

CICS Transaction Server for z/OS
Version 4 Release 1



System Definition Guide

CICS Transaction Server for z/OS
Version 4 Release 1



System Definition Guide

Note

Before using this information and the product it supports, read the information in "Notices" on page 429.

This edition applies to Version 4 Release 1 of CICS Transaction Server for z/OS (product number 5655-S97) and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright IBM Corporation 1989, 2011.**

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

Contents

| | |
|---|-----------|
| Preface | ix |
| What this book is about | ix |
| Who should read this book | ix |
| What you need to know to understand this book | ix |
| How to use this book | ix |
| Notes on terminology | ix |
| Book structure | ix |

Changes in CICS Transaction Server for z/OS, Version 4 Release 1 xi

Part 1. Defining data sets 1

Chapter 1. Setting up CICS data sets . . . 3

| | |
|--|----|
| Reviewing the CICS data sets. | 3 |
| Defining a data set naming convention | 5 |
| Defining data sets with multiple extents and volumes | 6 |
| Performance considerations of TS and TD buffers | 6 |
| Creating the CICS data sets | 7 |
| Sizing the MVS system data sets. | 7 |
| Planning data sets for XRF. | 8 |
| XRF shared and unique data sets | 9 |
| Defining and allocating XRF data sets | 10 |
| Defining backup while open (BWO) for VSAM files | 10 |
| Effect of disabling activity keypointing | 12 |
| Restrictions on BWO | 12 |
| XRF considerations | 12 |
| Using storage management facilities | 12 |
| Storage management subsystem (SMS) | 13 |
| Data facility hierarchical storage manager (DFHSM) | 13 |

Chapter 2. Setting up the temporary storage data set. 15

| | |
|---|----|
| Defining the temporary storage data set. | 15 |
| The control interval size | 16 |
| Number of control intervals | 17 |
| Number of VSAM buffers and strings | 17 |
| Defining temporary storage pools for temporary storage data sharing | 17 |
| Approximate storage calculations | 18 |

Chapter 3. Setting up data sets for transient data. 21

| | |
|--|----|
| Defining the intrapartition data set | 21 |
| What happens if the intrapartition data set fails to open. | 22 |
| Using multiple extents and multiple volumes | 23 |
| Size of the intrapartition data set | 23 |
| Intrapartition data set restrictions | 23 |
| Defining extrapartition data sets | 24 |
| The DFHCXRF data set | 24 |
| XRF considerations | 25 |

Chapter 4. Setting up CICS log streams 27

| | |
|--|----|
| Defining CICS system logs | 27 |
| Planning your CICS system log streams | 28 |
| Defining CICS general logs | 29 |
| Planning log streams for use by your user journals and autojournals. | 29 |
| Planning log streams for use by your forward recovery logs | 29 |
| Planning log streams for use by your log of logs (DFHLGLOG) | 30 |
| Naming journals. | 31 |
| System logs | 31 |
| Forward recovery logs. | 31 |
| User journals | 32 |
| Installing system log and journal names. | 33 |
| Defining JOURNALMODELS | 33 |
| Mapping log streams | 34 |
| Mapping of the system log stream. | 34 |
| Mapping of general log streams | 37 |
| Using the Journal utility program, DFHJUP | 38 |

Chapter 5. Setting up the CICS system definition data set 41

| | |
|--|----|
| Planning your CSD configuration | 41 |
| Calculating CSD disk space | 42 |
| Initializing the CSD. | 43 |
| Creating a larger CSD | 45 |
| Defining CSD attributes | 46 |
| Sharing the CSD in non-RLS mode | 47 |
| Shared user access from the same CICS region | 47 |
| Multiple users of the CSD within a CICS region (non-RLS) | 49 |
| Sharing a CSD by CICS regions within a single MVS image (non-RLS). | 49 |
| Sharing a CSD in a multi-MVS environment (non-RLS) | 50 |
| Multiple users of one CSD across CICS or batch regions (non-RLS) | 50 |
| Sharing the CSD between different releases of CICS | 51 |
| Other factors restricting CSD access | 52 |
| Sharing the CSD in RLS mode | 53 |
| Differences in CSD management between RLS and non-RLS access | 53 |
| Specifying file control attributes for the CSD | 54 |
| Effect of RLS on the CSD batch utility DFHCSDUP | 55 |
| Planning for backup and recovery. | 55 |
| Transaction backout during emergency restart | 58 |
| Dynamic backout for transactions | 58 |
| Other recovery considerations | 58 |
| Logging RDO commands. | 59 |
| Making the CSD available to CICS | 61 |
| Installing the RDO transactions. | 62 |

| | |
|--|----|
| Installing definitions for the Japanese language feature | 62 |
| CSD XRF considerations | 62 |

Chapter 6. Setting up the catalog data sets 65

| | |
|--|----|
| Defining the global catalog | 65 |
| Space calculations | 67 |
| Buffer space sizings. | 70 |
| Reusing the global catalog to perform a cold start | 70 |
| Defining the local catalog. | 71 |
| Initializing the local catalog | 73 |
| Domains that write to the local catalog | 75 |

Chapter 7. Setting up and using auxiliary trace data sets 77

| | |
|--|----|
| Starting and controlling auxiliary trace | 77 |
| Allocating auxiliary trace data sets | 78 |
| Space calculations | 79 |
| Job control statements for CICS execution | 79 |
| XRF considerations | 80 |
| Using the trace utility program (DFHTU660) | 80 |

Chapter 8. Defining dump data sets . . . 81

| | |
|---|----|
| System dumps | 82 |
| Suppressing system dumps that precede ASRx abends | 82 |
| The CICS transaction dump data sets. | 83 |
| Printing the transaction dump data sets | 83 |
| Defining the transaction dump data sets. | 84 |

Chapter 9. Defining the CICS availability manager data sets. 87

| | |
|---|----|
| The XRF data sets | 89 |
| Space calculations | 90 |
| Crucial and non-crucial messages | 91 |
| Effect of a full XRF message data set on the active CICS region | 91 |
| I/O error handling | 91 |

Chapter 10. Defining user files 93

| | |
|---|-----|
| VSAM data sets | 94 |
| VSAM bases and paths | 94 |
| Loading empty VSAM data sets | 95 |
| Reuse of data sets | 95 |
| VSAM record-level sharing (RLS) | 96 |
| BDAM data sets | 99 |
| Defining data sets to CICS | 100 |
| Using the DSNAMES and DISP file resource definition parameters. | 101 |
| Dynamic allocation using CEMT | 102 |
| Other forms of dynamic allocation | 102 |
| Opening VSAM or BDAM files | 102 |
| Closing VSAM or BDAM files. | 103 |
| Closing files normally | 103 |
| Closing files using the FORCE option | 104 |
| XRF considerations | 104 |
| CICS data tables | 104 |
| Opening data tables | 105 |

| | |
|---|-----|
| Loading data tables | 105 |
| Closing data tables | 105 |
| XRF considerations | 106 |
| Coupling facility data tables | 106 |
| Coupling facility data table models | 106 |
| Coupling facility data table structures and servers | 107 |
| Defining a coupling facility data table pool | 108 |

Chapter 11. Defining the CDBM GROUP command data set 113

| | |
|--|-----|
| Job control statements for CICS execution | 115 |
| Record layout in the CDBM GROUP command file | 115 |

Chapter 12. Defining the CMAC messages data set 117

| | |
|---|-----|
| Job control statements to define and load the messages data set | 117 |
|---|-----|

Chapter 13. Defining the EJB data sets 119

| | |
|---|-----|
| Defining EJB directory and object store data sets | 119 |
| Determining the object store space requirements | 121 |

Chapter 14. Defining the WS-AT data set 123

Chapter 15. Setting up the debugging profiles data sets. 125

| | |
|---|-----|
| Creating the debugging profiles data sets | 125 |
| Defining the debugging profiles data sets as VSAM RLS files | 127 |
| Defining the debugging profiles data sets as VSAM non-RLS files | 128 |
| Defining the debugging profiles data sets as remote files | 129 |

Part 2. CICS system initialization 131

Chapter 16. Specifying CICS system initialization parameters. 133

| | |
|--|-----|
| System initialization parameters for open TCBS | 134 |
| Open TCB pools | 136 |
| Specifying DFHSIT macro parameters | 137 |
| Defining CICS resource table and module keywords. | 138 |
| The system initialization parameter descriptions and summary | 139 |
| ADI | 147 |
| AIBRIDGE | 148 |
| AICONS | 148 |
| AIEXIT | 148 |
| AILDELAY | 149 |
| AIQMAX. | 150 |
| AIRDELAY | 150 |
| AKPFREQ | 150 |
| APPLID | 151 |
| AUTCONN | 152 |

| | | | |
|-------------------------|-----|--------------------------|-----|
| AUTODST | 153 | EUDSASZE | 181 |
| AUTORESETTIME | 153 | FCT | 182 |
| AUXTR | 153 | FCQRONLY | 182 |
| AUXTRSW | 154 | FEPI | 182 |
| BMS | 154 | FLDSEP | 183 |
| BRMAXKEEPTIME | 155 | FLDSTRT | 183 |
| CDSASZE | 156 | FORCEQR | 184 |
| CHKSTRM | 156 | FSSTAFF | 184 |
| CHKSTSK | 156 | FTIMEOUT | 185 |
| CICSSVC | 157 | GMTEXT | 185 |
| CILOCK | 157 | GMTRAN | 186 |
| CLINTCP | 157 | GNTRAN | 187 |
| CLSDSTP | 158 | GRNAME | 187 |
| CLT | 158 | GRPLIST | 188 |
| CMDPROT | 158 | GTFTR | 190 |
| CMDSEC | 159 | HPO | 190 |
| CONFDATA | 159 | ICP | 190 |
| CONFTXT | 161 | ICV | 190 |
| CPSMCONN | 161 | ICVR | 191 |
| CRLPROFILE | 162 | ICVTS | 192 |
| CSDACC | 163 | IIOPLISTENER | 192 |
| CSDBKUP | 163 | INFOCENTER | 192 |
| CSDBUFND | 164 | INITPARM | 193 |
| CSDBUFNI | 164 | INTR | 193 |
| CSDDISP | 164 | IRCSTRT | 193 |
| CSDDSN | 164 | ISC | 193 |
| CSDFRLOG | 165 | JESDI | 194 |
| CSDINTEG | 165 | JVMCCSIZE | 194 |
| CSDJID | 166 | JVMCCSTART | 194 |
| CSDLRNO | 166 | JVMxxxxTRACE | 195 |
| CSDRECOV | 167 | JVMLEVEL0TRACE | 195 |
| CSDRLS | 168 | JVMLEVEL1TRACE | 196 |
| CSDSTRNO | 168 | JVMLEVEL2TRACE | 196 |
| CWAKEY | 169 | JVMUSERTRACE | 196 |
| DAE | 169 | JVMPROFILEDIR | 196 |
| DATFORM | 169 | KEYRING | 197 |
| DB2CONN | 170 | LGDFINT | 197 |
| DBCTLCON | 170 | LGNMSG | 197 |
| DEBUGTOOL | 170 | LLACOPY | 198 |
| DFLTUSER | 171 | LOCALCCSID | 198 |
| DIP | 171 | LPA | 199 |
| DISMACP | 171 | MAXJVMTCBS | 199 |
| DOCCODEPAGE | 171 | MAXOPENTCBS | 199 |
| DSALIM | 172 | MAXSOCKETS | 200 |
| DSHIPIDL | 173 | MAXSSLTCBS | 200 |
| DSHIPINT | 173 | MAXXPTCBS | 200 |
| DSRTPGM | 174 | MCT | 200 |
| DTRPGM | 174 | MN | 201 |
| DTRTRAN | 175 | MNCONV | 201 |
| DUMP | 175 | MNEXC | 202 |
| DUMPDS | 175 | MNFREQ | 202 |
| DUMPSW | 176 | MNIDN | 202 |
| DURETRY | 176 | MNPER | 203 |
| EDSASZE | 177 | MNRES | 203 |
| EDSALIM | 177 | MNSYNC | 203 |
| EJBROLEPRFX | 178 | MNTIME | 203 |
| ENCRYPTION | 179 | MQCONN | 203 |
| EODI | 180 | MROBTCH | 204 |
| ERDSASZE | 180 | MROFSE | 205 |
| ESDSASZE | 181 | MROLRM | 205 |
| ESMEXITS | 181 | MSGCASE | 205 |

| | | | |
|-----------------------|-----|---------------------|-----|
| MSGVLV | 206 | SSLTCBS | 234 |
| MXT | 206 | START | 234 |
| NATLANG | 207 | STARTER. | 236 |
| NCPLDFT | 208 | STATEOD | 236 |
| NEWSIT | 209 | STATINT | 237 |
| NONRLSRECOV | 210 | STATRCD | 237 |
| OFFSITE | 210 | STGPROT | 238 |
| OPERTIM | 210 | STGRCVY | 238 |
| OPNDLIM | 211 | STNTR | 239 |
| PARMERR | 211 | STNTRxx | 239 |
| PDI | 211 | SUBTSKS | 242 |
| PDIR | 212 | SUFFIX | 243 |
| PGAICTLG | 212 | SYDUMAX | 243 |
| PGAEXIT | 212 | SYSIDNT | 243 |
| PGAIPGM | 212 | SYSTR | 244 |
| PGCHAIN | 213 | TAKEOVR | 244 |
| PGCOPY | 213 | TBEXITS | 245 |
| PGPURGE | 213 | TCAM. | 245 |
| PGRET | 213 | TCP | 245 |
| PLTPI | 214 | TCPIP | 246 |
| PLTPISEC | 214 | TCSACTN | 246 |
| PLTPIUSR | 214 | TCSWAIT | 247 |
| PLTSD. | 215 | TCT | 247 |
| PRGDLAY | 215 | TCTUAKEY | 247 |
| PRINT. | 215 | TCTUALOC | 248 |
| PRTYAGE | 216 | TD | 248 |
| PRVMOD | 217 | TDINTRA | 249 |
| PSBCHK | 217 | TDSUBTASK | 249 |
| PSDINT | 217 | TRANISO | 249 |
| PSTYPE | 218 | TRAP | 250 |
| PVDELAY | 219 | TRDUMAX | 250 |
| QUIESTIM | 219 | TRTABSZ. | 250 |
| RAMAX | 219 | TRTRANSZ | 251 |
| RAPOOL | 219 | TRTRANTY | 251 |
| RDSASZE | 220 | TS | 251 |
| RENTPGM | 221 | TST | 252 |
| RESP | 221 | UDSASZE | 252 |
| RESSEC | 221 | UOWNETQL | 252 |
| RLS | 222 | USERTR | 253 |
| RLSTOLSR | 222 | USRDELAY | 253 |
| RMTRAN | 223 | USSHOME | 254 |
| RRMS | 224 | VTAM. | 255 |
| RST | 224 | VTPREFIX | 255 |
| RSTSIGNOFF | 224 | WEBDELAY | 256 |
| RSTSIGNTIME | 225 | WRKAREA | 256 |
| RUWAPOL | 226 | XAPPC | 256 |
| SDSASZE. | 226 | XCMD | 257 |
| SDTRAN | 226 | XCFGROUP | 257 |
| SEC | 227 | XDB2 | 258 |
| SECPREFX. | 228 | XDCT | 258 |
| SIT | 228 | XEJB | 259 |
| SKRxxxx | 228 | XFCT | 259 |
| SNSCOPE | 229 | XHFS | 260 |
| SPCTR | 230 | XJCT | 260 |
| SPCTRxx | 230 | XLT | 261 |
| SPOOL | 233 | XPCT | 261 |
| SRBSVC | 233 | XPPT | 262 |
| SRT | 233 | XPSB | 262 |
| SRVERCP | 234 | XRES | 263 |
| SSLCACHE | 234 | XRF | 263 |
| SSLDELAY | 234 | XRFSOFF. | 264 |

| | |
|---|-----|
| XRFSTME | 264 |
| XTRAN | 264 |
| XTST | 264 |
| XUSER | 265 |
| System initialization parameters that can be defined only with the DFHSIT macro | 265 |
| ESMEXITS | 266 |
| HPO | 266 |
| STARTER. | 266 |
| SUFFIX | 266 |
| System initialization parameters that cannot be defined with the DFHSIT macro | 267 |
| CDSASZE | 267 |
| CHKSTRM | 267 |
| CHKSTSK | 268 |
| ECDSASZE | 268 |
| ERDSASZE | 268 |
| ESDSASZE | 269 |
| EUDSASZE | 269 |
| JVMLEVEL0TRACE | 269 |
| JVMLEVEL1TRACE | 269 |
| JVMLEVEL2TRACE | 270 |
| JVMUSERTRACE | 270 |
| NEWSIT | 270 |
| OFFSITE | 271 |
| PRVMOD | 271 |
| RDSASZE | 271 |
| SDSASZE. | 272 |
| SIT | 272 |
| SPCTRxx | 272 |
| STNTRxx. | 275 |
| UDSASZE | 278 |
| System initialization parameters that cannot be defined through the system operator's console | 278 |
| CMDSEC | 278 |
| CONFDATA. | 279 |
| CONFTXT | 281 |
| DFLTUSER | 281 |
| PLTPISEC | 281 |
| PLTPIUSR | 282 |
| PSBCHK | 282 |
| RESSEC | 283 |
| SEC | 283 |
| SECPREFX. | 284 |
| SNSCOPE | 284 |
| XAPPC | 285 |
| XCMD | 286 |
| XDCT | 286 |
| XEJB | 287 |
| XFCT | 287 |
| XHFS | 288 |
| XJCT | 288 |
| XPCT | 289 |
| XPPT | 290 |
| XPSB | 290 |
| XRES | 291 |
| XTRAN | 291 |
| XTST | 292 |
| XUSER | 292 |
| The default system initialization table | 293 |
| Assembling the SIT | 300 |

| | |
|---|-----|
| Selecting versions of CICS programs and tables | 301 |
| Using an explicit level of function to select programs. | 301 |
| Excluding unwanted programs | 301 |

Chapter 17. Processing system initialization parameters 303

| | |
|--|-----|
| Supplying system initialization parameters to CICS | 303 |
| Using system initialization control keywords | 304 |
| Processing the PARM parameter | 306 |
| Processing the SYSIN data set | 306 |
| Processing the console entries | 307 |
| Controlling start and restart | 308 |
| The role of the CICS catalogs | 308 |
| The START system initialization parameter | 309 |
| CICS startup and the VTAM session. | 314 |
| The CICS parameter manager domain | 315 |
| End of CICS startup | 316 |

Chapter 18. Retrieving information about system initialization parameters 317

Chapter 19. CICS startup 319

| | |
|---|-----|
| Using the sample startup job stream. | 320 |
| A sample CICS startup job | 321 |
| Storage requirements for a CICS region. | 333 |
| Storage protection | 334 |
| The dynamic storage areas and associated storage cushions | 337 |
| The sample statistics program, DFH0STAT | 340 |
| A sample CICS startup procedure | 341 |

Chapter 20. Preparing CICS for using debugging tools 343

| | |
|--|-----|
| Preparing your CICS region for debugging | 343 |
|--|-----|

Part 3. Initializing CICS data sharing servers 345

Chapter 21. Defining and starting AXM system services 347

Chapter 22. Setting up and running a temporary storage server 349

| | |
|---|-----|
| Overview of the temporary storage data sharing server | 349 |
| Defining TS server regions | 350 |
| Queue server REGION parameter | 351 |
| TS queue server parameters | 352 |
| Primary parameters | 352 |
| Automatic restart manager (ARM) parameters | 353 |
| List structure parameters | 354 |
| Debug trace parameters | 354 |
| Tuning parameters | 355 |
| Warning parameters | 356 |
| Automatic ALTER parameters | 357 |
| Queue server automatic ALTER processing | 357 |
| Shared TS queue server commands | 358 |

| | |
|---|-----|
| DISPLAY and PRINT keywords | 359 |
| The CANCEL command options | 360 |
| Unloading and reloading queue pools | 360 |

Chapter 23. Setting up and running a coupling facility data table server . . . 363

| | |
|--|-----|
| Overview of a coupling facility data table server | 363 |
| Defining and starting a coupling facility data table server region | 364 |
| Coupling facility data table server parameters | 365 |
| Avoiding structure full conditions | 374 |
| Coupling facility data table server automatic structure alter | 375 |
| Controlling coupling facility data table server regions | 375 |
| The SET command options | 376 |
| DISPLAY and PRINT command options | 378 |
| The CANCEL command options | 381 |
| Deleting or emptying coupling facility data table pools | 381 |
| Unloading and reloading coupling facility data table pools | 382 |

Chapter 24. Setting up and running a region status server 385

| | |
|--|-----|
| Defining a list structure for a region status server | 386 |
| Defining and starting a region status server region | 387 |
| Controlling region status servers | 388 |
| The SET command options | 389 |
| DISPLAY and PRINT command options | 391 |
| The CANCEL command options | 394 |
| Deleting region status server pools | 394 |

Chapter 25. Setting up and running a named counter server. 397

| | |
|---|-----|
| Named counter server overview | 397 |
| Defining a named counter options table | 399 |
| The options table parameters | 400 |
| Making an options table available to CICS | 402 |
| Defining a list structure | 402 |
| List structure storage considerations | 403 |
| Defining and starting a named counter server region | 403 |
| Named counter server parameters | 404 |
| Controlling named counter server regions | 408 |
| The SET command options | 409 |
| DISPLAY and PRINT command options | 409 |
| The CANCEL command options | 411 |
| Deleting or emptying named counter pools | 411 |
| Changing the size of named counter pools | 411 |

| | |
|---|-----|
| Unloading and reloading named counter pools | 412 |
| Unload JCL example | 413 |
| Reload JCL example | 413 |
| Dumping named counter pool list structures | 413 |

Chapter 26. Coupling facility server operations. 415

| | |
|--|-----|
| Monitoring coupling facility server messages | 415 |
| Server messages | 415 |
| AXM messages | 416 |
| Coupling facility storage management | 416 |
| Managing the pool structure | 417 |
| Monitoring pool structure usage levels | 417 |
| Operator messages reporting on pool structure usage | 418 |
| Use of CFRM automatic ALTER to increase pool structure size | 418 |
| Using system-managed rebuild to increase pool structure size | 418 |
| Increasing the number of data lists | 418 |
| Deleting or emptying the pool structure | 419 |
| Server connection management | 419 |
| Terminating server connections | 419 |
| Failed server connections | 420 |
| Restarting a server | 420 |

Chapter 27. CICS server support for system-managed processes 423

| | |
|---|-----|
| System-managed list structure rebuild | 423 |
| TS data sharing and CFDT servers | 424 |
| Named counter server | 424 |
| System-managed list structure duplexing | 425 |

Part 4. Appendixes 427

| |
|------------------------------|
| Notices 429 |
| Trademarks 430 |

Bibliography. 431

| | |
|--|-----|
| CICS books for CICS Transaction Server for z/OS | 431 |
| CICSplex SM books for CICS Transaction Server for z/OS | 432 |
| Other CICS publications | 432 |
| Other IBM publications | 432 |

Accessibility. 435

Index 437

Preface

What this book is about

This book is intended to help you specify and install the system definitions and resources for a CICS® system. It contains guidance about the system definitions required to run a CICS system in an IBM® MVS™ environment.

Who should read this book

This book is for system programmers responsible for specifying and installing the system definitions and resources for a CICS system.

What you need to know to understand this book

You should have experience of the MVS operating system, and either have previous experience of CICS, or at least be familiar with the concepts and terminology. To understand the jobs required to install CICS resource definitions, you should be familiar with MVS job control language (JCL) and cataloged procedures.

How to use this book

The parts and chapters of this book are self-contained. Use an individual part or chapter where it contains information about the particular task you are engaged in.

Notes on terminology

“CICS” is used throughout this book to mean the CICS element of the IBM CICS Transaction Server for z/OS®, Version 4 Release 1.

“RACF” is used to mean the IBM Resource Access Control Facility (RACF) or any other external security manager that provides equivalent function.

In the programming examples in this book, the dollar symbol (\$) is used as a national currency symbol and is assumed to be assigned the EBCDIC code point X'5B'. In some countries a different currency symbol, for example the pound symbol (£), or the yen symbol (¥), is assigned the same EBCDIC code point. In these countries, the appropriate currency symbol should be used instead of the dollar symbol.

“MVS” is used throughout this book to mean the MVS operating system.

Book structure

The book consists of the following parts.

Part1. Defining data sets

Describes the data sets needed by the various CICS facilities. Each chapter describes the facility, its function and usage, and the data sets required to implement it in a running CICS region.

Part 2. System initialization

Describes the system initialization parameters, that you can code to initialize a CICS region tailored for your installation.

Part 3. Initializing CICS data sharing servers

Describes how to set up and start CICS data sharing servers.

Changes in CICS Transaction Server for z/OS, Version 4 Release 1

For information about changes that have been made in this release, please refer to *What's New* in the information center, or the following publications:

- *CICS Transaction Server for z/OS What's New*
- *CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.2*
- *CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.1*
- *CICS Transaction Server for z/OS Upgrading from CICS TS Version 2.3*

Any technical changes that are made to the text after release are indicated by a vertical bar (|) to the left of each new or changed line of information.

Part 1. Defining data sets

This section of the book describes the CICS system data sets needed to support various CICS facilities, such as temporary storage, transient data, transaction dump, trace, and so on. Some of these data sets are optional, others are required in order to run CICS. If a data set needs preformatting, a job that you can use for this purpose is shown.

The CICS facilities and their data sets are dealt with in the following sections:

- Chapter 1, "Setting up CICS data sets," on page 3
- Chapter 2, "Setting up the temporary storage data set," on page 15
- Chapter 3, "Setting up data sets for transient data," on page 21
- Chapter 4, "Setting up CICS log streams," on page 27
- Chapter 5, "Setting up the CICS system definition data set," on page 41
- Chapter 6, "Setting up the catalog data sets," on page 65
- Chapter 7, "Setting up and using auxiliary trace data sets," on page 77
- Chapter 8, "Defining dump data sets," on page 81
- Chapter 9, "Defining the CICS availability manager data sets," on page 87
- Chapter 10, "Defining user files," on page 93
- Chapter 11, "Defining the CDBM GROUP command data set," on page 113
- Chapter 12, "Defining the CMAC messages data set," on page 117
- Chapter 13, "Defining the EJB data sets," on page 119
- Chapter 15, "Setting up the debugging profiles data sets," on page 125
- Chapter 27, "CICS server support for system-managed processes," on page 423

Chapter 1. Setting up CICS data sets

You must define a number of data sets to run CICS. Some of these data sets are mandatory, whereas others are needed only if you are using the corresponding facilities. You might also need to provide data set definitions for user files, DL/I databases, and terminals other than VTAM[®] terminals.

About this task

Procedure

1. Plan what CICS data sets you require.
 - a. Review the CICS facilities that you want in your CICS regions and their data set requirements.
 - b. Define a naming convention for your data sets.
 - c. Calculate the space to allocate to the data sets and the data definition statements to define them to the running CICS region.
2. Define and catalog the data sets that are used by CICS.
3. If required, initialize or preformat the data sets that are used by CICS.
4. Protect the data sets with an external security manager, such as RACF, to suit your security requirements.
5. Define the DD statements for the required data sets in the CICS startup job stream. You do not have to define DD statements for the following data sets:
 - User files for which you are using CICS dynamic allocation facilities
 - CICS system data sets that are managed by CICS file control and for which you are using CICS dynamic allocation facilities
 - DL/I databases you are accessing through CICS remote DL/I support or DBCTL

For more information about user file definitions, see Chapter 10, “Defining user files,” on page 93.

What to do next

When you have defined the CICS data sets, you can use CICS utility programs to perform postprocessing on the data sets. These utilities are described in the *CICS Operations and Utilities Guide*

Reviewing the CICS data sets

Review the list of mandatory and optional data sets to select which ones to create in your CICS regions.

Procedure

1. You must define the following CICS data sets:

| Data set | DDNAME | Block or CI size (bytes) | Record format | Data set organization |
|--|--------------------|--------------------------|---------------|-----------------------|
| Auxiliary trace (See Setting up and using auxiliary trace data sets) | DFHAUXT DFHBUXT | 4096 | F | Sequential |

| Data set | DDNAME | Block or CI size (bytes) | Record format | Data set organization |
|---|---|--|--|-----------------------|
| BTS local request queue. (See <i>CICS Business Transaction Services</i>) | DFHLRQ | 1024 and 2560 | VB | VSAM KSDS |
| Catalogs (See Setting up and using catalog data sets) | DFHGCD DFHLCD | 8192 & 2048 | VB | VSAM KSDS |
| CSD (See Setting up the CICS system definition data set) | DFHCSD | 8192 | VB | VSAM KSDS |
| Dump (See Defining dump data sets) | DFHDMPA DFHDMPB | 32 760 (tape) or 1 track (DASD) | V | Sequential |
| Messages and codes (See Defining the CMAC messages data set) | DFHCMACD | — | V | VSAM KSDS |
| Temporary storage (See Setting up the temporary storage data set) | DFHTEMP | See Control interval size of temporary data set | N/A | VSAM ESDS |
| Transient data extrapartition (See “Defining extrapartition data sets” on page 24) | From the DDNAME option in the resource definition | From the BLOCKSIZE option in the resource definition | From the RECORD FORMAT option in the resource definition | Sequential |
| Transient data intrapartition (See “Defining the intrapartition data set” on page 21) | DFHINTRA | See “Size of the intrapartition data set” on page 23 | N/A | VSAM ESDS |

Temporary storage and transient data intrapartition data sets use control interval (CI) processing and therefore the record format is not relevant.

- You must define the following CICS data sets if you plan to use the equivalent functions:

| Data set | DDNAME | Block or CI size (bytes) | Record format | Data set organization | Function |
|--------------------|-----------|--------------------------|---------------|-----------------------|---|
| CAVM control | DFHXRCTL | 4096 minimum | N/A | VSAM ESDS | Required if running CICS with XRF. See Defining the CICS availability manager data sets |
| CAVM message | DFHXRMMSG | 4096 minimum | N/A | VSAM ESDS | Required if running CICS with XRF. See Defining the CICS availability manager data sets |
| CDBM group command | DFHDBFK | 8192 | VB | VSAM KSDS | See Defining the CDBM GROUP command data set |
| DJAR mapping file | DFHADJM | 8192 | F | VSAM KSDS | Required for CICS EJB development deployment tool only. This data set must not be shared. See Chapter 13, “Defining the EJB data sets,” on page 119 |

| Data set | DDNAME | Block or CI size (bytes) | Record format | Data set organization | Function |
|------------------|----------|--------------------------|---------------|-----------------------|---|
| EJB directory | DFHEJDIR | 1024 | F | VSAM KSDS | Required only for use by the regions (listeners and AORs) in a logical EJB server. This data set must be shared by all the regions in the EJB server. See Chapter 13, "Defining the EJB data sets," on page 119 |
| EJB object store | DFHEJOS | 8192 | F | VSAM KSDS | Required only for use by the regions (listeners and AORs) in a logical EJB server. This data set must be shared by all the regions in the EJB server. See Chapter 13, "Defining the EJB data sets," on page 119 |

The CAVM control and message data sets use control interval (CI) processing and therefore the record format is not relevant.

Example

What to do next

Defining a data set naming convention

There are no restrictions on the data set names you choose for CICS data sets, other than MVS constraints. In this information, CICS41.CICS is used as the high-level qualifier, and the DD name as the lowest level. If you are running multiple CICS regions, and especially if you are running CICS with XRF, you can use the CICS APPLID as a second level qualifier.

About this task

If the data set is shared between an active CICS region and an alternate CICS region, use the generic APPLID, but if the data set is unique to either the active or the alternate CICS region, use the specific APPLID. For information about actively and passively shared data sets, see "Planning data sets for XRF" on page 8.

Procedure

- For 4-character names, use the *CTGI* naming convention, as described in *z/OS Parallel Sysplex Application Migration*. It is based on the 4-character *CTGI* symbol, where:
 - C identifies an entire CICSplex
 - T identifies the type of region
 - G identifies a group of regions
 - I identifies iterations of regions within a group
- For 8-character names, as for CICS APPLIDs, use the letters CICS for the first four characters, particularly for production regions. For example, if CICSHTH1 is the APPLID, the data set name for the CSD would be:

```
DFHCSD DD DSN=CICSTS41.CICS.CICSHTH1.DFHCSD,DISP=SHR
```

If the data set is shared between an active CICS region and an alternate CICS region, use the generic APPLID. If the data set is unique to either the active or the alternate CICS region, use the specific APPLID. For information about actively and passively shared data sets, see "Planning data sets for XRF" on page 8.

Defining data sets with multiple extents and volumes

You can define a temporary storage data set or a transient data destination data set, as a single extent defined on a single volume. That data set must be big enough to hold all your data.

Instead of defining one data set, which to cater for exceptional cases might have to be much larger than your average, you can define:

- Multiple extents on one volume
- One extent on each of multiple volumes
- Multiple extents on multiple volumes

When you define more than one extent, CICS uses the extra extents only when the primary extent is full. You could make your primary extent large enough to meet average demand, and then have smaller secondary extents for overflow. In this way, you save space until it is needed. When each extra extent becomes full, VSAM creates another. VSAM continues to create extra extents, as needed, up to a maximum of 123 extents. The use of multiple volumes has no effect on this limit.

To allocate additional extents in the same volume, code a secondary extent operand on the RECORDS parameter:

```
RECORDS(primary,secondary)
```

To use single extents on multiple volumes, code:

```
RECORDS(primary) -  
VOLUMES(volume1,volume2,volume3,.....)
```

For multiple extents on multiple volumes, combine both primary and secondary RECORDS operands with multiple VOLUMES operands:

```
RECORDS(primary,secondary) -  
VOLUMES(volume1,volume2,volume3,.....)
```

If a particular volume causes performance bottlenecks, try single extents on multiple volumes.

Use multiple extents over multiple volumes if there is a probability that a volume will exhaust its free space before VSAM reaches its limit on extra extents. If this occurs, VSAM continues to create extra extents on the next volume in the list.

Performance considerations of TS and TD buffers

When specifying the number of buffers for temporary storage and transient data, there might be a performance impact depending on how many you specify.

Using a large number of buffers means that for nonrecoverable queues processing can often be performed without calling VSAM. This improves CICS performance. However, at shutdown all non-empty buffers have to be flushed sequentially which can take a long time.

Creating the CICS data sets

You can use CICS-supplied jobs to create the CICS data sets.

About this task

When you ran the DFHISTAR job as part of the CICS installation, these jobs were tailored to your environment and stored in the library that you specified on the **LIB** parameter of the DFHISTAR job. The default is CICSTS41.XDFHINST).

You can generate several copies of these jobs by rerunning the DFHISTAR job, selecting the jobs that you want to copy. To generate new copies of these jobs, edit the DFHISTAR job to specify new values for the **DSINFO** and **SELECT** parameters. Only those jobs that you name by the **SELECT** parameter are regenerated.

Procedure

1. Run the DFHCOMDS job to create the CICS region definition data set, DFHCSD, and the SYSIN data set.
2. Run the DFHDEFDS job to create data sets used only by one CICS region. You must run a separate copy of this job to create the data sets for each CICS region. This job creates the following data sets:

| | |
|-----------|--|
| DFHADJM | Application deployment DJAR mapping file |
| DFHAUXT | non-VSAM auxiliary trace (A) data set |
| DFHBUXT | non-VSAM auxiliary trace (B) data set |
| DFHDMPA | non-VSAM dump (A) data set |
| DFHDMPB | non-VSAM dump (B) data set |
| DFHEJDIR | EJB directory |
| DFHEJOS | EJB object store |
| DFHGCD | CICS global catalog |
| DFHINTRA | intrapartition transient data set |
| DFHLCD | CICS local catalog |
| DFHLRQ | BTS local request queue |
| DFHTEMP | temporary storage data set |
| DFHXRCTL | XRF control data set |
| DFHXRMMSG | XRF message data set |
| FILEA | sample program file |

3. Run the DFHCMACI job to create the CICS messages data set, DFHCMACD, and load it with the data from the CICS-supplied file, DFHCMACD, in the CICSTS41.CICS.SDFHMSG target library.

Sizing the MVS system data sets

Besides its own system data sets, CICS also uses some MVS data sets.

About this task

These data sets are:

| Data set | Owned or used by | Other comments |
|-----------------|------------------|--|
| SDUMP data sets | MVS SDUMP macro | Used by CICS for system dumps through the MVS SDUMP macro. |

| Data set | Owned or used by | Other comments |
|---------------|----------------------------|---|
| SMF data sets | System management facility | Used by CICS monitoring and statistics domains for monitoring and statistics records. |
| GTF data sets | Generalized trace facility | Used by CICS trace domain for CICS trace entries. |

Procedure

1. Recalculate the size of these system data sets, taking into account the increased volumes of data that CICS generates. For example, for an SDUMP data set you must have at least 25 cylinders of a 3380 device, or the equivalent. For guidance information about calculating the size of SDUMP data sets, see the *z/OS MVS Initialization and Tuning Guide* manual.
2. To prevent the SDUMP data sets from becoming full of unwanted SDUMPs, suppress the SDUMPs as described in “Suppressing system dumps that precede ASRx abends” on page 82. SDUMPs can precede ASRA, ASRB, and ASRD abends after the message DFHAP0001.
3. If you are collecting CICS interval statistics frequently, or the volume of statistics at each interval is high, then you must take this into account when sizing your SMF data sets.

CICS can write records to SMF of up to 32 756 bytes, resulting in SMF writing spanned records to the SMF data sets. For more efficient use of DASD, you should consider creating the SMF data sets to be used by CICS with a control interval size of either 16 384 bytes (16 KB) or 8192 bytes (8 KB). If you use other control interval sizes you must consider the trade-off between efficient use of DASD, SMF data set I/O performance, and the possibility of data being lost due to insufficient SMF buffers.

4. If you are collecting CICS monitoring data, you must size the amount of data that is written when the monitoring classes are active. For background information about SMF, and about other SMF data set considerations, see the *z/OS MVS System Management Facilities (SMF)*.
5. If you are running CICS with GTF trace on, make allowance for CICS trace entries in the GTF data sets. For background information about GTF, see *z/OS MVS Diagnosis: Tools and Service Aids*.

Planning data sets for XRF

You must consider the type of data set sharing that takes place for CICS system data sets and user application data sets, between the active and alternate CICS regions, and the data set allocation and disposition.

About this task

Consider the following general points when running CICS in an XRF environment:

- An alternate CICS region can be started before an active CICS region terminates.
- The active and alternate CICS regions might be running in different MVS images.

The status and location of the data sets used by CICS become very important. In particular, consider the following points:

Procedure

1. For a given file name, do the active and alternate CICS regions:
 - Refer to separate data sets?
 - Refer to the same data set?
2. For a given data set, is it required by the alternate CICS region:
 - Before takeover occurs?
 - After takeover occurs?
3. For a given data set, is it allocated:
 - At job step initiation?
 - Dynamically?
4. What facilities of MVS global resource serialization (GRS) or JES3 are being used?

MVS does not prevent conflicting concurrent use of a data set residing on shared DASD by two or more jobs running in different MVS images. To prevent concurrent use, you can use either global resource serialization (GRS) or JES3, to provide global data set enqueueing in a multi-MVS environment.

XRF shared and unique data sets

A shared data set, in XRF terms, means one that is required by both the active and alternate CICS regions, though not necessarily concurrently. In an XRF environment, CICS classifies data sets as actively shared, passively shared, or unique.

Actively shared data sets

Actively shared data sets are required for use in both the active and alternate CICS regions. There are only two CICS system data sets in this category, called the CICS availability manager (CAVM) data sets. The active and the alternate CICS regions each open these data sets during CICS initialization, and the data sets are shared while both CICS regions are running. They are:

- CAVM control data set
- CAVM message data set

It is not usual for user application data sets to be actively shared.

Passively shared data sets

Passively shared data sets are required by both the active and alternate CICS regions, but not at the same time.

Initially, they are opened by the active CICS region, and are only opened by the alternate CICS region during or following takeover. Thus in an XRF environment, passively shared data sets are said to be “owned” by the active CICS region. The CICS system data sets in this category are:

- CICS system definition data set (DFHCSD)
- Global catalog data set (DFHGCD)
- Temporary storage data set (DFHTEMP)
- Transient data intrapartition data set (DFHINTRA)

User data sets managed by CICS file control, and DL/I data sets, are also passively shared.

Unique data sets

These data sets are unique to either the active or the alternate CICS region, and are not shared in any way.

The CICS system data sets in this category are:

- CICS local catalog data set (DFHLCD)
- Dump data sets (DFHDMPx)
- Auxiliary trace data sets (DFHAUXT and DFHBUXT)

User application data sets are not usually unique.

Defining and allocating XRF data sets

The allocation of data sets, and how you specify the **DISP** parameter, are important when running CICS with XRF.

Before you begin

About this task

Define the data sets to CICS and MVS using DD statements so that they are allocated at job step initiation. The value that you code for the **DISP** parameter is critical for all shared data sets. If you are using dynamic allocation, the alternate CICS region does not allocate any data sets dynamically before takeover occurs.

Procedure

- Specify **DISP=SHR** to allow data sets to be allocated concurrently by the active and alternate CICS regions. It also allows the data sets to be allocated by jobs other than the active and alternate CICS regions. If this risk proves unacceptable for BDAM and VSAM user files and for DL/I databases, consider using dynamic allocation with **DISP=OLD**, or enable security.
- Specify **DISP=NEW**, **DISP=OLD**, or **DISP=MOD** to request exclusive use of a data set. It might not be possible to start the alternate CICS region before the active region ends.

What to do next

When you run CICS with XRF, CICS always ensures (except for the CSD) that there is no conflicting concurrent use of data sets by an active CICS region and its alternate CICS region, even though they are running in different MVS images.

For more information about sharing the CSD, see “Sharing a CSD in a multi-MVS environment (non-RLS)” on page 50.

Defining backup while open (BWO) for VSAM files

CICS supports the backup while open (BWO) facility provided by DFSMSdss and DFSMSHsm. This support enables some types of VSAM data sets to be backed up by DFSMSdss while CICS is currently updating these data sets.

At the same time, CICS logs forward recovery images of any changes to these data sets on a forward recovery journal. At a later date the backup of the data set can

be restored using DSMSshm and brought to a point of consistency by applying the forward recovery logs using a forward recovery utility such as CICS VSAM Recovery.

Before you begin

BWO is available only for data sets accessed by CICS file control, which includes the CICS system definition (CSD) data set. You must decide which VSAM user data sets are eligible for BWO, subject to the restrictions detailed in “Restrictions on BWO” on page 12 that are applicable to heavily-updated KSDS data sets.

VSAM data sets that are to use this facility must reside on SMS-managed DASD, and must have an ICF catalog structure. Only VSAM ESDS, RRDS (both fixed and variable), and KSDS data sets are supported.

About this task

For DFSMS 1.3, there are two ways of defining BWO:

Procedure

- Define the cluster with parameter **BWO=TYPECICS**. This value means that the cluster is eligible for BWO in a CICS region. The file resource definition is ignored, even if it conflicts.

Clusters with data sets that are to be opened in RLS mode must have BWO specified in the cluster definition.

- If the BWO parameter is not defined, it defaults to UNDEFINED. In this case, CICS looks at the file resource definition.

CICS defines a data set as eligible for BWO when a file is defined using RDO. If BACKUPTYPE=DYNAMIC is specified for a VSAM file, the file is defined as eligible for BWO when the data set is opened.

If DFSMSdss is to back up a data set that is specified with BACKUPTYPE=STATIC, all CICS files currently open for update against that data set must be closed before the backup can start.

Results

CICS records the fact that a VSAM base cluster data set is eligible for BWO in its base cluster block. This is remembered when all files have closed against the VSAM base cluster and across CICS warm and emergency restarts. It is not remembered across CICS cold or initial starts.

What to do next

You must put appropriate procedures into place for BWO and for forward recovery. These procedures must include:

- Restoring the BWO backup and running the forward recovery utility to bring the data set to a point of consistency. The restore requires that users do not have access to the file during the recovery process.
- Restoring and forward recovery of data sets that might have been damaged while allocated to CICS. This operation might require backout of partially committed units of work, by CICS emergency restart.

Effect of disabling activity keypointing

If you disable activity keypointing in your CICS region by specifying the system initialization parameter `AKPFREQ=0`, there is a serious effect on BWO support for non-RLS activities.

When activity keypointing is disabled, no tie-up records (TURs) are written to the forward recovery logs and the data set recovery point is not updated. Therefore, forward recovery of a BWO backup must take place from the time that the data set was first opened for update. This requires that all forward recovery logs are kept since that time so that forward recovery can take place. If there are many inserts or records that change length, a lot of forward recovery could be required. If, however, a record is just updated and the length is unchanged, there is no CI split.

For information about TURs and recovery points, see the *CICS Recovery and Restart Guide*.

Restrictions on BWO

The following restrictions apply to VSAM KSDS data set types only.

If a VSAM control interval or control area split occurs while a BWO is in progress, the backup is unreliable and is discarded by DFHSM and DFDSS. During such a split, certain portions of the data set may be duplicated or not represented at all in the backup as DFDSS copies sequentially. MVS/DFP 3.2 indicates that a split has occurred in the ICF catalog. At the end of the backup, DFHSM and DFDSS check the ICF catalog, and if a split has occurred, or is still in progress, discard the backup. For this reason, certain heavily-updated VSAM KSDS data sets may not be eligible for BWO, or might be eligible only during periods of reduced activity (for example, overnight). For a KSDS data set to be eligible for BWO, the typical time between control interval or control area splits must be greater than the time taken for DFHSM and DFDSS to take a backup of the data set.

XRF considerations

CICS XRF regions take keypoints more frequently than non-XRF regions.

If BWO is used, extra activity occurs during keypoints, because:

- Extra “tie-up” records (TURs) are written to associate file names with their associated data set names, and
- The ICF catalog is updated to record the recovery time (the time from which the forward recovery utility must start applying log records).

These actions are performed approximately once every 30 minutes.

Using storage management facilities

BWO requires a number of storage management facilities.

About this task

These storage management facilities are :

- Storage management subsystem (SMS), part of MVS/DFP Version 3 Release 2 or later, product number 5665-XA3.
- Data facility hierarchical storage manager (DFHSM), product number 5665-329.
- Data facility data set services (DFDSS), product number 5665-327.

For more information about these facilities see the following sections.

Storage management subsystem (SMS)

SMS is the approach to DASD storage management in which CICS, by means of the storage management subsystem, determines data placement. In addition, an automatic data manager handles data backup, movement, space, and security.

This is sometimes referred to as DFSMS and complements functions of MVS/DFP and other individual products of the Data Facility product family. For more details about SMS, see the following publications:

- *MVS/DFP Version 3 Release 2: Storage Administration Reference*, SC26-4566

This describes storage administrator applications.

- *MVS/DFP Version 3 Release 2: General Information*, SC26-4552

This gives an overview of MVS/DFP and its requirements and describes concepts of SMS-managed storage.

