EXCP use in 133
for TSO browsing 133
space requirements, data sets 133
system
data sets
for online change 111
initializing IMS system data sets 111
system checkpoint data set example 510
system configuration
macro statements 270
system data sets
online change function 111
system definition
allowing for Fast Path 25
examples 253
DB/DC environment 253
for a Common Service Layer installation 205
high-level view 1
how it is related to installation 1
IMSCTRL macro 340
large system definition
storage requirements 18
large system definition environment 24
large systems 24
macros 5, 254, 299
maximum occurrences 299
output from stage 1 23
parameters changing 5
preprocessor
executing 20
large system definition 18
standard 18
storage requirements, estimating 18
preprocessor, use of 15
procedures
environments in which they apply 445
required macros for Fast Path 25
RSR enabling 340
specifying alternative version 23
specifying security options 108
storage requirements 19
supertasks 1
tracking subsystem
when separate definition required 108
type
guidelines for selecting 340
type of system definition required 5
types 23
when it is required 23
system definition macro statements
APPLCTN macro, for Fast Path 25
checking with the preprocessor 15
RTCODE macro, for Fast Path 25
TERMINAL macro, for Fast Path 25
TRANSACT macro, for Fast Path 25
system definition process
ALL 2
BATCH 2
CTLBLKS 2
JCLIN process 28
MODBLKS 2
MSVERIFY 2
NUCLEUS 2
ON-LINE 2
overview of 1
SMP/E maintenance 28
stage 1 21
stage 2 27
types of 2
system log data set 119
system modification program/extended (SMP/E)
used for maintenance 28
system services trace table (SSRV) 587
system set up
BPE external trace 244
CQS tracing 244
external trace environment 241
IMS Connect tracing 244
OM tracing 244
RM tracing 244
SCI tracing 244
specify SYSMDUMP statement 237
specify SYSUDUMP statement 237
writing trace tables 242
z/OS master trace table size 237
system set up for diagnosis
IMS Control Region EXEC 237
system trace table 237
System/3
LINEGRP macro statement, specifying 365
TERMINAL macro statement, specifying 391
System/7
LINEGRP macro statement, specifying 365
TERMINAL macro statement, specifying 391
SYSTEM= parameter 340
SYSUDUMP statement
specify 237

T

T= parameter 499
tables, writing trace 242
tailoring 171
execution procedures for Fast Path 174
TCORACF= parameter 499
TCP/IP
and creating the IMS Connect configuration member 782
and IMS Connect configuration 210
ECB parameter 782
KeepAlive function 782
keyword parameters 782
local option client communications updates in the z/OS PPT 210
user exit message 782
z/OS Program Properties Table (PPT)
local option client communication updates 210
TCP/IP updates 210
updating 210
TCPIP
parameters 793
statement 793
syntax 793
TERM= parameter 351
terminal devices, allocating in an online system 562
TERMINAL macro 133
COMM macro, and 322
description 391
label field 391