Message on recall of backed up data sets

If you use DFHSM to manage your VSAM data sets, you should consider carefully the period after which your CICS VSAM data sets are migrated to primary or secondary storage.

If a migrated data set has to be recalled for CICS, it can take several minutes from primary storage, or longer from secondary storage. While the recall is taking place, the user is locked out, and no other opens or closes on that data set can be performed until the data set has been recalled.

If a migrated data set has to be recalled, CICS issues message DFHFC0989 to the system console, to notify the user that a recall is taking place, and to indicate whether it is from primary or secondary storage.

Data facility hierarchical storage manager (DFHSM)

DFHSM is an IBM-licensed program to manage volumes and data sets.

For more details about DFHSM, see the *Data Facility Hierarchical Storage Manager Version 2 Release 5.0: General Information*, GH35-0092. This manual discusses the main features, options, and potential benefits of DFHSM, and addresses people who want to know what the DFHSM program can do for them.

Data facility data set services (DFDSS)

DFDSS is an IBM-licensed program used to copy, move, dump, and restore data sets and volumes.

For more details about DFDSS, see the *Data Facility Data Set Services Version 2 Release 5.0: General Information*, GC26-4123. This manual introduces DFDSS and helps you evaluate its use.

Chapter 2. Setting up the temporary storage data set

CICS provides the temporary storage facility to enable application programs to hold data, created by one transaction, for use later by the same transaction or by a different transaction.

About this task

You save data in temporary storage queues that are identified by symbolic names. Temporary storage queues can be in main storage, VSAM-managed auxiliary storage, or temporary storage pools in the coupling facility.

You would use main storage if:

- Data is needed for only short periods of time
- Data does not need to be recoverable
- Only small amounts of data are to be stored

You would use auxiliary temporary storage if:

- Large amounts of data are to be stored
- Data is to be kept for extended periods of time
- Data is to be maintained from one CICS run to the next

You define auxiliary temporary storage as a nonindexed VSAM data set. CICS uses control interval processing when storing or retrieving temporary storage records in this data set. A control interval usually contains several records. Temporary storage space within a control interval is reusable.

Temporary storage queues can also reside in queue pools in a coupling facility. This applies to nonrecoverable queues which might be written to and read from different CICS regions. For more information about temporary storage data sharing, see “Defining temporary storage pools for temporary storage data sharing” on page 17. For background information about CICS temporary storage, see the *CICS Application Programming Guide*.

Defining the temporary storage data set

You can either define a VSAM data set for auxiliary temporary storage as a single extent data set on a single volume using the sample job described here, or you can use the CICS-supplied job DFHDEFDS. DFHDEFDS creates the DFHTEMP data set as one of the data sets for a CICS region.

About this task

You must not define any extra associations for a temporary storage data set. Do not, for example, define a PATH. Doing so causes CICS startup to fail. Do not allocate the DFHTEMP data set from an SMS data class using extended addressability because CICS does not support this.

Procedure

1. Decide if defining the data set for a single-extent data set defined on a single volume, as described in the sample job, is going to have enough storage. Instead of defining one data set, which might have to be much larger than your

average needs to cater for exceptional cases, you can define multiple extents and multiple volumes. For more information about defining these, see “Defining data sets with multiple extents and volumes” on page 6.

2. Specify a **RECORDSIZE** value that is 7 bytes less than the **CONTROLINTERVALSIZE**. You must specify the amount of space for temporary storage in two values:
 - a. The control interval size. See “The control interval size” for information about how to calculate the space.
 - b. The number of control intervals in the data set. See “Number of control intervals” on page 17 for information about how to set the correct number of control intervals.
3. Add a data definition statement for the DFHTEMP data set to the startup job stream. The temporary storage data set is a passively shared data set, owned by the active CICS region, but allocated to both the active and alternate CICS regions. Although the alternate CICS region does not open this data set before takeover, it is allocated at job step initiation, so you must specify **DISP=SHR** on the DD statement to enable the alternate CICS region to start.

```
//DFHTEMP DD DSN=CICSTS41.CICS.app1id.DFHTEMP,DISP=SHR
```

Example

```
//DEFTS JOB accounting info,name
//AUXTEMP EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
  DEFINE CLUSTER(NAME(CICSTS41.CICS.CNTL.CICSqualifier.DFHTEMP)-
    RECORDSIZE(4089,4089) -
    RECORDS(200) -
    NONINDEXED -
    CONTROLINTERVALSIZE(4096) -
    SHAREOPTIONS(2 3) -
    VOLUMES(vol1id) -
    DATA(NAME(CICSTS41.CICS.CNTL.CICSqualifier.DFHTEMP.DATA) -
    UNIQUE)
/*
```

Figure 1. Sample job defining an auxiliary temporary storage data set

The control interval size

You specify the control interval size with the **CONTROLINTERVALSIZE** parameter in the VSAM CLUSTER definition. Because a control interval contains one or more temporary storage records, take the temporary storage record size into account when choosing the control interval size.

The following factors affect your choice:

- Each temporary storage record must have space for:
 - The data
 - 36 bytes (for the temporary storage header)

If you install BMS with 3270 support, the data length of the record is at least as large as the 3270 buffer size. For 3270 terminals with the alternate screen size facility, the data length is the larger of the two sizes.

The total number of bytes allocated for a temporary storage record (including the 36-byte header) is rounded up to a multiple of 64 (for control interval sizes less than, or equal to, 16 384), or a multiple of 128 (for larger control interval sizes).

- The control interval size must be large enough to hold at least one (rounded up) temporary storage record, including 64 bytes of VSAM control information for

control interval sizes less than or equal to, 16 384, or 128 bytes of control information for larger control interval sizes. The maximum control interval size is 32 KB.

Choose a control interval size large enough to hold the largest normally occurring temporary storage record, together with the VSAM control information. Oversize records are split across control intervals, but this might degrade performance. Control interval sizes must be multiples of 512 bytes when smaller than 8 KB and must be a multiple of 2 KB when equal to or larger than 8 KB.

Example

If you use BMS to write a 24 x 80 character screen to temporary storage, the data written occupies 1920 bytes. You require 36 bytes for the CICS temporary storage header, giving a total of 1956 bytes. Rounding this up to a multiple of 64 gives 1984 bytes. Finally, adding a further 64 bytes of VSAM control information gives a control interval size of 2048 bytes. Typically, the control interval size is larger than 2048 bytes to hold several records possibly differing in size.

Number of control intervals

VSAM uses the RECORDS and RECORDSIZE operands to allocate enough space for the data set to hold the number of records of the specified size.

You must code the same value for the two operands of the RECORDSIZE parameter (the average and maximum record sizes), and this value must be 7 bytes less than the CONTROLINTERVALSIZE. In this way, the specified number of VSAM records matches the number of control intervals available to temporary storage management. You thus specify, indirectly, the number of control intervals in the temporary storage data set. (Note that the RECORDS and RECORDSIZE parameters do not correspond to the temporary storage records as seen at the CICS temporary storage interface.)

The number of control intervals to be allocated depends on user and system requirements for temporary storage, up to the maximum number permitted of 65 535.

Number of VSAM buffers and strings

You can use the TS system initialization parameter to specify the number of CICS temporary storage buffers up to the maximum of 32 767.

The number of buffers that you specify may have an effect on CICS performance, as described in "Performance considerations of TS and TD buffers" on page 6. You should specify a value to suit your CICS region. If you specify `TS=(0)`, requests for auxiliary temporary storage are executed using main storage.

Defining temporary storage pools for temporary storage data sharing

Using TS data sharing means replacing main or auxiliary storage for your TS queues with one or more TS pools, where the scope and function of each TS pool is similar to a QOR.

About this task

Each TS pool is defined, using MVS cross-system extended services (XES), as a keyed list structure in a coupling facility. This means you must define the pool

using the coupling facility resource manager (CFRM) policy statements. Using the CFRM policy definition utility, IXCMIAPU, you specify the size of the list structures required, and their placement within a coupling facility. For an example of this utility, see member IXCCFRMP in the SYS1.SAMPLIB library, in the *z/OS MVS Setting Up a Sysplex* manual. An example of a definition statement is shown in Figure 2.

```
STRUCTURE NAME(DFHXQLS_PRODTSQ1)
  SIZE(1000)
  INITSIZE(500)
  PREFLIST(FACIL01,FACIL02)
```

Figure 2. Example of defining the estimated size of a list structure

The name of the list structure for a TS data sharing pool is created by appending the TS pool name to the prefix DFHXQLS_, giving DFHXQLS_*poolname*. When defined, you must activate the CFRM policy using the MVS operator command SETXCF START.

When a list structure is allocated, it may have an initial size and a maximum size specified in the CFRM policy. (All structure sizes are rounded up to the next multiple of 256K at allocation time). Provided that space is available in the coupling facility, a list structure can be dynamically expanded from its initial size towards its maximum size, or contracted to free up coupling facility space for other purposes. Note that if the initial structure allocation becomes full, the structure does not expand automatically, even if the structure allocated is less than the specified maximum size. To expand a list structure when the allocation becomes full, you can expand it (up to its maximum size) using the following SETXCF command:

```
SETXCF START,ALTER,STRNAME=DFHXQLS_poolname,SIZE=nnnn
```

Approximate storage calculations

A coupling facility structure contains not only stored data but also the information needed to manage and access that data, in a similar way to a key-sequenced data set.

The data for each entry in the coupling facility is stored as a chain of fixed size (usually 256-byte) elements, which means that the exact length for variable-length data has to be stored separately. CICS does this by including a length prefix in the stored data, so space calculations have to allow for each entry using a whole number of elements. The amount of internal control information depends on the level of functionality and performance of the coupling facility control code for the current CFLEVEL. The storage requirements can increase for a higher CFLEVEL.

The easiest way to calculate accurate structure storage requirements is to use the Web-based IBM CFSizer tool at <http://www.ibm.com/systems/support/z/cfsizer/>. The tool takes these factors into account and communicates with a coupling facility at a current CFLEVEL to perform the calculation. You need to enter some minimum input to get an accurate storage calculation as follows:

MAXQUEUES

specifies the maximum number of data lists that are reserved when the structure list is allocated by CICS. This determines the maximum number of large queues that can be stored in the structure and corresponds to the MAXQUEUES server parameter. A large queue is where the total size of

the data items exceeds 32K. It is stored in a separate list in the structure. Although you should specify a large enough number, specifying an excessively large number will use up an unnecessary amount of coupling facility storage for unused preallocated list headers. The valid range is from 1 to 999999. The default is 1000.

Average rounded item size

specifies the average amount of storage required for each TS queue item. Note that each item has a two-byte length prefix and is stored as one or more 256-byte elements. This number should range from 1 to 32768. This value determines the entry to element ratio that is used to calculate the required structure size.

If all queue items are approximately the same size, calculate this value by taking the average data size, adding two, and rounding up to the next multiple of 256. If queue items are different sizes, round up each size first before taking the average. For example, if half of the items are 100 bytes and half are 300 bytes, then the rounded sizes are 256 and 512. The average rounded item size is half way between those values, 384, which is more accurate than taking the overall average item size of 200 and rounding it up to 256.

Total number of items in all queues

specifies the total number of entries in all of the TS queues.

Target Usage Percent

specifies the percentage of the structure space which the given total number of items are expected to use. Specify a number in the range of 1 to 100. The default is 75. This leaves some free space for temporary expansion, and to give time to expand the structure in response to warning messages (which normally start at 80%) if the initial free space is not enough.

Max Expansion Percent

specifies the percentage that the structure can expand. If a non-zero value is specified here, the maximum structure size will be greater than the initial structure size by an amount sufficient to allow for a further expansion of the total amount of data by this percentage. For example, if the value 200 is specified, the initial size will be enough to store the specified total number of items, but the maximum size will be enough to store three times that number of items.

Alternatively you can use the following calculation, which applies up to CFLEVEL 11 of the coupling facility. If you have a coupling facility at CFLEVEL 12, you need to change the calculation to add 10% and 2MB to the sizings in the formula below.

Storage calculations

$$\text{Item entry size} = (170 + (\text{average item size, rounded up to next 256})) + 5\% \text{ extra for control information}$$

$$\begin{aligned} \text{Total size} &= 200\text{K} \\ &+ (\text{number of large queues} \times 1\text{K}) \\ &+ (\text{number of items in all queues} \times \text{item entry size}) \end{aligned}$$

The calculation determines the size of a structure that would be approximately 100% full for the specified level of usage. For practical operation, however, a reasonable proportion of free space must be available, not only to minimize the

risk of the structure becoming totally full but also to avoid triggering low space warning messages and additional activity to alter entry to element ratios. The maximum normal usage should therefore aim to be about 75% of the structure size. Expected usage values should therefore be adjusted upwards to allow for the required amount of free space (for example, by about one third to aim for 75% usage).

Note:

1. If you have a small queue with multiple items that does not exceed 32K, then all the queue items are stored as the data portion of the queue index entry. If the queue exceeds 32K, then it is converted to a form where one item per entry is stored in a separate list and referred to by the queue index entry.
2. The amount of element storage required is two bytes more than the data item size due to the length prefix on each item.
3. Defining the CFRM policy statements for a list structure does not create the list structure, this is done by a TS server during its initialization.

For information about the TS server system initialization parameters, see Chapter 22, "Setting up and running a temporary storage server," on page 349.

For information about defining list structures, see the following MVS publications:

- *z/OS MVS Setting Up a Sysplex*
- *z/OS MVS Programming: Sysplex Services Guide*
- *z/OS MVS Programming: Sysplex Services Reference*

Chapter 3. Setting up data sets for transient data

Data sets that are used for transient data queues can be intrapartition or extrapartition.

About this task

The transient data intrapartition data set is a VSAM entry-sequenced data set (ESDS) that is used for queuing messages and data within the CICS region. Transient data extrapartition data sets are sequential files, normally on disk or tape; you can use the queue either to send data outside the CICS region or to receive data from outside the region.

Messages or other data are addressed to a symbolic queue that you define as either intrapartition or extrapartition using the CEDA transaction. The queues can be used as indirect destinations to route messages or data to other queues. System messages that CICS produces are commonly sent to transient data queues, either intrapartition or extrapartition.

Procedure

1. Define all of the queues that CICS uses in your CICS region. Although the omission of any of the queues does not cause a CICS failure, you lose important information about your CICS region if CICS cannot write its data to the required queue.
 - a. Use the sample definitions in group DFHDCTG to define the required queues. See Required TDQueue definitions for a summary.
 - b. Take a backup copy of the changes that you made to DFHDCTG. When maintenance is applied, the DFHDCTG group might be refreshed, overwriting your changes. Taking a backup avoids this problem.
2. Define the intrapartition data set using job control statements. See “Defining the intrapartition data set” for details on how to do this step.
3. Define the extrapartition data sets using job control statements. See “Defining extrapartition data sets” on page 24 for details on how to do this step.
4. To print CICS system messages on a local printer as they occur, use the transient data write-to-terminal sample program, DFH\$TDWT. This sample program is supplied with the CICS in CICSTS41.CICS.SDFHLOAD and the assembler source is in CICSTS41.CICS.SDFHSAMP.

Defining the intrapartition data set

You can either use the sample job described here to define the transient data intrapartition data set, or you can use the CICS-supplied job, DFHDEFDS. DFHDEFDS creates the DFHINTRA data set as one of the data sets for a CICS region

About this task

The intrapartition data set must be big enough to hold all the data for intrapartition queues. If you are using the sample job to define the transient data intrapartition data set, perform the following steps:

Procedure

1. Decide if a single extent data set on a single volume is appropriate. If you define one extent on one volume, you might require a much larger data set than your average requirements to cater for exceptional cases. You can define multiple extents or multiple volumes or both for the data set. For details, see “Using multiple extents and multiple volumes” on page 23.
2. Specify a **CONTROLINTERVALSIZE** parameter that is large enough to hold the longest data record, in addition to the 32 bytes that CICS requires for its own purposes. The maximum control interval size is 32 KB. Space is allocated to queues in units of a control interval (CI). The first CI is reserved for CICS use; the remaining CIs are available to hold data. Data records are stored in CIs according to VSAM standards.
3. If you allocate space in records, rather than tracks or cylinders, you must specify a **RECORDSIZE** value. The value must be 7 bytes less than the **CONTROLINTERVALSIZE**.
4. Use the **TD** system initialization parameter to specify the number of CICS transient data buffers up to the maximum of 32 767. The number of buffers that you specify may have an effect on CICS performance, as described in “Performance considerations of TS and TD buffers” on page 6. You should specify a value to suit your CICS region.
5. Add the data definition statement for the intrapartition data set to the CICS startup job stream. The DD name for the intrapartition data set is DFHINTRA, and the DSN operand must be the name of the VSAM entry-sequenced data set. For example, you could specify:

```
//DFHINTRA DD DSN=CICSTS41.CICS.applid.DFHINTRA,DISP={OLD|SHR}
```

If you are using XRF, specify **DISP=SHR** on the DD statement. A transient data intrapartition data set is a passively shared data set, owned by the active CICS region, but allocated to both active and alternate CICS regions. Although the alternate CICS region does not open this data set before takeover, it is allocated at job step initiation.

Example

```
//DEFDS JOB accounting info,name,MSGCLASS=A
//TDINTRA EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
        DEFINE CLUSTER -
            ( NAME(CICSTS41.CICS.applid.DFHINTRA) -
              RECORDSIZE(1529,1529) -
              RECORDS(100) -
              NONINDEXED -
              CONTROLINTERVALSIZE(1536) -
              VOL(volid) -
            DATA -
              ( NAME(CICSTS41.CICS.applid.DATA.DFHINTRA))
/*
//
```

Figure 3. Sample job to define a transient data intrapartition data set

What happens if the intrapartition data set fails to open

The DFHINTRA data set is opened during CICS initialization, regardless of the presence or absence of any intrapartition queue definitions that might become active during GRPLIST installation.

If DFHINTRA fails to open during an initial or cold start of CICS, a message is issued informing you of this, and asks if you want to continue or if you want to cancel CICS.

If DFHINTRA opened successfully during a previous startup but fails to open during a subsequent warm or emergency restart, CICS is terminated.

If CICS initializes without a DFHINTRA data set, any attempts to install intrapartition data destinations for that run of CICS fails and appropriate error messages are issued.

Using multiple extents and multiple volumes

Instead of defining one data set, which might have to be much larger than your average needs to cater for exceptional cases, you can define multiple extents and multiple volumes.

The job control statements in Figure 3 on page 22 are for a single extent data set defined on a single volume. That data set must be big enough to hold all your data. For more information about defining these, see “Defining data sets with multiple extents and volumes” on page 6.

Size of the intrapartition data set

If all available control intervals are currently allocated to queues, further EXEC CICS WRITEQ TD requests receive a NOSPACE response until control intervals are released by READQ TD or DELETEQ TD requests.

The intrapartition data set should hold at least two control intervals. When a logically recoverable queue is read until a QZERO condition is returned, and the request is committed, CICS retains the last CI used by the queue (unless there is no further space between the end of the last record and the end of the CI). Retaining the final CI used by the queue, means that subsequent requests to write to the queue can be accommodated in the remaining space within the CI if they can fit there. This avoids the need to acquire a new CI whenever the first request is made to write to the queue following a QZERO condition, which benefits performance on subsequent write requests. However, the CI is left allocated to the queue, and so increases the usage of the data set, and the possibility of a NOSPACE condition being returned by CICS.

Intrapartition data set restrictions

A transient data intrapartition data set should be associated with one, and only one, CICS region.

(If you are running CICS with XRF, one region means a pair of active and alternate CICS regions.) The destination control table contains relative byte addresses (RBAs) of records written to an intrapartition data set, and care must be taken to preserve the RBAs during any VSAM export or import operation on the data set.

Data can be corrupted or lost if:

- You start CICS with the wrong intrapartition data set; that is, a data set that contains data from another CICS region.
- You use VSAM export or import services to:
 - Increase the available space by compressing the data set, or
 - Increase the control interval size

Defining extrapartition data sets

You can define each extrapartition data set either for input only or for output only, but not for both.

About this task

You should define transient data extrapartition data sets used as queues for CICS messages with a record length of 132 bytes and a record format of V or VB. If you use the FREE=CLOSE parameter for an input extrapartition entry, be aware that it will only be usable once in the CICS session. Any attempt to read the queue after it has been CLOSED and OPENED again will result in the IOERR condition being raised.

Example

```
//LOGUSR DD DSN=CICSTS41.CICS.applid.LOGUSR,DISP=(NEW,KEEP),
//      DCB=(DSORG=PS,RECFM=V,BLKSIZE=136),
//      VOL=SER=valid,UNIT=3380,SPACE=(CYL,2)
//MSGUSR DD SYSOUT=A
```

Figure 4. Sample JCL to define transient data extrapartition data sets

The DFHCXRF data set

Besides any extrapartition data sets that you might define, there is a special extrapartition queue that CICS creates dynamically. This queue has the identifier CXRF and is created by CICS early in the initialization process.

The DD name for this extrapartition data set is DFHCXRF. You cannot define CXRF or DFHCXRF. If you code DFHCXRF as a DSCNAME, or code CXRF as a destination identifier, an error message is issued. If you create a definition for CXRF in the CSD, CICS does not install the definition. This is because the CICS entry is hardcoded and cannot be removed or replaced.

Although the CXRF data set has special significance in an alternate CICS region when you are operating CICS with XRF, it is also available in an active CICS region, and CICS regions running with XRF=NO.

If an attempt is made to write to a CICS-defined transient data queue before CICS is fully initialized, a message is written to CXRF.

If, on an initial or cold start, a request is received to write a record to a queue that has not yet been installed as part of GRPLIST, the record is written to CXRF.

If an attempt is made to write to an intrapartition queue after the warm keypoint has been taken, the record is written to CXRF.

Active CICS regions

CICS uses the CXRF queue during CICS initialization as some CICS components might need to write to transient data queues.

If the queues are not available at initialization time, the request to write to these queues is redirected to CXRF. Any requests from CICS components to write to transient data before the CXRF queue definition has been installed fails with a QIDERR condition.

Any requests to write to an intrapartition transient data queue after the warm keypoint has been taken during a normal shutdown are routed to CXRF.

If you want to take advantage of the special CXRF queue, you must include a DD statement for DFHCXRF. If you omit the DD statement, transient data write requests redirected to CXRF fail with a NOTOPEN condition.

Alternate CICS regions

DFHCXRF has special significance for alternate CICS regions, because transient data initialization is suspended while the alternate CICS region is in standby mode, and is not completed until a takeover occurs. If you want to capture messages written to transient data by the alternate CICS region before takeover, you must include a DD statement defining the DFHCXRF data set. These messages include information about terminals that have been installed and logged on to the active CICS region. The alternate CICS region takes this information from the message data set and stores it in the CICS-generated extrapartition transient data queue, CXRF.

DFHCXRF DD statements

You can define the DFHCXRF data set to MVS in the same way as other transient data extrapartition data sets, either to a SYSOUT data set or a sequential data set on disk (or tape). For example, you could use either of the DD statements shown in the following example in a startup job stream for an alternate CICS region:

```
//DFHCXRF DD SYSOUT=*  
  
or  
//DFHCXRF DD DSN=CICSTS41.CICS.applid.DFHCXRF,DISP=(NEW,KEEP),  
//          DCB=(DSORG=PS,RECFM=V,BLKSIZE=136),  
//          VOL=SER=volid,UNIT=3380,SPACE=(TRK,5)
```

Figure 5. Sample DD statements for DFHCXRF

Before takeover occurs, the alternate CICS region assumes that the transient data queues are defined as indirect, and pointing to CXRF. CXRF is associated with the data set that has the DD name DFHCXRF.

XRF considerations

Except for DFHCXRF, an alternate CICS region does not open any extrapartition data sets before takeover.

Normally, when data sets are defined for output, you should have separate data sets for the active and alternate CICS regions; that is, they are unique data sets in CICS terms.

If you have separate data sets, you can code the following DISP operands in the DD statements that define the data sets:

- DISP=SHR

- DISP=NEW
- DISP=OLD
- DISP=MOD

Whatever you code on the DISP parameter, be aware that data might be lost when the alternate CICS region takes over from the active CICS region, because takeover involves abending or canceling the active CICS region.

If you do not have separate data sets, you should code DISP=SHR. Anything else implies exclusive use of the data set, and for this reason you could not start an alternate CICS region (in the same MVS image as the active CICS region) until the active CICS region terminates.

Data written by the active CICS region is lost when the alternate CICS region takes over and opens the data set.

Chapter 4. Setting up CICS log streams

This chapter describes how to define and create CICS log streams that exploit the MVS system logger to record journaling and logging information.

About this task

CICS log manager supports:

- The CICS system log, which is also used for dynamic transaction backout.
- User journals, forward recovery logs, and autojournals. These are general logs.

This chapter covers the following topics:

- “Defining CICS system logs”
- “Defining CICS general logs” on page 29
- “Naming journals” on page 31
- “Defining JOURNALMODELS” on page 33
- “Mapping log streams” on page 34
- “Using the Journal utility program, DFHJUP” on page 38

Defining CICS system logs

Each CICS region requires its own system log. The system log is implemented as two MVS system logger log streams - a primary and a secondary - but, together, they form a single logical log stream.

About this task

The system log is used for recovery purposes - for example, during dynamic transaction backout, or during emergency restart, and is not meant to be used for any other purpose.

CICS connects to its system log automatically during initialization (unless you specify a journal model definition that defines the system log as type DUMMY).

You must define a system log if you want to preserve data integrity in the event of unit of work failures and CICS failures. CICS needs a system log in order to perform:

- The backout of recoverable resources changed by failed units of work.
- Cold starts when CICS needs to recover conversation state data with remote partners.
- Warm restarts, when CICS needs to restore the region to its pre-shutdown state.
- Emergency restarts, when CICS needs to restore the region to its pre-shutdown state as well as recovering transactions to perform the backout of recoverable resources changed by units of work that were in-flight at the time of shutdown.

For information on how to define CICS system log streams, see Coupling facility log streams, coupling facility log streams and DASD-only log streams in the *CICS Transaction Server for z/OS Installation Guide*.

Planning your CICS system log streams

A CICS system log (which comprises two physical log streams) is unique to the region and must not be used by any other CICS region. The default log stream names *region_userid.applid.DFHLOG* and *region_userid.applid.DFHSHUNT* are designed to ensure unique names.

Using JOURNALMODELS to define the system log

You might want to use journal models for system logs if the CICS region userid changes between runs (for example, where a test CICS region is shared between application developers).

It would be wasteful to create log streams with a different high level qualifier for each user. Using the same system log stream regardless of the which programmer starts up the CICS region keeps the number of log streams to a minimum. The following example uses a specific JOURNALNAME, with symbols in the STREAMNAME, making this an explicit model for the primary log stream.

```
DEFINE GROUP(TEST) DESC('System logs for test CICS regions')
      JOURNALMODEL(DFHLOG) JOURNALNAME(DFHLOG) TYPE(MVS)
      STREAMNAME(TESTCICS.&APPLID..&JNAME.)
```

If you define JOURNALMODEL resource definitions to define log stream names for DFHLOG and DFHSHUNT, ensure that the resulting log stream names are unique. If you have some CICS regions that use the same applid, you must use some other qualifier in the log stream name to ensure uniqueness.

If you use JOURNALMODEL resource definitions for the system log, these resource definitions must be defined and added to the appropriate group list (using the CSD utility program, DFHCSDUP) before INITIAL-starting CICS.

System logs cannot be TYPE(SMF).

DFHLOG can be TYPE(DUMMY), but you can use this only if you always INITIAL start your CICS regions and there are no recoverable resources requiring transaction backout. CICS cannot perform a cold start, or warm or emergency restart if TYPE(DUMMY) is specified on the JOURNALMODEL definition.

If you do not want to use a system log, perhaps in a test or development region, define a JOURNALMODEL for DFHLOG with type DUMMY, as shown in the following example:

```
DEFINE JOURNALMODEL(DFHLOG) GROUP(CICSLOGS)
      JOURNALNAME(DFHLOG)
      TYPE(DUMMY)
```

To start a CICS region without a system log, you must ensure that a JOURNALMODEL definition, such as the one shown above, is included in the start-up group list. Use the DFHCSDUP batch utility program to define the required JOURNALMODEL and to add the group to the group list.

DFHSHUNT can be TYPE(DUMMY). This is not recommended, however, because it impairs the ability of CICS to manage the system log.

Effects of the AKPFREQ parameter

Review the activity keypoint frequency (AKPFREQ) defined for each CICS region. The larger the value, the more coupling facility space you need for the system logs, but you should not set AKPFREQ too low so that transactions last longer than an activity keypoint interval.

Defining CICS general logs

Journals on general log streams comprise user journals, forward recovery logs, and autojournals.

About this task

Planning log streams for use by your user journals and autojournals

General logs are identified as such by their MVS log stream names, which are differentiated from system log streams in that their names do not end with DFHLOG or DFHSHUNT.

Using JOURNALMODELS to define general logs

If you are running multiple cloned copies of your application-owning regions (AORs), it is probable that the logged data is common and that you would want to merge the data from all of the AORs to the same log stream.

The following JOURNALMODEL resource definition maps CICS journals of the same journal identifier to a shared log stream:

```
DEFINE GROUP(MERGE) DESC('Merge journals across cloned CICS regions')
    JOURNALMODEL(JRNLS) JOURNALNAME(DFHJ*) TYPE(MVS)
    STREAMNAME(&USERID..SHARED.&JNAME.)
```

In this example, the literal SHARED is used in place of the default CICS applid, which would require a unique log stream for each region.

You might want to use JOURNALMODELS to map journals to log streams if the CICS region userid changes between runs. This could be the case, for example, where CICS test regions are shared between groups of developers. It would be wasteful to create log streams with a different high level qualifier for each user and you might prefer to use the same log streams regardless of which developer starts up a CICS region. For example, the following generic JOURNALMODEL definition maps all journals not defined by more explicit definitions to the same log stream

```
DEFINE GROUP (TEST) DESC('Journals for test CICS regions')
    JOURNALMODEL(JRNLS) JOURNALNAME(*) TYPE(MVS)
    STREAMNAME(TESTCICS.&APPLID..&JNAME.)
```

You might want to merge data written by CICS regions using different journal names to a single log stream.

```
DEFINE GROUP (TEST) DESC('Merging journals 10 to 19')
    JOURNALMODEL(J10T019) JOURNALNAME(DFHJ1*) TYPE(MVS)
    STREAMNAME(&USERID..MERGED.JNLS)
DEFINE GROUP (TEST) DESC('Merging journalnames JNLxxxxx')
    JOURNALMODEL(JNLXXXXX) JOURNALNAME(JNL*) TYPE(MVS)
    STREAMNAME(&USERID..MERGED.JNLS)
```

The last qualifier of the stream name is used as the CICS resource name for dispatcher waits. Therefore, if it is self-explanatory, it can be helpful when interpreting monitoring information and CICS trace entries.

Planning log streams for use by your forward recovery logs

CICS performs the logging for RLS and non-RLS data sets.

You can share a forward recovery log stream between multiple data sets - you do not need to define a log stream for each forward-recoverable data set. Your

decision is a balance of transaction performance, rapid recovery, and the work involved in managing a large number of log streams.

For a data set open in RLS mode, the MVS logger merges all the forward recovery log records from the various CICS systems on to the shared forward recovery log.

Some points to consider are:

- All data sets used by one transaction should use the same log stream (to reduce the number of log streams written to at syncpoint).
- A good starting point is to use the same forward recovery log ID that you use in an earlier CICS release.
- Share a forward recovery log stream between data sets that:
 - Have similar security requirements
 - Have similar backup frequency
 - Are likely to need restoring in their entirety at the same time
- Log stream names should relate to the data sets. For example, PAYROLL.data_sets could be mapped to a forward recovery log named PAYROLL.FWDRECOV.PAYLOG.
- The last qualifier of the stream name is used as the CICS resource name for dispatcher waits. Therefore, if it is self-explanatory, it can be helpful when interpreting monitoring information and CICS trace entries.
- Don't mix high update frequency data sets with low update frequency data sets, because this causes a disproportionate amount of unwanted log data to be read during recovery of low frequency data sets.
- Don't put all high update frequency data sets on a single log stream because you could exceed the throughput capacity of the stream.
- If you define too many data sets to a single log stream, you could experience frequent structure-full events when the log stream can't keep up with data flow.
- Redundant data should be deleted from log streams periodically. On OS/390[®] Release 2 or earlier, this is a user responsibility, and must be done before the system logger inventory entry exceeds the limit of 168 data sets per log stream. On OS/390 Release 3 or later, you can specify that redundant data is to be deleted automatically from general log streams, by defining them with AUTODELETE(YES) RETPD(dddd). For information about the AUTODELETE and RETPD MVS parameters, see the *CICS Transaction Server for z/OS Installation Guide*.

Typically, for a forward recovery log, deletion of old data is related to the data backup frequency. For example, you might keep the 4 most recent generations of backup, and when you delete a redundant backup generation you should also delete the corresponding redundant forward recovery log records. These are the records older than the redundant backup because they are no longer needed for forward recovery.

For information about IBM CICS VSAM Recovery, see the *CICS VSAM Recovery User's Guide*.

Planning log streams for use by your log of logs (DFHLGLOG)

The log of logs is written by CICS to provide information to forward recovery programs such as CICS VSAM Recovery (CICSVR).

The log of logs is a form of user journal containing copies of the tie-up records written to forward recovery logs. Thus it provides a summary of which recoverable VSAM data sets CICS has used, when they were used, and to which log stream the forward recovery log records were written.

If you have a forward recovery product that can utilize the log of logs, you should ensure that all CICS regions sharing the recoverable data sets write to the same log of logs log stream.

```
DEFINE GROUP(JRNL) DESC('Merge log of logs')
    JOURNALMODEL(DFHLGLOG) JOURNALNAME(DFHLGLOG) TYPE(MVS)
    STREAMNAME(&USERID..CICSVR.DFHLGLOG)
```

Note: Note that this definition is supplied in group DFHLGMOD in DFHLIST.

If you don't have a forward recovery product that can utilize the log of logs you can use a dummy log stream:

```
DEFINE GROUP(JRNL) DESC('Dummy log of logs')
    JOURNALMODEL(DFHLGLOG) JOURNALNAME(DFHLGLOG) TYPE(DUMMY)
```

Do not share the log of logs between test and production CICS regions, because it could be misused to compromise the contents of production data sets during a restore.

Naming journals

The journals have the following naming conventions.

About this task

System logs

DFHLOG and DFHSHUNT are the journal names for the CICS system log.

CICS automatically creates journal table entries for DFHLOG and DFHSHUNT during initialization as shown in Table 1.

Table 1. Journal name entry for the CICS primary system log

| Journal table entry - CICS system log | Created during system initialization |
|---------------------------------------|---|
| Name: DFHLOG | Always DFHLOG for the primary log |
| Status: Enabled | Set when journal entry created |
| Type: MVS | The default, but it can be defined as DUMMY on JOURNALMODEL definition (DUMMY = no output) |
| LSN: log_stream_name | By default, log_stream_name resolves to ®userid.&applid..DFHLOG, but this can be user-defined on a JOURNALMODEL definition |

Forward recovery logs

For non-RLS data sets that have not specified their recovery attributes in the VSAM catalog, forward recovery log names are of the form DFHJ nm where nm is a number in the range 1–99.

You define the name of the forward recovery log in the forward recovery log id (FWDRECOVLOG) in the FILE resource definition.

User applications can use a forward recovery log through a user journal name that maps on to the same log stream name. In this case, the user records are merged on to the forward recovery log. See Table 2 on page 32 for an example of this.

Table 2. Example of journal name entry for non-RLS mode forward recovery log

| Journal table entry - forward recovery log | Entry created during file-open processing |
|--|---|
| Name: DFHJ01 | Name derived from FWDRECOVLOG identifier. For example, FWDRECOVLOG(01) = DFHJ01, thus FWDRECOVLOG(nn) = DFHJnn |
| Status: Enabled | Set when journal entry created |
| Type: MVS | The default, but it can be defined as DUMMY on JOURNALMODEL definition (DUMMY = no output) |
| LSN: log_stream_name | By default, log_stream_name resolves to ®userid.&applid..DFHJ01, but this can be user-defined on a JOURNALMODEL definition |

Note: There is no journal table entry for the forward recovery log of an RLS file. The recovery attributes and LSN are obtained directly from the VSAM catalog, and the LSN is referenced directly by CICS file control. Therefore there is no need for indirect mapping through a journal name.

You can also choose to specify the recovery attributes and LSN for a non-RLS file in the VSAM catalog.

User journals

CICS user journals are identified by their journal names (or number, in the case of DFHJnn names), which map on to MVS log streams.

You name your user journals using any 1-8 characters that conform to the rules of a data set qualifier name. Apart from the user journal names that begin with the letters DFHJ, followed by two numeric characters, you should avoid using names that begin with DFH. User journal names of the form DFHJnn are supported for compatibility with earlier CICS releases.

Although 8-character journal names offer considerable flexibility compared with the DFHJnn name format of previous releases, you are recommended not to create large numbers of journals (for example, by using the terminal name or userid as part of a program-generated name).

Journal name DFHLOG (on an EXEC CICS WRITE JOURNALNAME command) indicates that you want to write to the CICS system log.

When used in FILE and PROFILE resource definitions, the journal numbers 1 through 99 map on to the user journal names DFHJ01–99. You can map these journal names to specific MVS log stream names by specifying JOURNALMODEL resource definitions, or allow them to default. If you do not specify matching JOURNALMODEL definitions, by default user journals are mapped to LSNs of the form *userid.applid.DFHJnn*.

Table 3 shows an example of a user journal name table entry.

Table 3. Example of a user journal name entry for output to MVS

| Journal table entry - user journals | Entry created on first reference |
|-------------------------------------|---|
| Name: JRNL001 | Name derived from API WRITE JOURNALNAME command |

Table 3. Example of a user journal name entry for output to MVS (continued)

| Journal table entry - user journals | Entry created on first reference |
|-------------------------------------|---|
| Status: Enabled | Set when journal entry created |
| Type: MVS | This journal is defined for MVS output by a JOURNALMODEL that references the JRNL001 name |
| LSN: log_stream_name | By default, log_stream_name resolves to ®userid.&applid.JRNL001, but this can be user-defined on a JOURNALMODEL definition |

Installing system log and journal names

The journal control table of earlier releases is obsolete, and is replaced by a journal names table that CICS creates dynamically.

The CICS log manager needs the name of the log stream that corresponds to a CICS system log or general log, and the type - whether it is MVS, SMF, or a dummy. Except for VSAM forward recovery log stream names taken directly from the ICF catalog, CICS maintains this information in the journal names table, together with the corresponding log stream token that is returned by the MVS system logger when CICS successfully connects to the log stream.

Defining JOURNALMODELS

CICS uses JOURNALMODEL definitions to resolve log stream names at the following times:

About this task

System log

During initialization, on an initial start only.

On a cold, warm or emergency restart, CICS retrieves the log stream name from the CICS global catalog.

General logs (excluding log streams defined in the)

When a journal name is first referenced after the start of CICS, or when it is first referenced again after its log stream has been disconnected. Log stream disconnection, requiring further reference to a matching JOURNALMODEL resource definition, occurs as follows:

User journals

As soon as you issue a DISCARD JOURNALNAME command.

Any further references to the discarded journal name means that CICS must again resolve the log stream name by looking for a matching JOURNALMODEL resource definition. You can change the log stream name for a user journal by installing a modified JOURNALMODEL definition.

Auto journals for files

All files that are using the log stream for autojournaling are closed.

Forward recovery logs (excluding those defined in the)

All files that are using the log stream for forward recovery logging are closed.

A JOURNALMODEL definition generally specifies a generic journal name, thereby mapping to the same MVS log stream any journal names that match on the generic name. JOURNALMODEL definitions can also be specific models and, using JOURNALMODEL definitions, you can map many journals or forward recovery logs to the same MVS log stream, or assign them to SMF (see Figure 6).

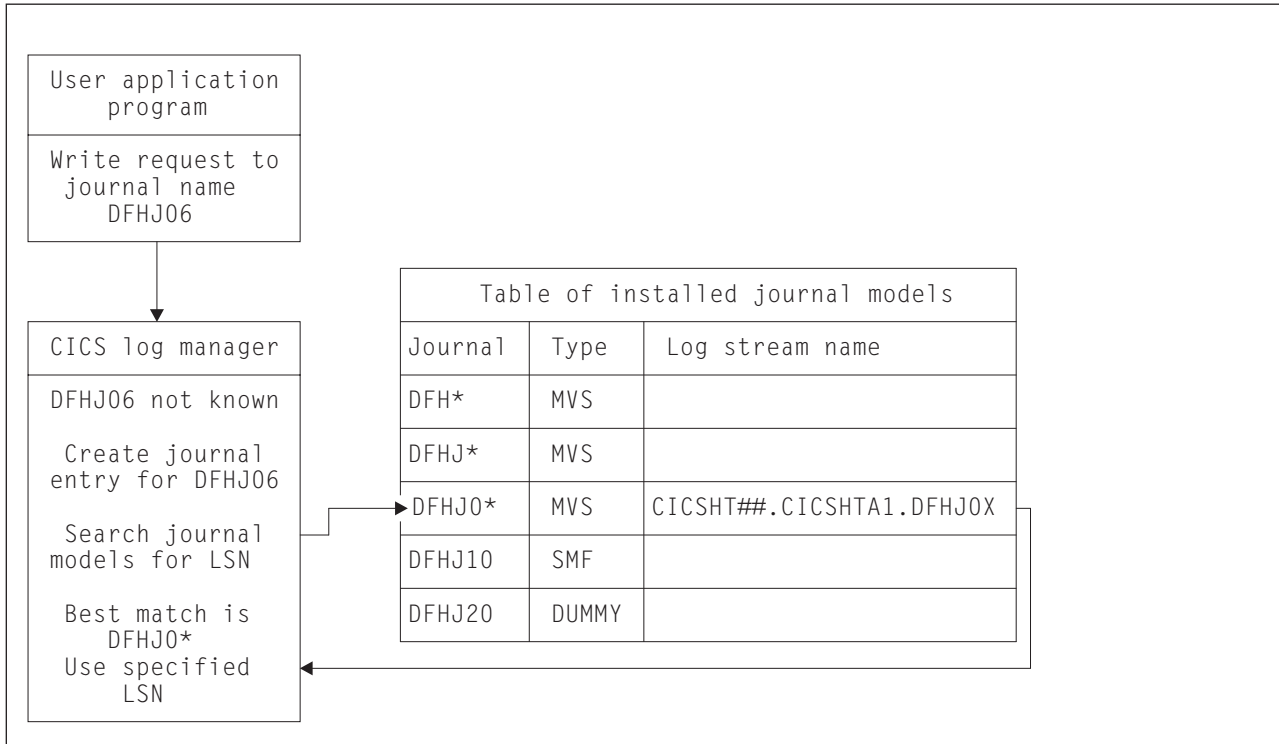


Figure 6. Looking for a journal model that matches a journal name. CICS searches the Table of installed journal models to find the log stream name that corresponds to the journal name, using a “best-match” algorithm.

Mapping log streams

Except for VSAM RLS forward recovery log stream names, which are obtained directly from the VSAM catalog, CICS maps the system log name or journal name to a corresponding log stream name.

About this task

CICS does this either by using a user-defined JOURNALMODEL resource definition (if one exists), or by using default names created by resolving symbolic names.

Mapping of the system log stream

On a cold start, or warm or emergency restart, CICS retrieves the system log stream name from the CICS global catalog.

CICS uses the default log stream names unless you provide a JOURNALMODEL resource definition for the system log.

If there are JOURNALMODEL definitions for your system logs (CICS finds JOURNALMODEL definitions with JOURNALNAME(DFHLOG) and

JOURNALNAME(DFHSHUNT)), it attempts to connect to the system log streams named in these definitions. System log stream names must be unique to the CICS region.

If you define JOURNALMODEL resource definitions for your system logs, ensure that:

- The log streams named in the definition are defined to the MVS system logger, or
- Suitable model log streams are defined so that they can be created dynamically.

If there are no suitable JOURNALMODEL definitions, CICS attempts to connect to system log streams, using the following default names:

- userid.applid.DFHLOG
- userid.applid.DFHSHUNT

where 'userid' is the RACF[®] userid under which the CICS address space is running, and 'applid' is the region's VTAM APPL name. The CSD group DFHLGMOD holds the CICS-supplied JOURNALMODEL definitions for the default DFHLOG and DFHSHUNT log streams.

Before you try to use these default log stream names, ensure that

- The default log streams are defined explicitly to the MVS system logger, or
- Suitable model log streams are defined so that they can be created dynamically.

If these log streams are not available (perhaps they have not been defined to MVS) or the definitions are not found (perhaps they have not been installed), CICS attempts to create system log streams using a model log stream named *&sysname.LSN_last_qualifier.MODEL*, where:

- *&sysname* is the MVS symbol that resolves to the system name of the MVS image
- *LSN_last_qualifier* is the last qualifier of the log stream name as specified on the JOURNALMODEL resource definition.

If you do not provide a JOURNALMODEL resource definition for DFHLOG and DFHSHUNT, or if you use the CICS definitions supplied in group DFHLGMOD, the model names default to *&sysname.DFHLOG.MODEL* and *&sysname.DFHSHUNT.MODEL*. Once these log streams have been created, CICS then connects to them.

Figure 7 on page 36 shows a graphical representation of the system log mapping process during an INITIAL start.

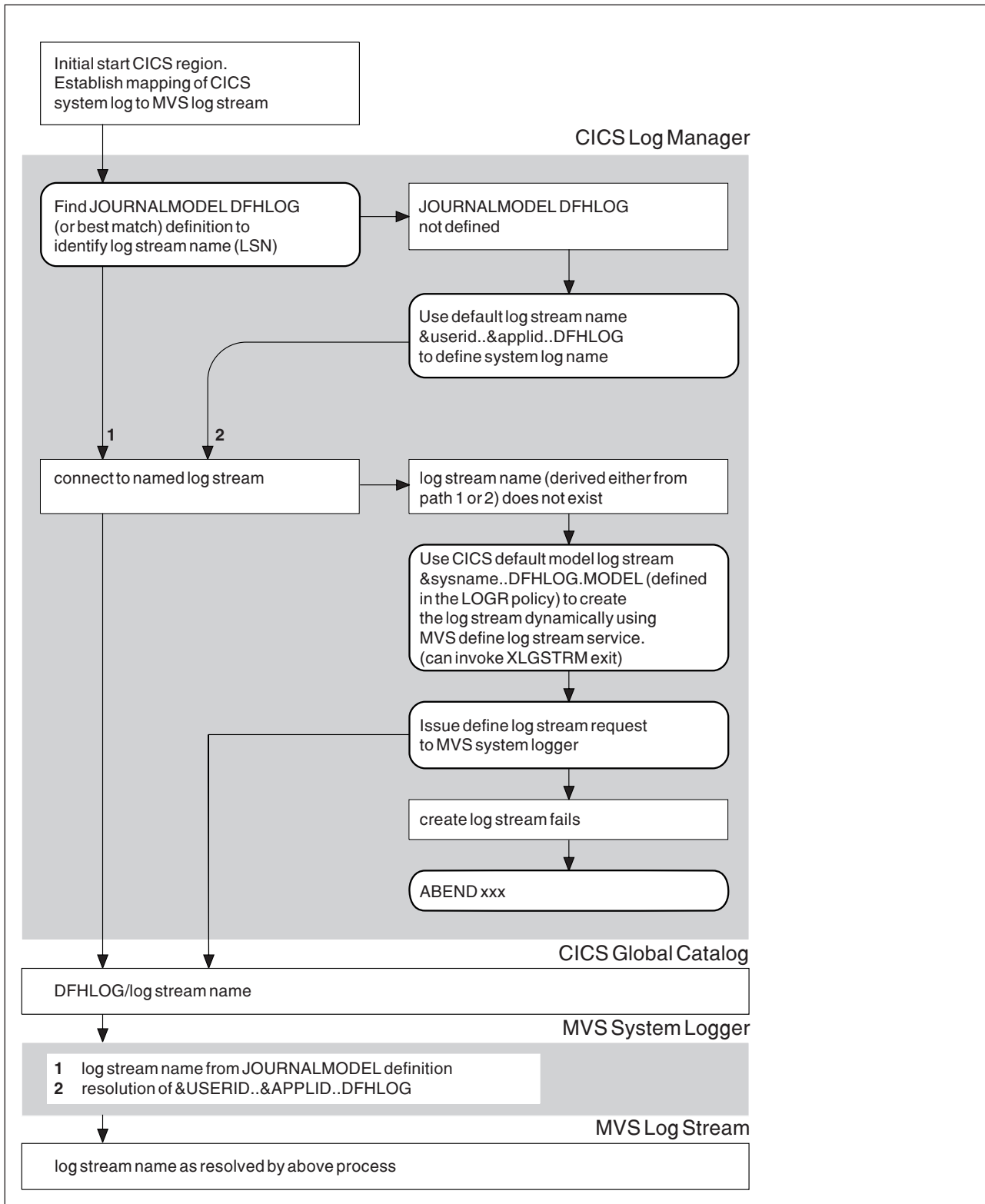


Figure 7. How CICS maps the system log (DFHLOG) to a log stream name (LSN) during an INITIAL start. CICS uses the same process for the secondary system log, DFHSHUNT.

Mapping of general log streams

CICS uses the default log stream names unless you provide a JOURNALMODEL resource definition for the journal or log.

If there is a JOURNALMODEL definition for the log, CICS attempts to connect to the log stream named in the definition.

If you define JOURNALMODEL resource definitions for your system logs, ensure that:

- The log streams named in the definition are defined to the MVS system logger, or
- Suitable model log streams are defined so that they can be created dynamically.

If CICS cannot connect to the log stream named in the JOURNALMODEL definition, it attempts to connect to a log stream, using the default name:
`userid.applid.journalname`

Before you try to use this default log stream name, ensure that

- The default log stream is defined explicitly to the MVS system logger, or
- A suitable model log stream is defined so that it can be created dynamically.

If the log stream is not available (perhaps it has not been defined to MVS) or the definition is not found (perhaps it has not been installed), CICS attempts to create a log stream using the default name:

`LSN_QUALIFIER1.LSN_QUALIFIER2.MODEL`

where the qualifier fields are based on the JOURNALMODEL definition streamname attribute, as follows:

- If the log stream being created has a qualified name consisting of only two names (*qualifier1.qualifier2*) or has an unqualified name, CICS constructs the model name as *qualifier1.MODEL* or *name.MODEL*.
- If the log stream being created has a qualified name consisting of 3 or more names (*qualifier1.qualifier2....qualifier_n*), CICS constructs the model name as *qualifier1.qualifier2.MODEL*.

Once the log stream has been created, CICS connects to it.

Figure 8 on page 38 shows a graphical representation of the mapping process for general logs.

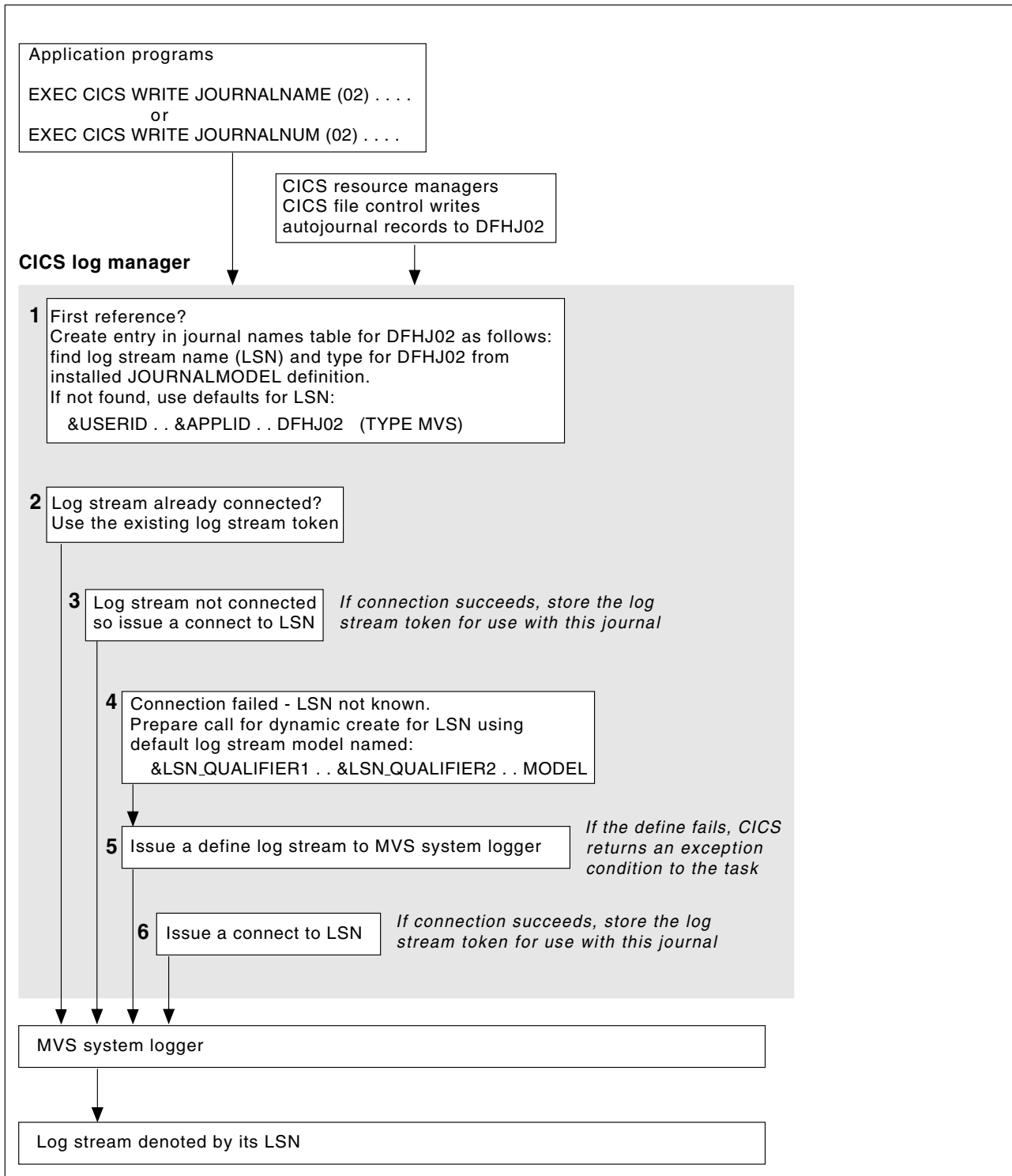


Figure 8. How a CICS journal is mapped to its log stream name (LSN). The name is DFHJ02, used here for user journaling and file control autojournaling.

Using the Journal utility program, DFHJUP

CICS provides a journal utility program, DFHJUP.

About this task

You can use the DFHJUP utility program which uses the SUBSYS=(LOGR... facility to select, print, or copy data held on MVS system logger log streams. Alternatively, you can use your own utility to use the SUBSYS=(LOGR... facility.

For information about running DFHJUP, and the SUBSYS=(LOGR.. , facility, see the *CICS Operations and Utilities Guide*.

Chapter 5. Setting up the CICS system definition data set

This section describes how to define and initialize the system definition data set (CSD) that CICS needs to store definitions of the resources that it uses. It also describes using the CEDA transaction, particularly when a CSD is being shared by more than one CICS region.

About this task

You might have used the CEDA transaction already, when running the interactive installation verification procedures (IVPs) after installing CICS. If you ran any of the IVPs (for example, the jobs called DFHIVPBT or DFHIVPOL), you also used a CSD. Note that the CSD created by the IVPs is limited in size, and initialized with the CICS-supplied resource definitions only.

A CSD is required for most resource definitions. If you are creating a CSD for the first time, go through the steps listed in “Planning your CSD configuration.” The remainder of this section describes these steps in more detail.

If you are already using a CSD with a previous release of CICS, upgrade your CSD to include CICS resource definitions new in CICS Transaction Server for z/OS, Version 4 Release 1. For information about upgrading your CSD, see the *CICS Resource Definition Guide*.

You can run the DFHCSDUP offline utility as a batch job to read from and write to the CSD. You should give UPDATE access to the CSD to **only** those users who are permitted to use the DFHCSDUP utility.

Planning your CSD configuration

Before you can create a CSD, you must plan your configuration.

Procedure

1. Decide how much disk space you require.
2. Decide whether you want to use the CSD in RLS or non-RLS mode. Having the CSD open in RLS mode allows more than one CICS region to update the CSD concurrently. However, if your CSD is defined as a recoverable data set, and you want update it using the batch utility, DFHCSDUP, you must quiesce the CSD in the CICS regions before running DFHCSDUP.

If you decide to use RLS for the CSD, specify CSDRLS=YES as a system initialization parameter. See “VSAM record-level sharing (RLS)” on page 96.

3. Decide what backup and recovery procedures you require for your CSD. The CSD can use backup-while-open (BWO), which means that DFSMS components can back up the CSD while the data set is open for update. To use BWO, ensure that DFSMSHsm and DFSMSdss components of DFSMS 1.2 or later are available. The CSD must have an ICF catalog entry and be defined in SMS-managed storage.
 - Define the recovery options for a CSD accessed in RLS mode:
 - Specify BWO(TYPECICS) in the ICF catalog to make the data set eligible for backup-while open.

- Make the CSD a recoverable data set by specifying the appropriate LOG parameter in the ICF catalog.
- Define the recovery options for a CSD accessed in non-RLS mode:
 - Make the data set eligible for backup-while-open by either specifying BWO(TYPECICS) in the catalog or setting the **CSDBKUP** system initialization parameter to DYNAMIC.
 - Make the data set recoverable by setting the appropriate LOG parameter in the ICF catalog entry or specifying the appropriate option on the **CSDRECOV** system initialization parameter.

By default, the recovery options in the catalog override the attributes in the FILE resource. If no recovery options are set in the catalog, CICS uses the attribute values of the FILE resource. You can set the **NONRLSRECOV** system initialization parameter to FILEDEF, if you want CICS to always use the recovery options on the FILE resource instead of the catalog.

4. Define and initialize the CSD.
5. Decide what CICS file processing attributes you want for your CSD. Although the CSD is a CICS file-control-managed data set, you define file control resource definitions for the CSD by specifying CSDxxxx system initialization parameters (see “Defining CSD attributes” on page 46).
6. Decide if you want to use command logs for RDO. See “Logging RDO commands” on page 59 for details of the CADL, CAIL, CRDI, CSDL, CSFL, CSKL, CSPL, and CSRL destinations that CICS uses for RDO command logs.
7. Make the CSD available to CICS, either by using dynamic allocation or by including the necessary DD statement in the CICS startup job stream. For dynamic allocation of the CSD, you name the fully qualified data set name, and the disposition, on the **CSDDSN** and the **CSDDISP** system initialization parameters. When you have started CICS, test the RDO transactions CEDA, CEDB, and CEDC. For information about these transactions, see the *CICS Supplied Transactions*.

Calculating CSD disk space

Before you can create the CSD, you must calculate the amount of space you need in your CSD for definition records.

Procedure

1. In your calculation, allow for approximately 1800 CICS-supplied resource definitions of various types, which are loaded into the CSD when you initialize the CSD with the utility program, DFHCSDUP. You need to consider:
 - a. Each resource definition (for example each program, transaction and terminal) needs one record. The sizes of these definition records are:

| Resource | Definition record size (maximum) |
|-------------|----------------------------------|
| ATOMSERVICE | 720 bytes |
| BUNDLE | 698 bytes |
| CONNECTION | 260 bytes |
| CORBASERVER | 1375 bytes |
| DB2CONN | 308 bytes |
| DB2ENTRY | 236 bytes |
| DB2TRAN | 198 bytes |
| DJAR | 445 bytes |
| DOCTEMPLATE | 567 bytes |
| ENQMODEL | 447 bytes |

| Resource | Definition record size (maximum) |
|--------------|----------------------------------|
| FILE | 369 bytes |
| IPCONN | 468 bytes |
| JOURNALMODEL | 222 bytes |
| JVMSEVER | 208 bytes |
| LIBRARY | 925 bytes |
| LSRPOOL | 425 bytes |
| MAPSET | 190 bytes |
| MQCONN | 240 bytes |
| PARTITIONSET | 190 bytes |
| PARTNER | 408 bytes |
| PIPELINE | 959 bytes |
| PROCESSTYPE | 206 bytes |
| PROFILE | 231 bytes |
| PROGRAM | 499 bytes |
| REQUESTMODEL | 1211 bytes |
| SESSION | 296 bytes |
| TCPIPSERVICE | 571 bytes |
| TDQUEUE | 331 bytes |
| TERMINAL | 327 bytes |
| TRANCLASS | 192 bytes |
| TRANSACTION | 545 bytes |
| TSMODEL | 308 bytes |
| TYPETERM | 402 bytes |
| URIMAP | 1443 bytes |
| WEBSERVICE | 708 bytes |

- b. Each group requires two 122-byte records
 - c. Each group list requires two 122-byte records
 - d. Each group name within a list requires one 68-byte record
2. Add a suitable contingency (approximately 25%) to the size that you have calculated so far.

What to do next

Use your final figure when you define the VSAM cluster for the CSD. (See the sample job in “Initializing the CSD.”)

Initializing the CSD

The INITIALIZE command initializes your CSD with definitions of the CICS-supplied resources. After initialization, you can migrate resource definitions from your CICS control tables and begin defining your resources interactively with CEDA. You use INITIALIZE only once in the lifetime of the CSD.

Before you begin

Before you can use the CSD, you must define it as a VSAM KSDS data set, and initialize it using the DFHCSDUP utility program.

About this task

Use the sample job to define and initialize the CSD.

Procedure

1. Code the KEYS parameter as shown in the sample job. The key length is 22 bytes.
2. Calculate the CSD disk space that is required. The average record size is 200 bytes for a CSD that contains only the CICS-supplied resource definitions (generated by the INITIALIZE and UPGRADE commands). If you create a larger proportion of terminal resource definition entries than are defined in the initial CSD, the average record size is higher because of the larger size of the terminal-type entries. The TERMINAL and TYPETERM definition record sizes are listed under “Calculating CSD disk space” on page 42. The maximum record size is 2000, as shown in the sample job.
3. Code the SHAREOPTIONS parameter as shown in the sample job.
4. Optional: You can specify the recovery attributes for the CSD in the ICF catalog instead of using the CSD system initialization parameters. If you decide to use the CSD in RLS mode, you must define the recovery attributes in the ICF catalog. You specify the recovery attributes as:
 - LOG(NONE) (Nonrecoverable data set)
 - LOG(UNDO) (For backout only)
 - LOG(ALL) (For both backout and forward recovery)If you specify LOG(ALL), you must also specify LOGSTREAMID to define the 26-character name of the MVS log stream to be used as the forward recovery log. If you specify recovery attributes in the ICF catalog, and also want to use BWO, specify LOG(ALL) and BWO(TYPECICS).
5. You must specify the DDNAME for the CSD as DFHCSD.

Example

```
//DEFINIT JOB accounting information
//DEFUCSD EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//AMSDUMP DD SYSOUT=A
//SYSIN DD *
  DEFINE CLUSTER -
    (NAME(CICSTS41.CICS.applid.DFHUCSD) -
     VOLUMES(vol1id) -
     KEYS(22 0) -
     INDEXED -
     RECORDS(n1 n2) -
     RECORDSIZE(200 2000) -
     FREESPACE(10 10) -
     SHAREOPTIONS(2) -
     LOG(ALL) -
     LOGSTREAMID(CICSTS41.CICS.CSD.FWDRECOV) -
     BWO(NO)
    DATA
      (NAME(CICSTS41.CICS.applid.DFHUCSD.DATA) -
       CONTROLINTERVALSIZE(8192))
    INDEX
      (NAME(CICSTS41.CICS.applid.DFHUCSD.INDEX))
  /*
//INIT EXEC PGM=DFHUCSDUP,REGION=300K
//STEPLIB DD DSN=CICSTS41.CICS.SDFHLOAD,DISP=SHR
//DFHUCSD DD DSN=CICSTS41.CICS.applid.DFHUCSD,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//SYSIN DD *
  INITIALIZE
  LIST ALL OBJECTS
/*
//
```

Figure 9. Sample job to define and initialize the CSD

What to do next

The command **LIST ALL OBJECTS** lists the CICS-supplied resources that are now in the CSD.

Creating a larger CSD

To avoid the CSD filling while CICS is running, ensure that you define the data set with primary and secondary space parameters, and that there is sufficient DASD space available for secondary extents.

If your CSD fills up while you are running a CEDDA transaction (or the offline utility), define a larger data set and use an AMS command, such as REPRO, to recover the contents of the CSD. If your CSD was dynamically allocated, you can close it, delete it, and redefine it as a larger data set. If your CSD was not dynamically allocated, you must shut down CICS to create a larger data set.

For a description of the commands that you can use for copying files, see the *MVS/ESA Integrated Catalog Administration: Access Method Services Reference*.

Defining CSD attributes

File processing attributes for the CSD are defined in a number of system initialization parameters.

About this task

Define suitable definitions for the following system initialization parameters:

Procedure

1. Define the type of access that is allowed using the **CSDACC** parameter.
2. Define whether the CSD is eligible for BWO using the **CSDBKUP** parameter. This parameter is ignored if you specify **CSDRLS=YES**. CICS uses the BWO parameter in the ICF catalog instead. By default, CICS also uses the BWO parameter in the ICF catalog for non-RLS mode CSDs if the LOG parameter in the ICF catalog specifies either UNDO or ALL. You can set the **NONRLSRECOV** system initialization parameter to FILEDEF if you want CICS to always use the **CSDBKUP** parameter over the BWO attribute.
3. Define the number of buffers for CSD data using the **CSDBUFND** parameter. This parameter is ignored if you specify **CSDRLS=YES**.
4. Define the number of buffers for the CSD index using the **CSDBUFNI** parameter. This parameter is ignored if you specify **CSDRLS=YES**.
5. Define the disposition of the CSD data set using the **CSDDISP** parameter.
6. Define the JCL data set name (DSNAME) of the CSD using the **CSDDSN** parameter.
7. Define a forward recovery journal identifier using the **CSDFRLOG** parameter. This parameter is ignored if you specify **CSDRLS=YES**, or if the recovery attributes are defined in the ICF catalog on the **LOG** parameter, in which case LOGSTREAMID from the ICF catalog is used instead. You can set the **NONRLSRECOV** system initialization parameter to FILEDEF if you want CICS to always use the **CSDFRLOG** parameter over the LOGSTREAMID attribute.
8. Define the level of read integrity for a CSD accessed in RLS mode using the **CSDINTEG** parameter.
9. Define an identifier for automatic journaling using the **CSDJID** parameter.
10. Define the VSAM local shared resource pool using the **CSDLRNO** parameter. This value is ignored if you specify **CSDRLS=YES**.
11. Define whether or not the CSD is recoverable using the **CSDRECOV** parameter. This parameter is ignored if you specify **CSDRLS=YES** and CICS uses the **LOG** parameter from the ICF catalog instead. If **LOG** is “undefined”, any attempt to open the CSD in RLS mode fails.
If **CSDRLS=NO**, this parameter is used only if **LOG** in the ICF catalog is “undefined.” By default, if **LOG** in the ICF catalog specifies NONE, UNDO, or ALL, the **LOG** parameter overrides the **CSDRECOV** value. You can set the **NONRLSRECOV** system initialization parameter to FILEDEF if you want CICS to always use the **CSDRECOV** parameter over the LOG attribute.
12. Define whether the CSD is accessed in RLS or non-RLS mode using the **CSDRLS** parameter.
13. Define the number of strings for concurrent requests using the **CSDSTRNO** parameter. This value is ignored and a value of 1024 is assumed if you specify **CSDRLS=YES**.

What to do next

These parameters are described in greater detail in Chapter 16, “Specifying CICS system initialization parameters,” on page 133.

Sharing the CSD in non-RLS mode

You can implement the sharing of a CSD that is accessed in a non-RLS mode using LSR or NSR. Sharing the CSD by several CICS regions enables those regions to use the same definitions, and means there is no need for duplicate data sets. This is particularly important in a parallel sysplex environment where a CICSplex might have a number of cloned regions, in which case it is essential that they use the same CSD.

About this task

You can optimize the sharing of a CSD as follows:

Procedure

- Enable more than one user to access the CSD at the same time in a CICS region. If you have specified read and write access for the CSD, all the CEDAs in a CICS region can perform read and write functions. CICS file control manages concurrent access for multiple users within a region, using the attributes specified in the **CSDACC** system initialization parameter.
- Enable several users in different CICS regions to access the CSD at the same time. See “Sharing user access from several CICS regions” for details.
- Enable multiple users to access the CSD in a CICS region.
- Share a CSD by CICS regions in a single MVS image.
- Share a CSD in a multi-MVS environment.
- Enable multiple users access one CSD across CICS or batch regions.
- Share the CSD between different releases of CICS.

Shared user access from the same CICS region

Several users in a CICS region can access the CSD at the same time.

If you have specified read/write access for the CSD, all the CEDAs in a CICS region can perform read and write functions. CICS file control manages concurrent access for multiple users within a region, using the attributes specified in the **CSDACC** system initialization parameter.

For more information, see “Multiple users of the CSD within a CICS region (non-RLS)” on page 49.

Sharing user access from several CICS regions

Several users in different CICS regions can access the CSD at the same time.

1. Give one CICS region read and write access to the CSD (**CSDACC=READWRITE** system initialization parameter). That CICS region should be at the latest level, to ensure that obsolete resource attributes from earlier release can still be updated safely. Other CICS regions should be given only read access to the CSD (**CSDACC=READONLY** system initialization parameter). This ensures that the CSD integrity is preserved for CICS regions in the same MVS image or different MVS images.

2. If you update your shared CSD from one region only, and use CEDA in all the other regions just to install into the required region, specify read/write access for the updating region and read-only for all other regions.
3. If you want to update the CSD from several CICS regions, you can use the CICS transaction routing facility, and MRO or ISC, to enable read-only CICS regions to update the CSD. The procedure to follow is:
 - a. Select one region that is to own the CSD (the CSD-owning region), and only in this region specify read/write access for the CSD.
 - b. Define the CSD as read-only to other CICS regions.
 - c. For all regions other than the CSD-owning region:
 - 1) Redefine the CEDB transaction as a remote transaction (to be run in the CSD-owning region).
 - 2) Install the definition and add the group to your group list for these regions.

You may then use the CEDB transaction from any region to change the contents of the CSD, and use CEDA to INSTALL into the invoking region. You cannot use CEDA to change the CSD in region(s) that do not own the CSD.

If the CSD-owning region fails, the CSD is not available through the CEDB transaction until emergency restart of the CSD-owning region has completed (when any backout processing on the CSD is done). If you try to install a CSD GROUP or LIST that is the target of backout processing, before emergency restart, you are warned that the GROUP or LIST is internally locked to another user. Do not run an offline VERIFY in this situation, because backout processing removes the internal lock when emergency restart is invoked in the CSD-owning region.

If you do not want to use the above method, but still want the CSD to be defined as a recoverable resource, then integrity of the CSD cannot be guaranteed. In this case, you must not specify CSDBKUP=DYNAMIC, because the CSD would not be suitable for BWO.

4. You can define several CICS regions with read/write access to the CSD, but this should only be considered if the CICS regions run in the same MVS image, and all are at the latest CICS level.
5. If you give several CICS regions read/write access to the same CSD, and those regions are in the same MVS image, integrity of the CSD is maintained by the SHAREOPTIONS(2) operand of the VSAM definition, as shown in the Figure 9 on page 45.
6. If you give several CICS regions read/write access to the same CSD, and those regions are in different MVS images, the VSAM SHAREOPTIONS(2) operand does not provide CSD integrity, because the VSAMs for those MVS images do not know about each other.

Shared access from CICS regions and DFHCSDUP:

If you want to use the DFHCSDUP utility program in read/write mode to update the CSD, you must ensure that no CICS users are using any of the CEDA, CEDB, or CEDC transactions.

For information about other factors that can restrict access to a CSD, see “Other factors restricting CSD access” on page 52.

For information about the system initialization parameters for controlling access to the CSD, see “Defining CSD attributes” on page 46.

Multiple users of the CSD within a CICS region (non-RLS)

If you have specified read/write access for the CSD, all the CEDA users in a CICS region can perform read and write functions. CICS file control manages concurrent access for multiple users within a region, using the attributes specified in the CSDACC system initialization parameter.

CICS protects individual resource definitions against concurrent updates by a series of internal locks on the CSD. CICS applies these locks at the group level. While CICS is executing a command that updates any element in a group, it uses the internal lock to prevent other RDO transactions within the region from updating the same group. CICS removes the lock record when the updating command completes execution. Operations on lists are also protected in this way.

The number of concurrent requests that may be processed against the CSD is defined by the CSDSTRNO system initialization parameter. Each user of CEDA (or CEDB or CEDC) requires two strings, so calculate the CSDSTRNO value by first estimating the number of users that may require concurrent access to the CSD, and then multiply the number by two.

CEDA issues a diagnostic message if the CSDSTRNO value is too small to satisfy the instantaneous demand on the CSD for concurrent requests. A subsequent attempt to reissue the command succeeds if the conflict has disappeared. If conflicts continue to occur, increase the CSDSTRNO value.

Sharing a CSD by CICS regions within a single MVS image (non-RLS)

The CSD may be shared by a number of CICS regions within the same MVS image.

You can maintain the integrity of the CSD in this situation by coding SHAREOPTIONS(2) on the VSAM definition, as shown in the Figure 9 on page 45. The CICS attributes of the CSD, as viewed by a given region, are defined in the system initialization parameters for that region.

You should consider defining:

- One CICS region with read/write access (CSDACC=READWRITE) to the CSD. That region can use all the functions of CEDA, CEDB, and CEDC.
- Other CICS regions with only read access (CSDACC=READONLY) to the CSD. Such CICS regions can use the CEDC transaction, and those functions of CEDA and CEDB that do not require write access to the CSD (for example, they can use INSTALL, EXPAND, and VIEW, but not DEFINE). You can enable such CICS regions to update the CSD, by using the procedure described in “Sharing user access from several CICS regions” on page 47.

Note: Read integrity is not guaranteed in a CICS region that has read-only access to a shared CSD. For example, if one CICS region that has full read/write access updates a shared CSD with new or changed definitions, another CICS region with read-only access might not obtain the updated information. This could happen if a control interval (CI) already held by a read-only region (before an update by a read/write region) is the same CI needed by the read-only region to obtain the updated definitions. In this situation, VSAM does not reread the data set, because it already holds the CI. However, you can minimize this VSAM restriction by specifying CSDLRNO=NONE, and the minimum values for CSDBUFNI and CSDBUFND, but at the expense of degraded performance.

See “Specifying read integrity for the CSD” on page 54 for information about read integrity in a data set accessed in RLS mode.

If you define several CICS regions with read/write access to the CSD, those regions should all be at the latest level. Only one CICS region with read/write access can use a CEDA, CEDB, or CEDC transaction to access the CSD, because the VSAM SHAREOPTIONS(2) definition prevents other regions from opening the CSD.

If you are running CICS with the CSD defined as a recoverable resource (CSDRECOV=ALL), see “Planning for backup and recovery” on page 55 for some special considerations.

You can use CEMT to change the file access attributes of the CSD, or you can use the **EXEC CICS SET FILE** command in an application program. However, ensure that the resulting attributes are at least equivalent to those defined either by CSDACC=READWRITE or CSDACC=READONLY. These parameters allow the following operations on the CSD:

READONLY

Read and browse.

READWRITE

Add, delete, update, read and browse.

Sharing a CSD in a multi-MVS environment (non-RLS)

If you need to share a CSD between CICS regions that are running in different MVS images, you should ensure that only one region has read/write access.

The VSAM SHAREOPTIONS(2) offers no integrity, because the VSAMs running in a multi-MVS environment do not know of each other.

Because of these limitations, active and alternate CICS regions running in different MVS images must not share the CSD with other CICS regions, unless you are using some form of global enqueueing (for example, with global resource serialization (GRS)).

These multi-MVS restrictions also apply to running the offline utility, DFHCSDUP.

Multiple users of one CSD across CICS or batch regions (non-RLS)

There are four types of activity that require access to the CSD.

The types of access needed in the four situations where the CSD are used are shown in Table 4.

Table 4. CSD access

| | Type of activity | Access |
|---|--|--|
| 1 | CICS region performing initialization (cold or initial start) | Read-only |
| 2 | CICS region running one or more CEDA, CEDB, or CEDC transactions | Read/write or read-only (as specified in the CSDACC parameter) |
| 3 | Batch region running utility program DFHCSDUP | Read/write or read only, depending on PARM parameter |

Table 4. CSD access (continued)

| | Type of activity | Access |
|---|---|------------|
| 4 | CICS regions performing emergency restart, and CSD file backout is required | Read/write |

Note the following limitations when the activities listed in Table 4 on page 50 are attempted concurrently:

1. You cannot run DFHCSDUP in read/write mode in a batch region if any CICS region using the same CSD is running one of the CEDA, CEDB, or CEDC transactions. (The exception is when the CEDx transactions accessing the CSD are in a region (or regions) for which the CSD is defined as read-only.)
2. None of the CEDx transactions runs if the CSD to be used is being accessed by the DFHCSDUP utility program in read/write mode. (This restriction does not apply if the transaction is run in a region for which the CSD is defined as read-only.)
3. None of the CEDx transactions runs in a CICS region whose CSD is defined for read-write access if any of the RDO transactions are running in another CICS region that has the CSD defined for read-write access.

A CICS region starting with an initial or cold start opens the CSD for read access only during initialization, regardless of the CSDACC operand. This enables a CICS region to be initialized even if a user on another region or the DFHCSDUP utility program is updating the CSD at the same time. After the group lists are installed, CICS leaves the CSD in a closed state.

On a warm or emergency start, the CSD is not opened at all during CICS initialization if CSDRECOV=NONE is coded as a system initialization parameter. However, if CSDRECOV=ALL is coded, and backout processing is pending on the CSD, the CSD is opened during CICS initialization on an emergency start.

Sharing the CSD between different releases of CICS

Resource attributes become obsolete when they have no relevance for a new release of CICS. CICS continues to display these on CEDx panels, but they are displayed as protected fields, indicating that they are not supported by this release.

Using the ALTER command on definitions that specify obsolete attributes does not cause the loss of these attributes, so you can safely update resource definitions using this release. If you are sharing the CSD with CICS regions at an earlier release, you can update the unsupported fields by using the PF2 function key to remove the protection when in ALTER mode. (PF2 is designated as the “compatibility” key (COM) on the CEDA or CEDB display panels.) Pressing PF2 converts protected fields to unprotected fields that you can modify. If you want to use this facility to enable you to share common resource definitions, the rule for sharing between different release levels of CICS is that you must update the CSD from the highest release level.

For information about using the CEDA and CEDB ALTER commands to update resource definitions in compatibility mode, see the *CICS Resource Definition Guide*.

You can also use the CSD utility program, DFHCSDUP, to update resources that specify obsolete attributes. A compatibility option is added for this purpose, which you must specify on the **PARM** parameter of the EXEC PGM=DFHCSDUP statement.

You indicate the compatibility option by specifying COMPAT or NOCOMPAT. The default is NOCOMPAT, which means that you cannot update obsolete attributes.

CICS regions that use DB2

If you share your CSD between different releases of CICS that use DB2[®], you must use the DB2 resource definitions appropriate for each release of CICS.

With those releases of CICS that ship the CICS DB2 attachment facility, you must use the CICS-supplied group called DFHDB2. This group is included in the CICS-supplied startup list, DFHLIST, and specifies different program names from the attachment facility provided by DB2.

For earlier releases of CICS that do not provide the DFHDB2 group, you must use your own resource definitions that specify the resource names appropriate for the release of CICS and DB2.

CICS-supplied compatibility groups

If you are sharing the CSD between CICS Transaction Server for z/OS, Version 4 Release 1 and an earlier release of CICS, you must ensure that the group list you specify on the **GRPLIST** system initialization parameter contains all the CICS-required standard definitions.

When you upgrade the CSD to the CICS Transaction Server for z/OS, Version 4 Release 1 level, some of the IBM groups referenced by your group list are deleted and the contents transferred to one of the compatibility groups, DFHCOMPx. To ensure that these continue to be available to your CICS regions of earlier releases, add the compatibility groups *after* all the other CICS-supplied definitions.

For information about upgrading your CSD, and about the compatibility groups in CICS Transaction Server for z/OS, Version 4 Release 1, see the *CICS Resource Definition Guide*.

Other factors restricting CSD access

Access to the CSD may also be restricted if it is left open after abnormal termination of a CEDA, CEDB, or CEDC transaction.

If the CSD is left open with write access, this prevents other address spaces from subsequently opening it for write access. This situation can be remedied by using CEMT to correct the status of the CSD.

Access to the CSD is not released until the RDO transaction using it is ended, so users of CEDA, CEDB, and CEDC should ensure that a terminal running any of these transactions is not left unattended. Always end the transaction with PF3 as soon as possible. Otherwise, users in other regions are unable to open the CSD.

There may be times when you cannot create definitions in a group or list. This situation arises if an internal lock record exists for the group or list you are trying to update. If you are running the DFHCSDUP utility program (or a CEDA transaction) when this occurs, CICS issues a message indicating that the group or list is locked. As described under “Multiple users of the CSD within a CICS region (non-RLS)” on page 49, this is normally a transient situation while another user within the same region is updating the same group or list. However, if a failure occurs, preventing a CEDA transaction from completing successfully, and CSDRECOV=NONE is coded, the internal lock is not removed and is left in force. (If CSDRECOV=ALL is coded, the CSD is recoverable and file backout occurs and frees the lock.) This could happen, for example, if a system failure occurs while a

CEDA transaction is running; it could also happen if the CSD becomes full. You can remedy this situation by running the DFHCSDUP utility program with the VERIFY command.

However, if you have coded CSDRECOV=ALL, make sure no backout processing is pending on the CSD before you run an offline VERIFY. The effect of coding CSDRECOV=ALL is discussed more fully under “Planning for backup and recovery” on page 55.

Sharing the CSD in RLS mode

This section discusses the use of VSAM RLS to enable the CSD to be shared between a number of CICS regions.

About this task

The reasons for, and benefits of, sharing the CSD are the same regardless of the access mode. However, there are some RLS-related factors that you need to consider if you decide to operate CICS with the CSD in RLS mode.

The following requirements and rules apply to using the CSD in RLS-mode:

- Your CICS regions must run in an RLS-capable environment. That is, all the CICS regions must reside in a parallel sysplex, and an SMSVSAM server must be running in each MVS image that supports one or more CICS regions.
- The CSD must reside in SMS-managed storage.
- You must specify CSDRLS=YES in all CICS regions that are sharing the CSD in RLS-mode, and RLS must be enabled in each region (by the RLS=YES system initialization parameter).
- As soon as the first CICS region opens the CSD in RLS mode, it can only be opened in RLS mode by other CICS regions. If a CICS region attempts to open the CSD in non-RLS mode when it is open in RLS mode by other regions, the non-RLS open request fails.

Note: This rule means that you cannot use a CSD in RLS mode on a CICS release that supports RLS and share it with CICS regions that do not support RLS. The sharing by non-RLS capable regions means that a CSD can only be use in non-RLS mode.

- All the rules governing the use of a data set in RLS mode apply also to the CSD—there are no special rules for the CSD because it is a CICS system data set.
- Any number of CICS regions can open the CSD in RLS mode and all can use CEDA to update the data set with full integrity. The CICS regions can reside in different MVS images, but the MVS images must be in the same sysplex. There is no need to restrict updating to only one CICS region as in the case of non-RLS sharing, and you can specify the CSDACC=READWRITE system initialization parameter for all CICS regions that specify CSDRLS=YES.

Differences in CSD management between RLS and non-RLS access

Although a CSD accessed in RLS mode is protected by VSAM RLS locking, this operates at the CICS file control level. It does not change the way the CEDA and CEDB transactions manage the integrity of CSD groups.

The CEDx transactions protect resource definitions in the same way for RLS mode and non-RLS mode CSDs. They protect individual resource definitions against

concurrent updates by a series of internal locks on the CSD. The RDO transactions apply these locks at the group level. While RDO transactions are executing a command that updates any element in a group, they use the internal lock to prevent other RDO transactions within a CICS region from updating the same group. The locks are freed only when the updating command completes execution. Operations on lists are protected in the same way. However, in an RLS environment, these internal locks affect all CICS regions that open the CSD in RLS mode. In the non-RLS case they apply only to the CICS region that has the data set open for update (which can only be a single region).

The use of a single buffer pool by the SMSVSAM server removes some of the problems of sharing data that you get with a non-RLS CSD.

Some other points to note are:

- If a CSD is defined with CSDACC=READWRITE and CSDRLS=YES, more than one CICS region can open the CSD concurrently. However, file control closes the CSD opened in RLS mode at termination of each CEDx transaction, in the same way as for a non-RLS CSD. CSDACC=READONLY is not necessary for a CSD accessed in RLS mode.
- The number of concurrent requests that can be processed against the CSD, is always 1024 for RLS. Diagnostic messages about CSDSTRNO value do not occur for RLS-mode CSDs.
- The VSAM cluster definition SHAREOPTIONS parameter is ignored by SMSVSAM when an application, such as CICS, opens a data set in RLS mode.
- A CSD accessed in RLS mode could give rise to RDO transaction failures that do not occur for a non-RLS mode CSD: For example:
 - An RDO transaction could be holding an RLS exclusive lock while it updates a record, which causes another RDO transaction to time out.
 - If the CSD is recoverable and CICS or MVS fails, update locks of failed-inflight RDO transactions are converted into retained locks. This could result an RDO transaction receiving a LOCKED response from VSAM which, in turn, would cause the RDO transaction to fail.

Specifying read integrity for the CSD

You can specify that you want read integrity for a CSD opened in RLS mode.

This ensures that CEDx INSTALL command always installs the latest version of a resource definition. The CEDA INSTALL command has to wait for a lock on any CSD record that it is trying to install if another CEDx transaction is updating the record. The install completes only when the updating task has finished updating the record and released its exclusive lock.

Although the CSDINTEG system initialization parameter supports both consistent and repeatable read integrity, consistent read integrity should provide all the benefit you need for your RDO operations.

Specifying file control attributes for the CSD

You specify file control attributes for the CSD using the CSDxxxx system initialization parameters, with a few exceptions.

These exceptions are as follows:

CSDBKUP

You specify backup-while-open support for the CSD using the VSAM **BWO** parameter in the ICF catalog.

CSDBUFND

Ignored.

CSDBUFNI

Ignored.

CSDFRLOG

You specify the forward recovery log stream for the CSD using the VSAM **LOGSTREAMID** parameter in the ICF catalog.

CSDINTEG

You specify read integrity for RDO transactions (CED x) using this system initialization parameter.

CSDLSRNO

Ignored.

CSDRECOV

You specify the recovery attributes for the CSD using the VSAM **LOG** parameter in the ICF catalog. If **LOG** is “undefined”, any attempt to open the CSD in RLS mode will fail.

CSDSTRNO

For RLS, the number of strings defaults to 1024.

Effect of RLS on the CSD batch utility DFHCSDUP

You can use DFHCSDUP to update a **non-recoverable** CSD in RLS mode while CICS also has the CSD open for update in RLS mode.

To enable DFHCSDUP to update the CSD in RLS mode, specify RLS=NRI or RLS=CR in the DD statement for the CSD in the DFHCSDUP JCL. Generally, DFHCSDUP does not perform as well in RLS mode as in non-RLS mode.

You cannot run DFHCSDUP while CICS regions have the CSD open in RLS mode if the CSD is defined as recoverable. This is because a non-CICS job, such as DFHCSDUP, is not allowed to open a recoverable data set for output in non-RLS mode while it is already open in RLS mode. Therefore, before you can run DFHCSDUP, you must quiesce the CSD by issuing a CEMT, or an EXEC CICS, SET DSNAME(...) QUIESCED command.

A recoverable CSD is unavailable to all CICS regions while DFHCSDUP is running until it is unquiesced, which makes it available again in RLS mode. To unquiesce the CSD at the end of the DFHCSDUP run, issue a CEMT, or an EXEC CICS, DSNAME(...) UNQUIESCED command.

For a recoverable CSD, the main factor to consider when planning whether to use RLS is how much you use DFHCSDUP compared with the CED x transactions. If you use DFHCSDUP frequently to update your production CSD, you may decide that it is better to use the CSD in non-RLS mode. On the other hand, if you use DFHCSDUP only occasionally, and you want the ability to update the CSD online from any CICS region, use RLS.

Planning for backup and recovery

To guard against system failures that affect your CSD, take a backup of the CSD at regular intervals. If the CSD is corrupted for any reason, you can restore it to its state at the last backup.

About this task

To keep the backup of the CSD as up-to-date as possible, make an image copy of your CSD before each period of update activity, either by an RDO transaction or DFHCSDUP.

Alternatively, because the CSD is open for update whenever RDO work is taking place, it is a good candidate for eligibility for BWO. If the CSD is specified as eligible for BWO, and the data set is corrupted, you can restore a BWO image of the CSD using DFSMSdss, then run forward recovery to the point of corruption using a forward recovery utility.

For a CSD opened in RLS mode, the recovery attributes must be defined in the ICF catalog entry for the CSD, and CICS uses the forward recovery log's log stream name (LSN) from the ICF catalog.

For a CSD opened in non-RLS mode, the recovery attributes can be defined in the ICF catalog entry for the CSD, or on the CSD system initialization parameters. The forward recovery log stream name (LSN) is retrieved from either CSDFRLOG or the ICF catalog. If LOG is defined in the catalog, the forward recovery log stream specified in the catalog is used. If LOG is not defined, the CSDFRLOG journal ID is used to determine the log stream name. If the **NONRLSRECOV** system initialization parameter is set to FILEDEF, the CSDFRLOG journal ID is always used to determine the log stream name. Any recovery attributes specified on the ICF catalog are ignored.

For a CSD opened in non-RLS mode, you can use the system initialization parameter CSDBKUP=DYNAMIC|STATIC to indicate whether the CSD is eligible for BWO. Specify CSDBKUP=DYNAMIC for BWO support, or STATIC (the default) for a "normal" quiesced backup. If you specify BWO support for the CSD you must also define it as forward recoverable. For more information about BWO, see "Defining backup while open (BWO) for VSAM files" on page 10.

For a CSD opened in RLS mode, you must specify all recovery attributes, which includes backup, in the ICF catalog. BWO backup eligibility is specified using BWO(TYPECICS).

If you specify forward recovery for the CSD, changes (after images) made by CICS to the CSD are logged in the forward recovery log stream. Using the latest backup, and the after images from forward recovery log stream, you can recover all the changes made by running a recovery program, such as the CICS VSAM forward recovery utility. After performing forward recovery, you must reenter any CEDA transactions that were running at the time of failure, as these are effectively backed out by the forward recovery process. You can find details of these in the CSDL transient data destination, which is the log for copies of all CEDA commands. See "Logging RDO commands" on page 59 for more information.

Recoverability, forward recovery log stream names, and BWO eligibility can be defined optionally in the ICF catalog for a non-RLS accessed CSD, but must be defined the ICF catalog if the CSD is accessed in RLS mode.

The CSDBKUP, CSDRECOV, and CSDFRLOG system initialization parameters interact according to how they are specified. Table 5 on page 57 and Table 6 on page 57 summarize their effects when the SIT is assembled and during CICS override processing, respectively.

Table 5. CSDBKUP and related parameters at SIT assembly time for CSDRLS=NO

| CSDRECOV | CSDFRLOG | CSDBKUP | Result |
|---------------------|---------------------------|--------------------------|--|
| ALL | FRLOG from 01 through 99. | Either DYNAMIC or STATIC | OK |
| ALL | NO | Either DYNAMIC or STATIC | SIT assembly fails with MNOTE stating that CSDRECOV=ALL implies that the CSDFRLOG option must be specified. |
| BACKOUTONLY or NONE | FRLOG from 01 through 99. | DYNAMIC | SIT assembly fails with assembler MNOTES stating that CSDBKUP=DYNAMIC requires CSDRECOV=ALL and that CSDFRLOG requires CSDRECOV=ALL. |
| BACKOUTONLY or NONE | NO | DYNAMIC | SIT assembly fails with an assembler MNOTE stating that CSDBKUP=DYNAMIC requires CSDRECOV=ALL. |
| BACKOUTONLY or NONE | NO | STATIC | OK |
| BACKOUTONLY or NONE | FRLOG from 01 through 99. | STATIC | SIT assembly warning MNOTE stating that CSDFRLOG is ignored unless CSDRECOV=ALL. |

Note:

1. When CSDBKUP=DYNAMIC, the CSD is eligible for BWO.
2. Backup and recovery attributes must be specified in the ICF catalog for a CSD opened in RLS mode (CSDRLS=YES).
3. Backup and recovery attributes can optionally be specified in the ICF catalog for a CSD opened in non-RLS mode (CSDRLS=NO), but you must still have a consistent set of parameters as defined in the table above.

Table 6. CSDBKUP and related system initialization parameters during CICS override processing (CSDRLS=NO)

| CSDRECOV | CSDFRLOG | CSDBKUP (see Notes) | Result |
|---------------------|---------------------------|---------------------------|---|
| ALL | FRLOG from 01 through 99. | Either DYNAMIC or STATIC. | OK |
| ALL | NO | Either DYNAMIC or STATIC. | Message DFHPA1944 is issued stating that CSDRECOV=ALL cannot be specified without a CSDFRLOG if CSDRLS=NO. CICS initialization is terminated. |
| BACKOUTONLY or NONE | FRLOG from 01 through 99. | DYNAMIC | Processing continues and messages DFHPA1929, stating that CSDBKUP has defaulted to STATIC, and DFHPA1930, stating that CSDFRLOG has been ignored, are issued. |

Table 6. CSDBKUP and related system initialization parameters during CICS override processing (CSDRLS=NO) (continued)

| CSDRECOV | CSDFRLOG | CSDBKUP (see Notes) | Result |
|---------------------|---------------------------|---------------------|---|
| BACKOUTONLY or NONE | NO | DYNAMIC | Processing continues and message DFHPA1929 is issued, stating that CSDBKUP has defaulted to STATIC. |
| BACKOUTONLY or NONE | NO | STATIC | OK |
| BACKOUTONLY or NONE | FRLOG from 01 through 99. | STATIC | Processing continues and message DFHPA1930 stating that CSDFRLOG has been ignored is issued. |

Note:

1. When CSDBKUP=DYNAMIC, the CSD is eligible for BWO.
2. Backup and recovery attributes must be specified in the ICF catalog for a CSD opened in RLS mode (CSDRLS=YES).
3. Backup and recovery attributes can optionally be specified in the ICF catalog for a CSD opened in non-RLS mode (CSDRLS=NO), but you must still have a consistent set of parameters as defined in the table above.