TERMINAL macro *(continued)*
MFS 142
parameters 391
BACKUP= 391
BUFSIZE= 391
COMPT= 391
COMPTx= 391
EDIT= 391
FEAT= 391
FPBUF= 391
LTERM= 391
MODEL= 391
MODETBL= 391
MSGDEL= 391
NAME= 391
OPTIONS= 391
OUTBUF= 391
PTRSIZE= 391
PU= 391
SEGSIZE= 391
SESSION= 391
SIZE= 391
TYPE= 391
UNIT= 391
syntax diagram 391
TERMINAL macro statement
for Fast Path 25
terminal network 101
terminals
defining 95
dynamic 103
dynamic allocation 103
effect of system definition process on 103
static 103
static definition 103
TEST= parameter 499
time-of-day clock 125
TIME= parameter 726
TIMEOUT= parameter 817
timer
counting unused IOVF control intervals 473
TIMER= parameter 817
TLIM= parameter 499
TMINAME= parameter 340, 499, 730
TMS
installing 231
TOD clock 125
TPNAME= parameter 649
TRACE command
DASD log activity, tracing 755
DB2 subsystem connection, tracing 759
DBF entries from FP, tracing 756
dispatcher activity, tracing 755
DL/I activity, tracing 755, 756
latch activity, tracing 756
lock activity, tracing 757
options 754
OTMA control, tracing 758
queue manager, tracing 758
RSR log router, tracing 757
scheduler, tracing 758
shared queues interface, tracing 758
storage manager calls, tracing 759
trace entries
options, defining 752
storage requirements 759
trace table
external trace environment
starting and stopping 241
sizes
z/OS master 237
z/OS system 237
TRACE= parameter 500
traces
controlling the volume 241
CQS 244
Fast Path 242
IMS Connect 244
offloading trace data set 242
OM 244
RM 244
SCI 244
writing entries to external data sets 587
TRACK= parameter 500
tracking subsystem
system definition
when separate definition required 108
trademarks 823
TRANSACT macro
description 420
parameters 426
AOI= 420
CODE= 420
DCLWA= 420
EDIT= 420
FPATH= 420
INQ= 420
INQUIRY= 420
MAXRGN= 420
MODE= 420
MSGTYPE= 420
PARLIM= 420
PROCLIM= 420
PRTY= 420
ROUTING= 420
SCHD= 420
SEGNO= 420
SEGSIZE= 420
SERIAL= 420
SPA= 420
SYSID= 420
WFI= 420
syntax diagram 420
TRANSACT macro statement
for Fast Path 25
INQUIRY keyword 83
keywords for Fast Path transactions 83
MSGTYPE keyword 83
PROCLIM keyword 83
PRTY keyword 86
SCHD keyword 93
transaction codes
specifying 420
transactions
configuring 81
deleting 58
deleting dynamically 58
transactions *(continued)*
deleting with DRD 58
examples of grouping 84
Fast Path
defining characteristics 83
recommended keywords 81
scheduling in parallel 92
scheduling using suspend queue 90
stopped due to 10 U3303 abends 89
stopping 85
storage in resource structure 198
Transport Manager subsystem
defining 231
VTAM definition 231
Transport Manager Subsystem
installing 231
TRCLEV
BPE trace table statements 587
BPE trace table types 587
members of the IMS PROCLIB data set 587
parameters 587
RM trace table types 587
TRKMODS= parameter 730
TRN= parameter 500
TRUNC= parameter 657
TSO
IMS Connect
allocating log record data sets 220
TSO browsing, IMS support of 133
TSR= parameter 500
tuning considerations
control program address spaces 71
Type 2 SVC 101
Type 4 SVC 101
TYPE macro
description 441
MFS 142
parameters 442
BACKUP= 441
EDIT= 441
OPTIONS= 441
UNITYPE= 441
syntax diagram 441
TYPE= parameter 223, 376, 380, 386, 391
DATASET 325
DFSDCMON 325
DFSMDA 325
FINAL 325
FPDEDB 325
IMSACB 325
INITIAL 325
OLCSTAT 325
OLDS 325
RECON 325
SLDS 325

U

U3303 abend
stopping transactions and PSBs after ten U3303 abends 89
UHASH= parameter 501
UHTS= parameter 501
UJCL1= parameter 351
UJCL2= parameter 351
UJCL3= parameter 351