Write and test procedures for backing up and recovering your CSD before beginning to operate a production CICS region.

Forward recovery of the CSD is not possible if CSD updates are made outside CICS. To enable recovery of the updates made outside CICS, you need to use an image copy. If you update the CSD from outside CICS, do not use CEDA to update the CSD until an image copy has been made.

Transaction backout during emergency restart

If you define the CSD as a recoverable resource, by coding the **CSDRECOV** system initialization parameter, the same rules apply to the CSD as to any other CICS recoverable resource.

If you code CSDRECOV=ALL (or BACKOUTONLY) as a system initialization parameter, and have to perform an emergency restart following a failure, CICS backs out any incomplete RDO transactions that were in-flight at the time of failure.

Dynamic backout for transactions

CICS performs dynamic transaction backout for any RDO transaction abends.

You cannot decide whether you want dynamic transaction backout by coding an attribute on transaction definitions in the CSD; CICS assumes this requirement for all transactions. (Defining the CSD as non-recoverable has the effect of preventing backout, but this is not recommended.)

Other recovery considerations

When you are deciding what recoverability options to specify, you must consider a number of factors.

These factors are as follows:

- CEDA command syncpoint criteria
- Sharing the CSD with another CICS region
- Accessing the CSD by the offline utility program DFHCSDUP

For information about CEDA command syncpoint criteria, see “CEDA command syncpoint criteria.” For information about sharing the CSD between CICS regions, see “Sharing the CSD in non-RLS mode” on page 47. For information about using the DFHCSDUP utility to access the CSD, see “Accessing the CSD by the offline utility program, DFHCSDUP.”

CEDA command syncpoint criteria

You can issue a CEDA command on the command line, or issue the command as a series of single commands from an EXPAND or DISPLAY panel.

Commands that change the contents of the CSD commit or back out changes at the single command level. The exception to this rule is a generic ALTER command. A generic ALTER command is committed or backed out at the single resource level.

The replacement of an existing resource definition by an INSTALL command only occurs if the resource is not in use. If any of the resources in the group being installed are in use, the install will fail.

Changes made to the following resource definitions by an INSTALL command are committed at the resource level and are not backed out if the install fails:

- AUTOINSTALL MODEL,FILE, LSRPOOL, MAPSET, PARTITIONSET, PARTNER, PROFILE, PROGRAM, TDQUEUE, and TRANSACTION,

Changes made to the following resource definitions by an INSTALL command are committed at the group level and are backed out if the install fails:

- CONNECTION, SESSION, TERMINAL, and TYPETERM

Accessing the CSD by the offline utility program, DFHCSDUP

Changes made to the CSD by the offline utility program DFHCSDUP are not recoverable.

Also consider the effects of using commands provided by this program, before emergency restart of a failing CICS region, that:

1. Change the contents of lists or groups that are the target of backout processing.
2. Remove internal locks (by using VERIFY, for example).

These situations are analogous to the problems met when using multiple read/write regions, and are discussed above.

Logging RDO commands

If you want to record RDO commands, use the sample job to create definitions for the extrapartition queues CADL, CAIL, CRDI, CSDL, CSFL, CSKL, CSLB, CSPL, and CSRL.

About this task

These queues are used as follows:

CADL Logs VTAM resources installed in the active CICS region. CICS records in this log all terminal entries installed in the TCT, entries deleted from the

TCT, and dynamically installed entries that are discarded. This log includes autoinstalled terminal definitions, terminal definitions installed explicitly by the CEDA INSTALL command, and terminal definitions installed from a group list during system initialization.

- CAIL** Logs autoinstall terminal model entries installed in the TCT, and entries deleted from the TCT.
- CRDI** Logs installed resource definitions, such as programs, transactions, mapsets, profiles, partition sets, files, and LSR pools.
- CSDL** Logs RDO commands that affect the CSD.
- CSFL** Logs file resources installed in the active CICS region. That is, all file entries installed in the FCT, entries deleted from the FCT, dynamically installed entries that are discarded, and messages from dynamic allocation of data sets and from loading CICS data tables.
- CSKL** Logs transaction and profile resources installed in the active CICS region. That is, all transaction and profile entries installed in the PCT, entries deleted from the PCT, and dynamically installed entries that are discarded.
- CSLB** Logs changes to the dynamic LIBRARY resources installed in the active CICS region. That is, install and discard changes, and any changes to the enablement, ranking or critical status of the LIBRARY. If no dynamic LIBRARY resources are installed, no audit logging will take place.
- CSPL** Logs program resources installed in the active CICS region. That is, all program entries installed in the PPT, entries deleted from the PPT, and dynamically installed entries that are discarded.
- CSRL** Logs changes to the set of partner resources installed in the active CICS region. That is, all operations that install or discard partner resources.

If you want these RDO command logs sent to the same destination (CSSL) as the messages, you can use the definitions shown in Figure 10 on page 61. If you like, you can direct these logs to any other transient data queue, or define them as extrapartition data sets.

```

*
DEFINE TDQUEUE (CSSL)          GROUP(DFHDCTG)
  DESCRIPTION(USED FOR MESSAGES)
  TYPE(EXTRA)                  TYPEFILE(OUTPUT)
  RECORDSIZE(132)              BLOCKSIZE(136)
  RECORDFORMAT(VARIABLE)      BLOCKFORMAT(UNBLOCKED)
                               DDNAME(MSGUSR)
*
DEFINE TDQUEUE (CADL)          GROUP(DFHDCTG)
  DESCRIPTION(CEDA VTAM RESOURCE LOGGING)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CAIL)          GROUP(DFHDCTG)
  DESCRIPTION(AITM MESSAGES)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CRDI)          GROUP(DFHDCTG)
  DESCRIPTION(RDO INSTALL LOG)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CSLB)          GROUP(DFHDCTG)
  DESCRIPTION(CICS LD Domain LIBRARY Audit Trail)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CSDL)          GROUP(DFHDCTG)
  DESCRIPTION(CEDA COMMAND LOGGING)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CSFL)          GROUP(DFHDCTG)
  DESCRIPTION(FILE ALLOCATION MESSAGES)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CSKL)          GROUP(DFHDCTG)
  DESCRIPTION(TRANSACTION MANAGER MESSAGES)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CSPL)          GROUP(DFHDCTG)
  DESCRIPTION(PROGRAM MANAGER MESSAGES)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)
*
DEFINE TDQUEUE (CSRL)          GROUP(DFHDCTG)
  DESCRIPTION(PARTNER RESOURCE MANAGER)
  TYPE(INDIRECT)              INDIRECTNAME(CSSL)

```

Figure 10. Definitions for RDO command logs sent to CSSL

Making the CSD available to CICS

To make your CSD available to CICS, you can either include a DD statement in the CICS startup job or use dynamic allocation.

About this task

Procedure

1. Include the following DD statement in your CICS startup job stream:

```
//DFHCSD DD DSN=CICSTS41.CICS.applid.DFHCSD,DISP=SHR
```

You usually need the CSD DD statement to include DISP=SHR. (See “Sharing the CSD in non-RLS mode” on page 47.) If you include a DD statement for the

CSD in the CICS startup job, the CSD is allocated at the time of CICS job step initiation and remains allocated for the duration of the CICS job step.

2. To dynamically allocate the CSD, specify the data set name (DSNAME) and disposition (DISP) of the CSD, using one of the following methods:
 - The **CSDDSN** and **CSDDISP** system initialization parameters
 - The **CEMT SET FILE** command
 - The **EXEC CICS SET FILE** command

Do not provide a DD statement for the CSD in the startup job stream. If there is a CSD DD statement, it is used instead of dynamic allocation. CICS uses the full data set name (DSNAME) to allocate the CSD as part of OPEN processing. The CSD is automatically deallocated when the last entry associated with it is closed.

What to do next

For more information about OPEN processing, see Chapter 10, “Defining user files,” on page 93. For information about the parameters that you can code for the CSD in the SIT, see Chapter 16, “Specifying CICS system initialization parameters,” on page 133.

Installing the RDO transactions

The RDO transactions, CEDA, CEDB, and CEDC are defined in the CICS-supplied group, DFHSPI.

This group is also included in DFHLIST, the CICS group list. Ensure that a copy of DFHSPI is included in the group list that you use for your CICS startup. You specify the group list on the GRPLIST system initialization parameter.

For information about the CEDA, CEDB, and CEDC transactions, see *CICS Supplied Transactions*.

Installing definitions for the Japanese language feature

If you have the Japanese language feature, install the definitions for the feature in the CSD, by running DFHCSDUP.

When you run DFHCSDUP, specify the following option:

```
UPGRADE USING(DFHRDJPN)
```

For information about the DFHCSDUP utility program and the available commands, see the *CICS Resource Definition Guide*.

CSD XRF considerations

If you are running CICS with XRF, both the active and alternate CICS regions must refer to the same CICS system definition data set (that is, the CSD must be passively shared).

About this task

The active and alternate CICS regions can share the same CSD even if they are running on different MVS images. A CICS region running with XRF=YES may also

share the CSD with other CICS regions within the same MVS image. (For a definition of passively and actively shared data sets in an XRF environment, see “Planning data sets for XRF” on page 8.)

The CSD is allocated by MVS when the CICS job step is initiated. This means the DD statements in the CICS startup job streams defining the CSD for the active and alternate CICS regions must specify DISP=SHR.

The alternate CICS region does not open the CSD during initialization, or before takeover occurs. The alternate CICS region does not even open the CSD during takeover, if the CSD was not changed at any time by the active CICS region. (For example, the CSD might have been used only to install a group list at CICS startup, and subsequently by read-only operations.) However, if you use the CEDA transaction in an active CICS region to alter resource definitions, the CSD might be opened at takeover, to perform any file backout that is necessary. To enable file backout to occur, you must define the CSD as a recoverable resource by the system initialization parameter CSDRECOV; see “Defining CSD attributes” on page 46.

For more information about using the CSD as a recoverable file, see “Planning for backup and recovery” on page 55.

Chapter 6. Setting up the catalog data sets

You must define and initialize new CICS catalogs for CICS Transaction Server for z/OS, Version 4 Release 1.

About this task

You must define the CICS global catalog data set and the CICS local catalog data set, which CICS requires to catalog CICS system information. Throughout the documentation, these data sets are referred to as the global catalog and the local catalog. The CICS catalog data sets are not connected with MVS system catalogs and contain data that is unique to CICS.

For information on how CICS uses the catalogs, including startup and restart, see “The role of the CICS catalogs” on page 308

To estimate the amount of space needed in your global catalog, see “Space calculations” on page 67

To set up the catalog data sets:

Procedure

1. Define the global catalog.
2. Define the local catalog.

Defining the global catalog

The global catalog is a VSAM key-sequenced data set (KSDS) is used to store start type information, location of the CICS system log, resource definitions, terminal control information and profiles. In an XRF environment, there is only one global catalog. It is shared passively between the active and the alternate CICS regions.

About this task

CICS uses the global catalog to perform the following activities:

- To record information that governs the possible types of start and the location of the CICS system log.
- During the running of CICS, to hold the resource definitions that are installed during initialization when CICS installs the group list, by the RDO **CEDA INSTALL** command or by the **EXEC CICS CREATE** command.
- During a normal shutdown, to record terminal control information and profiles. All other warm keypoint information is written to the CICS system log.

You must ensure that the REGION parameter on your CICS jobs is high enough to cope with the increase in buffer storage used for the global catalog, because this storage comes out of region storage not EDSA.

You can define and initialize the CICS global catalog in two ways. You can use the sample job as described below, or you can use the CICS-supplied job, DFHDEFDS.

Edit the sample job:

Procedure

1. Edit the data set name in the CLUSTER definition to be the same as the DSN parameter in the DD statement for the global catalog in the CICS startup job stream.
2. The primary and secondary extent sizes are shown as *n1* and *n2* cylinders. Calculate the size required to meet your installation requirements, and substitute your values for *n1* and *n2*. Whichever **IDCAMS** parameter you use for the global catalog data set space allocation (CYLINDERS, TRACKS, or RECORDS), make sure that you specify a secondary extent. CICS abends if your global catalog data set fills and VSAM cannot create a secondary extent. For information about record sizes, see Table 7 on page 68
3. Specify the REUSE option on the **DEFINE CLUSTER** command. This option enables the global catalog to be opened repeatedly as a reusable cluster. Also specify REUSE if you intend to use the COLD_COPY input parameter of the DFHRMUTL utility.
4. Edit the *CONTROLINTERVALSIZE* values for the VSAM definition if required. This job does not specify a minimum or maximum buffer size explicitly, but accepts the default that is set by VSAM. You can code an explicit value if you want to define buffers of a specific size. See “Buffer space sizings” on page 70 for more information.
5. Use the recovery manager utility program, DFHRMUTL, to initialize the data set. Specify this utility in the job step INITGCD. DFHRMUTL writes a record to the data set, specifying that on its next run using this global catalog, if START=AUTO is specified, CICS is to perform an initial start and not prompt the operator for confirmation. This record is called the autostart override record.
6. Add a job step to run the DFHCCUTL utility. Adding this step means that the global and local catalogs do not get out of step.
7. Define the data definition statement for CICS as:

```
//DFHGCD DD DSN=CICSTS41.CICS.applid.DFHGCD,DISP=OLD
```

This example shows the minimum specification for a global catalog for use by a single CICS region.

- a. Add the relevant **AMP** subparameters to help improve restart and shutdown time. The **AMP** parameter is described in *z/OS MVS JCL Reference* and an example is shown in the CICS startup job stream in Chapter 19, “CICS startup,” on page 319.
- b. Specify DISP=SHR if you are running CICS with XRF.

Example

```
| //GLOCAT JOB accounting info,,CLASS=A
| //DEFGCD EXEC PGM=IDCAMS
| //SYSPRINT DD SYSOUT=*
| //SYSIN DD *
| DEFINE CLUSTER -
| (NAME(CICSTS41.CICS.app1id.DFHGCD) -
| INDEXED -
| CYLINDERS(n1 n2) -
| FREESPACE(10 10) -
| SHAREOPTIONS(2) -
| RECORDSIZE(4089 32760) -
| REUSE -
| VOLUMES(vol1id) -
| DATA -
| (NAME(CICSTS41.CICS.app1id.DFHGCD.DATA) -
| CONTROLINTERVALSIZE(32768) -
| KEYS(52 0)) -
| INDEX -
| (NAME(CICSTS41.CICS.app1id.DFHGCD.INDEX) )
| /*
| //INITGCD EXEC PGM=DFHRMUTL,REGION=1M
| //STEPLIB DD DSNAME=CICSTS41.CICS.SDFHLOAD,DISP=SHR
| //SYSPRINT DD SYSOUT=A
| //DFHGCD DD DSNAME=CICSTS41.CICS.app1id.DFHGCD,DISP=OLD
| //SYSIN DD *
| SET_AUTO_START=AUTOINIT
| /*
```

Figure 11. Example job to define and initialize the global catalog

Space calculations

Estimate the amount of space needed in your global catalog to keypoint installed resource definitions, table entries, and control blocks. Each global catalog record has a 52-byte key. Each entry is one VSAM record, and the records for each type of table have different keys.

You must regularly review your space usage to ensure that extents are not running too close to the limit for your environment. Use the sizes specified in Table 7 on page 68 to estimate the amount of space needed in your global catalog.

The space requirements for a VSAM KSDS such as DFHGCD can vary for different CICS cold starts. They can vary when no changes have been made to the CICS definitions that are going to be stored on the VSAM KSDS, because VSAM uses the space in the data set differently depending on whether the data set has just initialized or has data from a previous run of CICS. CICS calls VSAM to perform sequential writes. VSAM honors the **freespace** value specified on the data set's definition if the keys of the records being added sequentially are higher than the highest existing key. However, if the data set contains existing records with a higher key than the ones being inserted, the **freespace** value is only honored after a CI split has occurred.

The size of the index portion of the data set can also vary depending on the number of CI and CA splits that have occurred. The size affects the index sequence set.

When you are initializing the global catalog, you can use the **COLD_COPY** parameter; for example, SET_AUTO_START=AUTOCOLD,COLD_COPY. The cold copy creates a reduced copy of the global catalog data set, which improves the performance of the cold start. The CI splits cease after the first cold start and the data set will not expand into additional extents. Alternatively, you can reorganize or reinitialize the data set from time to time.

Note: In Table 1, the **Number of bytes per entry** column includes the 52-byte key length.

Table 7. Sizes for entries in the global catalog

| Installed definition, table entry, or control block | Number of bytes per entry | Size in previous release if different |
|---|---|---------------------------------------|
| Installed ATOMSERVICE definition | 660 bytes | |
| Installed BUNDLE definition 1 | 300 bytes up to 800 bytes approximately | |
| Installed CONNECTION definition | 528 bytes | 440 bytes |
| Installed CORBASERVER definition | 1304 bytes | 1216 bytes |
| Installed DB2CONN definition | 1548 bytes | 1460 bytes |
| Installed DB2 ENTRY definition | 332 bytes | 244 bytes |
| Installed DB2TRAN definition | 160 bytes | 62 bytes |
| Installed DJAR definition | 432 bytes | 344 bytes |
| Installed DOCTEMPLATE definition | 284 bytes | 196 bytes |
| Installed ENQMODEL definition | 152 bytes | 64 bytes |
| Installed EVENTBINDING and CAPTURESPEC definitions 2 | 4000 bytes approximately | |
| Installed extrapartition queue definition | 392 bytes | 296 bytes |
| Installed FILE definition | 588 bytes | 500 bytes |
| Installed indirect queue definition | 180 bytes | 92 bytes |
| Installed intrapartition queues definition | 328 bytes | 240 bytes |
| Installed IPCONN definition | 402 bytes | 312 bytes |
| Installed JOURNALMODEL definition | 168 bytes | 80 bytes |
| Installed JVM Program definition | 168 bytes up to 307 bytes | 80 bytes up to 219 bytes |
| Installed JVMSERVER definition | 146 bytes | |
| Installed LIBRARY definition | 852 bytes | 764 bytes |
| Installed MQCONN definition | 620 bytes | |
| Installed MQINI definition | 212 bytes | |
| Installed PARTNER definition | 148 bytes | 124 bytes |
| Installed PIPELINE definition | 1500 bytes | 1412 bytes |
| Installed PROCESSTYPE definition | 148 bytes | 60 bytes |
| Installed PROFILE definition | 158 bytes | 70 bytes |
| Installed PROGRAM definition | 168 bytes | 44 bytes |
| Installed REQUESTMODEL definition | 226 bytes | 138 bytes |
| Installed remote queue definition | 172 bytes | 84 bytes |
| Installed TCPIPSERVICE definition | 924 bytes | 580 bytes |
| Installed model TERMINAL definitions 3 | 634 bytes | 610 bytes |
| Installed TRANCLASS definitions | 124 bytes | 36 bytes |
| Installed TRANSACTION definitions (without TPNAME) | 244 bytes | 140 bytes |
| Installed TRANSACTION definitions (with TPNAME or XTPNAME) | 388 bytes | 204 bytes |
| Installed TSMODEL definition | 236 bytes | 148 bytes |
| Installed TYPETERM definitions 3 | 634 bytes | 610 bytes |

Table 7. Sizes for entries in the global catalog (continued)

| Installed definition, table entry, or control block | Number of bytes per entry | Size in previous release if different |
|--|---------------------------|---------------------------------------|
| Installed VSAM file (or data table) definition | 312 bytes | 288 bytes |
| Installed URIMAP definition | 1316 bytes | 1220 bytes |
| Installed WEBSERVICE definition | 1040 bytes | 936 bytes |
| BDAM file control table entry (FCT) | 284 bytes | 146 bytes |
| BDAM data control blocks | 156 bytes | 132 bytes |
| VSAM LSR share control blocks 4 | 1224 bytes | 1184 bytes |
| Data set names (JCL or dynamically allocated) 5 | 96 bytes | 72 bytes |
| Data set name blocks | 194 bytes | 170 bytes |
| Eventprocess status | 89 bytes | |
| File control recovery blocks 6 | 149 bytes | 125 bytes |
| Terminal control table entry (TCT) | 1552 bytes | 1500 bytes |
| | approximately | approximately |
| Dump table entry | 100 bytes | 76 bytes |
| Interval control element (ICE) | 120 bytes | 96 bytes |
| Automatic initiator descriptor (AID) | 120 bytes | 96 bytes |
| Deferred work element (DWE) 7 | 132 bytes | 108 bytes |
| Installed journal | 111 bytes | 88 bytes |
| Recovery manager remote names | 158 bytes | 134 bytes |
| Transient data destination record | 70 bytes | 46 bytes |
| Transient data destination auxiliary record | 58 bytes | 34 bytes |
| Loader program definitions | 68 bytes | 44 bytes |
| Session TCTTEs | 918 bytes | 894 bytes |
| Log streams | 112 bytes | 88 bytes |
| Uri virtual hosts | 180 bytes | 156 bytes |

Notes

1. The size of a bundle catalog record depends on the number of parts that are included in the bundle, the size of the bundle directory, and the length of the scope. The minimum size for a bundle with no scope and small directory is approximately 300 bytes. However, with a scope and large directory, the number of bytes can increase to 800. For each bundle part, the size can also range from 300 to 800 bytes.
2. The size of an event binding catalog record depends on the number of capture specifications in the event binding and the number of filters and capture data items in each capture specification. If required, you might have multiple catalog records for each event binding.
3. The TYPETERM and model TERMINAL definitions are present if you are using autoinstall. They are stored directly in the global catalog when the definitions are installed, either by a CEDA transaction, or as members of a group installed through a group list. For example, if you start CICS with the startup parameter GRPLIST=DFHLIST, the CICS-supplied TYPETERM and model terminal definitions, defined in the groups DFHTERM and DFHTYPE, are recorded in the global catalog. Allocate space in your calculations for all autoinstall resources installed in your CICS region.
4. One for each LSR pool; that is, 8.
5. If you open a VSAM path, you get two of these; for BDAM or VSAM base data sets you get one.

6. You will have these if you use the VSAM RLS SHCDS option NONRLSUPDATEPERMITTED. In this case, for each data set for which you have specified NONRLSUPDATEPERMITTED, you can have an upper limit. This limit is the number of different file names through which you access the data set multiplied by the number of tasks that update the data set. You will typically have only a few, if any, of these control blocks.
7. The value given is for a DWE chained off an LU6.1 session or an APPC session.

Buffer space sizings

Use the buffer space parameters to define buffers of a specific size in the VSAM definition for the global catalog data set.

You must ensure that the REGION parameter on your CICS jobs is high enough to cope with the increase in buffer storage used for the global catalog, because this storage comes out of region storage not EDSA.

By default, VSAM uses a buffer space equal to twice the control interval size of the data component and the control interval size of the index. In the sample job, this calculation gives a default of 69632 bytes. A larger minimum buffer size can improve cold start and warm restart times, and might significantly reduce CICS shutdown times.

You can use two parameters to control the buffer size. The **BUFFERSPACE** parameter defines the minimum buffer space size that is allowed. The **BUFSP** parameter defines the maximum buffer size. You can add either parameter to the sample job to set an appropriate buffer size.

For performance reasons, CICS defines a STRNO (number of strings) value of 32. Based on the sample job, the minimum value of BUFSP is calculated as follows:

```

BUFND = (STRNO + 1)
BUFNI = STRNO
BUFSP = 33 * 32768 (BUFND * CI size) + 32 * 1024 (BUFNI * CI size) =
        1114112 bytes

```

Another way to define buffer space for the global catalog data set is to use the **AMP** parameter on the DD statement in the CICS startup job stream, which you can use to override the default or defined value. If the **AMP BUFSP** specifies fewer bytes than the **BUFFERSPACE** parameter of the access method services DEFINE command, the **BUFFERSPACE** number overrides the **BUFSP** number.

Reusing the global catalog to perform a cold start

If you need to perform a cold start, **do not** delete and redefine the global catalog data set.

If you were to delete and redefine the global catalog, CICS would perform an *initial* start, and all recovery information for remote systems would be lost. When remote systems reconnected, CICS would inform them that it had lost any information that they needed to resynchronize their units of work, and messages would be produced to record the fact, on both the local and the remote systems.

Instead, to specify that the next start should be cold, use the DFHRMUTL utility with the SET_AUTO_START=AUTOCOLD option. This has the following advantages:

- You do not have to reset the START system initialization parameter from AUTO to COLD, and back again.

- Because sufficient information is preserved on the global catalog and the system log, CICS is able to recover information for remote systems from the log, and to reply to remote systems in a way that enables them to resynchronize their units of work.

You can speed up a cold start by using the DFHRMUTL COLD_COPY option to copy only those records that are needed for the cold start to another catalog data set. If the return code set by DFHRMUTL indicates that the copy was successful, a subsequent job-step can copy the new (largely empty) catalog back to the original catalog data set. The performance gain occurs because, at startup, CICS does not have to spend time deleting all the definitional records from the catalog. This technique will also speed up initial starts, for the same reason. Figure 12 is an example of this technique.

Note: Before you use COLD_COPY, you should be certain that you want to perform a cold or initial start. As a safeguard, make a backup copy of the original global catalog before you copy the new catalog output by DFHRMUTL over it. For more information about the use of the global catalog in a cold start of CICS, see “Controlling start and restart” on page 308.

```
//RMUTL EXEC PGM=DFHRMUTL,REGION=1M
//STEPLIB DD DSN=CICSTS41.CICS.SDFHLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=A
//DFHGCD DD DSN=CICSTS41.CICS.app1id.DFHGCD,DISP=OLD
//NEWGCD DD DSN=CICSTS41.CICS.app1id.COPY.DFHGCD,DISP=OLD
//SYSIN DD *
SET_AUTO_START=AUTOCOLD,COLD_COPY
/*
// IF (RMUTL.RC<16) THEN
//* Steps to be performed if RMUTL was a success
//COPY EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//DFHGCD DD DSN=CICSTS41.CICS.app1id.DFHGCD,DISP=OLD
//NEWGCD DD DSN=CICSTS41.CICS.app1id.COPY.DFHGCD,DISP=OLD
//SYSIN DD *
REPRO INFILE(NEWGCD) OUTFILE(DFHGCD) REUSE
/*
//* End of steps to be performed if RMUTL was a success
// ENDIF
```

Figure 12. DFHRMUTL example—setting the global catalog for a cold start. The COLD_COPY option is used to improve startup performance. Note that the NEWGCD and DFHGCD data sets must have been defined with the REUSE attribute.

Defining the local catalog

The local catalog is a VSAM key-sequenced data set (KSDS). The local catalog is not shared by any other CICS region, such as an alternate CICS in an XRF environment. If you are running CICS with XRF, you must define a unique local catalog for the active CICS region, and another for the alternate CICS region.

Before you begin

Unlike the global catalog, which must be defined with enough space to cope with any increase in installed resource definitions, the size of the local catalog is relatively static. The following section describes the information held on the local catalog.

CICS Transaction Server for z/OS is divided into functional areas (or components) known as domains. These domains communicate through a central component, the CICS kernel, and their initialization and termination is controlled by the domain manager. All the domains require an individual domain parameter record, and these are stored in the local catalog. The CICS domains use the local catalog to save some of their information between CICS runs and to preserve this information across a cold start. For further guidance information about what is written to the local catalog, and about how CICS uses the local catalog for startup and restart, see “Controlling start and restart” on page 308.

About this task

You can define and initialize the CICS local catalog in two ways. You can use the sample job as described below or you can use the CICS-supplied job, DFHDEFDS, to create a local catalog for an active CICS region.

Edit the sample job as follows:

Procedure

1. If you are defining local catalogs for multiple CICS regions, identify the clusters uniquely by making the specific APPLID of each CICS one of the data set qualifiers. For example, you might use the following names for the clusters of active and alternate CICS regions, where DBDCCIC1 and DBDCCIC2 are the specific APPLIDs:

```
DEFINE CLUSTER -  
    (NAME( CICSTS41.CICS.DBDCCIC1.DFHLCD)
```

```
DEFINE CLUSTER -  
    (NAME( CICSTS41.CICS.DBDCCIC2.DFHLCD)
```

2. Space for 200 records is probably adequate for the local catalog. However, specify space for secondary extents as a contingency allowance.
3. The local catalog records are small in comparison with the global catalog. Use the record sizes shown, which, with the number of records specified, ensure enough space for the data set.
4. Define the data definition statement for the local catalog as follows:

```
//DFHLCD DD DSN=CICSTS41.CICS.app1id.DFHLCD,DISP=OLD
```

Example

```
//LOCAT    JOB accounting info,,CLASS=A
//DEFLCD   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
//*
      DEFINE CLUSTER -
            (NAME( CICS41.CICS.applid.DFHLCD) - 1
             INDEXED                               -
             RECORDS( 200 10 )                     -           2
             FREESPACE(10 10)                      -
             SHAREOPTIONS( 2 )                     -
             REUSE                                  -
             VOLUMES( volid ))                    -
      DATA
            (NAME( CICS41.CICS.applid.DFHLCD.DATA ) -
             KEYS( 52 0 )                          -
             RECORDSIZE( 70 2041 )                 -           3
             CONTROLINTERVALSIZE( 2048 ))         -
      INDEX (NAME( CICS41.CICS.applid.DFHLCD.INDEX ) )
//*
//*****
//INITLCD EXEC PGM=DFHCCUTL
//*
//*          INITIALIZE THE CICS LOCAL CATALOG
//*
//STEPLIB  DD DSN=CICS41.CICS.SDFHLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DFHLCD   DD DSN=CICS41.CICS.applid.DFHLCD,DISP=SHR
//*
//
```

Figure 13. Sample job to define and initialize the local catalog

Initializing the local catalog

Before the local catalog can be used to start a CICS region, you must initialize it with data from the domain manager parameter records and three loader domain parameter records.

About this task

The domain manager parameter records contain information relating to the CICS domains and are identified by their domain names. See Table 8 on page 74 for a complete list.

The three loader domain parameter records contain information relating to these resources:

- DFHDMP, the CSD file manager
- DFHEITSP, the RDO language definition table
- DFHPUP, the CSD parameter utility program

Procedure

1. To enable you to initialize the local catalog correctly, with all the records in the correct sequence, run the CICS-supplied utility DFHCCUTL immediately after you have defined the VSAM data set. This utility writes the necessary data to the local catalog from the CICS domains.

2. The loader domain also writes a program definition record for each CICS nucleus module. The number of records varies depending on the level of function that you have included in your CICS region. Allow for at least 75 of these loader-domain records.
3. Use the CICS-supplied utility DFHSMUTL to add records to the local catalog to enable the CICS self-tuning mechanism for storage manager domain subpools.

Example

Table 8. List of domain names

| Domain name in catalog | Description |
|------------------------|--|
| DFHAP | Application domain |
| DFHBA | Business application manager |
| DFHCC | CICS local catalog domain |
| DFHDD | Directory manager domain |
| DFHDDH | Document handler domain |
| DFHDM | Domain manager domain |
| DFHDP | Debugging profiles domain |
| DFHDS | Dispatcher domain |
| DFHDU | Dump domain |
| DFHEJ | Enterprise Java domain |
| DFHEM | Event manager domain |
| DFHGC | CICS global catalog domain |
| DFHIE | ECI over TCP/IP domain |
| DFHII | IIOP domain |
| DFHKE | Kernel domain |
| DFHLD | Loader domain |
| DFHLG | Log manager domain |
| DFHLM | Lock manager domain |
| DFHME | Message domain |
| DFHML | XML services domain |
| DFHMN | Monitoring domain |
| DFHNQ | Enqueue manager domain |
| DFHOT | Object transaction domain |
| DFHPA | System initialization parameter domain |
| DFHPG | Program manager domain |
| DFHPI | Pipeline manager domain |
| DFHPT | Partner management domain |
| DFHRL | Resource life-cycle domain |
| DFHRM | Recovery manager domain |
| DFHRS | Region status domain |
| DFHRX | RRMS domain |
| DFHRZ | Request streams domain |

Table 8. List of domain names (continued)

| Domain name in catalog | Description |
|------------------------|----------------------------|
| DFHSH | Scheduler services domain |
| DFHSJ | JVM domain |
| DFHSM | Storage manager domain |
| DFHSO | Sockets domain |
| DFHST | Statistics domain |
| DFHTI | Timer domain |
| DFHTR | Trace domain |
| DFHTS | Temporary storage domain |
| DFHUS | User domain |
| DFHWB | Web domain |
| DFHXM | Transaction manager domain |
| DFHXS | Security domain |

Domains that write to the local catalog

Some domains write a domain status record to the local catalog, for use in a warm or emergency restart. For example, in the case of the dump domain, the status record indicates which transaction dump data set was in use during the previous run.

The domains that write to the local catalog are:

- Dispatcher domain
- Dump domain
- Loader domain
- Message domain
- Parameter manager domain
- Storage manager domain
- Transient data

Chapter 7. Setting up and using auxiliary trace data sets

This section describes the auxiliary trace data sets controlled by CICS.

About this task

Several types of tracing are available in CICS to help you with problem determination. Among the various types of trace, the CICS tracing handled by the CICS trace domain allows you to control the amount of tracing that is done, and also to choose from any of three destinations for the trace data. Any combination of these three destinations can be active at any time:

1. The internal trace table, in main storage above the 16MB line in the CICS address space.
2. The auxiliary trace data sets, defined as BSAM data sets on disk or tape.
3. The MVS generalized trace facility (GTF) data sets.

For information about GTF, see the *OS/390 MVS Diagnosis: Tools and Service Aids* manual. For information about using CICS tracing for problem determination, see the *CICS Problem Determination Guide*.

If you decide to use auxiliary trace, you must define one or two sequential data sets, on either disk or tape. If you specify automatic switching for your auxiliary trace data sets, define two data sets. If you specify autoswitch for auxiliary trace, and define only one data set, auxiliary trace is stopped and CICS takes a dump.

The DD names of the auxiliary trace data sets are defined by CICS as DFHAUXT and DFHBUXT. If you define a single data set only, its DD name must be DFHAUXT. You can allocate and catalog the auxiliary trace data sets before starting CICS.

If you use tape for recording auxiliary trace output, use unlabeled tape. Using standard-labeled tape, whether on a single tape drive or on two tape drives, stops you processing the contents of any of the volumes with the DFHTU660 utility until after the CICS step has been completed. If you use standard-labeled tape, make sure all the output produced in the CICS run fits on the one (or two) volumes mounted.

You cannot catalog data sets that are on unlabeled tapes.

Starting and controlling auxiliary trace

You might need to use auxiliary trace data sets to avoid the loss of diagnostic information, because internal trace table entries wrap around. When the end of the internal trace table is reached, subsequent entries overwrite those at the start of the table.

About this task

To get a trace of CICS activity in which trace entries are not overwritten, use the auxiliary trace data sets. This can be particularly useful if you are using CICS trace during startup, because of the high volume of trace entries written when CICS is initializing.

You can start CICS tracing at initialization by coding system initialization parameters; you can also specify which of the three destinations you want CICS to use. The following is a summary of the trace system initialization parameters that you can code:

| Keyword | Description |
|----------------|--|
| AUXTR | Switches auxiliary trace on or off at CICS startup. |
| AUXTRSW | Specifies automatic switching for auxiliary trace data sets when full. |
| INTTR | Switches internal trace on or off at CICS startup. |
| GTFTR | Specifies whether CICS is to use GTF as a destination for CICS trace data. |
| SPCTR | Specifies the level of special tracing. |
| SPCTRxx | Specifies the level of special tracing for the “xx” component. |
| STNTR | Specifies the level of CICS standard tracing. |
| STNTRxx | Specifies the level of standard tracing for the “xx” component. |
| SYSTR | Switches the system master trace flag on or off at CICS startup. |
| TRTABSZ | Defines the size of the CICS internal trace table. |
| USERTR | Switches the user trace flag on or off at CICS startup. |

For more information about these system initialization parameters, and how to code them, see Chapter 16, “Specifying CICS system initialization parameters,” on page 133.

You can also control CICS tracing by means of the CICS-supplied transactions CETR and CEMT. (Note that you cannot use CETR through an MVS console.) For information about the CICS control options available with these transactions, see *CICS Supplied Transactions*.

Allocating auxiliary trace data sets

If you are defining auxiliary trace data sets on disk, you can use the following sample job to allocate and catalog them before running CICS.

About this task

Alternatively, you can run the CICS-supplied job DFHDEFDS to create the auxiliary trace data sets for an active CICS region or the CICS-supplied job DFHALTDS to create them for an alternate CICS region. For information about the jobs Creating data sets unique to each CICS region, DFHDEFDS jobDFHDEFDS and Defining CICS regions as applications to VTAMDFHALTDS, see the *CICS Transaction Server for z/OS Installation Guide*.

Procedure

1. The DCB subparameters shown in this sample job specify the required DCB attributes for the CICS auxiliary trace data sets. As an alternative to this job,

you can specify (NEW,CATLG) on the DD statements in the CICS startup job stream, omit the DCB parameter, and let CICS open the data sets with the same default values.

2. Change the space allocations in this sample job stream to suit your installation's needs.

Example

```
//DEFTRCDS JOB (accounting information),
//          MSGCLASS=A,MSGLEVEL=(1,1),
//          CLASS=A,NOTIFY=userid
//*****
//*          Create auxiliary trace data sets
//*****
//ALLOCD   EXEC PGM=IEFB14
//DFHAUX   DD DSN=CICSTS41.CICS.applid.DFHAUX,UNIT=3380,VOL=SER=valid,
//          DISP=(NEW,CATLG),DCB=(BLKSIZE=4096,RECFM=F,LRECL=4096),  1
//          SPACE=(CYL,(5,1))                                         2
//DFHBUX   DD DSN=CICSTS41.CICS.applid.DFHBUX,UNIT=3380,VOL=SER=valid,
//          DISP=(NEW,CATLG),DCB=(BLKSIZE=4096,RECFM=F,LRECL=4096),  1
//          SPACE=(CYL,(5,1))                                         2
//
```

Figure 14. Sample job to define auxiliary trace data sets on disk

Space calculations

Trace entries are of variable length, but the physical record length (block size) of the data written to the auxiliary trace data sets is fixed at 4096 bytes.

As a rough guide, each block contains an average of 40 entries, although the actual number of entries depends on the processing being performed.

Job control statements for CICS execution

If you allocate and catalog the auxiliary trace data sets on disk, you can define them to CICS in the startup job stream.

Use the following DD statements:

```
//DFHAUX   DD DSN=CICSTS41.CICS.applid.DFHAUX,DCB=BUFNO=n,DISP=SHR
//DFHBUX   DD DSN=CICSTS41.CICS.applid.DFHBUX,DCB=BUFNO=n,DISP=SHR
```

If you specify BUFNO greater than 1, you can reduce the I/O overhead involved in writing auxiliary trace records. A value between 4 and 10 can greatly reduce the I/O overhead when running with auxiliary trace on.

DISP=SHR allows the simultaneous processing of a data set by the DFHTU660 offline utility program after a switch to the other data set has taken place.

For auxiliary trace data sets on unlabeled tapes, use the following sample DD statements:

```
//DFHAUX   DD DSN=CICSTS41.CICS.applid.DFHAUX,UNIT=3400,VOL=SER=valid,
//          DISP=(NEW,KEEP),LABEL=(,NL)
//DFHBUX   DD DSN=CICSTS41.CICS.applid.DFHBUX,UNIT=3400,VOL=SER=valid,
//          DISP=(NEW,KEEP),LABEL=(,NL)
```

If you are using tape for the auxiliary data sets, assign tape units and mount the tapes before entering the command to start auxiliary trace. If you specify AUXTR=ON as a system initialization parameter, ensure the tape is mounted before starting CICS.

XRF considerations

The active and the alternate CICS regions must refer to different auxiliary trace data sets; that is, they must be unique data sets. This means that you can capture auxiliary trace data for the active CICS region, while the alternate CICS region is running but before takeover occurs.

About this task

For the active CICS region, you can use CETR or CEMT to control auxiliary trace data sets. For the alternate CICS region, you can use CEBT. For information about using these transactions, see *CICS Supplied Transactions*.

Using the trace utility program (DFHTU660)

If you write trace entries to CICS auxiliary trace data sets you can use the trace utility program, DFHTU660, to extract all or selected trace entries, and format and print the data.

About this task

To process the separate trace data sets for active and alternate CICS regions, you need separate utility jobs for each set of data sets. For information about DFHTU660, see the *CICS Operations and Utilities Guide*.

Chapter 8. Defining dump data sets

CICS uses the dump data sets for recording dumps as a consequence of a failure that is detected when CICS is running, or upon explicit request.

About this task

You must define two types of dump data sets:

1. MVS system dump data sets, for recording system dumps that CICS requests using the MVS SDUMP macro.
2. CICS transaction dump data sets, for recording transaction dumps.

Procedure

1. For the initial installation of CICS, define a dump data set of between 5 and 10 MB. When normal operations begin, you can adjust this to suit your own installation's requirements.
2. Change the SDUMP options using the **MVS CHNGDUMP** command. You must thoroughly review your use of the **CHNGDUMP** command when setting up your CICS region.
 - a. Use the MERGE function to ensure that the areas selected by CICS to dump are included in the MVS dump data set output.
 - b. Use the ADD option to replace any options specified by CICS when issuing the SDUMP. This can result in partial dumps being taken to the MVS dump data set.
 - c. If you are running CICS with XRF, use the **MVS CHNGDUMP** command to set QUIESCE=NO. Setting this value prevents SDUMPs of other address spaces from causing unnecessary takeovers when the alternate CICS is running on a different MVS image. The surveillance signal of the active CICS region stops during an MVS SDUMP of the active CICS region's address space, which could lead to unnecessary takeovers being initiated if the alternate delay interval (ADI) is set too low.
3. Define the CICS transaction dump data sets, as described in "Defining the transaction dump data sets" on page 84.

Results

When you start CICS for the first time, CICS uses system default values for the dump table options, and continues to use the system default values until you modify them.

CICS has a dump table facility that enables you to control dumps. The dump table lets you:

- Specify the type of dump, or dumps, you want CICS to record.
- Suppress dumping entirely.
- Specify the maximum number of dumps to be taken during a CICS run.
- Control whether CICS is to terminate as a result of a failure that results in a dump.

Related information

The dump code options you can specify

System dumps

CICS produces a system dump using the MVS SDUMP macro.

The MVS SDUMP dump results from CICS issuing an MVS SDUMP macro. It includes almost the entire CICS address space, that is, the MVS nucleus and other common areas, as well as the CICS private storage areas. The SDUMP dump is written to an MVS dump data set, which you can process using the interactive problem control system (IPCS), either online under TSO, or by submitting a batch job to print it. IPCS is described in the *z/OS MVS IPCS User's Guide*. For information about the IPCS **VERBEXIT** parameters that you use with the CICS IPCS dump exit, see the *CICS Operations and Utilities Guide*. For information about the SDUMP macro, and the associated MVS dump data sets, see *z/OS MVS Diagnosis: Procedures*.

The SDUMP macros issued by CICS normally contain the QUIESCE=NO parameter. They might not if the SDUMP is taken because of an abend in CICS SVC code or when altering MRO control blocks. This parameter allows the MVS system to remain dispatchable while the SDUMP is being taken, thus reducing the impact on the system. However if QUIESCE=YES is specified as an MVS system default it will override that specified by CICS. These defaults can be altered by using the **MVS CHNGDUMP** command.

Suppressing system dumps that precede ASRx abends

The MVS system dump data sets can become full with unwanted SDUMPs that precede ASRA, ASRB, and ASRD abends (after either message DFHAP0001 or DFHSR0001).

If CICS storage protection is active, you can suppress the system dumps caused by errors in application programs (after message DFHSR0001), while retaining the dumps caused by errors in CICS code (after message DFHAP0001). To do this, use either a **CEMT SET SYDUMPCODE** command, or an **EXEC CICS SET SYSDUMPCODE** command to suppress system dumps for system dump code SR0001.

```
CEMT SET SYDUMPCODE(SR0001) ADD NOSYSDUMP
```

CICS uses dump code SR0001 if an application program was executing in user-key at the time of the program check or MVS abend. This is only possible if storage protection is active. If the program was executing in CICS-key, dump code AP0001 is used instead.

Where storage protection is not active, SDUMPs can be suppressed by suppressing dump code AP0001. However, note that this suppresses dumps for errors in both application *and* CICS programs. The XDUREQ global user exit can be used to distinguish between AP0001 situations in application and CICS programs.

For more information about the storage protection facilities available in CICS, see "Storage protection" on page 334.

If you want SDUMPs for one of these transaction abends but not the other, select the one you want by using either a **CEMT TRDUMPCODE** or an **EXEC CICS**

TRANDUMPCODE command. This specifies, on an entry in the dump table, that SDUMPs are to be taken for either ASRA, ASRB, or ASRD abends. For example, specifying:

```
CEMT SET TRDUMPCODE(ASRB) ADD SYSDUMP
```

adds an entry to the dump table and ensures that SDUMPs are taken for ASRB abends. However, in this case the SDUMPs are taken at a later point than SDUMPs usually taken for system dump code AP0001 and SR0001.

For information about the DFHAP0001 and DFHSR0001 messages, see the *CICS Messages and Codes* and the *CICS Problem Determination Guide*.

The CICS transaction dump data sets

You can optionally specify DCB parameters for dump data sets if you want to copy the data sets to tape or disk. When CICS opens the dump data set, it issues an MVS DEVTYPE macro. This macro returns the track size for direct access devices, or 32760 for magnetic tape.

The maximum block size used for a transaction dump is the lesser of the values returned from the DEVTYPE macro and 4096. As this usually results in a block size of 4096 (because devices generally have a track size greater than this), CICS writes multiple blocks per track. After writing each block, MVS returns the amount of space remaining on the current track. If the space remaining is 256 bytes or more, then the size of the next block written is the lesser of the values returned by MVS and 4096.

If the space remaining is less than 256 bytes, the next block is written to the next track.

There are four global user exits that you can use with the transaction dump data sets:

1. XDUCLE, after the dump domain has closed a transaction dump data set
2. XDUREQ, before the dump domain takes a transaction dump
3. XDUREQC, after the dump domain takes a transaction dump
4. XDUOUT, before the dump domain writes a record to the transaction dump data set

For programming information about the global user exits, see the *CICS Customization Guide*

Printing the transaction dump data sets

With two data sets, you can print transaction dumps from one data set while CICS is running.