UJCL4= parameter 351
UJCL5= parameter 351
UJCLx= parameter 351
UM (undefined record format) 133
UMAC0= parameter 351
UMAC1= parameter 351
UMAC2= parameter 351
UMAC3= parameter 351
undefined record format 133
UNIT= parameter 391, 695, 723, 727, 729
  DFSMDA TYPE=DFSCMON statement 329
UNITYPE= parameter 365, 441, 695
Universal drivers
  configuring IMS support 32
UOM= parameter 649
UPDTPRT= parameter 351
user descriptors
  format 689, 701
  parameters 689, 701
  syntax 689, 701
user exit list member of the IMS PROCLIB data set
  BPE exit list member 605
  sample BPE 605
  sample CQS 605
  sample IMS Connect 605
  sample of combined 605
  sample OM 605
  sample RM 605
  sample SCI 605
user exit routine trace table (USRX) 587
user exit routines
  DFSDFxxx 687
user exits
  Structured Call Interface
    initialization parameters 205, 644
    List member of the IMS PROCLIB data set 205, 644
    sample CSLSIxxx 645
    sample list member of the IMS PROCLIB data set 205, 646
User Queue Space Notification Exit routine 126
user-supplied exit routines
  specifying owner's type 605
USERID= parameter 817
USERLIB= parameter 351
USERVAR= parameter 501, 713
USRX (user exit routine trace table) 587
utilities
  Dynamic Allocation 325

V

VACBOPN= parameter 657
VALCK= parameter 501
VAUT, online execution parameter 151
VAUT= parameter 501
verifying stage 1 input 23
VFREE= parameter 501
VSAM
  buffer pools, defining 179, 180, 741
    dynamically 180, 182
  buffers
    adjusting 179
  performance options, defining 746
VSAM *(continued)*
  subpools
    defining 182
  subpools, defining 743
VSAM (Virtual Storage Access Method)
  data sets 129
VSFX= parameter 501
VSPEC= parameter 501
VTAM
  COMM macro, and 317, 320
  data communication macros
    COMM 95
    NAME 95
    SUBPOOL 95
    TERMINAL 95
    TYPE 95
    VTAMPOOL 95
  macro statements 259, 261, 265, 267, 268
  Transport Manager system definition 231
  using multi-node persistent sessions (MNPS) to take over terminals 709
VTAM terminals
  dynamic 103
  dynamic allocation 103
  effect of system definition process on 103
  static 103
  static definition 103
  using multi-node persistent sessions (MNPS) to take over 665
  when multi-node persistent sessions (MNPS) are used rather than XRF 317
VTAMPOOL macro 443

W

WADS (write-ahead data set)
  allocating 119, 123
  definition 123
WADS= parameter 159, 502, 763
WADSDEF control statement 123
WADSDEF parameter 763
wait-for-input (WFI) mode 88
warm start
  impact on resource definitions 41
  resuming an online reorganization for HALDBs 770
WFI (wait-for-input) mode 88
WFI= parameter 420
WKAP= parameter 502
write-ahead data set 119
writing trace tables 242
WTORUSID= parameter 657

X

X'22' log records 38
XIBAREA 782
XML
  adapter 216
  converter 216
  data conversion
    into COBOL 216
XML *(continued)*
  data conversion *(continued)*
    into PL/I 216
  IMS Connect
    conversion support 216
  XML-to-COBOL conversion support
    configuring 216
    example 216
    prerequisites 216
    restrictions 216
  XML-to-PL/I conversion support
    configuring 216
    example 216
    prerequisites 216
    restrictions 216
XPLINK= parameter 502
XRF
  COMM macro
    APPLID parameter 317
    PASSWD parameter 320
  DFSHSBxx PROCLIB member 709
  restrictions 126
XRF (Extended Recovery Facility)
  allocation of data sets 137
  data set placement requirements 137
  impact on other data sets 137
  replicate data sets 137
  shared data sets
    tracking phases 137

Y

YEAR4= parameter 502

Z

z/OS
  IMS procedures, and 183
  IMSRDR procedures, and 183
  master trace table
    size recommendations 235
  program properties table
    updating 28
  system trace table
    size recommendations 235
z/OS setup recommendations
  Automatic Dump Data set Allocation 236
  CHNGDUMP MAXSPACE 236
  common storage tracker 235
z/OS system trace table 237

Product Number: 5635-A02

Printed in USA

GC19-2444-05

Spine information:

IMS **Version 11** **System Definition**