About this task

You can either print transaction dumps explicitly or use automatic switching.

Procedure

- To print transaction dumps explicitly:
 1. Use the command **CEMT SET DUMP SWITCH** to switch the data sets. CICS closes the current data set after any transaction dump being recorded has been completed, and opens the other data set.

2. Print the completed data set using the DFHDU660 dump utility program. For information about the DFHDU660 dump utility program, see the *CICS Operations and Utilities Guide*.
- To print transaction dumps using automatic switching:
 1. Use the command **CEMT SET DUMP AUTO** to cause automatic switching when the current data set becomes full. This command permits **one** switch only. When a transaction dump data set is full, CICS closes the data set and issues console messages as follows:


```
DFHDU0303I applid Transaction Dump Data set dataset closed.
DFHDU0304I applid Transaction Dump Data set dataset opened.
DFHDU0305I applid Transaction Dump Data set switched to ddname.
```
 2. If you specified DISP=SHR for the dump data set, you can print the completed data set with the DFHDU660 utility program and then reissue the **CEMT SET DUMP AUTO** command. This again switches data sets automatically (once only) when the current data set is full.

Defining the transaction dump data sets

You must define the data sets in the CICS startup job stream with the DD names DFHDMPA and DFHDMPB. If you define a single data set only, its DD name must be DFHDMPA.

About this task

You can either define DFHDMPA and DFHDMPB as temporary data sets for each CICS run, or allocate and catalog the data sets in advance to reuse them repeatedly.

Procedure

1. Code the **DUMPDS** system initialization parameter to specify which transaction dump data set is to be opened during CICS initialization. If you specify **DUMPDS=AUTO**, CICS opens, on a warm or emergency start, the data set that was not in use when CICS was last terminated. This lets you restart CICS after an abnormal termination without waiting to print the dump data set that was in use at termination.
2. Allocate the dump data sets. You can do this in one of two ways:
 - Use the sample data definition statements to allocate and catalog dump data sets on disk.
 - Use the CICS-supplied job DFHDEFDS to allocate and catalog the dump data sets.

If you select to use the sample job, edit the statements as follows:

- a. Change the space allocations in this sample job stream to suit your own installation's needs.
- b. If you are running CICS with XRF, allocate different data sets for the alternate.
- c. If you use tape for recording dump output, use unlabeled tape. Standard-labeled tape, whether on a single tape drive or on two tape drives, stops you processing the contents of any of the volumes with the DFHDU660 utility until after the CICS step has been completed. If you want to use standard-labeled tape, make sure that all the output produced in the CICS run fits on the one or two volumes mounted.

You cannot catalog dump data sets defined on unlabeled tapes. Your data set definitions must be in the CICS startup job stream each time CICS is run.

```
//DFHDMPA DD DSN=CICSTS41.CICS.applid.DFHDMPA,DISP=(NEW,CATLG),
//          UNIT=3380,VOL=SER=valid,SPACE=(CYL,(5,1))
//DFHDMPB DD DSN=CICSTS41.CICS.applid.DFHDMPB,DISP=(NEW,CATLG),
//          UNIT=3380,VOL=SER=valid,SPACE=(CYL,(5,1))
```

Figure 15. Sample job control statements for defining disk dump data sets

3. Optional: To copy dump data sets to tape or disk, specify DCB parameters on the DD statements when allocating and cataloging the dump data sets, as follows:

```
//          DCB=(RECFM=VB,BLKSIZE=4096,LRECL=4092)
```

4. Include the following DD statements in the CICS startup job stream.

- If you have cataloged the transaction dump data sets, add the following DD statement:

```
//DFHDMPA DD DSN=CICSTS41.CICS.applid.DFHDMPA,DISP=SHR
//DFHDMPB DD DSN=CICSTS41.CICS.applid.DFHDMPB,DISP=SHR
```

DISP=SHR enables each data set, if held on disk, to be processed by the DFHDU660 offline utility after the switch to the other data set has taken place.

- If you have put the transaction dump data sets on unlabeled tapes, add the following DD statement:

```
//DFHDMPA DD DSN=CICSTS41.CICS.applid.DFHDMPA,UNIT=3400,VOL=SER=valid1,
//          DISP=(NEW,KEEP),LABEL=(,NL)
//DFHDMPB DD DSN=CICSTS41.CICS.applid.DFHDMPB,UNIT=3400,VOL=SER=valid2,
//          DISP=(NEW,KEEP),LABEL=(,NL)
```

Results

CICS always attempts to open at least one transaction dump data set during initialization. If you do not include a DD statement for at least one transaction dump data set in your CICS job, initialization continues after the DFHDU0306 message is sent to the console.

When CICS opens the dump data set, it issues an MVS DEVTYPE macro. This returns the track size for direct access devices, or 32760 for magnetic tape. The maximum block size used for a transaction dump is the lesser of the values returned from the DEVTYPE macro and 4096. As this usually results in a block size of 4096 (because devices generally have a track size greater than this), CICS writes multiple blocks per track. After writing each block, MVS returns the amount of space remaining on the current track. If the space remaining is 256 bytes or more, then the size of the next block written is the lesser of the values returned by MVS and 4096.

If the space remaining is less than 256 bytes, the next block is written to the next track.

There are four global user exits that you can use with the transaction dump data sets:

1. XDUCLOSE, after the dump domain has closed a transaction dump data set

2. XDUREQ, before the dump domain takes a transaction dump
3. XDUREQC, after the dump domain takes a transaction dump
4. XDUOUT, before the dump domain writes a record to the transaction dump data set

For programming information about the global user exits, see the *CICS Customization Guide*

Chapter 9. Defining the CICS availability manager data sets

The CICS availability manager (CAVM) data sets support the CAVM mechanism, that enables active and alternate CICS regions to coordinate their processing when XRF=YES is coded as a system initialization parameter. If you code XRF=NO, these data sets are not used.

About this task

You must define a separate pair of data sets for each generic APPLID in use. Both the active and alternate CICS regions must refer to the same pair of data sets. You define these data sets, but must not initialize them. If possible, place the data sets on separate volumes. You can define the data sets in one of two ways:

- Use the CICS-supplied job DFHDEFDS to define the XRF data sets as some of the data sets for a CICS region.
- Use the sample jobs, as described below, to define the data sets.

Procedure

1. Select the correct volume for the XRF data sets. Access to the XRF control data set must be serialized during the critical sections of CAVM signon, sign-off, and takeover processing. The correct choice of volume is important because this serialization is provided by RESERVE/RELEASE (DEQ) logic. For example, do not place an XRF control data set on the same volume as the JES checkpoint data set. If you use global resource serialization (GRS) you must not convert this RESERVE, which uses the qname SYSCICSX, to a global ENQ.
2. Use the following sample job to define the XRF control data set. Edit the job as follows:

```
//CICSCTL JOB 'accounting info',name,MSGCLASS=A
//XRCTL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
        DEFINE CLUSTER -
            (NAME(CICSTS41.CICS.applid.DFHXRCTL) -
             RECORDSIZE(4089 4089) -
             CONTROLINTERVALSIZE(4096) -
             RECORDS(4) -
             NIXD -
             SHAREOPTIONS(3,3) -
             VOLUMES(vol1d1)) -
            DATA
            (NAME(CICSTS41.CICS.applid.DFHXRCTL.DATA))
/*
//
```

Figure 16. Sample job to define the XRF control data set

- a. Specify a suitable value for RECORDSIZE. It must be at least 4089.
- b. The control interval sizes of the XRF control data set and the XRF message data to the same value and at least 4096 bytes. Only four control intervals are required.
- c. Specify the SHAREOPTIONS value as 3,3.
- d. The data set must be VSAM-ESDS.

3. Use the following sample job to define the XRF message data set. Define the XRF message data set on a volume that is not subject to RESERVE activity and do not locate it where a single failure can make both it and the XRF control data set inaccessible. This reduces the risk of the surveillance signal being stopped accidentally while CICS is still running normally. Edit the job as follows:

```

//CICSMMSG JOB 'accounting info',name,MSGCLASS=A
//XRMSG EXEC PGM=IDCAMS
//DDNAME2 DD DISP=OLD,UNIT=3380,VOL=SER=vol1id2
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
        DEFINE CLUSTER -
            (NAME(CICSTS41.CICS.app1id.DFHXRMSG) -
             RECORDSIZE(4089 4089) -
             CONTROLINTERVALSIZE(4096) -
             RECORDS(1500) -
             NIXD -
             SHAREOPTIONS(3,3) -
             VOL(vol1id2) -
             FILE(DDNAME2)) -
            DATA
            (NAME(CICSTS41.CICS.app1id.DFHXRMSG.DATA))
/*
//

```

Figure 17. Sample job to define the XRF message data set

- a. Specify a suitable value for RECORDSIZE. It must be at least 4089.
 - b. Define the control interval sizes of the XRF control data set and the XRF message data to the same value and at least 4096 bytes. Only four control intervals are required. If the CI size of the XRF message data set is greater than 4096, the CI buffers occupy more real storage and virtual storage above the 16MB line, although fewer I/O operations occur during the catch-up phase.
 - c. Specify the SHAREOPTIONS value as 3,3.
 - d. The data set must not be indexed.
4. Define the data definition statements for the XRF control data sets on the CICS startup job.
 - a. Define the DFHXRCTL DD statement for the XRF control data set as follows:

```
//DFHXRCTL DD DSN=CICSTS41.CICS.app1id.DFHXRCTL,DISP=SHR
```
 - b. Define the DFHXRMSG DD statement for the XRF message data set as follows:

```
//DFHXRMSG DD DSN=CICSTS41.CICS.app1id.DFHXRMSG,DISP=SHR
```
 5. Protect your XRF data sets using RACF to ensure that the integrity and security of your CICS regions and terminal network are not compromised. When you have done so, give each CICS region CONTROL access to its own pair of data sets. If you are running your XRF systems with an overseer program, make sure that it has READ access to all the CAVM data sets. All other users must be denied access to the data sets.

What to do next

You do not need to take backup copies of these data sets because when neither of the active or alternate CICS regions is running, you can always start with a fresh pair of data sets.

The first time they are used, CICS recognizes them as a new pair of data sets. If the data sets are new, CICS initializes them in such a way that, from then on, they can be used only as a pair with the original generic APPLID and for their original purpose (that is, as either an XRF message data set or an XRF control data set). If you need to redefine either data set, for any reason, you must redefine both of them.

The XRF data sets

The CAVM requires two data sets: the XRF control data set and the XRF message data set.

This pair of data sets is logically a single entity that contains:

- State data whose main purpose is to ensure that, at any given time, only one job is allowed to fulfill the active role for a particular generic APPLID.
- Primary and secondary surveillance signals of active and alternate CICS regions, so that each CICS region can tell whether its partner is working correctly.
- Messages about the state of particular resources in use on the active CICS region, that are written by the active CICS region, and read and processed by the alternate CICS region.

Why have two data sets?

Because of RESERVE commands issued by other MVS images in a multi-MVS environment, a shared DASD volume might become inaccessible for periods ranging from milliseconds to perhaps a minute. By making use of two data sets, placed on different volumes, CAVM can greatly reduce the risk that, by preventing surveillance signals from being written, normal RESERVE activity might cause the unnecessary takeover of a CICS region that was running normally.

If the access paths to the two volumes are separate, CAVM is also less vulnerable to hardware failures.

The XRF control data set

The XRF control data set records the presence or absence, identities, and current states of the active and alternate CICS regions' jobs. It is also used for the primary surveillance signals of the active and the alternate CICS regions.

CAVM rejects a request from a CICS job to sign on as the active CICS region if the XRF control data set shows that an active CICS region is already present, or that a takeover is in progress. This ensures that the integrity of files and databases cannot be lost as a result of uncontrolled concurrent updating by two or more active CICS regions. As soon as an active or alternate CICS regions signs on, it starts to write its own surveillance signals, and to look for its partner's surveillance signals.

The XRF message data set

The XRF message data is used to pass messages about the current states of specific resources from the active to the alternate CICS region. It is also used for the secondary surveillance signals of the active and alternate CICS regions, when the control data set is unavailable for this purpose, either because the last write has not completed yet or because of I/O errors.

Space calculations

The size required for the XRF message data set depends on the length and number of messages that have been sent by the active CICS region but not yet received by the alternate CICS region.

The XRF message data set is written and read cyclically. When the alternate CICS region has read a message, that space becomes available for another message on the next cycle. It is important to make the data set large enough to store the backlog of messages that accumulates if the alternate CICS region is held up for any reason. If the data set is too small, you run the risk of the alternate CICS region being unable to read the data set correctly, and thereby becoming incapable of taking over. However, the active CICS region does not write messages to the data set until it has been notified that an alternate CICS region is present (signed on to CAVM), and able to receive them.

The peak in message traffic usually occurs during the catch-up phase shortly after the active CICS region detects the presence of the alternate CICS region. You might be able to estimate the amount of space you need from the number of terminal resources you have. The active CICS region sends messages about these resources:

- Installed:
 - VTAM terminals
 - ISC connections
 - MRO connections
 - Consoles
- Autoinstalled terminals
- Bound VTAM terminals that are XRF-eligible

Table 9 lists the sizes of the various messages sent to the data set.

Table 9. Sizes of messages sent to the XRF message data set

| <i>Type of TCT entry</i> | <i>Bytes per install</i> |
|---|--|
| The CICS-generated TCT entries (2 only) | 629 |
| VTAM terminals | 710 |
| Non-3270 devices with pipeline logical units and TASKNO= operand (or TASKLIMIT if RDO) is specified | 581 x TASKNO value |
| MVS consoles | 389 |
| LUTYPE6.2 connection | 2083 |
| LUTYPE6.2 mode | 169 + (837 x maximum number of sessions) |
| LUTYPE6.1 connection | 226 + (732 x number of sessions) |
| IRC | 237 + (520 x number of sessions) |
| IRCBCH | 240 + (565 x number of sessions) |

For VTAM terminals only, you should also make allowance for the following:

Table 10. Additional space requirements (VTAM terminals only)

| <i>Bytes per logon</i> | <i>Bytes per logoff</i> | <i>Bytes per signon</i> | <i>Bytes per sign-off</i> |
|------------------------|-------------------------|-------------------------|---------------------------|
| 70 | 35 | 45 | 29 |

The alternate CICS region issues some messages that can help you with your sizing. The following messages issued by the alternate CICS region can give you an idea of the rate of message transfer:

```
DFHTC1041I applid TERMINAL CONTROL TRACKING STARTED
DFHTC1040I applid TERMINAL CONTROL TRACKING RECORDS RECEIVED
DFHTC1043I applid TERMINAL CONTROL TRACKING ENDED - nnn RECORDS RECEIVED
```

The following messages may indicate that the XRF message data set is not large enough:

```
DFHXG6447I NON CRUCIAL XRF MESSAGE(S) DISCARDED
DFHXA6541I XRF HAS FAILED. THE XRF MESSAGE READER IN THE ALTERNATE
SYSTEM HAS FALLEN TOO FAR BEHIND
```

Crucial and non-crucial messages

The active CICS region classifies its messages as crucial or non-crucial.

An example of a crucial message is an autoinstall message that the alternate CICS region must receive if it is to remain eligible to take over. An example of a non-crucial message is a logon message. The alternate CICS region can tolerate the loss of such a message, and the loss only results in some degradation at takeover; no standby session is established for that terminal and it must be logged on again. Install messages that form part of the initial description are also treated as non-crucial, because the active CICS region can try to send them again later, and the alternate CICS region can construct its tables from the CICS catalog if it does not receive a complete initial description.

The active discards non-crucial messages if it decides that sending them may overwrite messages that the alternate CICS region has not yet read, thereby making it ineligible to take over. It issues message DFHXG6447I for the first such discard. The active CICS region always sends crucial messages. If this causes an unread message to be overwritten, the alternate CICS region detects it and terminates after issuing message DFHXA6541I.

Effect of a full XRF message data set on the active CICS region

The active CICS region is not affected by the state of the XRF message data set.

It continues running even when the data set is full; only the alternate CICS region fails. Further, the XRF message data set is only “full” to the alternate CICS region that fails; you can start a new alternate CICS region, using the same XRF message data set, and the active CICS region resends all the messages for the new alternate CICS region to begin tracking. If the first failure was caused by some unusual condition, you may not need to increase the size of the XRF message data set.

However, if messages DFHXG6447I or DFHXA6541I occur too often, you must stop the active CICS region so that you can change to a larger data set.

I/O error handling

While the active CICS region can write its surveillance signals successfully to either the XRF control data set or the XRF message data set, it keeps running in spite of I/O errors.

About this task

However, if the active CICS region has an I/O error while writing a message to the XRF message data set, the alternate CICS region cannot function correctly, so the active CICS region disables it to prevent it from taking over. If the active CICS region is unable to write to either the XRF control data set or the XRF message data set, it can neither disable the alternate CICS region nor keep it properly synchronized, and so the active CICS region fails.

While an alternate CICS region can receive the active CICS region's surveillance signals and tracking messages successfully, in addition to writing its own surveillance signals to either the XRF control data set or the XRF message data set, it keeps running in spite of some types of I/O error. However, an isolated I/O error that would have no effect during tracking, may cause failure of the alternate CICS region if it occurs during takeover.

Note: When the active and alternate CICS regions are running in different MVS images, they are not necessarily affected in the same way by the failure of a control unit or channel path that provides access to an CAVM XRF data set.

Chapter 10. Defining user files

This chapter tells you how to define user files and how to access VSAM data sets, BDAM data sets, data tables, and coupling facility data tables.

About this task

CICS application programs process files, which, to CICS, are logical views of a physical data set or data table. For data tables, the file provides a view of the data table, which resides either in data space storage or in a coupling facility structure. Except in the case of coupling facility data tables, for which an underlying physical data set is optional, a data table is also associated with a source data set from which the table is loaded. For non-data-table files, the file provides a view of the data set.

A file is identified to CICS by a **file name** of up to eight characters, and there can be many files defined to CICS that refer to the same physical data set or data table. This has the following effect, depending on the type of object the file is defining:

- For non data table files, if more than one file refers to the same data set, each file refers to the same physical data.
- For user-maintained data tables, if more than one file refers to the same data set, each file represents a view of a unique data table.
- For CICS-maintained data tables, if more than one file refers to the same data set, only one can be defined as a CMT. The other files access data from the CMT created by the CMT file definition.
- For coupling facility data tables, if more than one file refers to the same data set, each file represents a view of a unique coupling facility data table in a CFDT pool (unless each file specifies the same tablename and poolname, in which case each they provide a separate view of the same table.

A data set, identified by a data set name (DSNAME) of up to 44 characters, is a collection of data held on disk. CICS file control processes only VSAM or BDAM data. Any data sets referred to by CICS files must be created and cataloged, so that they are known to MVS before any CICS job refers to them. Also, the data sets are usually initialized by being preloaded with at least some data before being used by CICS transactions.

You can use CICS-maintained or user-maintained data tables to improve the performance and function of CICS regions using files that refer to VSAM data sets. Data tables offer a method of constructing, maintaining, and gaining rapid access to data records contained in tables held in data space storage, above 16MB. Each data table is associated with a VSAM KSDS, known as its **source data set**. For more information about data tables, see “Defining data sets with multiple extents and volumes” on page 6.

You can use coupling facility data tables to share data across a sysplex, using the CICS file control API, subject to some restrictions, such as a 16 byte key length.

You can use RLS access mode to share VSAM data sets between CICS application-owning regions throughout a sysplex. See “VSAM record-level sharing (RLS)” on page 96 for further information.

Each of the above methods is discussed under the following topics:

- “VSAM data sets”
- “BDAM data sets” on page 99
- “Defining data sets to CICS” on page 100
- “Opening VSAM or BDAM files” on page 102
- “Closing VSAM or BDAM files” on page 103
- “XRF considerations” on page 104
- “CICS data tables” on page 104
- “Coupling facility data tables” on page 106.

VSAM data sets

You create a VSAM data set by running the Access Methods Services (AMS) utility program IDCAMS in a batch job, or by using the TSO DEFINE command in a TSO session.

About this task

The DEFINE command specifies to VSAM and MVS the VSAM attributes and characteristics of your data set. You can also use it to identify the catalog in which your data set is to be defined.

If required, you can load the data set with data, again using IDCAMS. You use the AMS REPRO command to copy data from an existing data set into the newly created one.

You can also load an empty VSAM data set from a CICS transaction. You do this by defining the data set to CICS (by allocating the data set to a CICS file), and then writing data to the data set, regardless of its empty state. See “Loading empty VSAM data sets” on page 95.

When you create a data set, you can define a data set name of up to 44 characters. If you choose not to define a name, VSAM assigns the name for you. This name, known as the data set name (or DSNNAME), uniquely identifies the data set to your MVS system.

You can define VSAM data sets accessed by user files under CICS file control as eligible to be backed up while CICS is currently updating these data sets. For more information about backing up VSAM files open for update, see “Defining backup while open (BWO) for VSAM files” on page 10.

VSAM bases and paths

You store data in data sets, and retrieve data from data sets, using application programs that reference the data at the record level.

Depending on the type of data set, you can identify a record for retrieval by its key (a unique value in a predefined field in the record), by its relative byte address, or by its relative record number.

Access to records through these methods of primary identification is known as access through the base.

Sometimes you may need to identify and access your records by a secondary or alternate key. With VSAM, you can build one or more alternate indexes over a single base data set, so that you do not need to keep multiple copies of the same information organized in different ways for different applications. Using this method, you create an **alternate index path** (or paths), that links the alternate index (or indexes) with the base. You can then use the alternate key to access the records by specifying the path as the data set to be accessed, that is by allocating the path data set to a CICS file.

When you create a path you give it a name of up to 44 characters, in the same way as a base data set. A CICS application program does not need to know whether it is accessing your data through a path or a base; except that it may be necessary to allow for duplicate keys if the alternate index was specified to have non-unique keys.

Loading empty VSAM data sets

There are two ways you can load data into an empty VSAM data set.

An empty data set can be loaded using either of the following methods:

- Running the AMS utility program, IDCAMS
- Writing records to the data set using CICS transactions

Although VSAM imposes some restrictions during initial data set load processing, when the data-set is said to be in **load mode**, these do not affect CICS transactions. For files opened in non-RLS mode, CICS file control “hides” load mode processing from your application programs. For files opened in RLS mode against an empty data set, load mode processing is hidden from CICS by VSAM, and all VSAM requests are allowed.

Using IDCAMS

If you have a large amount of data to load into a new data set, run the AMS utility program IDCAMS as a batch job, using the REPRO command to copy data from an existing data set to the empty data set.

When you have loaded the data set with IDCAMS, it can be used by CICS in the normal way.

A data set in VSAM load mode cannot have alternate indexes in the upgrade set. If you want to create and load a data set with alternate indexes, you must use AMS, or some other suitable batch program, to load the data set and invoke BLDINDEX to create the alternate indexes.

Using CICS applications

If the amount of data to be loaded is small, and there is no upgrade set, you may load an empty data set by using standard CICS file WRITE requests.

When the first write, or series of writes (mass insert), to the file is completed, CICS closes the file and leaves it closed and enabled, so that it will be reopened for normal processing when next referenced. If you attempt to read from a file in load mode, CICS returns a NOTFOUND condition.

Reuse of data sets

If you define a data set with the AMS REUSE attribute, it can also be emptied of data during a CICS run.

This allows it to be used as a work file. When the status of a file referencing the data set is CLOSED and DISABLED (or UNENABLED), you can use the SET EMPTY command, either from an application program using the EXEC CICS command-level interface, or from a master terminal using the master terminal CEMT command. This command sets an indicator in the installed file definition so that when the file is next opened, the VSAM high-used relative byte address (RBA) is set to zero, and the contents of the data set are effectively cleared.

If you define a data set to VSAM with the average and maximum record lengths equal, and define a file to CICS with fixed length records to reference that data set, the size of the records written to the data set **must** be of the defined size. For example, if a record in a data set has been read for update, you get errors when rewriting the record if, for example, you:

- Defined the record sizes to VSAM as 250 bytes, with the parameter RECORDSIZE(250 250)
- Defined the file to CICS with the parameter RECFORM=FIXED
- Loaded the data set with records that are only 200 bytes long

VSAM record-level sharing (RLS)

Record-level sharing (RLS) is an access mode for VSAM data sets supported by DFSMS 1.3 and later releases. RLS enables VSAM data to be shared, with full update capability, between many applications running in many CICS regions.

With RLS, CICS regions that share VSAM data sets can reside in one or more MVS images within an MVS parallel sysplex. This concept, in a parallel sysplex with VSAM RLS supporting a CICSplex, is illustrated in Figure 18 on page 97.

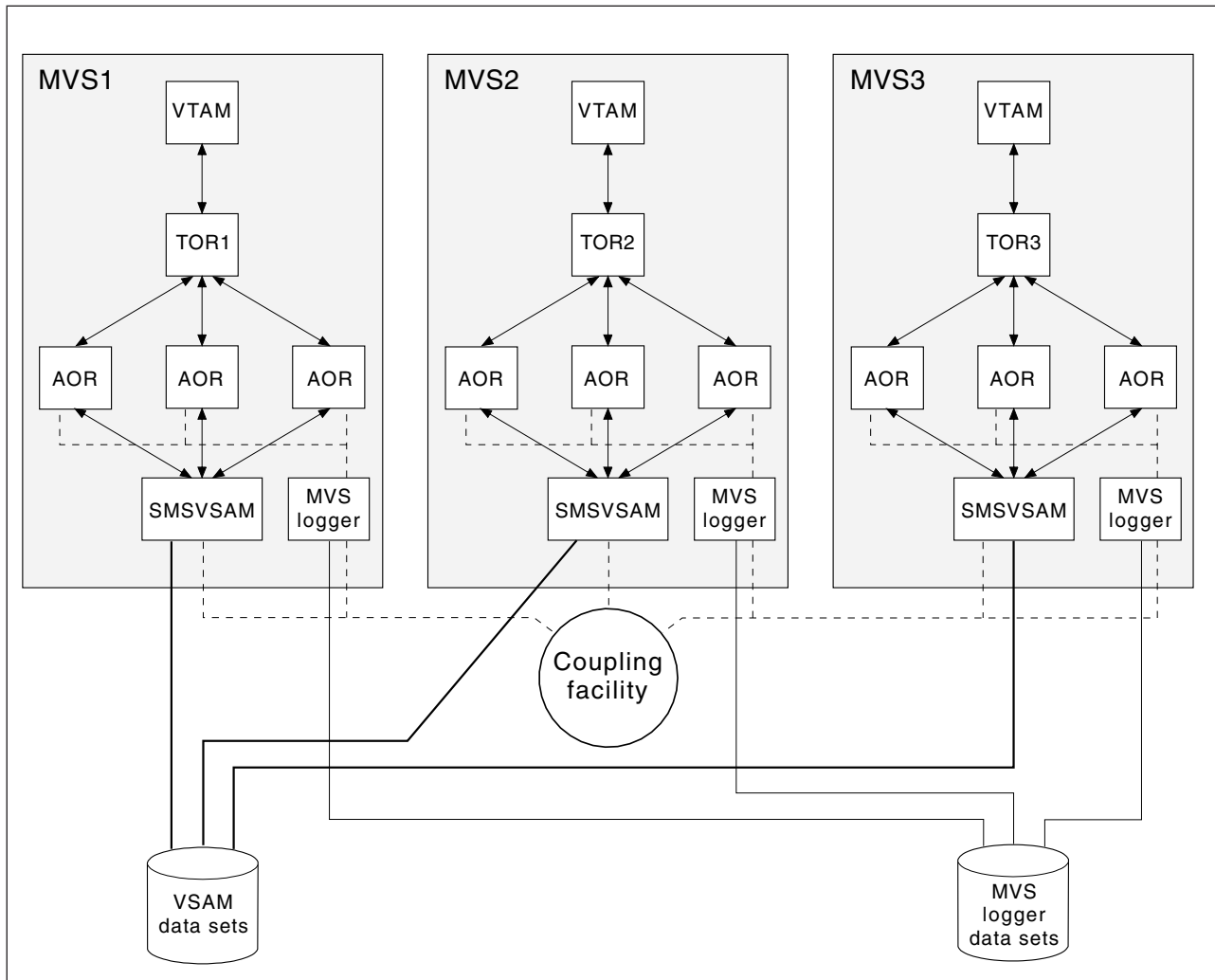


Figure 18. Diagram illustrating a Parallel Sysplex with RLS. This view of RLS shows multiple CICS regions using VSAM RLS, through the services of a SMSVSAM server in each MVS image.

Without RLS support (RLS=NO system initialization parameter), more than one CICS region cannot open the same VSAM data set concurrently using a non-RLS mode (such as LSR or NSR). These access modes mean that to share VSAM data between CICS regions, you must either:

- Use shared data tables,
- or
- Allocate the VSAM data sets to one CICS region, a file-owning region (FOR), and function ship file requests from the applications to the FOR using either MRO or APPC connections.

With RLS support, multiple CICS regions can open the same data set concurrently. To use RLS:

- You need a level of DFSMS that supports RLS, and RLS=YES specified as a CICS system initialization parameter
- The CICS regions must all run in the same parallel sysplex
- There must be one SMSVSAM server started in each MVS image
- Specify RLSACCESS(YES) in the CICS file resource definition to provide full update capability for data sets accessed by multiple CICS regions.

You can specify RLS access for all files supported by CICS file control, except for the following:

- Key range data sets are not supported.
- VSAM clusters defined with the IMBED attribute are not supported. However, you can remove the IMBED attribute from the cluster definition without loss of function. Use the access method services REPRO function to move the data into a new cluster defined without the IMBED attribute. You can then use RLS access mode for files that reference the new cluster. (IMBED is a performance option that is generally unnecessary with modern caching disk controllers.)
- Opening individual components of a VSAM cluster (which is not supported by CICS for any mode of access).
- Temporary data sets are not supported.
- Key-sequence data sets (KSDS) in relative byte address (RBA) mode (OPTCDE=ADR) are not supported. Application programs that specify the RBA keyword on file control API commands for a KSDS opened RLS mode receive an INVREQ with RESP2 51 exception condition.
- Direct open of alternate index (AIX[®]) data is not supported in RLS access mode. However, path access to data is supported.
- VSAM catalogs and VVDS data sets are not supported.

Although you can specify RLS access for entry-sequenced data sets (ESDS), it is not recommended, because it can have a negative effect on the performance and availability of the data set when you are adding records. See the *CICS Performance Guide*.

For details of all the steps necessary to set up support for VSAM RLS, see the *CICS Transaction Server for z/OS Installation Guide*.

Mixed-mode operation for VSAM data sets

Generally, you choose which data sets need to be shared and updated in RLS mode by multiple CICS regions. When you have made this choice, you are recommended always to update these data sets in RLS mode.

However, with RLS support, data sets can be shared in mixed access mode, between CICS regions and batch jobs. Mixed access mode means that a data set is open in RLS mode and a non-RLS mode concurrently by different users.

Although data sets can be open in different modes at different times, all the data sets within a VSAM sphere normally should be opened in the same mode. (A sphere is the collection of all the components—the base, index, any alternate indexes and alternate index paths—associated with a given VSAM base data set.) However, VSAM does permit mixed-mode operations on a sphere by different applications, subject to some CICS restrictions. In the following discussion about mixed-mode operation, references to a data set refer to any component of the sphere.

SMSVSAM operation of mixed mode:

SMSVSAM permits a data set to be opened in different modes concurrently, by different applications in a sysplex.

There are some sharing rules and limitations:

- A data set that is open in RLS mode to a number of CICS regions can also be opened in non-RLS mode for **read-only** operations

- Read-integrity is not guaranteed for the non-RLS application
- A data set to be opened concurrently in RLS and non-RLS mode must be defined with cross-region SHAREOPTIONS(2)

CICS restrictions:

You can open a file in RLS mode or non-RLS mode in a CICS region when the referenced data set is already open in a different mode by another user (CICS region or batch job).

However, in addition to the above VSAM rules, a data set cannot be open in different modes concurrently within the same CICS region. This ensures that CICS maintains a consistent view of data within the CICS region.

The CICS restrictions operate as follows:

- If a data set is opened in RLS mode in a CICS region, it cannot be opened, through another file, in non-RLS mode in the same CICS region.
A non-RLS mode file open request fails with message DFHFC0512 if CICS already has the data set open in RLS mode.
- If a data set is opened in non-RLS mode in a CICS region, it cannot be opened, through another file, in RLS mode in the same CICS region.
An RLS mode file open request fails with message DFHFC0511 if CICS already has the data set open in non-RLS mode.
- If a CICS region has unresolved recovery work for a data set it cannot be opened, through another file, in non-RLS mode in the same CICS region.
A non-RLS mode file open request fails with message DFHFC0513 if CICS has outstanding recovery work for the data set.

BDAM data sets

CICS supports access to keyed and nonkeyed BDAM data sets. To construct and format such data sets, you use BDAM.

About this task

A BDAM data set must contain data before it is used in a CICS run. You load the data set using a batch program that writes the records sequentially. An example of this is Figure 19.

Notes:

```
//BDAM EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=CICSTS41.bdam.user.file.init,DISP=SHR 1
//SYSUT2 DD DSN=CICSTS41.bdam.user.file,DISP=(,CATLG), 2
// SPACE=(TRK,(1,1)),UNIT=3380,VOL=SER=volid,
// DCB=(RECFM=F,LRECL=80,BLKSIZE=80,DSORG=DA) 3
```

Figure 19. Sample JCL to create and load a BDAM data set

1. The input data set (called SYSUT1 in this example) should be physically sequential and have attributes that are compatible with the DCB for the output data set (called SYSUT2 in this example; see note 3). In particular:

- If RECFM=F is specified for the output data set, then the input data set must have RECFM=F or RECFM=FB specified, and the value of LRECL should be the same for both data sets.
 - If RECFM=V or RECFM=U is specified for the output data set, then the value of LRECL for the input data set must not be greater than that specified on the output data set.
2. When you create a data set, you define a data set name (DSNAME) of up to 44 characters. This data set name uniquely identifies the data set to your MVS system.
 3. The DCB parameter for the output data set should specify the following:
 - DSORG=DA. This identifies the data set as a BDAM data set.
 - BLKSIZE. This should have the same value as specified for BLKSIZE in the associated file control table (FCT) entry.
 - RECFM. This can take the values F (fixed), V (variable), or U (undefined), and correspond to the first subparameter of the RECFORM operand in the associated FCT entry.

These options are specified on the DFHFCT TYPE=FILE definition. The *CICS Resource Definition Guide* gives information about defining files using DFHFCT TYPE=FILE options. A data set created by this example, and loaded with data such as that shown in Figure 20, would have the following attributes specified in its FCT entry:

- BLKSIZE=80
- LRECL=40
- RECFORM=(FIXED BLOCKED)
- KEYLEN=8

```

RECORD 1 DATA FOR RECORD 1    RECORD 2 DATA FOR RECORD 2
RECORD 3 DATA FOR RECORD 3    RECORD 4 DATA FOR RECORD 4

RECORD98 DATA FOR RECORD 98   RECORD99 DATA FOR RECORD 99
1----+-----2----+-----3----+-----4----+-----5----+-----6----+-----7----+-----8

```

Figure 20. Sample data for loading a BDAM data set

Defining data sets to CICS

Before CICS can open a file referencing a data set, there must be an installed file definition for that file. The definition can be installed either from the CSD or from a file control table (FCT).

You can define VSAM files **only** through the CSD, and BDAM files **only** through the FCT. Definitions other than for BDAM in the FCT will not be installed, but are still supported because the DFHFCT macro is used by the migration utility that migrates file definitions to the CSD.

A file is identified by its file name, which you specify when you define the file. CICS uses this name when an application program, or the master terminal operator using the CEMT command, refers to the associated data set.

Each file must also be associated with its data set in one of the following ways:

- Using JCL in the startup job stream
- Using the DSNAME and DISP parameters of the FILE resource definitions

- Using dynamic allocation with CEMT
- Using dynamic allocation with an application program

The number of VSAM files that can be allocated to a CICS address space is about 10000. However, VSAM maintains a table entry for each file opened, and table space limits the number of files opened to about 8189.

Using JCL

You can define the data set in a DD statement in the JCL of the CICS startup job. The DD name must be the same as the file name that refers to the data set.

For example, the following DD statements would correspond to file definitions for the file names VSAM1A and BDAMFILE:

```
//VSAM1A DD DSN=CICSTS41.CICS.vsam.user.file,DISP=OLD
//BDAMFILE DD DSN=CICSTS41.CICS.bdam.user.file,DISP=SHR
```

If you define a data set to CICS in this way it is allocated to CICS by MVS at CICS startup time, and it normally remains allocated until the end of the CICS run. Also, the physical data set is associated with the installed file definition throughout the CICS run.

If you use JCL to define a user data set to the CICS system, the DD statement must not include the FREE=CLOSE operand.

If you use the RLS=CR or RLS=NRI option on your DD statement, it will be ignored. The access mode for the file (RLS or non-RLS) and any read integrity options must be specified in the file definition.

When you are running CICS with XRF, you must specify DISP=SHR for data sets defined in JCL, so that the alternate CICS region can start while the active CICS region's job is also in progress.

Using the DSNNAME and DISP file resource definition parameters

You can define a data set to CICS by specifying the **DSNAME** and **DISP** parameters when you define the file.

You can specify these parameters either using RDO for files, or by using DFHFCT macros (BDAM files only). If you want to use DSNNAME and DISP from the file definition, do **not** provide a DD statement for the data set in the startup job stream, because the attributes in the DD statement will override those in the CICS resource definition.

If you use DSNNAME and DISP on the file definition, CICS allocates the data set dynamically at the time the first file referencing that data set is opened; that is, immediately before the file is opened. At this stage, CICS associates the file name with the data set.

When CICS applications subsequently refer to the data set, they do so by specifying the file name. When you define a data set in this way, it is automatically deallocated by CICS when the file is closed.

For information about using the **DSNAME** and **DISP** parameters, see the *CICS Resource Definition Guide*

Dynamic allocation using CEMT

You can set the data set name dynamically in an installed file definition by using the master terminal CEMT command.

```
CEMT SET FILE(filename) DSNAME(datasetname) SHARE|OLD
```

When you use this command, CICS allocates the data set as part of OPEN processing as described above. The data set is automatically deallocated when the last file entry associated with the data set is closed. Before you can dynamically allocate a file using the CEMT command, the file status must be CLOSED, and also be DISABLED or UNENABLED.

This method of defining the data set to CICS allows a file definition to be associated with different data sets at different times. Alternatively, you can close the file and deallocate the data set and then reallocate and open the same file with a different DISP setting. For example, you could do this to enable the physical data set to be shared with a batch program, which reads the data set.

For information about the CEMT SET command, see *CICS Supplied Transactions*.

Other forms of dynamic allocation

You are recommended to use only those methods of dynamic allocation that are part of CICS file control, and are described in the previous sections.

Do **not** use the CICS dynamic allocation transaction, ADYN, which invokes the sample CICS utility program, DFH99, for dynamic allocation of VSAM and BDAM **user files**. Use of the ADYN transaction may conflict with the dynamic allocation methods used within CICS file control, and can give unpredictable results.

Restrict the use of the ADYN transaction to those data sets not managed by CICS file control, such as auxiliary trace and CICS transaction dump data sets.

For information about the CICS samples, see the *CICS/ESA 4.1 Sample Applications Guide*.

Opening VSAM or BDAM files

Before your application programs can access a file, CICS must first have opened the file using the installed file definition referenced by your program.

About this task

Part of the process of opening a file is to ensure that the control blocks and buffers required for subsequent processing of the data set are available. If you defined the file to use VSAM local shared resources (LSR), these control blocks and buffers are allocated from the pool of resources. If the LSR pool does not exist at the time of the opening, CICS calculates the requirements and builds the pool before the file is opened. If you defined the file to use nonshared resources, the required control blocks and buffers are allocated by VSAM as part of OPEN processing. If you defined the file to be opened in RLS access mode, VSAM allocates control blocks and buffers in the SMSVSAM address space and associated data set. RLS mode also uses a CF cache structure, to which the data set is bound when the first file that references it is opened.

You may need to access a single VSAM data set either through the base or through one or more paths for different access requests. In this case, CICS uses a separate

file definition (that is, a separate file), for each of the access routes. Each file definition must be associated with the corresponding data set name (a path is also assigned a data set name). Each file must be open before CICS can access the file using the attributes in its installed file definition. This is because opening **one** file for a data set that is logically defined as two or more files with different attributes does not mean that the data set is then available for all access routes.

CICS permits more than one file definition to be associated with the same physical data set name. For example, you may want to define files with different processing attributes that refer to the same data set.

CICS allows or denies access to data in a file, depending on whether the state of the file is ENABLED. An **enabled** file that is closed is opened by CICS automatically when the first access request is made. The file remains open until an explicit CLOSE request or until the end of the CICS job.

You can also open a file explicitly by using either of the commands

```
CEMT SET FILE(filename) OPEN
EXEC CICS SET FILE(filename) OPEN
```

When you use one of these commands, the file is opened irrespective of whether its state is enabled or disabled. You may choose this method to avoid the overhead associated with opening the file being borne by the first transaction to access the file.

You can also specify that you want CICS to open a file immediately after initialization by specifying the RDO OPENTIME(STARTUP) attribute (or the FILSTAT=OPENED parameter in the DFHFCT macro). If you specify that you want CICS to open the file after startup, and if the file status is ENABLED or DISABLED, the CICS file utility transaction CSFU opens the file. (CSFU does not open files that are: defined as UNENABLED the status of these remains CLOSED, UNENABLED.) CSFU is initiated automatically, immediately before the completion of CICS initialization. CICS opens each file with a separate OPEN request. If a user transaction starts while CSFU is still running, it can reference and open a file that CSFU has not yet opened; it does not have to wait for CSFU to finish.

Closing VSAM or BDAM files

You can close files with a CLOSE command, with or without the FORCE option.

About this task

Closing files normally

You can close a file explicitly using the **SET FILE(filename) CLOSED** command.

The file is closed immediately if there are no transactions using the file at the time of the request. The file is also disabled as part of the close operation, this form of disablement showing as UNENABLED on a CEMT display. This prevents subsequent requests to access the file implicitly reopening it.

A transaction in the process of executing a VSAM or BDAM request, or executing a series of connected requests, is said to be a user of the file. For example, a transaction is a user during the execution of the following requests:

```
READ UPDATE ---- REWRITE
STARTBROWSE ---- READNEXT ... ---- ENDBROWSE
```

A transaction is also a user of a file if it completes a recoverable change to the file but has not yet reached a sync point or the end of the transaction.

If there are users at the time of the close request, the file is not closed immediately. CICS waits for all current users to complete their use of the file. The file is placed in an UNENABLING state to deny access to new users but allow existing users to complete their use of the file. When the last user has finished with the file, the file is closed and UNENABLED. If a transaction has made recoverable changes to a file and then suffered a failure during syncpoint, the unit of work is shunted, and the file can be closed at this point.

Closing files using the FORCE option

You can close a file using the FORCE option on the **SET FILE(filename)** command.

Any transactions that are current users of the file are abended and allowed to back out any changes as necessary, and the file is then closed and UNENABLED. A file UNENABLED as a result of a CLOSE request can be reenabled subsequently if an explicit OPEN request is made.

Closing a file using the FORCE option causes tasks of any current users of the file to be terminated immediately by the CICS task FORCEPURGE mechanism. Data integrity is not guaranteed with this mechanism. In some extreme cases (for example, if an error occurs during backout processing), CICS might terminate abnormally. For this reason, closing files using the FORCE option should be restricted to exceptional circumstances.

XRF considerations

For both VSAM and BDAM files, the active and alternate CICS region must refer to the same data sets.

About this task

To ensure that they do, perform the following steps:

Procedure

1. Define the files using the **DSNAME** and **DISP** parameters, and allow CICS to allocate the data sets dynamically. Omitting DD statements in the job streams for the active and alternate CICS regions minimizes the risk of inconsistency in data set naming. The alternate CICS region does not open the data sets before takeover occurs.
2. If you allocate the data sets at job step initiation, specify **DISP=SHR** in the JCL for the data sets.

CICS data tables

You can define a data table by using the **CEDA DEFINE FILE** command.

About this task

When a table is opened, CICS builds it by extracting data from the tables corresponding source VSAM data set and loading it into an MVS data space owned by the CICS data tables server region, and constructing an index in CICS

virtual storage above the 16 MB line. The commands used to access these tables are the file control commands of the CICS application programming interface (API).

For information about defining CICS data tables, see the *CICS Resource Definition Guide*. For programming information about the file control commands of the application programming interface, see the *CICS Application Programming Reference*. CICS supports two types of data table :

- **CICS-maintained data tables** that CICS keeps in synchronization with their source data sets.
- **User-maintained data tables** that are completely detached from their source data sets after being loaded.

For either type, a global user exit can be used to select which records from the source data set should be included in the data table.

For programming interface information about global user exits, see the *CICS Customization Guide*. For further information on CICS data tables, see the *CICS Shared Data Tables Guide*.

Opening data tables

A data table must be opened before its entries can be accessed by an application.

You can open a data table explicitly with an OPEN request, implicitly on first reference, or by the CSFU task just after startup, if OPENTIME(STARTUP) was specified in the file definition. When a data table is opened, CICS reads the complete source data set, copying the records into a data space and building an index.

A global user exit can be invoked for each record copied into the data table. This copying is subject to any selection criteria of the user-written exit.

The commands used to open data tables, and the rules and options concerning their implicit and immediate opening are the same as those described in “Opening VSAM or BDAM files” on page 102.

Loading data tables

A data table is built automatically when it is opened.

An index is constructed to provide rapid access to the records. See the *CICS Shared Data Tables Guide* for more details.

For a user-maintained data table, the ACB for the source data set is closed when loading has been completed. The data set is deallocated if it was originally dynamically allocated and there are no other ACBs open for it.

Closing data tables

You can close a data table with a CLOSE command, with or without the FORCE option.

When a data table is closed, the data space storage that was used to hold the records and the address space storage used for the associated index, is freed as part of the CLOSE operation.

The commands used to close data tables, and the rules concerning current users of a data table are the same as those described in “Closing VSAM or BDAM files” on page 103.

XRF considerations

After an XRF takeover, a data table must be reloaded from its source data set when the data table is opened.

For a CICS-maintained data table, the effect is to restore the data table to its final state in the previous active CICS region, because CICS keeps data tables and source data sets in step. For a user-maintained data table, the relationship of the current contents of the source data set to the contents of the data table when the previous active CICS region terminated is application-dependent.

Coupling facility data tables

Coupling facility data tables provide a method of file data sharing, using CICS file control, without the need for a file-owning region, and without the need for VSAM RLS support. CICS coupling facility data table support is designed to provide rapid sharing of working data within a sysplex, with update integrity.

About this task

The data is held in a coupling facility, in a table that is similar in many ways to a shared user-maintained data table. You must define the resources required for coupling facility data tables in an MVS coupling facility resource management (CFRM) policy.

Because read access and write access have similar performance, this form of table is particularly useful for scratchpad data. Typical uses might include sharing scratchpad data between CICS regions across a sysplex, or sharing of files for which changes do not have to be permanently saved. There are many different requirements for scratchpad data, and most of these can be implemented using coupling facility data tables. Coupling facility data tables are particularly useful for grouping data into different tables, where the items can be identified and retrieved by their keys. For example, you could use a field in a coupling facility data table to maintain the next free order number for use by an order processing application, or you could maintain a list of the numbers of lost credit cards in a coupling facility data table.

To an application, a coupling facility data table (CFDT) appears much like a sysplex-wide user-maintained data table, because it is accessed in the same way using the file control API. However, in a CFDT there is a maximum key-length restriction of 16 bytes.

Coupling facility data table models

There are two models of coupling facility data table.

These models are as follows:

- The contention model, which gives optimal performance but generally requires programs written to exploit it. This is because the CHANGED condition code (indicating that the data has been changed since the application program issued a read-for-update request) is specifically for this model, and programs not written for the contention model may not be able to handle this condition correctly. The CHANGED response can occur on a REWRITE or DELETE

command. There is also a situation with the contention model in which the NOTFND response can be returned on a REWRITE or DELETE.

This model is nonrecoverable: CFDT updates are not backed out if a unit of work fails.

- The locking model is API-compatible with programs that conform to the UMT subset of the file control API (this subset is nearly, but not quite, the full file control API).

This model can either be:

- **Non-recoverable:** locks do not last until syncpoint, and CFDT updates are not backed out if a unit of work fails, or
- **Recoverable:** coupling facility data tables are recoverable in the event of a unit of work failure and in the event of a CICS region failure (in that updates made by units of work that were in-flight at the time of the CICS failure are backed out).

The recoverable locking model supports indoubt and backout failures: if a unit of work fails when backing out an update to the CFDT or if it fails indoubt during syncpoint processing the locks are converted to retained locks and the unit of work is shunted.

You specify the model you want for each table on its file resource definition, enabling different tables to use different models.

Coupling facility data table structures and servers

Coupling facility data tables are held in coupling facility structures.

Access to a coupling facility data table is through a named server, which can be thought of as similar to a shared data tables file-owning region (see the *CICS Shared Data Tables Guide* for information about shared data tables support).

Coupling facility data tables support allows you to separate related groups of coupling facility data tables by storing them in separate pools. For example, you might want to have one pool for production and another for test.

A coupling facility data table pool is a coupling facility list structure, and access to it is provided by a coupling facility data table server. Within each MVS image, there must be one CFDT server for each CFDT pool accessed by CICS regions in the MVS image. The names of the servers are formed by adding the prefix DFHCF to the pool name, giving DFHCF.poolname. Coupling facility data table pools are defined in the coupling facility resource management (CFRM) policy. The pool name is then specified in the start-up JCL for the table server.

Access using file control API

A coupling facility data table is accessed from CICS through file control commands.

The file name specified on the command indicates the name of the table and pool in which it resides. The table name is either specified on the file definition or is the same as the file name, and the pool name is specified on the file definition. The table is uniquely identified by the pool name and table name, so that two tables with the same name may exist in different pools, and will be entirely separate entities.

Automatic connection to coupling facility data table pools

CICS automatically connects to the coupling facility data table server for a given pool the first time that a coupling facility data table within that pool is referenced. CICS also automatically reconnects to the coupling facility data table server when the server restarts after a failure.

Coupling facility data table servers are protected against misuse by CICS regions that call them, thus ensuring system integrity. In particular, protection is provided to prevent calling region from being able to modify sensitive parameters to authorized functions.

Likewise, CICS is protected from any side effects if a coupling facility data table server fails. If a CICS region issues a file control request to a coupling facility data table server that has failed, the resulting MVS abend is trapped and returned to the application program as a SYSIDERR condition.

Creating coupling facility data tables

CICS automatically creates a coupling facility data table (CFDT) when a first reference requires the CFDT to be opened. This CFDT is then used by the same region, or other CICS regions, that issue subsequent open requests of other files that name the same coupling facility data table.

CICS can optionally load the coupling facility data table automatically from a source VSAM (KSDS) data set when it is first opened. Unlike user-maintained data tables, with coupling facility data tables you can specify that there is no associated source data set, allowing you to create an empty CFDT.

Your application programs have access to a coupling facility data table as soon as it is created, although there are some restrictions on the keys that can be accessed while data is being loaded.

When a CFDT is loaded, it becomes an independent entity, separate from the behavior of the CICS regions that access the table or caused the table to be loaded. Even when all CICS regions have terminated, either normally or abnormally, a CFDT continues to remain in the coupling facility until you either take explicit action to delete the structure, or the coupling facility. You can delete the CFDT contents or structure with a `MODIFY cfdt_server, DELETE TABLE=name` command.

Administering coupling facility data tables

CICS provides some utility functions that allow you to obtain, from a coupling facility data table server, summary information and periodic statistics on coupling facility data tables defined within a pool.

This information is designed to help you to administer coupling facility data table pools, and to help you to evaluate capacity. See “Coupling facility data table server parameters” on page 365 for details.

Defining a coupling facility data table pool

You define the list structure for a coupling facility data table in a coupling facility resource manager (CFRM) policy in a sysplex couple data set.

A coupling facility data table pool, containing one or more coupling facility data tables, is accessed by CICS through a server region using cross-memory services.

(See Chapter 23, “Setting up and running a coupling facility data table server,” on page 363 for information about setting up and starting a coupling facility data table region.)

From the application point of view, a pool and its server are similar to a file-owning region, and the pool can contain any number of tables provided that each one has a unique table name.

Before a coupling facility data table server can use its pool, the active CFRM policy must contain a definition of the list structure to be used for the pool. To achieve this, add a statement that specifies the list structure to a selected CFRM policy, and then activate the policy.

The CFRM structure definition specifies the size of the list structure and the preference list of coupling facilities in which it can be stored. You create the name of the list structure for a coupling facility data table pool by adding the prefix DFHCFLS_ to the pool name, giving DFHCFLS_*poolname*.

Using IXCMIAPU to update a policy

To update an administrative policy in the CFRM couple data set, use the administrative data utility, IXCMIAPU.

The utility adds or changes policy data in the administrative policies only: it does not change any information in the system's copy of the active CFRM policy. For an example of a job to run this utility, see member IXCCFRMP in the SYS1.SAMPLIB library. An example of a policy statement for a coupling facility data table pool is shown in Figure 21.

```
STRUCTURE NAME(DFHCFLS_PRODCT1)
SIZE(1000)
INITSIZE(500)
PREFLIST(FACIL01,FACIL02)
```

Figure 21. Example of defining a coupling facility data table structure

Activating a CFRM policy:

When you have defined the list structure in a CFRM policy, activate the policy using the MVS command SETXCF START,POLICY,POLNAME=*polycyname*,TYPE=CFRM.

Activating a CFRM policy that contains a definition of a list structure does not create the structure. It is created the first time an attempt is made to connect to it, which occurs when the first coupling facility data table server that refers to the corresponding pool is started.

When the server creates a list structure, it is allocated with an initial size, which can be increased up to a maximum size as specified in the CFRM policy. All structure sizes are rounded up to the next multiple of 256 KB at allocation time. Provided that space is available in the coupling facility, you can dynamically expand a list structure from its initial size up to its maximum size, or contract it to free up coupling facility space for other purposes. Note that if the initial structure allocation becomes full, the structure does not expand automatically, even if the structure allocated is less than the specified maximum size. To expand a list structure when the allocation becomes full, you can expand it (up to its maximum size) using the following SETXCF command:

```
SETXCF START,ALTER,STRNAME=DFHCFLS_poolname,SIZE=nnnn
```

Note that if you dynamically increase the size of a list structure in this way, also update the INITSIZE parameter in the policy to reflect the new size, so that the structure does not revert to its original size if you subsequently recreate or reload it.

Calculating the structure size for a pool

A coupling facility structure contains not only stored data but also the information needed to manage and access that data, in a similar way to a key-sequenced data set.

The data for each entry in the coupling facility is stored as a chain of fixed size (usually 256-byte) elements, which means that the exact length for variable-length data has to be stored separately. CICS does this by including a length prefix in the stored data, so space calculations have to allow for each entry using a whole number of elements. The amount of internal control information depends on the level of functionality and performance of the coupling facility control code for the current CFLEVEL. The storage requirements can increase for a higher CFLEVEL. For more information about how the coupling facility structure storage is organized, see “Coupling facility storage management” on page 416.

The easiest way to calculate accurate structure storage requirements is to use the web-based IBM CFSizer tool at <http://www.ibm.com/systems/support/z/cfsizer/>. The tool takes these factors into account and communicates with a coupling facility at a current CFLEVEL to perform the calculation. You need to enter some minimum input to get an accurate storage calculation as follows:

MAXQUEUES

specifies the maximum number of data lists that are reserved when the structure list is allocated by CICS. This determines the maximum number of large queues that can be stored in the structure and corresponds to the MAXQUEUES server parameter. A large queue is where the total size of the data items exceeds 32K. It is stored in a separate list in the structure. Although you should specify a large enough number, specifying an excessively large number will use up an unnecessary amount of coupling facility storage for unused preallocated list headers. The valid range is from 1 to 999999. The default is 1000.

Average rounded item size

specifies the average amount of storage required for each TS queue item. Note that each item has a two-byte length prefix and is stored as one or more 256-byte elements. This number should range from 1 to 32768. This value determines the entry to element ratio that is used to calculate the required structure size.

If all queue items are approximately the same size, calculate this value by taking the average data size, adding two, and rounding up to the next multiple of 256. If queue items are different sizes, round up each size first before taking the average. For example, if half of the items are 100 bytes and half are 300 bytes, then the rounded sizes are 256 and 512. The average rounded item size is half way between those values, 384, which is more accurate than taking the overall average item size of 200 and rounding it up to 256.

Total number of items in all queues

specifies the total number of entries in all of the TS queues.

Target Usage Percent

specifies the percentage of the structure space which the given total number of items are expected to use. Specify a number in the range of 1 to

100. The default is 75. This leaves some free space for temporary expansion, and to give time to expand the structure in response to warning messages (which normally start at 80%) if the initial free space is not enough.

Max Expansion Percent

specifies the percentage that the structure can expand. If a non-zero value is specified here, the maximum structure size will be greater than the initial structure size by an amount sufficient to allow for a further expansion of the total amount of data by this percentage. For example, if the value 200 is specified, the initial size will be enough to store the specified total number of items, but the maximum size will be enough to store three times that number of items.

Alternatively you can use the following calculation, which applies up to CFLEVEL 11 of the coupling facility. If you have a coupling facility at CFLEVEL 12, you need to change the calculation to add 10% and 2MB to the sizings in the formula below.

Storage calculations

Data entry size = (170 + (*average record data size*¹))
+ 5% extra for control information

¹ Average record data size must have a 2-byte prefix added and be rounded up to a multiple of 256 bytes.

Total size = 200KB
+ (*number of tables* x 1KB)
+ (*number of records in all tables* x *data entry size*)

The above calculation assumes that the structure is allocated at its maximum size. If it is allocated at less than its maximum size, the same amount of control information is still required, so the percentage of space occupied by control information is correspondingly increased. For example, if a structure is allocated at one third of its maximum size, the overhead for control information increases to around fifteen per cent.

Note: The list headers for a table use about two-thirds of the 1KB in the storage formula shown above. If you let the MAXTABLES parameter take the default value, about 700KB of the list structure size is taken up by the list headers alone.

The calculation determines the size of a structure that would be approximately 100% full for the specified level of usage. For practical operation, however, a reasonable proportion of free space must be available, not only to minimize the risk of the structure becoming totally full but also to avoid triggering low space warning messages and additional activity to alter entry to element ratios. The maximum normal usage should therefore aim to be about 75% of the structure size. Expected usage values should therefore be adjusted upwards to allow for the required amount of free space (for example, by about one third to aim for 75% usage).

For information about the reserved space parameters you can use to enable the server to avoid a structure full condition, see “Reserved space parameters” on page 373 and “Avoiding structure full conditions” on page 374.

Chapter 11. Defining the CDBM GROUP command data set

The CDBM GROUP command data set, DFHDBFK, is a VSAM key-sequenced data set (KSDS). The CDBM transaction uses this data set to provide a repository for stored groups of DBCTL commands, .

About this task

Procedure

1. Create the DFHDBFK data set by running an IDCAMS job, an example of which is shown in Figure 22 on page 114.

```

//DBFKJOB JOB 'accounting information',name,MSGCLASS=A
//*
//DBFKDEF EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT DD SYSOUT=*
//AMSDUMP DD SYSOUT=*
//SYSIN DD *
DELETE CICS41.CICS.DFHDBFK
SET MAXCC=0
DEFINE CLUSTER (
    NAME( CICS41.CICS.DFHDBFK ) -
    INDEXED -
    RECORDS(100 20) -
    KEYS(22,0) -
    RECORDSIZE(1428 1428) -
)
INDEX
(
    NAME( CICS41.CICS.DFHDBFK.INDEX ) -
    CONTROLINTERVALSIZE(512) -
)
DATA
(
    NAME( CICS41.CICS.DFHDBFK.DATA ) -
    CONTROLINTERVALSIZE(2048) -
)
/*
/* The next two job steps are optional.
/*
//DBFKINID EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DELETE CICS41.CICS.DBFKINIT
/*
//DBFKINIF EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD DSN=CICS41.CICS.DBFKINIT,DISP=(NEW,CATLG),
// UNIT=dbfkunit,VOL=SER=dbfkvol,SPACE=(TRK,(1,1)),
// DCB=(RECFM=FB,LRECL=40,BLKSIZE=6160)
/* Place the definitions you want to load after SYSUT1. For example:
//SYSUT1 DD *
SAMPLE DIS DB DI21PART
SAMPLE STA DB DI21PART
SAMPLE STO DB DI21PART
/*
//SYSIN DD *
GENERATE MAXFLDS=1
RECORD FIELD=(40)
/*
//DBFKLOAD EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT DD SYSOUT=*
//AMSDUMP DD SYSOUT=*
//SYS01 DD DSN=CICS41.CICS.DBFKINIT,DISP=SHR
//DFHDBFK DD DSN=CICS41.CICS.DFHDBKF,DISP=SHR
//SYSIN DD *
REPRO INFILE (SYS01) -
OUTFILE (DFHDBFK)
/*
//

```

where *dbfkvol* is the volume on which the DFHDBFK data set is to be created and *dbfkunit* is the unit type for that volume.

Figure 22. Sample job to define and initialize the DFHDBFK data set

2. Use this job to load IMS™ commands or use the maintenance function within the CDBM transaction.

Job control statements for CICS execution

About this task

If you define the DFHDBFK data set using the sample JCL shown in Figure 22 on page 114, the data definition statement for the CICS execution is as follows:

```
//DFHDBFK DD DSN=CICSTS41.CICS.DFHDBFK,DISP=SHR
```

Alternatively, if you want to use dynamic file allocation, add the fully-qualified data set name to the DFHDBFK file resource definition.

Record layout in the CDBM GROUP command file

Each record in the DFHDBFK file may be up to 1428 characters long, as follows:

About this task

| field | length | content | description |
|-------|--------|------------------------|--|
| 1 | 12 | Group | a 12-character field containing your chosen name for this group. The acceptable characters are A-Z 0-9 \$ @ and #. Leading or embedded blanks are not allowed, but trailing blanks are acceptable. |
| 2 | 10 | IMS Command | a 10-character field containing any of the IMS command verbs that are valid for CDBM (see the <i>Connection, disconnection, and inquiry transactions for the CICS DBCTL interface</i> topic in the <i>CICS IMS Database Control Guide</i> for details). Leading or embedded blanks are not allowed, but trailing blanks are acceptable. Note: The validity of the IMS command verb is not checked by CDBM. Invalid values will be reported by IMS when the command is attempted. |
| 3 | 1406 | IMS Command parameters | Up to 1406 characters of parameters appropriate to the chosen IMS command verb. (This will often consist of lists of databases.) Note: Wildcard characters may not be used in the parameters stored in the CDBM Group command file. This is unlike the other functions of the CDBM transaction which permit the use of wildcard characters to describe multiple similarly named databases. |

Chapter 12. Defining the CMAC messages data set

The CMAC messages data set is a VSAM key-sequenced data set (KSDS) called DFHMACD. The CMAC transaction uses DFHMACD to provide online descriptions of the CICS messages and codes.

About this task

Procedure

1. Create the DFHMACD data set and load it with the CICS-supplied messages and codes data. You can create the data set in one of the following ways:
 - Run the DFHMACI job. For more information about the DFHMACI, see the *CICS Transaction Server for z/OS Installation Guide*.
 - Use the supplied sample job, as described in “Job control statements to define and load the messages data set.”
2. If you use the supplied sample job to define the messages data set, add the following data definition statement to your CICS start up JCL:

```
//DFHMACD DD DSN=CICSTS41.CICS.DFHMACD,DISP=SHR
```
3. The DFHMACD data set is accessed by the file CMAC, which is managed by CICS file control. You must create a definition for this file in the CSD or FCT. The CICS-supplied definition for the CMAC file and other resources required by the CICS messages facility are in the CSD group DFHMAC. The CICS IVPs have a DD statement for the CMAC file, but for dynamic allocation you must copy the supplied resource definition for the CMAC file and add the DSNAME option.
4. To use the CICS messages facility in your CICS region, you must create your own CSD group list to include the CICS-supplied group list DFHLIST, the DFHMAC group for the CICS messages facility, and any other groups of resources that your CICS region needs. You must specify this group list by using the system initialization parameter **GRPLIST** when you start up your CICS region.

Job control statements to define and load the messages data set

Before its first use, the DFHMACD data set should be defined and loaded as a VSAM key sequenced data set (KSDS). The following sample job shows you how to do this.

About this task

Note that `cmacvol` is the volume on which the DFHMACD data set is to be created.

```

//CMACJOB JOB 'accounting information',name,MSGCLASS=A
//CMACDEF EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT DD SYSOUT=*
//AMSDUMP DD SYSOUT=*
//SYSIN DD *
DELETE CICSTS41.CICS.DFHCMACD
SET MAXCC=0
DEFINE CLUSTER (
    NAME( CICSTS41.CICS.DFHCMACD ) -
    CYL(2,1) -
    KEYS( 9 0 ) -
    INDEXED -
    VOLUME ( cmacvo1) -
    RECORDSIZE( 8192 30646 ) -
    FREESPACE( 5 5 ) -
    SHAREOPTIONS( 2 ) -
)
INDEX (
    NAME( CICSTS41.CICS.DFHCMACD.INDEX ) -
)
DATA (
    NAME( CICSTS41.CICS.DFHCMACD.DATA ) -
)
/*
//CMACLOAD EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT DD SYSOUT=*
//AMSDUMP DD SYSOUT=*
//SYS01 DD DSN=CICSTS41.CICS.SDFHMSGSD(DFHCMACD),DISP=SHR
//DFHCMACD DD DSN=CICSTS41.CICS.DFHCMACD,DISP=SHR
//SYSIN DD *
REPRO INFILE (SYS01) -
    OUTFILE (DFHCMACD)
/*
//

```

Figure 23. Sample job to define and initialize the CMAC data set

Chapter 13. Defining the EJB data sets

This chapter describes the VSAM key-sequenced data sets you need for CICS EJB support.

These data sets are:

- The EJB directory, DFHEJDIR
- The EJB object store, DFHEJOS

These data sets, and the IDCAMS definition statements for creating them, are described in the following topic.

Defining EJB directory and object store data sets

DFHEJDIR is a file containing a request streams directory that is shared by all the regions (listeners and AORs) in a logical EJB server. DFHEJOS is a file of passivated stateful session beans. It is shared by all the AORs in the logical EJB server.

The EJB directory and object store data sets are CICS file-control-managed data sets that require resource definitions in the CSD. Because the data sets have to be shared by all the CICS regions that form a logical EJB server, you can define each of them to CICS file control as one of the following:

- An ordinary VSAM data set, to be opened in either LSR or NSR mode (that is, with the LSRPOOLID attribute specified with a pool number, or as NONE). In this case, the data set has to be owned by one CICS region (a file-owning region) to which the others in the EJB server can function ship file requests.

Definitions for both data sets are supplied in the CICS CSD group called DFHEJVS, with the LSRPOOLID default value of 1. Make a copy of this CSD group, rename it, and edit the file definitions to add details such as the data set name for dynamic allocation, a specific LSRPOOLID to match an explicit LSRPOOL resource definition, or the remote system attributes.

- A VSAM data set to be opened in RLS mode. Definitions for both data sets are supplied in the CICS CSD group called DFHEJVR with RLSACCESS(YES) specified. Make a copy of this CSD group, rename it, and edit the file definitions to add attributes such as the data set name for dynamic allocation.
- A coupling facility data table. Definitions for both data sets are supplied in the CICS CSD group called DFHEJCF. Make a copy of this CSD group, rename it, and edit the file definitions to modify the CFDT details, such as the pool name and the table name.

Figure 24 on page 120 shows an example of the JCL you can use to define an EJB directory data set.

```

//EJBDIR  JOB accounting info,,CLASS=A
//DEFEJB  EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN   DD   *
/*
/* DEFINE A DIRECTORY
/* DATASET
/*
  DEFINE CLUSTER(NAME(CICSTS41.CICS.DFHEJDIR)-
    INDEXED-
    LOG(UNDO)-
    CYL(2 1)-
    VOLUME(valid)-
    RECORDSIZE( 1017 1017 )-
    KEYS( 16 0 )-
    FREESPACE ( 10 10 )-
    SHAREOPTIONS( 2 3 )-
    DATA (NAME(CICSTS41.CICS.DFHEJDIR.DATA) -
    CONTROLINTERVALSIZE(1024)) -
    INDEX (NAME(CICSTS41.CICS.DFHEJDIR.INDEX))
/*
/*

```

Figure 24. Example JCL to define an EJB directory data set

1. The backout recovery attribute is defined in the ICF catalog so that the data set defined by this job can be used in either RLS or non-RLS mode. For data sets used in RLS mode, the recovery attributes must be defined in the ICF catalog, and they override any that are specified in the CICS file resource definition. The EJB directory data set must be defined as recoverable.
2. Specify your own value for the VOLUME parameter, or remove it altogether if you are using SMS-managed storage.
3. The average and maximum record size must both be defined as 1017 bytes.

Note: You should not need to change any of the definition values shown in Figure 24: DFHEJDIR only contains one record, its control record. However, ensure that runtime settings are specified (such as BUFND, BUFNI and STRNO subparameters on the AMP parameter on the DD statement, or the equivalent in the CICS file resource definition) to support the maximum number of requests that could be active at any one time.

Figure 25 on page 121 shows an example of the JCL you can use to define an EJB object store.

```

//EJBOS   JOB accounting info,,CLASS=A
//DEFEJB  EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN   DD   *
/*
/* DEFINE AN OBJECT STORE
/* DATASET
/*
  DEFINE CLUSTER(NAME(CICSTS41.CICS.DFHEJOS) -
    INDEXED -
    LOG(NONE) -
    RECORDS(1000,100) -
    VOLUME(valid) -
    RECORDSIZE( 8185 8185 ) -
    KEYS( 16 0 ) -
    FREESPACE ( 10 10 ) -
    SHAREOPTIONS( 2 3 )) -
  DATA (NAME(CICSTS41.CICS.DFHEJOS.DATA) -
    CONTROLINTERVALSIZE(8192)) -
  INDEX (NAME(CICSTS41.CICS.DFHEJOS.INDEX))
/*
/*

```

Figure 25. Example JCL to define an EJB object store data set

1. The backout recovery attribute is defined in the ICF catalog so that the data set defined by this job can be used in either RLS or non-RLS mode. For data sets used in RLS mode, the recovery attributes must be defined in the ICF catalog, and they override any that are specified in the CICS file resource definition. The EJB object store data set must be defined as non-recoverable.
2. The number of records that the file can hold. VSAM automatically calculates space requirements.
3. Specify your own value for the VOLUME parameter, or remove it altogether if you are using SMS-managed storage.
4. The average and maximum record size are both shown as 8185 bytes in this sample job, but you will need to calculate your own size as described in “Determining the object store space requirements.”

Determining the object store space requirements

Follow these guidelines when calculating the size of your object store data set:

1. Find the average size of a stateful bean. To do this you can write a method that serializes a bean and writes it to a file. The filesize is the size of the bean. Repeat for each bean, and then calculate the average filesize.

Note: The size of the bean depends on what state the bean has. For example, if a stateful bean contains a hashtable populated with 100 KB of non-transient data then the serialized object must also contain this information.

2. Round the figure at Step 1 up to nearest multiple of 512 bytes (subject to a minimum of 1 024 bytes).
3. Estimate the maximum number of records, and use this as the primary allocation in the RECORDS parameter (see 2).
4. Estimate the requirements for the secondary allocation, for example by halving the primary allocation.

Modify the supplied IDCAMS step in DFHDEFDS to create an object store that meets your needs.

Chapter 14. Defining the WS-AT data set

The data set you require to enable CICS Web Services Atomic Transaction (WS-AT) support is the WS-AT directory data set, DFHPIDIR. The WS-AT directory data set, DFHPIDIR, is a file containing a mapping between contexts and tasks.

The WS-AT directory and object store data set, DFHPIDIR, is a CICS file-control-managed data set that requires a resource definition in the CSD. Because the data set is shared across CICS regions that together provide a WS-AT capable Web service provider, you can define it to CICS file control in these ways:

- An ordinary VSAM data set, to be opened in either LSR or NSR mode; that is, with the LSRPOOLID attribute specified with a pool number, or as NONE. In this case, the data set has to be owned by one CICS region (a file-owning region) to which the other regions can function ship file requests.

A definition for the data set is supplied in the CICS CSD group DFHPIVS, with the LSRPOOLID default value of 1. Make a copy of this CSD group, rename it, and edit the file definitions to add details such as the data set name for dynamic allocation, a specific LSRPOOLID to match an explicit LSRPOOL resource definition, or the remote system attributes.

- A VSAM data set to be opened in RLS mode. A definition for the data set is supplied in the CICS CSD group DFHPIVR with RLSACCESS(YES) specified. Make a copy of this CSD group, rename it, and edit the file definitions to add attributes such as the data set name for dynamic allocation.
- A coupling facility data table. A definition for the data set is supplied in the CICS CSD group DFHPICF. Make a copy of this CSD group, rename it, and edit the file definitions to modify the CFDT details, such as the pool name and the table name.

Figure 26 shows the statements used to define DFHPIDIR.

```
DEFINE CLUSTER(NAME(@dsindex@.CICS@regname@.DFHPIDIR)-
INDEXED-
LOG(UNDO)-                1
CYL(2 1)-
VOLUME(@dsvol@)-          2
RECORDSIZE( 1017 1017 )-  3
KEYS( 16 0 )-
FREESPACE ( 10 10 )-
SHAREOPTIONS( 2 3 )-
DATA (NAME(@dsindex@.CICS@regname@.DFHPIDIR.DATA) -
CONTROLINTERVALSIZE(1024)) -
INDEX (NAME(@dsindex@.CICS@regname@.DFHPIDIR.INDEX))
```

Figure 26. Example JCL to define the WS-AT directory data set

1 Define the backout recovery attribute in the ICF catalog, so that the data set defined by this job can be used in either RLS or non-RLS mode. For data sets used in RLS mode, you define the recovery attributes in the ICF catalog, and those attributes override any that are specified in the CICS file resource definition. You must define the PI directory data set as recoverable.

2 Specify your own value for the VOLUME parameter, or remove it completely if you are using SMS-managed storage.

3 The default record size is 1 KB, which can be changed.

Note: You do not change any of the definition values shown in Figure 26 on page 123. DFHPIDIR contains only one record, its control record. However, ensure that you specify runtime settings (such as BUFND, BUFNI; and STRNO subparameters on the AMP parameter on the DD statement, or the equivalent in the CICS file resource definition) to support the maximum number of requests that might be active at any one time.

Chapter 15. Setting up the debugging profiles data sets

Application programmers who use certain debugging tools with CICS create debugging profiles which are stored in a VSAM key sequenced data set with an alternative index.

To set up the debugging profiles data sets:

1. Use the IDCAMS utility to create and initialize the VSAM data sets.
2. Create file definitions for the data sets. The data sets can be shared by more than one CICS region, and can be defined as:

VSAM RLS files

Define VSAM RLS files when you want to share profiles between several CICS regions in the same sysplex

VSAM non-RLS files

Define VSAM non-RLS files when you do not need to share profiles between regions in the same sysplex

Remote files

Define remote files when you want to use profiles that are stored in a region that is connected using MRO or ISC.

Creating the debugging profiles data sets

You can create the debugging profiles data sets in one of two ways. You can either use an IDCAMS utility to create and initialize the data sets or you can run the CICS-supplied job, DFHDEFDS, to create the data sets for a CICS region

Use the IDCAMS utility to create and initialize the following VSAM data sets:

DFHDPFMB

The debugging profiles base data set.

DFHDPFMP

The debugging profiles path data set.

DFHDPFMX

The debugging profiles alternate index data set.

Use the JCL in Figure 27 on page 126.

```

//DPFM JOB 'accounting information',name,MSGCLASS=A
//DEFINE EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
    DELETE CICSTS41.CICS.DFHDPFMB

    DEFINE CLUSTER (RECORDS(1000)-
        NAME (CICSTS41.CICS.DFHDPFMB) -
        SHAREOPTIONS(2 3) -
        LOG(NONE) -
        VOLUME (&DSVOL) -
        IXD) -
    DATA -
        (RECSZ(2560,2560) -
        CONTROLINTERVALSIZE(3072) -
        NAME (CICSTS41.CICS.DFHDPFMB.DATA) -
        KEYS(17 1) -
        FREESPACE(10 10) -
        BUFFERSPACE (8192)) -
    INDEX -
        (NAME(CICSTS41.CICS.DFHDPFMB.INDX))
//INITDP EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
    REPRO INFILE ( SYS01 ) -
        OUTDATASET(CICSTS41.CICS.DFHDPFMB)
//SYS01 DD *
    DDUMMY RECORD !! DO NOT ALTER !!
    EEXAMPLE RECORD REMOVE THIS LINE IF SAMPLES NOT REQUIRED
/*
//DEFAULT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
    DEFINE ALTERNATEINDEX -
        ( NAME(CICSTS41.CICS.DFHDPFMX ) -
        RECORDS(1000) -
        VOLUME(&DSVOL) -
        KEYS(12 20) -
        RELATE(CICSTS41.CICS.DFHDPFMB) -
        RECORDSIZE(200 200) -
        SHAREOPTIONS(2 3) -
        UPGRADE ) -
    DATA -
        ( NAME(CICSTS41.CICS.DFHDPFMX.DATA) ) -
    INDEX -
        ( NAME(CICSTS41.CICS.DFHDPFMX.INDEX) )
    DEFINE PATH -
        ( NAME(CICSTS41.CICS.DFHDPFMP) -
        PATHENTRY(CICSTS41.CICS.DFHDPFMX) )
/*
//BLDDP EXEC PGM=IDCAMS
//BDSET1 DD DSN=CICSTS41.CICS.DFHDPFMB,DISP=SHR
//ADSET1 DD DSN=CICSTS41.CICS.DFHDPFMX,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
    BLDINDEX -
        INFILE(BDSET1) -
        OUTFILE(ADSET1)
/*
//*/

```

Figure 27. Sample JCL to create the debugging profiles data sets

The sample JCL creates data sets which contain example debugging profiles. To create empty data sets, remove the following line:

```
EEXAMPLE RECORD REMOVE THIS LINE IF SAMPLES NOT REQUIRED
```

Defining the debugging profiles data sets as VSAM RLS files

Define VSAM RLS files when you want to share profiles between several CICS regions in the same sysplex.

CICS provides the following sample definitions:

```

*-----*
*   Define base file for Debugging Profiles (RLS)   *
*-----*
DEFINE FILE(DFHDPFMB) GROUP(DFHDPVR)
DESCRIPTION(Debugging Profiles base file - VSAM RLS)
  RLSACCESS(YES)          TABLE(NO)
  LSRPOOLID(1)            DSNSHARING(ALLREQS)
  STRINGS(5)              STATUS(ENABLED)
  OPENTIME(FIRSTREF)     DISPOSITION(SHARE)
  DATABUFFERS(3)         INDEXBUFFERS(2)
  RECORDFORMAT(V)        READINTEG(REPEATABLE)
  ADD(YES)                BROWSE(NO)
  DELETE(YES)            READ(YES)
  UPDATE(YES)            JOURNAL(NO)
  JNLREAD(NONE)          JNLSYNCREAD(NO)
  JNLUPDATE(NO)          JNLADD(NONE)
  JNLSYNCWRITE(YES)      RECOVERY(NONE)
  FWDRECOVLOG(NO)        BACKUPTYPE(STATIC)
*-----*
*   Define path file for Debugging Profiles (RLS)   *
*-----*
DEFINE FILE(DFHDPFMP) GROUP(DFHDPVR)
DESCRIPTION(Debugging Profiles path file - VSAM RLS)
  RLSACCESS(YES)          TABLE(NO)
  LSRPOOLID(1)            DSNSHARING(ALLREQS)
  STRINGS(10)             STATUS(ENABLED)
  OPENTIME(FIRSTREF)     DISPOSITION(SHARE)
  DATABUFFERS(11)        INDEXBUFFERS(10)
  RECORDFORMAT(V)        READINTEG(REPEATABLE)
  ADD(YES)                BROWSE(NO)
  DELETE(YES)            READ(YES)
  UPDATE(YES)            JOURNAL(NO)
  JNLREAD(NONE)          JNLSYNCREAD(NO)
  JNLUPDATE(NO)          JNLADD(NONE)
  JNLSYNCWRITE(YES)      RECOVERY(NONE)
  FWDRECOVLOG(NO)        BACKUPTYPE(STATIC)

```

Figure 28. Resource definitions for debugging profiles data sets defined as VSAM RLS files

1. Copy the sample FILE definitions for DFHDPFMB and DFHDPFMP resource to another group.
2. Add the DSNAME attribute:
 - For DFHDPFMB, specify the name of the debugging profiles base data set.
For example:
DSNAME ==> CICSTS41.CICS.DFHDPFMB
 - For DFHDPFMP, specify the name of the debugging profiles base data set.
For example:
DSNAME ==> CICSTS41.CICS.DFHDPFMP

Alternatively, you can omit the DSNAME attribute, and include a DD card in the CICS startup JCL. For example:

```
//DFHDPFMB DD DSN=CICSTS41.CICS.DFHDPFMB,DISP=SHR
//DFHDPFMP DD DSN=CICSTS41.CICS.DFHDPFMP,DISP=SHR
```

3. Install the FILE definitions.

Defining the debugging profiles data sets as VSAM non-RLS files

Define VSAM non-RLS files when you do not need to share profiles between regions in the same sysplex.

CICS provides the following sample definitions:

```
*-----*
*   Define base file for Debugging Profiles (non-RLS)   *
*-----*
DEFINE FILE(DFHDPFMB) GROUP(DFHDPVSL)
DESCRIPTION(Debugging Profiles Base File)
  RLSACCESS(NO)          LSRPOOLID(1)
  READINTEG(UNCOMMITTED) DSNSHARING(ALLREQS)
  STRINGS(10)           STATUS(ENABLED)
  OPENTIME(FIRSTREF)    DISPOSITION(SHARE)
  DATABUFFERS(11)       INDEXBUFFERS(10)
  TABLE(NO)            RECORDFORMAT(V)
  ADD(YES)              BROWSE(YES)
  DELETE(YES)           READ(YES)
  UPDATE(YES)           JOURNAL(NO)
  JNLREAD(NONE)         JNLSYNCREAD(NO)
  JNLUPDATE(NO)         JNLADD(NONE)
  JNLSYNCWRITE(NO)      RECOVERY(NONE)
  FWDRECOVLOG(NO)       BACKUPTYPE(STATIC)
*-----*
*   Define path file for Debugging Profiles (non-RLS)   *
*-----*
DEFINE FILE(DFHDPFMP) GROUP(DFHDPVSL)
DESCRIPTION(Debugging Profiles Path File)
  RLSACCESS(NO)          LSRPOOLID(1)
  READINTEG(UNCOMMITTED) DSNSHARING(ALLREQS)
  STRINGS(10)           STATUS(ENABLED)
  OPENTIME(FIRSTREF)    DISPOSITION(SHARE)
  DATABUFFERS(11)       INDEXBUFFERS(10)
  TABLE(NO)            RECORDFORMAT(V)
  ADD(YES)              BROWSE(YES)
  DELETE(YES)           READ(YES)
  UPDATE(YES)           JOURNAL(NO)
  JNLREAD(NONE)         JNLSYNCREAD(NO)
  JNLUPDATE(NO)         JNLADD(NONE)
  JNLSYNCWRITE(NO)      RECOVERY(NONE)
  FWDRECOVLOG(NO)       BACKUPTYPE(STATIC)
```

Figure 29. Resource definitions for debugging profiles data sets defined as VSAM non-RLS files

1. Copy the sample FILE definitions for DFHDPFMB and DFHDPFMP resource to another group.
2. Add the DSNNAME attribute:
 - For DFHDPFMB, specify the name of the debugging profiles base data set.
For example:
DSNAME ==> CICSTS41.CICS.DFHDPFMB
 - For DFHDPFMP, specify the name of the debugging profiles base data set.
For example:
DSNAME ==> CICSTS41.CICS.DFHDPFMP

Alternatively, you can omit the DSNNAME attribute, and include a DD card in the CICS startup JCL. For example:

```
//DFHDPFMB DD DSN=CICSTS41.CICS.DFHDPFMB,DISP=SHR
//DFHDPFMP DD DSN=CICSTS41.CICS.DFHDPFMP,DISP=SHR
```

3. Install the FILE definitions.

Defining the debugging profiles data sets as remote files

Define remote files when you want to use profiles that are stored in a region that is connected using MRO or ISC.

CICS provides the following sample definitions:

```
*-----*
*   Define base file for Debugging Profiles (non-RLS remote)   *
*-----*
DEFINE FILE(DFHDPFMB) GROUP(DFHDPVSR)
DESCRIPTION(Debugging Profile Base File - VSAM Remote)
          REMOTESYSTEM(CICA)          REMOTENAME(DFHDPFMB)
*-----*
*   Define path file for Debugging Profiles (non-RLS remote)  *
*-----*
DEFINE FILE(DFHDPFMP) GROUP(DFHDPVSR)
DESCRIPTION(Debugging Profile Path File - VSAM Remote)
          REMOTESYSTEM(CICA)          REMOTENAME(DFHDPFMP)
```

Figure 30. Resource definitions for debugging profiles data sets defined as remote files

1. Copy the sample FILE definitions for DFHDPFMB and DFHDPFMP resource to another group.
2. Add the DSNAME attribute:
 - For DFHDPFMB, specify the name of the debugging profiles base data set.
For example:
DSNAME ==> CICSTS41.CICS.DFHDPFMB
 - For DFHDPFMP, specify the name of the debugging profiles base data set.
For example:
DSNAME ==> CICSTS41.CICS.DFHDPFMP

Alternatively, you can omit the DSNAME attribute, and include a DD card in the CICS startup JCL. For example:

```
//DFHDPFMB DD DSN=CICSTS41.CICS.DFHDPFMB,DISP=SHR
//DFHDPFMP DD DSN=CICSTS41.CICS.DFHDPFMP,DISP=SHR
```

3. Install the FILE definitions.

If you define remote files, you will need to install corresponding file definitions in the remote system.

Part 2. CICS system initialization

This section of the book describes how to define CICS system initialization parameters and how they are processed by CICS. It also describes a startup job stream you can use to start a CICS system.

- Chapter 16, “Specifying CICS system initialization parameters,” on page 133 describes the CICS system initialization parameters.
- Chapter 17, “Processing system initialization parameters,” on page 303 describes the use of the PARM parameter, the SYSIN data set, and the system console for supplying system initialization parameters, and how these are processed by CICS.
- Chapter 19, “CICS startup,” on page 319 describes a sample startup job stream, and a sample procedure for use as a started task.

Chapter 16. Specifying CICS system initialization parameters

This section describes the CICS system initialization parameters, which you can use to modify CICS system attributes when you start your CICS regions. It gives the syntax and a detailed description of each system initialization parameter, and describes the methods that you can use to define the parameters to CICS.

The information defined by system initialization parameters can be grouped into three categories:

1. Information used to initialize and control CICS system functions (for example, information such as the dynamic storage area limits and the region exit time interval)
2. Module suffixes used to load your own versions of CICS control tables (for example, DFHMCTxx)
3. Special information used to control the initialization process

The primary method of providing system initialization parameters is with a system initialization table (SIT). The parameters of the SIT, which you assemble as a load table, supply the system initialization program with most of the information necessary to initialize the system to suit your unique environment. You can generate more than one SIT, and at the time of system initialization select the one that is appropriate to your needs.

You can also specify other system initialization parameters, which cannot be coded in the SIT. You can specify which SIT you want, and other system initialization parameters (with a few exceptions), in any of three ways:

1. In the PARM parameter of the EXEC PGM=DFHSIP statement
2. In the SYSIN data set defined in the startup job stream
3. Through the system operator's console

You can also use these methods of input to the system initialization process to override most of the system initialization parameters assembled in the SIT.

There are some system initialization parameters that cannot be coded in the SIT; other parameters cannot be specified in any other way. Finally, some system initialization parameters cannot be specified through the system operator's console. For more information, see:

- "System initialization parameters that cannot be defined with the DFHSIT macro" on page 267
- "System initialization parameters that can be defined only with the DFHSIT macro" on page 265
- "System initialization parameters that cannot be defined through the system operator's console" on page 278

When you upgrade to a new CICS release

When you upgrade to a new CICS release, if you have existing system initialization tables, you must modify them. Remove all obsolete parameters, and specify the required values for new or changed parameters if you want to run with

other than the defaults. When you have made the necessary changes, reassemble the tables using the CICS Transaction Server for z/OS, Version 4 Release 1 macro libraries.

If you have system initialization parameters defined in CICS start-up procedures, you must modify these also.

To avoid generating specific system initialization tables for each CICS region, a simple solution is to let CICS load the default, unsuffixed table (DFHSIT) at start-up, and supply the system initialization parameters for each region in a SYSIN data set. For more information about the source of the default system initialization table, see “The default system initialization table” on page 293.

This chapter describes:

- “Specifying DFHSIT macro parameters” on page 137
- “The default system initialization table” on page 293
- “The system initialization parameter descriptions and summary” on page 139
- “Assembling the SIT” on page 300
- “Selecting versions of CICS programs and tables” on page 301

System initialization parameters for open TCBs

The open transaction environment (OTE) is an environment where CICS application code can use non-CICS services (facilities outside the scope of the CICS API) inside the CICS address space, without interference with other transactions.

Applications that exploit the open transaction environment run on their own open TCB, rather than on the QR TCB. Unlike the QR TCB, CICS does not perform subdispatching on an open TCB. If the application running on an open TCB calls a non-CICS service that blocks the TCB, the TCB blocking does not affect other CICS tasks.

TCB modes

There are a number of open TCB *modes*. Each mode has a 2-character identifier to indicate its specific purpose, and is handled by CICS in a different way:

J8 mode TCBs and J9 mode TCBs

Both these TCBs are used to run Java programs under a Java Virtual Machine (JVM). The JVM is created on the TCB. J8 TCBs are used for JVMs in CICS key, and J9 mode TCBs are used for JVMs in user key. The priority of the J8 and J9 mode TCBs is set lower than that of the main CICS QR TCB to ensure that the J8 and J9 TCB activity does not affect the main CICS workload that is being processed on the CICS QR TCB. *Java Applications in CICS* has more information about how CICS manages JVMs and their TCBs.

L8 mode TCBs and L9 mode TCBs

Both these TCBs are used to run OPENAPI programs; that is, those defined as OPENAPI by their PROGRAM resource definition.

- L8 mode TCBs are used for CICSKEY OPENAPI application programs.
- L9 mode TCBs are used for USERKEY OPENAPI application programs.

L8 mode TCBs

These TCBs are also used when programs need access to a resource

manager through a task-related user exit (TRUE) enabled using the OPENAPI option on the **ENABLE PROGRAM** command.

When CICS is connected to DB2, the CICS DB2 task-related user exit operates in OPENAPI mode; it is an open API TRUE. In this situation, the CICS DB2 attachment facility uses L8 TCBS for DB2 request processing. The *CICS DB2 Guide* describes how CICS uses open TCBS as thread TCBS for the CICS DB2 attachment facility and explains what your CICS DB2 application programs must do to gain performance benefits by continuing to run on the L8 mode TCB after the DB2 request has been completed.

L8 mode TCBS are also used by CICS itself, because CICS uses OPENAPI CICSKEY programs that run on L8 TCBS:

- When accessing document templates and HTTP static responses that are stored on the z/OS UNIX file system.
- When processing Web service requests and parsing XML.

SP mode TCB and S8 mode TCBS

These TCBS are used by CICS to manage SSL connections and requests to LDAP using the DFHDDAPX XPI interface. The S8 TCBS run in a single enclave, which is owned by the SP TCB and also contains the SSL cache. An S8 TCB is allocated to a task from the SSL pool, but is locked only for the period of time that it takes to perform functions such as an SSL handshake or an LDAP request. After this function is complete, the TCB is released back into the SSL pool to be reused.

In UNIX System Services, the **MAXTHREADS** and **MAXTHREADTASKS** parameters can be used to restrict the number of pthreads that a USS process can own. Each SSL TCB requires a pthread and an MVS task. You must therefore ensure that the values of these USS parameters exceed the **MAXSSLTCBS** system initialization parameter. If you do not set a large enough value for **MAXTHREADS** or **MAXTHREADTASKS** and CICS reaches one of these limits while attempting to attach an SSL TCB, CICS issues error message DFHDS0002 severe error code X'0137' from DFHDSIT.

TP mode TCB and T8 mode TCBS

These TCBS are used by a JVM server to perform system processing. The JVM server provides a mechanism for CICS to use the same JVM for multiple tasks concurrently. The TP mode TCB owns the Language Environment enclave and the pool of T8 TCBS. There is one TP TCB for each JVM server that is running in the CICS region. Each JVM server can have up to 256 T8 TCBS. A T8 TCB is allocated to a task from the THRD pool of the appropriate JVM server, but is locked only for the period of time that it takes to perform the system processing. T8 TCBS are not shared between JVM servers.

Each T8 TCB requires a pthread and an MVS task. The maximum number of T8 TCBS that is allowed for the CICS region is 1024. In UNIX System Services, you can use the **MAXTHREADS** and **MAXTHREADTASKS** parameters to restrict the number of pthreads that a USS process can own. You must therefore ensure that the values of these USS parameters exceed the maximum number of T8 TCBS. If you do not set a large enough value for **MAXTHREADS** or **MAXTHREADTASKS** and CICS reaches one of these limits while attempting to attach a T8 TCB, CICS issues error message DFHDS0002 severe error code X'0137' from DFHDSIT.

X8 mode TCBS and X9 mode TCBS

Both these TCBS are used to run C and C++ programs compiled with the XPLINK option. X8 TCBS are used for programs in CICS key, and X9 mode

TCBs are used for programs in user key. Each instance of an XPLink program uses one X8 or X9 TCB. The *CICS Application Programming Guide* has more information about using XPLink.

A CICS task is allowed as many J8, J9, X8, and X9 TCBs as it requires, and these TCBs are kept only until the program finishes. However, each CICS task is allowed at most one L8 and one L9 TCB and it keeps an L8 and an L9 TCB from the time it is allocated to the end of the task. The TCBs then become free, and CICS can allocate them to another task or destroy them.

Open TCB pools

CICS manages open TCBs in *pools*. A pool contains open TCBs that are used for the same purpose; for example, there is a pool of J8 and J9 mode open TCBs that is known as the JVM pool.

The maximum number of TCBs allowed in each pool is usually specified by a MAXxxxTCBS parameter:

- The “MAXOPENTCBS” on page 199 parameter limits the number of TCBs in the pool of L8 and L9 mode open TCBs.
- The “MAXJVMTCBS” on page 199 parameter limits the number of TCBs in the pool of J8 and J9 mode open TCBs. It applies to the maximum total number of J8 and J9 mode TCBs in the JVM pool, and CICS decides how many of them will be J8 TCBs and how many will be J9 TCBs, according to the number of requests that specify each execution key.
- The “MAXSSLTCBS” on page 200 parameter limits the number of TCBs in the pool of S8 mode open TCBs.
- The THREADLIMIT attribute on the JVMSERVER resource limits the number of TCBs in the pool of T8 mode open TCBs. There is one pool for every JVM server in a CICS region.
- The “MAXXPTCBS” on page 200 parameter limits the number of TCBs in the pool of X8 and X9 mode open TCBs.

The minimum permitted value for any of the MAXxxxTCBS parameters is 1, meaning that CICS is always able to create at least one open TCB in each mode. CICS can create or attach open TCBs in each pool up to the limit set by the corresponding MAXxxxTCBS parameter.

At any one time, a pool can consist of some TCBs that are allocated to tasks and others that have been freed by applications and are available for reuse. For example, if the maximum number of open TCBs for JVMs (J8 or J9 mode) is set to 10, at a particular time the pool might consist of 5 TCBs, not all of which are allocated.

How CICS attaches a TCB

When an application makes a request that involves the use of an open TCB, CICS first tries to find a suitable TCB that is available for reuse in the appropriate pool of open TCBs. CICS can match a request with an available TCB of the correct mode only if the TCB has matching attributes. For example, in the case of a request for a J8 or J9 mode TCB, a free JVM TCB can be allocated only if the JVM profile names also match.

CICS attaches a new TCB if it cannot find a suitable match with a free TCB, provided that the MAXxxx TCBS limit for the pool has not been reached. For the

JVM pool, where the creation of a JVM can involve a large amount of MVS storage, an additional safeguard is provided by the CICS storage monitor for MVS storage, which prevents the creation of new JVMs if MVS storage is severely constrained.

If CICS cannot find a suitable match, and the MAXxxxxTCBS limit for the pool has been reached, CICS might fulfil the request by destroying a free TCB that has the wrong attributes, and replacing it with a TCB that has the right attributes. This technique is called stealing. Stealing can be costly on performance, depending on the type of open TCB, so CICS avoids it where it makes sense to do so. For L8 mode TCBs, used by task-related user exits enabled with OPENAPI, the cost of stealing a TCB is low, so CICS always steals a TCB if it receives a request for which it cannot find a suitable match with a free TCB or attach a new TCB. However, for J8 and J9 mode (JVM) TCBs, the cost of TCB stealing is high, because CICS needs to destroy and reinitialize the JVM as well as the TCB; so CICS has a selection mechanism to decide if stealing the TCB is worthwhile, or if the request will be made to wait. CICS maintains statistics of excess TCB management and TCB stealing activities.

How CICS handles storage

To minimize the impact on storage, CICS attempts to balance the number of open TCBs in each pool against current needs by reducing the number of free TCBs. If CICS finds that there are free TCBs in a pool, it gradually removes the excess number by detaching them, thereby freeing the resources used by the excess TCBs.

When specifying any of the MAXxxxxTCBS parameters, take into account TCB storage requirements. All TCBs use real storage, and virtual storage below 16 MB, so the number of open TCBs that a CICS region can support is restricted by the amount of storage available both above and below 16 MB. JVMs, which run on J8 and J9 mode TCBs, use a large amount of storage above 16 MB in addition to the cost of the TCB. The *CICS Performance Guide* has guidance for calculating the number of JVMs that your CICS region can support.

Specifying DFHSIT macro parameters

You can specify most system initialization parameters in a DFHSIT macro.

About this task

Procedure

1. Code the following macro parameter first:

TYPE={CSECT|DSECT}

This parameter specifies the type of SIT to be generated.

CSECT

A regular control section that is normally used.

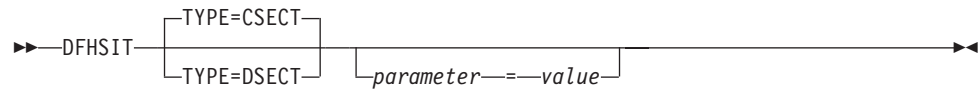
DSECT

A dummy control section.

For example:

```
DFHSIT TYPE=CSECT, *
```

2. Use the following syntax to add system initialization parameters to the macro:



where *parameter* is the system initialization parameter that you want to code, and *value* is the appropriate parameter value. You should code the parameters and keywords in uppercase, except for parameters where case is important. For example, any parameter that specifies the name of a z/OS UNIX directory is case sensitive. The system initialization parameters are described in detail in “The system initialization parameter descriptions and summary” on page 139.

3. Terminate your macro parameters with the following statement:
END DFHSITBA

Example

Defining CICS resource table and module keywords

About this task

The following table shows the system initialization keywords for those CICS resources that:

- Have a suffix option, or
- Result in a dummy program or table if you code resource=NO, or
- You can COLD start individually

Table 11. Summary of resources with a suffix, a dummy load module, or a COLD option

| DFHSIT keyword 1 | Default 2 | Suffix 3 | Dummy 4 | COLD start 5 |
|------------------|-----------|----------|---------|--------------|
| BMS | FULL | - | - | COLD |
| CLT | - | xx | - | - |
| DIP | NO | - | program | - |
| FCT | YES | xx | - | - |
| ICP | - | - | - | COLD |
| MCT | NO | xx | 6 | - |
| PDIR | NO | xx | - | - |
| PLTPI | NO | xx | - | - |
| PLTSD | NO | xx | - | - |
| RST | NO | xx | - | - |
| SRT | YES | xx | - | - |
| TCT | YES | xx | table | - |
| TS | - | - | - | COLD |
| TST | NO | xx | - | - |
| XLT | NO | xx | - | - |

Note:

1. When the DFHSIT keyword is specified with more than one value, these values must be enclosed within parentheses: for example, BMS=(FULL,COLD).
2. The **Default** column indicates the default value for the keyword in the DFHSIT macro.

For keywords with the suffix option, if you code YES in the SIT, an unsuffixed version of the table or program is loaded. For example, TCT=YES results in a

table called DFHTCT being loaded. You can also select an unsuffixed module or table at CICS startup by specifying *keyword=*, or *keyword=YES*. For example, if you code:

```
FCT=, or FCT=YES
```

blanks are appended to DFHFCT, and these unsuffixed names are used during initialization.

The result of specifying *keyword=*, as a system initialization parameter through PARM, SYSIN, or CONSOLE is not necessarily the same as in the DFHSIT macro. For example, TST=, (or omitted altogether) when coding the DFHSIT macro is taken to mean TST=NO, but TST=, through any of the other three methods is the same as TST=YES.

3. The **Suffix** column indicates whether you can code a suffix. (xx indicates that a suffix can be coded.)

A suffix can be any 1 or 2 characters, but you must not use DY, and you cannot use NO as a suffix.

If you code a suffix, a table or program with that suffix appended to the standard name is loaded.

4. The **Dummy** column indicates whether a dummy version of the program or table is loaded if you code NO. In some cases, coding NO for the operand associated with the **table** results in a dummy **program**. For more information about the effect of this option, see “Selecting versions of CICS programs and tables” on page 301.

5. The **COLD start** column indicates whether the resource can be forced to start COLD. (COLD indicates that the resource can be cold started individually). TST and cold start ensure that you cold start temporary storage or the whole system if you make any change to the TST.

If COLD is coded, it can be overridden only by coding START=(...,ALL) as a system initialization parameter. For more information about this option, see page ALL.

For more information about CICS table and program selection, see “Selecting versions of CICS programs and tables” on page 301.

6. If you code MCT=NO, the CICS monitoring domain builds dynamically a default monitoring control table. This ensures that default monitoring control table entries are always available for use when monitoring is on and a monitoring class is active.

The system initialization parameter descriptions and summary

There are a number of ways that system initialization parameters can be defined to CICS.

You can define system initialization parameters in any of the following ways:

1. In a DFHSIT macro
2. In a PARM parameter on the EXEC PGM=DFHSIP statement
3. In the SYSIN data set of the CICS startup job stream
4. Through the system console. For parameters that require mixed case values, type / on an SDSF command line to open the System Command Extension facility. Use single quotes around the parameter and value that you want to specify to preserve the mixed case. The command is still echoed in uppercase in the job log, but SDSF processes the value in mixed case.

Default values are **underscored**; for example, ADI=30. This notation applies to the SIT macro parameters only.

The table lists the parameters and default values, along with whether the parameter is specified in a DFHSIT macro, in a PARM parameter, in the SYSIN data set, or through the system console.

Table 12. System initialization parameters with override options and default settings

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|-----------|------|-------|----------------|--------|--------------------|--|
| ADI | YES | YES | YES | YES | 30 | XRF(B) - Alternate delay interval |
| AIBRIDGE | YES | YES | YES | YES | AUTO | Bridge Autoinstall URM |
| AICONS | YES | YES | YES | YES | NO | No autoinstall for MVS CONSOLES |
| AIEXIT | YES | YES | YES | YES | DFHZATDX | Autoinstall user program name |
| AILDELAY | YES | YES | YES | YES | 0 | Delete delay period for AI terminals |
| AIQMAX | YES | YES | YES | YES | 100 | Maximum number of terminals queued for AI |
| AIRDELAY | YES | YES | YES | YES | 700 | Restart delay period for AI terminals |
| AKPFREQ | YES | YES | YES | YES | 4000 | Activity keypoint frequency |
| APPLID | YES | YES | YES | YES | DBDCCICS | VTAM APPL identifier |
| AUTCONN | YES | YES | YES | YES | 0 | Autoconnect delay |
| AUTODST | YES | YES | YES | YES | NO | Language Environment [®] automatic storage tuning |
| AUTORESET | YES | YES | YES | YES | NO | Time-of-day synchronization |
| AUXTR | YES | YES | YES | YES | OFF | Auxiliary trace option |
| AUXTRSW | YES | YES | YES | YES | NO | Auxiliary trace autoswitch facility |
| BMS | YES | YES | YES | YES | FULL, UNALIGN, DDS | Basic Mapping Support options |
| BRMAXKEEP | YES | YES | YES | YES | 86400 | Bridge Max Keeptime |
| CDSASZE | YES | YES | YES | NO | 0 | The size of CDSA |
| CHKSTRM | YES | YES | YES | NO | NONE | Activation of terminal storage-violation checking |
| CHKSTSK | YES | YES | YES | NO | NONE | Activation of task storage-violation checking |
| CICSSVC | YES | YES | YES | YES | 216 | The CICS SVC number |
| CILOCK | YES | YES | YES | YES | NO | Do not keep CI lock after read update |
| CLSDSTP | YES | YES | YES | YES | NOTIFY | Notification for ISSUE PASS command |
| CLT | YES | YES | YES | YES | No default | The command list table option or suffix |
| CMDPROT | YES | YES | YES | YES | YES | Exec storage command checking |
| CMDSEC | YES | YES | NO | YES | ASIS | API command security checking |
| CONFDATA | YES | YES | NO | YES | SHOW | Show confidential data in dump and trace |
| CONFTXT | YES | YES | NO | YES | NO | Do not prevent VTAM tracing user data |
| CPSMCONN | YES | YES | YES | YES | NO | Do not connect to CPSM |

Table 12. System initialization parameters with override options and default settings (continued)

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|-------------|------|-------|----------------|--------|---------------|---|
| CRLPROFILE | YES | YES | YES | YES | No default | Name of profile that allows CICS to access certificate revocation lists |
| CSDACC | YES | YES | YES | YES | READWRITE | CSD access |
| CSDBKUP | YES | YES | YES | YES | STATIC | Backup type of CSD (STATIC or DYNAMIC) |
| CSDBUFND | YES | YES | YES | YES | No default | Number of data buffers for the CSD |
| CSDBUFNI | YES | YES | YES | YES | No default | Number of index buffers for the CSD |
| CSDDISP | YES | YES | YES | YES | No default | CSD disposition for dynamic allocation |
| CSDDSN | YES | YES | YES | YES | No default | CSD data set name for dynamic allocation |
| CSDFRLOG | YES | YES | YES | YES | NO | Journal ID for CSD forward recovery |
| CSDINTEG | YES | YES | YES | YES | UNCOMMITTED | Read integrity = uncommitted |
| CSDJID | YES | YES | YES | YES | NO | Journal ID for CSD auto journaling |
| CSDLRNO | YES | YES | YES | YES | 1 | The VSAM LSR pool number for the CSD |
| CSDRECOV | YES | YES | YES | YES | NONE | CSD recoverable file option |
| CSDRLS | YES | YES | YES | YES | NO | Use traditional VSAM |
| CSDSTRNO | YES | YES | YES | YES | 6 | CSD number of strings |
| CWAKEY | YES | YES | YES | YES | USER | CWA storage key |
| DAE | YES | YES | YES | YES | NO | SDUMPS will not be suppressed by DAE |
| DATFORM | YES | YES | YES | YES | MMDDYY | CSA date format |
| DB2CONN | YES | YES | YES | YES | NO | Do not connect to DB2 at CICS startup |
| DBCTLCON | YES | YES | YES | YES | NO | Do not connect to DBCTL at CICS start |
| DEBUGTOOL | YES | YES | YES | YES | NO | No Debugging Tool access |
| DFLTUSER | YES | YES | NO | YES | CICSUSER | Default user |
| DIP | YES | YES | YES | YES | NO | Batch data interchange program |
| DISMACP | YES | YES | YES | YES | YES | Disable macro programs |
| DOCCODEPAGE | YES | YES | YES | YES | 037 | Default host code page |
| DSALIM | YES | YES | YES | YES | 5M | Upper limit of DSA below 16 MB line |
| DSHIPIDL | YES | YES | YES | YES | 020000 | Delete shipped idle time |
| DSHIPINT | YES | YES | YES | YES | 120000 | Delete shipped interval |
| DSRTPGM | YES | YES | YES | YES | NONE | Distributed routing program |
| DTRPGM | YES | YES | YES | YES | DFHDYP | Dynamic routing program |
| DTRTRAN | YES | YES | YES | YES | CRTX | Default dynamic transaction routing transid |
| DUMP | YES | YES | YES | YES | YES | Dump option |
| DUMPPDS | YES | YES | YES | YES | AUTO | CICS dump data set opening option |
| DUMPSW | YES | YES | YES | YES | NO | Dump data set autoswitch option |

Table 12. System initialization parameters with override options and default settings (continued)

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|---------------|------|-------|----------------|--------|-------------------|--|
| DURETRY | YES | YES | YES | YES | 30 | SDUMP total retry time (in seconds) |
| ECDSASZE | YES | YES | YES | NO | 0 | Size of the ECDSA |
| EDSALIM | YES | YES | YES | YES | 48M | Upper limit of DSA above 16 MB line |
| EJBROLEPREFIX | YES | YES | YES | YES | No default | EJB ROLE PREFIX |
| ENCRYPTION | YES | YES | YES | YES | STRONG | Level of encryption for SSL |
| EODI | YES | YES | YES | YES | E0 | End-of-data indicator for sequential devices |
| ERDSASZE | YES | YES | YES | NO | 0 | Size of the ERDSA |
| ESDSASZE | YES | YES | YES | NO | 0 | Size of the ESDSA |
| ESMEXITS | NO | NO | NO | YES | NOINSTLN | External security manager exits |
| EUDSASZE | YES | YES | YES | NO | 0 | Size of the EUDSA |
| FCT | YES | YES | YES | YES | NO | File control table option or suffix |
| FCQRONLY | YES | YES | YES | YES | YES | Threadsafe FC runs on QR TCB |
| FEPI | YES | YES | YES | YES | NO | Front-End Programming Interface |
| FLDSEP | YES | YES | YES | YES | ' ' (four blanks) | End-of-field separator characters |
| FLDSTRT | YES | YES | YES | YES | ' ' (one blank) | Field start character for built-in function |
| FORCEQR | YES | YES | YES | YES | NO | Do not force QR for threadsafe programs |
| FSSTAFF | YES | YES | YES | YES | NO | Function-shipped START affinity option |
| FTIMEOUT | YES | YES | YES | YES | 30 | File timeout 30 seconds |
| GMTEXT | YES | YES | YES | YES | 'WELCOME TO CICS' | Good-morning message text |
| GMTRAN | YES | YES | YES | YES | CSGM | Initial transaction |
| GNTRAN | YES | YES | YES | YES | NO | Signoff transaction |
| GRNAME | YES | YES | YES | YES | No default | Generic resource name for CICS TORs |
| GRPLIST | YES | YES | YES | YES | DFHLIST | List name of CSD groups for startup |
| GTFTR | YES | YES | YES | YES | OFF | GTF trace option |
| HPO | NO | NO | NO | YES | NO | VTAM High Performance Option (HPO) |
| ICP | YES | YES | YES | YES | COLD | Interval control program start option |
| ICV | YES | YES | YES | YES | 1000 | Region exit interval (milliseconds) |
| ICVR | YES | YES | YES | YES | 5000 | Runaway task interval (milliseconds) |
| ICVTSD | YES | YES | YES | YES | 500 | Terminal scan delay interval (") |
| IIOPLISTENER | YES | YES | YES | YES | YES | Whether the CICS region is to function as an IIOP listener region |
| INFOCENTER | YES | YES | YES | YES | No default | The server name of where the CICS Information Center is installed and the port number that it uses to run in server mode |
| INITPARM | YES | YES | YES | YES | No default | Initialization parameters for programs |

Table 12. System initialization parameters with override options and default settings (continued)

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|---------------|------|-------|----------------|--------|--|--|
| INTTR | YES | YES | YES | YES | ON | CICS internal trace option |
| IRCSTRT | YES | YES | YES | YES | NO | Interregion communication start |
| ISC | YES | YES | YES | YES | NO | Intersystem communication option |
| JESDI | YES | YES | YES | YES | 30 | JES delay interval for XRF alternate |
| JVMCCSIZE | YES | YES | YES | YES | 24M | Shared Class Cache size |
| JVMCCSTART | YES | YES | YES | YES | AUTO | Start Shared Class Cache when needed |
| JVMxxxxTRACE | YES | YES | YES | NO | No default | JVM trace, specifies level of trace required |
| JVMPROFILEDIR | YES | YES | YES | YES | /usr/lpp/cicsts /cicsts41/JVMProfiles | JVM profile directory |
| KEYRING | YES | YES | YES | YES | No default | Key ring to be used by SSL support |
| LGDFINT | YES | YES | YES | YES | 5 | Log defer interval in Log Manager |
| LGNMSG | YES | YES | YES | YES | NO | Extract VTAM logon data |
| LLACOPY | YES | YES | YES | YES | YES | Use MVS LLACOPY support |
| LOCALCCSID | YES | YES | YES | YES | 037 | The default CCSID for the local region |
| LPA | YES | YES | YES | YES | NO | Use-LPA option for CICS/user modules |
| MAXJVMTCBS | YES | YES | YES | YES | 5 | Maximum number of JVM open TCBs |
| MAXOPENTCBS | YES | YES | YES | YES | 12 | Maximum number of open TCBs |
| MAXSOCKETS | YES | YES | YES | YES | 65535 | Maximum number of IP sockets |
| MAXSSLTCBS | YES | YES | YES | YES | 8 | Limit on number of SSL TCBs |
| MAXXPTCBS | YES | YES | YES | YES | 5 | Limit on number of XP TCBs |
| MCT | YES | YES | YES | YES | NO | Monitoring control table option or suffix |
| MN | YES | YES | YES | YES | OFF | CICS monitoring option |
| MNCONV | YES | YES | YES | YES | NO | Monitoring converse recording option |
| MNEXC | YES | YES | YES | YES | OFF | Monitoring exception class option |
| MNFREQ | YES | YES | YES | YES | 0 | Monitoring frequency period |
| MNIDN | YES | YES | YES | YES | OFF | Monitoring identity class option |
| MNPER | YES | YES | YES | YES | OFF | Monitoring performance class option |
| MNRES | YES | YES | YES | YES | OFF | Monitoring resource class option |
| MNSYNC | YES | YES | YES | YES | NO | Monitoring syncpoint recording option |
| MNTIME | YES | YES | YES | YES | GMT | Monitoring timestamp (GMT or LOCAL) |
| MQCONN | YES | YES | YES | YES | NO | Do not connect to MQ at startup |
| MROBTCH | YES | YES | YES | YES | 1 | Number of MRO requests to batch |
| MROFSE | YES | YES | YES | YES | NO | Extend lifetime of long-running mirror |

Table 12. System initialization parameters with override options and default settings (continued)

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|---------------|------|-------|----------------|--------|---------------|--|
| MROLRM | YES | YES | YES | YES | NO | Long-running mirror task option |
| MSGCASE | YES | YES | YES | YES | MIXED | CICS messages in mixed case |
| MSGLVL | YES | YES | YES | YES | 1 | System console MSG level option |
| MXT | YES | YES | YES | YES | 5 | Maximum number of tasks in CICS |
| NATLANG | YES | YES | YES | YES | E | List of national languages |
| NCPLDFT | YES | YES | YES | YES | DFHNC001 | Named counter default pool name |
| NEWSIT | YES | YES | YES | NO | NO | Load a specified SIT and enforce use of all system initialization parameters |
| NONRLSRECOVER | YES | YES | YES | YES | VSAMCAT | Select location of recovery options for non-RLS files |
| OFFSITE | YES | YES | YES | NO | NO | Restart in off-site recovery mode |
| OPERTIM | | | | | 120 | Write to operator timeout (seconds) |
| OPNDLIM | YES | YES | YES | YES | 10 | OPNDST/CLSDST request limit |
| PARMERR | YES | YES | YES | YES | INTERACT | System initialization parameter errors option |
| PDI | YES | YES | YES | YES | 30 | Primary delay interval - XRF active |
| PDIR | YES | YES | YES | YES | NO | DL/I PSB directory option or suffix |
| PGAICTLG | YES | YES | YES | YES | MODIFY | PG autoinstall catalog state |
| PGAIXIT | YES | YES | YES | YES | DFHPGADX | PG autoinstall exit program |
| PGAIPGM | YES | YES | YES | YES | INACTIVE | PG autoinstall state |
| PGCHAIN | YES | YES | YES | YES | No default | BMS CHAIN command |
| PGCOPY | YES | YES | YES | YES | No default | BMS COPY command |
| PGPURGE | YES | YES | YES | YES | No default | BMS PURGE command |
| PGRET | YES | YES | YES | YES | No default | BMS RETURN command |
| PLTPI | YES | YES | YES | YES | NO | Program list table PI option or suffix |
| PLTPISEC | YES | YES | NO | YES | NONE | No PLT security checks on PI programs |
| PLTPIUSR | YES | YES | NO | YES | No default | PLT PI userid = CICS region userid |
| PLTSD | YES | YES | YES | YES | NO | Program list table SD option or suffix |
| PRGDLAY | YES | YES | YES | YES | 0 | BMS purge delay interval |
| PRINT | YES | YES | YES | YES | NO | Print key option |
| PRTYAGE | YES | YES | YES | YES | 32768 | Dispatcher priority aging value |
| PRVMOD | YES | YES | YES | NO | name | Names of modules that are not to be used from the LPA |
| PSBCHK | YES | YES | NO | YES | NO | PSB resource checking required |
| PSDINT | YES | YES | YES | YES | 0 | Persistent session delay interval |
| PSTYPE | YES | YES | YES | YES | SNPS | VTAM single node persistent Sessions |
| PVDELAY | YES | YES | YES | YES | 30 | Timeout value for LUIT table |
| QUIESTIM | YES | YES | YES | YES | 240 | Timeout value for quiesce requests |
| RAMAX | YES | YES | YES | YES | 256 | Maximum I/O area for RECEIVE ANY |

Table 12. System initialization parameters with override options and default settings (continued)

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|-------------|------|-------|----------------|--------|---------------|---|
| RAPOOL | YES | YES | YES | YES | 50 | Maximum RECEIVE ANY request parm. lists |
| RDSASZE | YES | YES | YES | NO | 0 | Size of the RDSA |
| RENTPGM | YES | YES | YES | YES | PROTECT | Reentrant program write protection |
| RESP | YES | YES | YES | YES | FME | Logical unit response type |
| RESSEC | YES | YES | NO | YES | ASIS | Resource security check |
| RLS | YES | YES | YES | YES | NO | RLS option |
| RLSTOLSR | YES | YES | YES | YES | NO | RLS files in LSRPOOL build calculation |
| RMTRAN | YES | YES | YES | YES | CSGM | XRF alternate recovery transaction |
| RRMS | YES | YES | YES | YES | NO | Recoverable resource management services |
| RST | YES | YES | YES | YES | NO | Recovery service table (XRF-DBCTL) |
| RSTSIGNOFF | YES | YES | YES | YES | NOFORCE | XRF - Re-sign on after takeover |
| RSTSIGNTIME | YES | YES | YES | YES | 500 | XRF - sign off timeout value |
| RUWAPool | YES | YES | YES | YES | NO | Allocating storage pool for Language Environment |
| SDSASZE | YES | YES | YES | NO | 0 | Size of the SDSA |
| SDTRAN | YES | YES | YES | YES | CESD | Shutdown transaction |
| SEC | YES | YES | NO | YES | YES | External security manager option |
| SECPRFX | YES | YES | NO | YES | NO | Security prefix |
| SIT | YES | YES | YES | NO | No default | Suffix of the system initialization table that you want loading at the start of system initialization |
| SKRxxxx | YES | YES | YES | YES | No default | Single keystroke retrieval operation required |
| SNSCOPE | YES | YES | NO | YES | NONE | Multiple CICS sessions per user ID |
| SPCTR | YES | YES | YES | YES | (1,2) | Levels of special tracing required for CICS as a whole |
| SPCTRxx | YES | YES | YES | NO | (1,2) | The level of special tracing for a particular CICS component used by a transaction, terminal, or both |
| SPOOL | YES | YES | YES | YES | NO | System spooling interface option |
| SRBSVC | YES | YES | YES | YES | 215 | HPO Type 6 SVC number |
| SRT | YES | YES | YES | YES | 1\$ | System recovery table option or suffix |
| SRVERCP | YES | YES | YES | YES | 037 | Default server code page to be used by the DFHCNV data conversion table (if the SRVERCP parameter in the DFHCNV macro is set to SYSDEF) |
| SSLCACHE | YES | YES | YES | YES | CICS | SSL session ID caching |
| SSLDELAY | YES | YES | YES | YES | 600 | SSL timeout value |
| START | YES | YES | YES | YES | AUTO | CICS system initialization option |

Table 12. System initialization parameters with override options and default settings (continued)

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|-----------|------|-------|----------------|--------|---------------|---|
| STARTER | NO | NO | NO | YES | YES | Starter (\$ and #) suffixes option Note: The default is NO but the parameter must be set to YES here to enable the SIT to assemble correctly. |
| STATEOD | YES | YES | YES | YES | 0 | Statistics end-of-day time |
| STATINT | YES | YES | YES | YES | 030000 | Statistics interval time |
| STATRCD | YES | YES | YES | YES | OFF | Statistics recording status |
| STGPROT | YES | YES | YES | YES | NO | Storage protection facility |
| STGRCVY | YES | YES | YES | YES | NO | Storage recovery option |
| STNTR | YES | YES | YES | YES | 1 | Level of standard tracing required for CICS as a whole |
| STNTRxx | YES | YES | YES | NO | 1 | Level of standard tracing you require for a particular CICS component |
| SUBTSKS | YES | YES | YES | YES | 0 | Number of concurrent mode TCBs |
| SUFFIX | NO | NO | NO | YES | \$\$ | Suffix of this SIT |
| SYDUMAX | YES | YES | YES | YES | 999 | Number of SYSDUMPS to be taken |
| SYSIDNT | YES | YES | YES | YES | CICS | Local system identifier |
| SYSTR | YES | YES | YES | YES | ON | Master system trace flag |
| TAKEOVR | YES | YES | YES | YES | MANUAL | XRF alternate takeover option |
| TBEXITS | YES | YES | YES | YES | No default | Backout exit programs |
| TCP | YES | YES | YES | YES | YES | Terminal control program option or suffix |
| TCPIP | YES | YES | YES | YES | NO | TCP/IP support |
| TCSACTN | YES | YES | YES | YES | NONE | TC Shutdown action |
| TCSWAIT | YES | YES | YES | YES | 4 | TC Shutdown wait |
| TCT | YES | YES | YES | YES | NO | Terminal control table option or suffix |
| TCTUAKEY | YES | YES | YES | YES | USER | TCT user area storage key |
| TCTUALOC | YES | YES | YES | YES | BELOW | TCT user area below 16MB |
| TD | YES | YES | YES | YES | (3,3) | Transient data buffers and strings |
| TDINTRA | YES | YES | YES | YES | NOEMPTY | Initial state of transient data queues |
| TDSUBTASK | YES | YES | YES | YES | OFF | No TD subtasking |
| TRANISO | YES | YES | YES | YES | NO | Transaction Isolation |
| TRAP | YES | YES | YES | YES | OFF | F.E. global trap exit option |
| TRDUMAX | YES | YES | YES | YES | 999 | Number of TRANDUMPS to be taken |
| TRTABSZ | YES | YES | YES | YES | 4096 | Internal trace table size in 1K bytes |
| TRTRANSZ | YES | YES | YES | YES | 16 | Transaction dump trace table size |
| TRTRANTY | YES | YES | YES | YES | TRAN | Transaction dump trace option |
| TS | YES | YES | YES | YES | (3,3) | Temporary storage buffers and strings |
| TST | YES | YES | YES | YES | NO | Temporary storage table option or suffix |
| UDSASZE | YES | YES | YES | NO | 0 | Size of the UDSA |

Table 12. System initialization parameters with override options and default settings (continued)

| Parameter | PARM | SYSIN | System console | DFHSIT | Default value | Description |
|-----------|------|-------|----------------|--------|--------------------------|--|
| UOWNETQL | YES | YES | YES | YES | No default | Qualifier for NETUOWID |
| USERTR | YES | YES | YES | YES | ON | Master user trace flag |
| USRDELAY | YES | YES | YES | YES | 30 | Timeout value for user directory entries |
| USSHOME | YES | YES | YES | YES | /usr/lpp/cicsts/cicsts41 | The name and path of the root directory for CICS files on z/OS UNIX |
| VTAM | YES | YES | YES | YES | YES | VTAM access method option |
| VTPREFIX | YES | YES | YES | YES | \ | Client virtual terminal prefix |
| WEBDELAY | YES | YES | YES | YES | (5,60) | Web timer values |
| WRKAREA | YES | YES | YES | YES | 512 | Common work area (CWA) size in bytes |
| XAPPC | YES | YES | NO | YES | NO | RACF class APPCLU required |
| XCFGROUP | YES | YES | YES | YES | DFHIR000 | XCF group to use for MRO communications |
| XCMD | YES | YES | NO | YES | YES | SPI use default name for RACF check |
| XDCT | YES | YES | NO | YES | YES | Security check for transient data queues |
| XDB2 | YES | YES | YES | YES | NO | Security check for DB2ENTRY resources |
| XEJB | YES | YES | NO | YES | YES | EJB security required |
| XFCT | YES | YES | NO | YES | YES | Security check for files |
| XHFS | YES | YES | NO | YES | YES | Security check for z/OS UNIX files |
| XJCT | YES | YES | NO | YES | YES | Security check for journals |
| XLT | YES | YES | YES | YES | NO | Transaction list table option or suffix |
| XPCT | YES | YES | NO | YES | YES | Security check for started transactions |
| XPPT | YES | YES | NO | YES | YES | Security check for programs |
| XPSB | YES | YES | NO | YES | YES | Security check for DL/I PSBs |
| XRES | YES | YES | NO | YES | YES | For resources subject to XRES security, checks use the default name for the RACF check. For a list of resources subject to XRES security checks, see the <i>CICS RACF Security Guide</i> . |
| XRF | YES | YES | YES | YES | NO | Extended recovery feature (XRF) option |
| XTRAN | YES | YES | NO | YES | YES | Security check for transaction-attach |
| XTST | YES | YES | NO | YES | YES | Security check for temporary storage queues |
| XUSER | YES | YES | NO | YES | YES | Surrogate user checking to be done |

ADI

The **ADI** parameter specifies the alternate delay interval in seconds for an alternate CICS region when you are running CICS with XRF.

ADI={30|number}

The minimum delay that you can specify is 5 seconds. This is the time that must elapse between the (apparent) loss of the surveillance signal in the active CICS region, and any reaction by the alternate CICS region. The corresponding parameter for the active is PDI. ADI and PDI need not have the same value.

Note: You must give careful consideration to the values you specify for the parameters ADI and JESDI so that they do not conflict with your installation's policy on PR/SM™ RESETTIME and the XCF INTERVAL and OPNOTIFY intervals. You should ensure that the sum of the interval you specify for ADI plus JESDI exceeds the interval specified by the XCF INTERVAL and the PR/SM policy interval RESETTIME.

AIBRIDGE

The **AIBRIDGE** parameter specifies whether the autoinstall user replaceable module (URM) is to be called when creating bridge facilities (virtual terminals) used by the 3270 bridge mechanism.

AIBRIDGE={AUTO|YES}

Valid values are as follows:

AUTO

This is the default, and specifies that bridge facilities are defined automatically by CICS. The autoinstall URM is not called.

YES Specifies that the autoinstall URM is to be called for all new bridge facilities.

See the *CICS Customization Guide* for information about writing an autoinstall user replaceable module.

AICONS

The **AICONS** parameter specifies whether you want autoinstall support for consoles.

AICONS={NO|YES|AUTO}

You can also set the state of autoinstall support for consoles dynamically using the **SET AUTOINSTALL** command. Valid values for this parameter are as follows:

NO This is the default, and specifies that the CICS regions does not support autoinstall for consoles.

YES Specifies that console autoinstall is active and CICS is to call the autoinstall control program, as part of the autoinstall process, when an undefined console issues an MVS MODIFY command to CICS.

AUTO

Specifies that console autoinstall is active but CICS is not to call the autoinstall control program when an undefined console issues an MVS MODIFY command to CICS. CICS is to autoinstall undefined consoles automatically without any input from the autoinstall control program. The 4-character termid required for the console's TCT entry is generated by CICS, beginning with a ~ (logical not) symbol.

See the *CICS Customization Guide* for information about writing an autoinstall control program that supports consoles.

AIEXIT

The **AIEXIT** parameter specifies the name of the autoinstall user-replaceable program that you want CICS to use when autoinstalling local VTAM terminals, APPC connections, virtual terminals, and shipped terminals and connections.

AIEXIT={DFHZATDX|DFHZATDY|name}

Autoinstall is the process of installing resource definitions automatically, using VTAM logon or BIND data, model definitions, and an autoinstall program.

You can specify only one user-replaceable program on the **AIEXIT** parameter. Which of the CICS-supplied programs (or customized versions thereof) that you choose depends on what combination of resources you need to autoinstall.

For background information about autoinstall, see the *CICS Resource Definition Guide*. Valid values for this parameter are as follows:

DFHZATDX

A CICS-supplied autoinstall user program. This value is the default. It installs definitions for:

- Locally-attached VTAM terminals
- Virtual terminals used by the CICS Client products
- Remote shipped terminals
- Remote shipped connections

DFHZATDY

A CICS-supplied autoinstall user program. It installs definitions for:

- Locally-attached VTAM terminals
- Local APPC connections
- Virtual terminals used by the CICS Client products
- Remote shipped terminals
- Remote shipped connections

name The name of your own customized autoinstall program, which can be based on one of the supplied sample programs. For programming information about writing your own autoinstall program, see the *CICS Customization Guide*.

AILDELAY

The **AILDELAY** parameter specifies the delay period that elapses after all sessions between CICS and an autoinstalled terminal, APPC device, or APPC system are ended, before the terminal or connection entry is deleted.

AILDELAY={0|hhmmss}

All sessions are ended when the terminal or system logs off, or when a transaction disconnects it from CICS.

The **AILDELAY** parameter does not apply to the following types of autoinstalled APPC connection, which are not deleted:

- Sync level 2-capable connections (for example, CICS-to-CICS connections)
- Sync level 1-only, limited resource connections installed on a CICS that is a member of a generic resource group

Valid values for this parameter are as follows:

hhmmss

A 1 to 6-digit number. The default is 0. For non-LU6.2 terminals and LU6.2 single-session connections installed by a CINIT, 0 means that the terminal entry is deleted as soon as the session is ended. For LU6.2 connections installed by a BIND, 0 means that the connection is deleted as soon as all sessions are ended, but is reusable if a new BIND occurs before the deletion starts.

If you leave out the leading zeros, they are supplied (for example, 123 becomes 000123—that is, 1 minute 23 seconds).

AIQMAX

The **AIQMAX** parameter specifies the maximum number of VTAM terminals and APPC connections that can be queued concurrently for autoinstall.

AIQMAX={100|number}

The value for this parameter must be a number in the range 0 through 999. The default is 100. A zero value disables the autoinstall function.

Specify a number that is large enough to allow for both APPC connections and terminals.

Note: This value does not limit the total number of terminals that can be autoinstalled. If you have a large number of terminals autoinstalled, shutdown can fail due to the **MXT** system initialization parameter being reached or CICS becoming short on storage. For information about preventing this possible cause of shutdown failure, see the *CICS Performance Guide*.

AIRDELAY

The **AIRDELAY** parameter specifies the delay period that elapses after an emergency restart before autoinstalled terminal and APPC connection entries that are not in session are deleted.

AIRDELAY={700|hmmss}

The **AIRDELAY** parameter also applies when you issue a **CEMT SET VTAM OPEN** command after a VTAM abend and **PSTYPE=MNPS** is coded. This causes autoinstalled resources to be deleted, if the session was not restored and has not been used since the ACB was opened.

The **AIRDELAY** parameter does not apply to the following types of autoinstalled APPC connection, which are always written to the CICS global catalog and recovered during a warm or emergency start:

- Sync level 2-capable connections (for example, CICS-to-CICS connections)
- Sync level 1-capable, limited resource connections installed on a CICS that is a member of a generic resource group

hmmss

A 1-to 6-digit number. If you leave out the leading zeros, they are supplied. The default is 700, meaning a delay of 7 minutes. A value of 0 means that autoinstalled definitions are not written to the global catalog and therefore are not restored at an emergency restart.

For guidance about the performance implications of setting different **AIRDELAY** values, see the *CICS Customization Guide*.

Note: If you are running CICS with XRF, set the same value on the **AIRDELAY** parameter for both the active and the alternate CICS regions. It is particularly important, if you want autoinstall sessions to be reestablished after a takeover, that you avoid coding a zero on this parameter for either the active or the alternate CICS regions.

AKPFREQ

The **AKPFREQ** parameter specifies the number of write requests to the CICS system log stream output buffer required before CICS writes an activity keypoint.

AKPFREQ={4000|number}

4000 This is the default. You are recommended to allow **AKPFREQ** to assume its default value.

number

number can be 0 (zero) or any value in the range 200 through 65535. You cannot specify a number in the range 1—199. If you specify **AKPFREQ=0**, no activity keypoints are written, with the following consequences:

- The CICS system log automatic deletion mechanism will not work so efficiently in this situation. The average system log occupancy would merely increase, maybe quite dramatically for some users. Without efficient automatic deletion, the log stream will spill onto secondary storage, and from there onto tertiary storage (unless you control the size of the log stream yourself).
- Emergency restarts are not prevented, but the absence of activity keypoints on the system log affects the performance of emergency restarts because CICS has to read backwards through the entire log stream.
- Backout-while-open (BWO) support is seriously affected, because without activity keypointing, tie-up records are not written to the forward recovery logs and the data set recovery point is not updated. Therefore, for forward recovery to take place, all forward recovery logs must be kept since the data set was first opened for update after the last image copy. For more information about the effect of **AKPFREQ=0** on BWO, see “Effect of disabling activity keypointing” on page 12.

For more information about activity keypointing, see the *CICS Performance Guide*

APPLID

The **APPLID** parameter specifies the VTAM application identifier for this CICS region.

APPLID={DBDCCICS|applid}

Valid values are as follows:

applid This name, 1 through 8 characters, identifies the CICS region in the VTAM network. It must match the name field specified in the APPL statement of the VTAM VBUILD TYPE=APPL definition. For an example, see the *CICS Transaction Server for z/OS Installation Guide*.

If CICS is running in a sysplex, its applid must be unique within the sysplex. Note that, if the CICS extended recovery facility (XRF) is used by any of the regions in the sysplex, the specified applid must not duplicate the specific APPLID of any XRF CICS region. If, on CICS startup, the specified APPLID is found to duplicate the (specific or only) APPLID of any other CICS region currently active in the sysplex, CICS issues message DFHPA1946 and fails to initialize.

This parameter can be used also as the application identifier of this CICS region on IPIC connections.

When you define this CICS region to another CICS region, in an MRO or ISC over SNA CONNECTION definition you specify the applid as the NETNAME; in an IPIC IPCONN definition you specify the applid as the APPLID.

When sharing a DL/I database with a batch region, the applid is used by the batch region to identify the CICS region.

If the CICS region uses XRF, the form of the **APPLID** parameter is:

APPLID=(generic_applid,specific_applid)

specifies the generic and specific XRF APPLIDs for the CICS region. Both APPLIDs must be 1 through 8 characters long and the specific APPLID must be unique within the sysplex. If, on CICS startup, the specified specific APPLID is found to duplicate the (specific or only) APPLID of any other CICS region currently active in the sysplex, CICS issues message DFHPA1946 and fails to initialize.

generic_applid

The generic applid for both the active and the alternate CICS regions. Therefore, you must specify the same name for *generic_applid* on the APPLID system initialization parameter for both CICS regions. Because IRC uses the *generic_applid* to identify the CICS regions, there can be no IRC connection for an alternate CICS region until takeover has occurred and the alternate CICS region becomes the active CICS region.

When you define this XRF pair to another CICS region, in an MRO or ISC over SNA CONNECTION definition you specify the generic APPLID as the NETNAME; in an IPIC IPCONN definition you specify the generic APPLID as the APPLID.

When sharing a DL/I database with a batch region, this name is used by the batch region to identify the CICS region. CICS passes the generic applid to DBRC, because the alternate system does not sign on to DBRC until it has completed takeover.

Do not confuse the term *generic applid* with *generic resource name*. Generic APPLIDs apply only to CICS regions that use XRF. Generic resource names apply only to VTAM generic resource groups.

specific_applid

specifies the CICS region in the VTAM network. It must match the label specified in the VTAM VBUILD TYPE=APPL definition. You must specify a different *specific_applid* on the **APPLID** system initialization parameter for the active and for the alternate CICS region. Also, *generic_applid* and *specific_applid* must be different.

The active and alternate CICS regions use the **VTAM MODIFY USERVAR** command to set a user application name variable, so end users do not need to know which CICS region is active at any instant.

AUTCONN

The **AUTCONN** parameter specifies that the reconnection of terminals after an XRF takeover is to be delayed, to allow time for manual switching.

AUTCONN={0|hhmmss} (alternate)

The delay is *hh* hours, *mm* minutes, and *ss* seconds. The default value of zero means that there is no delay in the attempted reconnection.

The interval specified is the delay before the CXRE transaction runs. CXRE tries to reacquire any XRF-capable (class 1) terminal session that failed to get a backup session, or failed the switch for some other reason. CXRE tries to reacquire other terminals that were in session at the time of the takeover.

Note that the same delay interval applies to the connection of terminals with AUTOCONNECT(YES) specified in the TYPETERM definition, at a warm or emergency restart, whether or not you have coded XRF=YES.

AUTODST

The **AUTODST** parameter specifies whether CICS is to activate automatic dynamic storage tuning for application programs.

AUTODST={NO|YES}

Valid values are as follows:

- NO** Automatic dynamic storage tuning is not required and CICS does not request this support from Language Environment.
- YES** Automatic dynamic storage tuning is required. This is activated during CICS startup when Language Environment is being initialized. CICS indicates to Language Environment that it is able to support dynamic storage tuning, and if Language Environment responds by indicating that it also supports the facility, CICS and Language Environment are synchronized to provide the required support.

For more information, see the appropriate z/OS Language Environment manual.

AUTOSETTIME

The **AUTOSETTIME** parameter specifies the action CICS should take if, at the next local midnight, the CICS time-of-day differs from the system time-of-day by more than 30 minutes; for example, setting clocks forward or back to adjust for summer and winter time.

AUTOSETTIME={NO|YES}

Valid values are as follows:

- NO** CICS issues message DFHAP1500 to indicate that a **CEMT PERFORM RESET** command is required to synchronize the CICS time-of-day with the system time-of-day.
- YES** CICS issues a **PERFORM RESET** command to synchronize the CICS time-of-day with the system time-of-day

Note: Setting clocks back might cause end-of-day statistics to be written twice.

AUXTR

The **AUXTR** parameter specifies whether the auxiliary trace destination is to be activated at system initialization.

AUXTR={OFF|ON}

This parameter controls whether any of the three types of CICS trace entry are written to the auxiliary trace data set. The three types are: CICS system trace (see the SYSTR parameter), user trace (see the USERTR parameter), and exception trace entries (that are always made and are not controlled by a system initialization parameter).

- OFF** Do not activate auxiliary trace.
- ON** Activate auxiliary trace.

For details of internal tracing in main storage, see the INTTR parameter on page INTTR.

AUXTRSW

The **AUXTRSW** parameter specifies whether you want the auxiliary trace autoswitch facility.

AUXTRSW={NO|ALL|NEXT}

Valid values are as follows:

NO Disables the autoswitch facility.

NEXT Enables the autoswitch facility to switch to the next data set at end of file of the first data set used for auxiliary trace. Coding NEXT permits one switch only, and when the second data set is full, auxiliary trace is switched off.

ALL Enable the autoswitch facility to switch to the inactive data set at every end of file. Coding ALL permits continuous switching between the two auxiliary trace data sets, DFHAUXT and DFHBUXT, and whenever a data set is full, it is closed and the other data set is opened.

BMS

The **BMS** system initialization parameter specifies which version of basic mapping support you require in CICS.

BMS={({MINIMUM|STANDARD|FULL }[,COLD][, {UNALIGN |ALIGN}] [, { DDS|NODDS}])

The function included in each version of **BMS** is shown in the *CICS Application Programming Guide*. The parameter **BMS** can be overridden during CICS initialization.

You must use full or standard function **BMS** if you are using XRF and have specified **MESSAGE** for **RECOVNOTIFY** on any of your **TYPETERM** definitions.

MINIMUM

The minimum version of **BMS** is included.

STANDARD

The standard version of **BMS** is included.

FULL The full version of **BMS** is included. This value is the default.

COLD

CICS deletes delayed messages from temporary storage, and destroys their interval control elements (ICEs). COLD forces the deletion of messages regardless of the value in effect for **START**. If COLD is not specified, the availability of messages depend on the values in effect for the **START** and **TS** parameters.

UNALIGN

Specifies that all **BMS** maps assembled before CICS/OS/VS Version 1 Release 6 are unaligned. Results are unpredictable if the stated alignment does not match the actual alignment.

ALIGN

All **BMS** maps assembled before CICS/OS/VS Version 1 Release 6 are aligned.

DDS

BMS is to load suffixed versions of map sets and partition sets. **BMS** first tries to load a version that has the alternate suffix (if the transaction uses the alternate screen size). If the load fails, **BMS** tries to load a version that has the default map suffix. If this fails too, **BMS**

tries to load the unsuffixed version. DDS, which stands for “device dependent suffixing”, is the default.

You need to use map suffixes only if the same transaction is to be run on terminals with different characteristics (in particular, different screen sizes). If you do not use suffixed versions of map sets and partition sets, CICS need not test for them.

NODDS

BMS is not to load suffixed versions of map sets and partition sets. Specifying NODDS avoids the search for suffixed versions, saving processor time.

Table 13. Versions of BMS

| BMS version | Devices supported | Function provided |
|-------------|--|--|
| MINIMUM | All 3270 system display units and printers except SNA character string printers, which are defined as DEVICE(SCSPRINT) on the RDO TYPETERM definition or as TRMTYPE=SCSPRT in DFHTCT | SEND MAP command, RECEIVE MAP command, SEND CONTROL command. Default and alternate screens; extended attributes; map set suffixes; screen coordination with null maps; and block data |
| STANDARD | All devices are supported by BMS. These are listed in the the <i>CICS Application Programming Guide</i> | All function of MINIMUM, as well as outboard formats, partitions, controlling a magnetic slot reader, NLEOM mode for 3270 system printers, SEND TEXT command, and Subsystem LDC controls. |
| FULL | All devices supported by BMS. These are listed in the the <i>CICS Application Programming Guide</i> | Same as STANDARD, as well as terminal operator paging, cumulative mapping, page overflow, cumulative text processing, routing, message switching returning BMS-generated data stream to program before output. |

BRMAXKEEPTIME

The **BRMAXKEEPTIME** parameter specifies the maximum time (in seconds) that bridge facilities (virtual terminals used by the 3270 bridge) are kept if they are not used.

BRMAXKEEPTIME={86400|number}

The client application can specify this timeout value when it sends a request to run a transaction using the Link3270 bridge. If the client specifies a larger value than the BRMAXKEEPTIME value in the AOR, then CICS will change this parameter in the link parameter list.

number

The maximum timeout value that a client can specify (in seconds), before an unused bridge facility is deleted. The value specified must be in the range 0 to 86400. A value of 0 means that bridge facilities are never kept at the end of a transaction, therefore CICS will not be able

to run pseudoconversational transactions. This may be useful if the region is only used for inquiry transactions. The default value is 24 hours (86400 seconds).

CDSASZE

The **CDSASZE** system initialization parameter specifies the size of the CDSA.

CDSASZE={0K|**number**}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify **number** as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256KB). If the size specified is not a multiple of 256KB, CICS rounds the value up to the next multiple.

You can specify **number** in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions You can specify the **CDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

CHKSTRM

The **CHKSTRM** parameter specifies that terminal storage-violation checking is to be activated or deactivated.

CHKSTRM={**CURRENT**|**NONE**}

Valid values are as follows:

CURRENT

TIOA storage violations are to be checked.

NONE

TIOA storage-violation checking is to be deactivated.

You can also use the CICS-supplied transaction, CSFE, to switch terminal storage-violation checking on and off.

For information about checking for storage violations, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the **CHKSTRM** parameter in PARM, SYSIN, or CONSOLE only.

CHKSTSK

The **CHKSTSK** parameter specifies that task storage-violation checking at startup is to be activated or deactivated.

CHKSTSK={**CURRENT**|**NONE**}

Valid values are as follows:

CURRENT

All storage areas on the transaction storage chain for the current task only are to be checked.

NONE

Task storage-violation checking is to be deactivated.

You can also use the CICS-supplied transaction, CSFE, to switch task storage-violation checking on and off.

For information about checking for storage violations, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the **CHKSTSK** parameter in PARM, SYSIN, or CONSOLE only.

CICSSVC

The **CICSSVC** parameter specifies the number that you have assigned to the CICS type 3 SVC.

CICSSVC={216|number}

The default number is 216. A CICS type 3 SVC with the specified or default number must be installed in the LPA. For information about installing the CICS SVC, see the *CICS Transaction Server for z/OS Installation Guide*.

CICS checks if the SVC number supplied corresponds to the correct level of the CICS Type 3 SVC module, DFHCSVC. If the SVC number does not correspond to the correct level of DFHCSVC, the following can happen, depending on the value specified for the **PARMERR** system initialization parameter:

- CICS is terminated with a system dump
- The operator is allowed to retry using a different SVC number

For details of the **PARMERR** system initialization parameter, see PARMERR.

CILOCK

The **CILOCK** parameter specifies whether or not the control interval lock of a non-RLS VSAM file is to be kept after a successful read-for-update request.

CILOCK={NO|YES}

Valid values are as follows:

NO is the default and specifies that the control interval is to be freed. This allows other tasks to access other records in the same control interval, without an exclusive control conflict occurring. In these cases throughput should be greater. Note that the record lock on the record for which the read-for-update was first issued, still prevents other tasks from updating this record, even though the control interval lock has been released. When the record is rewritten or deleted, the read-for-update is reissued to VSAM as part of the update processing.

If a WRITE is issued by another task during a READ UPDATE, the WRITE receives a DUPREC condition.

YES specifies that the control interval is not to be freed. This means that a subsequent rewrite or delete request does not need to reissue the read-for-update request to VSAM. However, if other tasks attempt to access other records in the same control interval, an exclusive control conflict occurs on this control interval, forcing these tasks to wait until the update request completes.

CLINTCP

The **CLINTCP** parameter specifies the default client code page to be used by the DFHCNV data conversion table, but only if the **CLINTCP** parameter in the DFHCNV macro is set to SYSDEF.

CLINTCP={437|codepage}

The *codepage* is a field of up to 8 characters and can take the values supported

by the **CLINTCP** parameter in the DFHCNV macro. See the *CICS Family: Communicating from CICS on System/390* for the list of valid code pages. The default is 437.

CLSDSTP

The CLSDSTP system initialization parameter specifies the notification required for an EXEC CICS ISSUE PASS command.

CLSDSTP={NOTIFY|NONOTIFY}

This parameter is applicable to both autoinstalled and non-autoinstalled terminals. You can use the notification in a user-written node error program to reestablish the CICS session when a VTAM CLSDST PASS request resulting from an EXEC CICS ISSUE PASS command fails. For more information about the EXEC CICS ISSUE PASS command, see the *CICS Application Programming Reference*.

NOTIFY

CICS requests notification from VTAM when the EXEC CICS ISSUE PASS command is executed.

NONOTIFY

CICS does not request notification from VTAM.

CLT

The **CLT** parameter specifies the suffix for the command list table (CLT), if this SIT is used by an alternate XRF system.

CLT=xx (alternate)

The name of the table is DFHCLTxx. For information about coding the macros for this table, see the *CICS Resource Definition Guide*.

CMDPROT

The **CMDPROT** parameter specifies that you want to allow, or inhibit, CICS validation of start addresses of storage referenced as output parameters on EXEC CICS commands.

CMDPROT={YES|NO}

Valid values are as follows:

YES CICS validates the initial byte at the start of any storage that is referenced as an output parameter on EXEC CICS commands to ensure that the application program has write access to the storage. This ensures that CICS does not overwrite storage on behalf of the application program when the program itself cannot do so. If CICS detects that an application program has asked CICS to write into an area to which the application does not have addressability, CICS abends the task with an AEYD abend.

The level of protection against bad addresses depends on the level of storage protection in the CICS environment. The various levels of protection provided when you specify CMDPROT=YES are shown in Table 14 on page 159.

NO CICS does not perform any validation of addresses of the storage referenced by EXEC CICS commands. This means that an application program could cause CICS to overwrite storage to which the application program itself does not have write access.

Table 14. Levels of protection provided by CICS validation of application-supplied addresses

| Environment | Execution key of affected programs | Types of storage referenced by applications that cause AEYD abends |
|---|------------------------------------|--|
| Read-only storage (RENTPGM=PROTECT) | CICS-key and user-key | CICS key 0 read-only storage (RDSA and ERDSA). |
| Subsystem storage protection (STGPROT=YES) | User-key | All CICS-key storage (CDSA and ECDSA) |
| Transaction isolation (TRANISO=YES) | User-key and ISOLATE(YES) | Task-lifetime storage of all other transactions |
| Transaction isolation (TRANISO=NO) | User-key and ISOLATE(NO) | Task-lifetime storage of all except other user key and ISOLATE(NO) transactions |
| Base CICS (all storage is CICS key 8 storage) (RENTPGM=NOPROTECT; STGPROT=NO; and TRANISO=NO) | CICS-key and user-key | MVS storage only |

CMDSEC

The **CMDSEC** parameter specifies whether or not you want CICS to honor the CMDSEC option specified on a transaction's resource definition.

CMDSEC={ASIS|ALWAYS}

Valid values are as follows:

ASIS means that CICS honors the CMDSEC option defined in a transaction's resource definition. CICS calls its command security checking routine only when CMDSEC(YES) is specified in a transaction resource definition.

ALWAYS

CICS overrides the CMDSEC option, and always calls its command security checking routine to issue the appropriate call to the SAF interface.

Note:

1. Specify ALWAYS when you want to control the use of the SPI in all your transactions. Be aware that this might incur additional overhead. The additional overhead is caused by CICS issuing the command security calls on every eligible EXEC CICS command, which are *all* the system programming interface (SPI) commands.
2. If you specify ALWAYS, command checking applies to CICS-supplied transactions such as CESN and CESF. You must authorize all users of CICS-supplied transactions to use the internal CICS resources for the transactions, otherwise you will get unexpected results in CICS-supplied transactions.

Restrictions You can specify the CMDSEC parameter in the SIT, PARM, or SYSIN only.

CONFDATA

The **CONFDATA** parameter specifies whether CICS is to suppress user data that might otherwise appear in CICS trace entries or in dumps.

CONFDATA={SHOW|HIDETC}

This option applies to initial input data received on:

- A VTAM RECEIVE ANY operation
- An MRO connection
- An IPIC connection
- FEPI screens and RPLAREAs

This option also applies to the CICS client use of a Virtual Terminal. Data is traced before and after codepage conversion and is suppressed if HIDETC is used in combination with CONFDATA YES in the transaction.

SHOW

Data suppression is not in effect. User data is traced regardless of the CONFDATA option specified in transaction resource definitions. This option overrides the CONFDATA option in transaction resource definitions.

HIDETC

CICS is to 'hide' user transport data from CICS trace entries. The action taken by CICS is subject to the individual CONFDATA attribute on the transaction resource definition (see Table 15 on page 161).

If you specify CONFDATA=HIDETC, CICS processes VTAM, MRO, IPIC, and FEPI user data as follows:

- **VTAM:** CICS clears the VTAM RAIA containing initial input as soon as it has been processed, and before the target transaction has been identified.

The normal trace entries (FC90 and FC91) are created on completion of the RECEIVE ANY operation with the text "SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT" replacing all the user data except the first 4 bytes of normal data, or the first 8 bytes of function management headers (FMHs).

CICS then identifies the target transaction for the data. If the transaction definition specifies CONFDATA(NO), CICS traces the user data that it suppressed from the FC90 trace in the trace entry AP FC92. This trace entry is not created if the transaction is defined with CONFDATA(YES).

- **MRO:** CICS does not trace the initial input received on an MRO link.

The normal trace entries (DD16, DD23, and DD25) are created with the text SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT replacing all the user data.

CICS then identifies the target transaction for the data. If the transaction definition specifies CONFDATA(NO), CICS traces the user data that it suppressed from DD16 in the trace entry AP FC92. This special trace entry is not created if the transaction is defined with CONFDATA(YES).

- **IPIC:** Trace points SO 0201 and SO 0202 suppress buffer data with the message "Trace data suppressed because it may contain sensitive data". Subsequent trace point SO 029D (buffer continuation) and buffer data from trace points WB 0700 and WB 0701 is suppressed.

If the transaction definition specifies CONFDATA(NO), IS trace entries are created with the user data, as normal.

If the transaction definition specifies CONFDATA(YES), user data from IS trace points IS 0602, IS 0702, and IS 0906 is replaced with

"SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT". Data from IS trace points IS 0603 and IS 0703 is not shown.

- **FEPI:** FEPI screens and RPL data areas (RPLAREAs) areas are suppressed from all FEPI trace points if CONFDATA(YES) is specified in the transaction resource definition. The user data in the FEPI trace points AP 1243, AP 1244, AP 145E, AP 145F, AP 1460, AP 1461, AP 1595, AP 1596, AP 1597, AP 1598, and AP 1599 is replaced with the message SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT. If the transaction definition specifies CONFDATA(NO), the FEPI trace entries are created with the user data as normal.

Mirror transactions: The CICS-supplied mirror transaction definitions are specified with CONFDATA(YES). This ensures that, when you specify CONFDATA=HIDETC as a system initialization parameter, CICS regions running mirror transactions suppress user data as described for VTAM, MRO, and IS data.

Modified data: By waiting until the transaction has been identified to determine the CONFDATA option, VTAM, MRO, or IS data may have been modified (for example, it may have been translated to upper case).

The interaction between the **CONFDATA** system initialization parameter and the CONFDATA attribute on the transaction resource definition is shown in Table 15.

Table 15. Effect of CONFDATA system initialization and transaction definition parameters

| CONFDATA on transaction | CONFDATA system initialization parameter | |
|-------------------------|--|-----------------|
| | SHOW | HIDETC |
| NO | Data not suppressed | Data suppressed |
| YES | Data not suppressed | Data suppressed |

You cannot modify the CONFDATA option while CICS is running. You must restart CICS to apply a change.

Restrictions: You can specify the **CONFDATA** parameter in the SIT, PARM, and SYSIN only.

CONFTXT

The **CONFTXT** system initialization parameter specifies whether CICS is to prevent VTAM from tracing user data.

CONFTXT={NO|YES}

Valid values are as follows:

NO CICS does not prevent VTAM from tracing user data.

YES CICS prevents VTAM from tracing user data.

Restrictions You can specify the **CONFTXT** parameter in the SIT, PARM, and SYSIN only.

CPSMCONN

The **CPSMCONN** parameter specifies whether you want CICS to invoke the specified CICSplex[®] SM component to initialize the region.

CPSMCONN={NO|CMAS|LMAS|WUI}

You can initialize the region as one of the following:

- A CICSplex SM address space (CMAS)

- A CICSplex SM managed application system (MAS)
- A CICSplex SM Web User Interface server

NO Do not invoke any CICSplex SM initialization code in this region.

CMAS

Invoke CICSplex SM code automatically during CICS initialization to initialize the region as a CMAS. The other information CICSplex SM needs for a CMAS is taken from the CMAS parameters read from the EYUPARM data set, and from resource definitions installed from the CSD from group list EYU410L0.

Specifying **CPSMCONN=CMAS** is the recommended alternative to specifying the CICSplex SM CMAS initialization program in a CICS post-initialization program list table (PLTPI).

Note: If you specify **CPSMCONN=CMAS**, ensure that your CICS region startup JCL EXEC statement specifies the name of the CICSplex SM CMAS program, EYU9XECS. For example:

```
//CMAS EXEC PGM=EYU9XECS,.....
```

LMAS Invoke CICSplex SM code automatically during CICS initialization to initialize the region as a local MAS. The other information CICSplex SM needs for a MAS is taken from the MAS parameters read from the EYUPARM data set.

Specifying **CPSMCONN=LMAS** is the recommended alternative to specifying the CICSplex SM MAS initialization program in a CICS post-initialization program list table (PLTPI).

WUI Invoke CICSplex SM code automatically during CICS initialization to initialize the region as a CICSplex SM Web User Interface server. The other information CICSplex SM needs is taken from the MAS and WUI parameters read from the EYUPARM and EYUWUI data sets respectively.

Specifying **CPSMCONN=WUI** is the recommended alternative to specifying the CICSplex SM MAS and WUI initialization and shutdown programs in initialization and shutdown program list tables (PLTPI and PLTSD).

Note that using the **CPSMCONN** parameter has the same effect as specifying the relevant CICSplex SM program in a program list table. This means that MASPLTWAIT and other PLT-related CICSplex SM parameters are still valid and should be specified as necessary.

For information about starting CICSplex SM address spaces, see the *CICS Transaction Server for z/OS Installation Guide*.

CRLPROFILE

The **CRLPROFILE** parameter specifies the name of the profile that is used to authorize CICS to access the certification revocation lists (CRLs) that are stored in an LDAP server.

CRLPROFILE=PROFILENAME

The profile name is specified in the external security manager's LDAPBIND general resource class that contains bind information for an LDAP server. The profile name must be uppercase and can be up to 246 characters in length.

The profile must contain the name of the LDAP server and the distinguished name and password of a user who is authorized to extract certification revocation lists from it. For more information about setting up the profile, see the *CICS RACF Security Guide*.

Specifying this parameter means that CICS checks each client certificate during the SSL negotiation for a revoked status using the certificate revocation lists in the LDAP server. If the certificate is revoked, CICS closes the connection immediately.

If the **CRLPROFILE** parameter is omitted or invalid, or the specified profile contains invalid data, or if the LDAP server identified by the profile is unavailable, CICS does not check the revoked status of certificates during SSL handshakes.

CSDACC

The **CSDACC** parameter specifies the type of access to the CSD to be permitted to this CICS region.

CSDACC={READWRITE | READONLY}

This parameter is effective only when you start CICS with a **START=COLD** parameter. If you code **START=AUTO**, and CICS performs a warm or emergency restart, the file resource definitions for the CSD are recovered from the CICS global catalog. However, you can redefine the type of access permitted to the CSD dynamically with a **CEMT SET FILE** or **EXEC CICS SET FILE** command.

READWRITE

Read/write access is allowed, permitting the full range of CEDA, CEDB, and CEDC functions to be used.

READONLY

Read access only is allowed, limiting the CEDA and CEDB transactions to only those functions that do not require write access.

CSDBKUP

The **CSDBKUP** parameter specifies whether or not the CSD is eligible for BWO.

CSDBKUP={STATIC | DYNAMIC}

If you want to use BWO, specify **CSDBKUP=DYNAMIC**.

The **CSDBKUP**, **CSDRECOV**, and **CSDFRLOG** system initialization parameters interact according to how they are specified. For information about their effects when the SIT is assembled and during CICS override processing, see “Planning for backup and recovery” on page 55.

STATIC

All CICS files open for update against the CSD data set must be quiesced before a DFHSM and DFDSS backup of the CSD data set. The files must remain quiesced during the backup.

DYNAMIC

DFHSM and DFDSS are allowed to make a data set back up copy while CICS is updating the CSD.

Note that **CSDBKUP=DYNAMIC** is valid only if you have also specified **CSDRECOV=ALL**.

CSDBUFND

The **CSDBUFND** parameter specifies the number of buffers to be used for CSD data.

CSDBUFND=number

The minimum you should specify is the number of strings coded on the **CSDSTRNO** parameter plus 1, up to a maximum of 32768. Note that this parameter is used only if you have also coded **CSDLSRNO=NONE**; if you have coded **CSDLSRNO=number**, **CSDBUFND** is ignored.

If you specify a value for **CSDBUFND** that is less than the required minimum (the **CSDSTRNO** value plus 1), VSAM automatically changes the number of buffers to the number of strings plus 1 when CICS issues the OPEN macro for the CSD.

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDBUFNI

The **CSDBUFNI** parameter specifies the number of buffers to be used for the CSD index.

CSDBUFNI=number

The minimum you should specify is the number of strings coded on the **CSDSTRNO** parameter, up to a maximum of 32768. This parameter is used only if you have also coded **CSDLSRNO=NONE**; if you have coded **CSDLSRNO=number**, **CSDBUFNI** is ignored.

If you specify a value for **CSDBUFNI** that is less than the required minimum (the **CSDSTRNO** value), VSAM automatically changes the number of buffers to the number of strings when CICS issues the OPEN macro for the CSD.

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDDISP

The **CSDDISP** parameter specifies the disposition of the data set to be allocated to the CSD.

CSDDISP={OLD|SHR}

If no JCL statement for the CSD exists when it is opened, the open is preceded by a dynamic allocation of the CSD using this disposition. If a DD statement exists in the JCL of the CICS startup job, it takes precedence over this disposition.

OLD The disposition of the CSD is set to OLD if dynamic allocation is performed.

SHR The disposition of the CSD is set to SHR if dynamic allocation is performed.

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDDSN

The **CSDDSN** parameter specifies the 1-44 character JCL data set name (DSNAME) to be used for the CSD.

CSDDSN=name

If no JCL statement exists for the CSD when it is opened, the open is preceded by a dynamic allocation of the CSD using this DSNNAME. If a DD statement exists in the JCL of the CICS startup job, it takes precedence over this DSNNAME.

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDFRLOG

The **CSDFRLOG** parameter specifies a number that corresponds to the journal name that CICS uses to identify the forward recovery log stream for the CSD.

CSDFRLOG=number

This parameter is meaningful only if CSDRECOV=ALL and CSDRLS=NO are specified, otherwise it is ignored. If you specify CSDRLS=NO and CSDRECOV=ALL, but omit CSDFRLOG (or specify CSDFRLOG=NO), the SIT assembly fails. However, if you specify an invalid combination as SIT overrides, CICS initialization will fail.

CSDBKUP, **CSDRECOV** and **CSDFRLOG** are ignored if CSDRLS=YES is specified. This is because recovery attributes (that is, the recoverability, the forward recovery LSN, and the BWO eligibility) must be specified in the ICF catalog for data sets that are opened in RLS mode.

The recovery attributes can also be specified (optionally) in the ICF catalog when you specify CSDRLS=NO. If you specify recovery attributes in both the ICF catalog and as system initialization parameters, the ICF catalog values are used (but see the next paragraph).

For a CSD opened in a non-RLS mode (CSDRLS=NO), the **CSDBKUP**, **CSDRECOV** and **CSDFRLOG** system initialization parameters interact according to how they are specified. For information about their effects when the SIT is assembled and during CICS override processing, see “Planning for backup and recovery” on page 55.

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

number

The journal number that identifies the user journal that CICS is to use for forward recovery of the CSD. CICS journal names are of the form DFHJnn where nn is a number in the range 1 through 99. CICS maps the resulting journal name (DFHJ01—DFHJ99) to an MVS log stream.

CSDINTEG

The **CSDINTEG** parameter specifies the level of read integrity for the CSD if it is accessed in RLS mode.

CSDINTEG={UNCOMMITTED|CONSISTENT|REPEATABLE}

If the CSD is not accessed in RLS mode (CSDRLS=NO), a value for CSDINTEG of CONSISTENT or REPEATABLE will be changed to UNCOMMITTED.

UNCOMMITTED

The CSD is read without read integrity. For each read request, CICS obtains the current value of the record as known to VSAM. No attempt is made to serialize this read request with any concurrent update

activity for the same record. The record returned may be a version updated by another RDO task but not yet committed, and this record could change if the update is subsequently backed out.

CONSISTENT

CICS reads the CSD with consistent read integrity. If a record is being modified by another RDO task, the READ request waits until the update is complete, the timing of which depends on whether the CSD is recoverable or non-recoverable:

- For a recoverable CSD, the READ request completes when the updating transaction completes its next syncpoint or rollback.
- For a non-recoverable CSD, the READ completes as soon as the VSAM request performing the update completes.

REPEATABLE

CICS reads the CSD with repeatable read integrity. If the record is being modified by another RDO task, the READ request waits until the update is complete, the timing of which depends on whether the CSD is recoverable or non-recoverable:

- For a recoverable CSD, the READ request completes when the updating transaction completes its next syncpoint or rollback.
- For a non-recoverable CSD, the READ completes as soon as the VSAM request performing the update completes.

After the CSD read completes, a shared lock remains held until syncpoint. This guarantees that a CSD record read within an RDO task cannot be modified until the end of the task (for example, a CEDA transaction) that is reading the CSD.

CSDJID

The **CSDJID** parameter specifies the journal identifier of the journal that you want CICS to use for automatic journaling of file requests against the CSD.

CSDJID={NO|number}

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

NO You do **not** want automatic journaling for the CSD. This is the default.

number

A number in the range 1 through 99 to identify the journal that CICS is to use for automatic journaling for the CSD. Mapping to a log stream works in the same way that CSDFRLOG does, that is, *nm* maps to DFHJnn. 01 no longer maps to the system log.

The automatic journaling options enforced for the CSD when you code CSDJID=number are JNLADD=BEFORE and JNLUPDATE=YES. These options are sufficient to record enough information for a user-written forward recovery utility. No other automatic journaling options are available for the CSD. For information about the options JNLADD=BEFORE and JNLUPDATE=YES, see the *CICS Resource Definition Guide*.

CSDLSRNO

The **CSDLSRNO** system initialization parameter specifies whether the CSD is to be associated with a local shared resource (LSR) pool.

CSDLSRNO={1|number|NONE|NO}

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog. However, you can redefine the LSR pool attribute for the CSD dynamically with an EXEC CICS SET FILE command.

1 The default LSR pool number is 1.

number

The number of the LSR pool the CSD is to be associated with. The number of the pool must be in the range 1 through 8.

NONE|NO

The CSD is not to be associated with a local shared resource pool.

CSDRECOV

The **CSDRECOV** system initialization parameter specifies whether the CSD is a recoverable file.

CSDRECOV={NONE|ALL|BACKOUTONLY}

The **CSDBKUP**, **CSDRECOV**, and **CSDFRLOG** system initialization parameters interact according to how they are specified, if **CSDRLS=NO** is specified. If **CSDRLS=YES** is specified, these parameters are ignored, because the recovery attributes must be specified in the VSAM catalog (using the **BWO**, **LOG**, and **LOGSTREAMID** parameters on **DEFINE CLUSTER** or **ALTER CLUSTER**). If **CSDRLS=NO** is specified but **LOG** has been specified in the VSAM catalog, the recovery attributes are taken from the VSAM catalog, and **CSDBKUP**, **CSDRECOV**, and **CSDFRLOG** do not need to be specified. If they are specified, however, the rules given in “Planning for backup and recovery” on page 55 must still be followed.

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

NONE

The CSD is not recoverable.

ALL You want both forward recovery and backout for the CSD. If you code **ALL**, also specify **CSDFRLOG** with the journal identification of the journal to be used for forward recovery of the CSD.

Note: If the journal you specify for logstreams associated with CSD recovery (**CSDJID**, **CSDFRLOG**, and possibly the log of logs, **DFGLGLOG**) is a DASD-only log stream, there can be delays when you use the CEDA transaction if the log stream requires a new connection. This delay is because the MVS system logger is formatting the staging dataset. Symptoms of the problem are:

```
DFHLG0771 07/08/01 03:30:42 IY0T1 A temporary error condition occurred
during MVS logger operation IXGWRITE for logstream xxxxxx.yyyyyy.zzzzzz.
MVS logger codes: X'00000008', X'00000868'.
```

If the CSD is the only file using those logstreams, CICS disconnects from the log when you end the CEDA transaction. The next time you run a CEDA transaction, CICS reconnects to the log stream and the MVS system logger allocates and formats a new staging data set.

BACKOUTONLY

CSD recovery is limited to file backout only. If you specify backout for the CSD, CICS uses the system log to record before images for backout purposes.

CSDRLS

The **CSDRLS** system initialization parameter specifies whether CICS is to access the CSD in RLS mode.

CSDRLS={NO|YES}

Valid values are as follows:

NO The CSD is opened in non-RLS mode, as specified on the **CSDLSRNO** parameter.

YES The CSD is opened in RLS mode. This enables you to update the CSD concurrently from several CICS regions, provided all the regions specify **CSDRLS=YES**. If a CICS region opens the CSD in RLS mode, another CICS region cannot open it in non-RLS mode. The first CICS region to open the CSD in a sysplex with SMSVSAM determines the access mode for all regions.

Your CSD must be defined to support RLS access: the **IMBED** option must not be specified, and recovery attributes must be defined in the VSAM catalog. The *CICS Transaction Server for z/OS Installation Guide* explains the data set characteristics required to support RLS access. If your CSD does not meet these requirements, it will fail to open.

If you specify both RLS and local shared resource (**CSDLSRNO=number**), RLS takes precedence.

If you specify **CSDRLS=YES**, the **CSDRECOV**, **CSDFRLOG**, and **CSDJID** parameters are ignored. You must specify the recovery attributes for an RLS-mode CSD in the ICF catalog entry for the CSD.

Note: If you define a recoverable CSD for RLS-mode access, you have to quiesce all RLS activity against the CSD before you can update the CSD using the batch utility program, **DFHCSDUP**. You can use the **SET DSNAME QUIESCE** command to do this, to ensure that no **CEDA**, **CEDB**, or **CEDC** transactions can run until you unquiesce the data set on completion of the batch job.

CSDSTRNO

The **CSDSTRNO** system initialization parameter specifies the number of concurrent requests that can be processed against the CSD.

CSDSTRNO={6|number}

When the number of requests reaches the **STRNO** value, CICS automatically queues any additional requests until one of the active requests terminates.

CICS requires two strings per CSD user, and you can increase the **CSDSTRNO** value, in multiples of two, to allow more than one concurrent **CEDA** user.

See “Multiple users of the CSD within a CICS region (non-RLS)” on page 49 before you code this parameter.

This parameter is effective only on a CICS cold or initial start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog. However, you can redefine the number of strings for the CSD dynamically with an **EXEC CICS SET FILE** command.

6 The minimum number of concurrent requests for the CSD is 6.

number

This number must be a multiple of 2, in the range 6 through 254.

CWAKEY

The **CWAKEY** system initialization parameter specifies the storage key for the common work area (CWA) if you are operating CICS with storage protection (STGPROT=YES).

CWAKEY={USER|CICS}

You specify how much storage you want for the CWA on the **WRKAREA** parameter. The permitted values are **USER** (the default), or **CICS**:

USER CICS obtains storage for the CWA in user key. This allows a user program executing in any key to modify the CWA.

CICS CICS obtains storage for the CWA in CICS key. This means that only programs executing in CICS key can modify the CWA, and user-key programs have read-only access.

If CICS is running without storage protection, the **CWAKEY** parameter is ignored, and the CWA is always allocated from CICS-key storage.

DAE

The **DAE** system initialization parameter specifies the default DAE action when new system dump table entries are created.

DAE={NO|YES}

Valid values are as follows:

NO New system dump table entries will be created with **DAEOPTION(NODAE)**. This means that the system dump will not be suppressed by the MVS Dump Analysis and Elimination (DAE) component.

YES New system dump table entries will be created with **DAEOPTION(DAE)**. This means that the system dump is eligible for suppression by the MVS DAE component.

For more information about the **DAEOPTION** option, see the *CICS System Programming Reference*.

DATFORM

The **DATFORM** system initialization parameter specifies the external date display standard that you want to use for CICS date displays.

DATFORM={MMDDYY|DDMMYY|YYMMDD}

An appropriate indicator setting is made in the CSA. It is examined by CICS supplied system service programs that display a Gregorian date. CICS maintains the date in the form 0CYDDDD in the CSA (where C=0 for years 19xx, 1 for years 20xx, and so on; YY=year of century; and DDD=day of year), and converts it to the standard you specify for display.

The **DATFORM** option selects the order in which the date is to be displayed. It does not select the format of the year. Both YY and YYYY formats are displayed.

MMDDYY

The date is in the form of month-day-year, MMDDYY and MMDDYYYY.

DDMMYY

The date is in the form of day-month-year, DDMMYY and DDMMYYYY.

YYMMDD

The date is in the form of year-month-day, YYMMDD and YYYYMMDD.

DB2CONN

The **DB2CONN** system initialization parameter specifies whether you want CICS to start the DB2 connection automatically during initialization.

DB2CONN={NO|YES}

Valid values are as follows:

NO Do not automatically invoke DFHD2CM0, the CICS DB2 attach program, during initialization.

YES Invoke the CICS DB2 attach program, DFHD2CM0, automatically during CICS initialization. The other information CICS needs for starting the attachment is taken from CICS DB2 connection resource definitions installed from the CSD.

Specifying DB2CONN=YES is the recommended alternative to specifying the CICS DB2 attach programs in the CICS post-initialization program list table (PLT).

DBCTLCON

The **DBCTLCON** system initialization parameter specifies whether you want CICS to start the DBCTL connection automatically during initialization.

DBCTLCON={NO|YES}

Valid values are as follows:

NO Do not automatically invoke DFHDBCON, the CICS DBCTL attach program, during initialization.

YES Invoke the CICS DBCTL attach program, DFHDBCON, automatically during CICS initialization. The other information CICS needs for starting the attachment, such as the DRA startup table suffix or the DBCTL subsystem name, is taken from an INITPARM system initialization parameter.

Specifying DBCTLCON=YES means you don't need to define the DBCTL attach program in the CICS post-initialization program list table (PLT).

DEBUGTOOL

The **DEBUGTOOL** system initialization parameter specifies whether you want to use debugging profiles to select the programs that will run under the control of a debugging tool.

DEBUGTOOL={NO|YES}

The following debugging tools use debugging profiles:

- Debug Tool, for compiled language application programs (programs written in COBOL, PL/I, C, C++ and Assembler)

- Remote debugging tools (for compiled language application programs and Java programs)

Other debugging mechanisms, such as the CICS Execution Diagnostic Facility (CEDF) do not use debugging profiles.

NO Specifies that you do not want to use CICS debugging profiles to select the programs that will run under the control of a debugger tool.

YES

Specifies that you want to use CICS debugging profiles to select the programs that will run under the control of a debugger tool.

For more information, see the *CICS Application Programming Guide*.

DFLTUSER

The **PLTPISEC** system initialization parameter specifies the RACF userid of the default user; that is, the user whose security attributes are used to protect CICS resources in the absence of other, more specific, user identification.

DFLTUSER={CICSUSER|userid}

For example, except in the case of terminals defined with preset security, the security attributes of the default user are assigned to terminal users who do not sign on.

The specified userid must be defined to RACF if you are using external security (that is, you have specified the system initialization parameter **SEC=YES**).

The specified userid is signed on during CICS initialization. If it cannot be signed on, CICS fails to initialize.

Restrictions You can specify the **DFLTUSER** parameter in the **SIT**, **PARM**, or **SYSIN** only.

DIP

The **DIP** system initialization parameter specifies whether the batch data interchange program, **DFHDIP**, is to be included.

DIP={NO|YES}

This program supports the batch controller functions of the IBM 3790 Communication System and the IBM 3770 Data Communication System. Support is provided for the transmit, print, message, user, and dump data sets of the 3790 system. For the effect of this parameter, see “Defining CICS resource table and module keywords” on page 138.

DISMACP

The **DISMACP** system initialization parameter specifies whether CICS is to disable any transaction that terminates abnormally with an **ASRD** or **ASRE** abend.

DISMACP={YES|NO}

DISMACP=YES has no effect if the **ASRD** or **ASRE** abend is handled by an active abend exit. An abend might be caused by a user program invoking a CICS macro, or referencing the **CSA**, or the **TCA**.

DOCCODEPAGE

The **DOCCODEPAGE** system initialization parameter specifies the default host code page to be used by the document domain.

DOCCODEPAGE={037|codepage}

The *codepage* is a field of up to 8 characters. If *codepage* value is not specified, the default *doccodepage* is set to 037.

The standard CICS form of a host code page name consists of the code page number (or more generally CCSID) written using 3 to 5 decimal digits as necessary then padded with trailing spaces to 8 characters. For code page 37, which is fewer than 3 digits, the standard form is 037. CICS accepts any decimal number of up to 8 digits (padded with trailing spaces) in the range 1 to 65535 as a code page name, even if it is not in the standard form.

The DOCCODEPAGE parameter must specify an EBCDIC-based code page if any symbol processing is required, as the delimiters used for symbol and symbol list processing are assumed to be in EBCDIC.

DSALIM

The **DSALIM** system initialization parameter specifies the upper limit of the total amount of storage within which CICS can allocate the individual dynamic storage areas (DSAs) that reside below the 16 MB boundary.

DSALIM={5M|number}

Valid values are as follows

5M The default DSA limit is 5 MB (5 242 880).

number

This is the amount of storage in the range 2 MB to 16 MB (2 097 152 bytes to 16 777 216 bytes) in multiples of 262 144 bytes (256 KB). If the size specified is not a multiple of 256 KB, CICS rounds the value up to the next multiple.

You can specify *number* in bytes (for example, 4 194 304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

From the storage size that you specify on the **DSALIM** parameter, CICS allocates the following dynamic storage areas:

The user DSA (UDSA)

The user-key storage area for all user-key task-lifetime storage below the 16MB boundary.

The read-only DSA (RDSA)

The key-0 storage area for all reentrant programs and tables below the 16MB boundary.

The shared DSA (SDSA)

The user-key storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing EXEC CICS GETMAIN commands for storage below the 16 MB boundary with the SHARED option.

The CICS DSA (CDSA)

The CICS-key storage area for all non-reentrant CICS-key RMODE(24) programs, all CICS-key task-lifetime storage below the 16 MB boundary, and for CICS control blocks that reside below the 16 MB boundary.

Note:

1. CICS allocates the UDSA in multiples of 1 MB when transaction isolation is active, but in multiples of 256 KB in CICS regions without transaction

isolation. The other DSAs below 16 MB are allocated in multiples of 256 KB, with or without transaction isolation. The maximum you can specify depends on a number of factors, such as how you have configured your MVS storage (which governs how much private storage remains below the line) and how much private storage you must leave free to satisfy MVS GETMAIN requests for storage outside the DSAs.

2. For information about calculating the amount of storage to specify on the **DSALIM** parameter, see the *CICS Performance Guide*.
3. Dynamic changes to the DSA limit are cataloged in the local catalog, and override the **DSALIM** parameter, if it is specified in the system initialization table, during all forms of restart-initial, cold, and warm. The cataloged value is not used if:
 - You specify startup values as system initialization parameters overrides (for example, in SYSIN).
 - You re-initialize the CICS catalog data sets.

DSHIPIDL

The **DSHIPIDL** system initialization parameter specifies the minimum time, in hours, minutes, and seconds, that an inactive shipped terminal definition must remain installed in this region.

DSHIPIDL={020000|hhmmss}

When the timeout delete mechanism is invoked, only those shipped definitions that have been inactive for longer than the specified time are deleted.

You can use this parameter in a transaction routing environment, on the application-owning and intermediate regions, to prevent terminal definitions having to be reshipped because they have been deleted prematurely.

The default minimum idle time is 2 hours.

hhmmss

A 1- to 6-digit number in the range 0-995959. Numbers that have fewer than six digits are padded with leading zeros.

DSHIPINT

The **DSHIPINT** system initialization parameter specifies the interval between invocations of the timeout delete mechanism.

DSHIPINT={120000|0|hhmmss}

The timeout delete mechanism removes any shipped terminal definitions that have not been used for longer than the time specified by the **DSHIPIDL** parameter.

You can use this parameter in a transaction routing environment, on the application-owning and intermediate regions, to control:

- How often the timeout delete mechanism is invoked
- The approximate time of day at which a mass delete operation is to take place, relative to CICS startup

Note: For more flexible control over when mass delete operations take place, you can use a CEMT SET DELETSHIPPED or EXEC CICS SET DELETSHIPPED command to reset the interval. (The revised interval starts from the time the command is issued, **not** from the time the remote delete mechanism was last invoked, nor from CICS startup.)

0 The timeout delete mechanism is not invoked. You might set this value in a terminal-owning region, or if you are not using shipped definitions.

hhmmss

A 1- to 6-digit number in the range 1-995959. Numbers that have fewer than six digits are padded with leading zeros.

DSRTPGM

The **DSRTPGM** system initialization parameter specifies the name of a distributed routing program.

DSRTPGM={NONE|DFHDSRP|**program-name**|EYU9XLOP}

The program can dynamically route:

- Eligible CICS business transaction services (BTS) processes and activities
For information about which BTS processes and activities are eligible for dynamic routing, see *CICS Business Transaction Services*.
- Eligible non-terminal-related EXEC CICS START requests.
For information about which non-terminal-related START requests are eligible for dynamic routing, see the *CICS Intercommunication Guide*.

DFHDSRP

The CICS sample distributed routing program.

EYU9XLOP

The CICSplex SM routing program.

NONE

For eligible CICS BTS processes and activities, no routing program is invoked. BTS processes and activities cannot be dynamically routed.

For eligible non-terminal-related START requests, the CICS sample distributed routing program, DFHDSRP, is invoked.

program-name

The name of a user-written program.

Note: See also the DTRPGM parameter, used to name the dynamic routing program.

DTRPGM

The **DTRPGM** system initialization parameter specifies the name of a dynamic routing program.

DTRPGM={DFHDYP|**program-name**}

The program can dynamically route transactions initiated from user terminals, transactions initiated by eligible terminal-related EXEC CICS START commands, and eligible program-link requests.

DFHDYP, the default, is the name of the CICS-supplied program. For information about which transactions started by EXEC CICS START commands, and which program-link requests, are eligible for dynamic routing, see the *CICS Intercommunication Guide*.

Note: See also the **DSRTPGM** parameter, used to name the distributed routing program.

DTRTRAN

The **DTRTRAN** system initialization parameter specifies the name of the transaction definition that you want CICS to use for dynamic transaction routing.

DTRTRAN={CRTX|*name*|**NO**}

This is intended primarily for use in a CICS terminal-owning region, although you can also use it in an application-owning region when you want to daisy-chain transaction routing requests. In a dynamic transaction routing environment, the transaction defined for **DTRTRAN** must be installed in the CICS terminal-owning regions if you want to eliminate the need for resource definitions for individual transactions.

Note: DTRTRAN does not apply to non-terminal **EXEC CICS START** requests where the distributed routing program is invoked.

The transaction name is stored in the catalog for recovery during CICS restarts.

CRTX This is the default dynamic transaction definition. It is the name of the CICS-supplied sample transaction resource definition provided in the CSD group DFHISC.

name The name of your own dynamic transaction resource definition that you want CICS to use for dynamic transaction routing.

NO The dynamic transaction routing program is not invoked when a transaction definition cannot be found.

For information about the CICS-supplied sample transaction resource definition, CRTX, and about defining your own dynamic transaction routing definition, see in the *CICS Intercommunication Guide*.

DUMP

The **DUMP** system initialization parameter specifies whether the CICS dump domain is to take SDUMPs.

DUMP={YES|**NO**} (**active and alternate**)

Valid values are as follows:

YES SDUMPs are produced, unless suppressed by the options specified in the CICS system dump table or by the MVS system defaults.

NO SDUMPs are suppressed.

Note: This does not prevent the CICS kernel from taking SDUMPs.

For more information about SDUMPs, see “System dumps” on page 82.

DUMPDS

The **DUMPDS** system initialization parameter specifies the transaction dump data set that is to be opened during CICS initialization.

DUMPDS={AUTO|**A**|**B**}

Valid values are as follows:

AUTO

For all emergency or warm starts, CICS opens the transaction dump data set that was **not** in use when the previous CICS run terminated. This information is obtained from the CICS local catalog.

If you specify AUTO, or let it default, code DD statements for both of the transaction dump data sets, DFHDMPA and DFHDMPB, in your CICS startup job stream.

- A CICS opens transaction dump data set DFHDMPA.
- B CICS opens transaction dump data set DFHDMPB.

DUMPSW

The **DUMPSW** system initialization parameter specifies whether you want CICS to switch automatically to the next dump data set when the first is full.

DUMPSW={NO|NEXT}

Valid values are as follows:

- NO** Disables the CICS autoswitch facility. If the transaction dump data set opened during initialization becomes full, CICS issues a console message to notify the operator. If you want to switch to the alternate data set, you must do so manually using the CEMT or EXEC CICS SET DUMPDS SWITCH command.
- NEXT** Enables the autoswitch facility to switch to the next data set at end of file of the data set opened during initialization. Coding NEXT permits one switch only. If you want to switch to the alternate data set again, you must do so manually using CEMT or EXEC CICS SET DUMPDS SWITCH command. If you specify NEXT, code DD statements for both of the transaction dump data sets, DFHDMPA and DFHDMPB, in your CICS startup job stream.

For more information about transaction dump data sets, see page “Printing the transaction dump data sets” on page 83.

DURETRY

The **DURETRY** system initialization parameter specifies, in seconds, the total time that CICS is to continue trying to obtain a system dump using the SDUMP macro.

DURETRY={30|number-of-seconds|0}

DURETRY allows you to control whether, and for how long, CICS is to reissue the SDUMP macro if another address space in the same MVS system is already taking an SDUMP when CICS issues an SDUMP request.

In the event of an SDUMP failure, CICS responds, depending on the reason for the failure, as follows:

- If MVS is already taking an SDUMP for another address space, and the DURETRY parameter is nonzero, CICS issues an MVS STIMER macro to wait for five seconds, before retrying the SDUMP. CICS issues a message to say that it is waiting for five seconds before retrying the SDUMP. After five seconds CICS issues another message to say that it is retrying the SDUMP request.
 - If the SDUMP fails for any other reason, such as no SYS1.DUMP data sets being available, or I/O errors preventing completion of the dump, CICS issues a message to inform you that the SDUMP has failed, and to give the reason why.
- 30** 30 seconds allows CICS to retry up to 6 times (once every 5 seconds), if the cause of failure is that another region is taking an SDUMP.

number-of-seconds

Code the total number of seconds (up to 32767) during which you

want CICS to continue retrying the SDUMP macro if the reason for failure is that another region is taking an SDUMP. CICS retries the SDUMP, once every five seconds, until successful or until retries have been made over a period equal to or greater than the DURETRY value.

0 Code a zero value if you do not want CICS to retry the SDUMP macro.

ECDSASZE

The **ECDSASZE** system initialization parameter specifies the size of the ECDSA.

ECDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify *number* as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1 MB). If the size specified is not a multiple of 1 MB, CICS rounds the value up to the next multiple.

You can specify *number* in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Note: For the DS domain function CHANGE_MODE, a trace entry will be generated if DS level 2, 3, or ALL tracing is active.

Restrictions You can specify the **ECDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

EDSALIM

The **EDSALIM** system initialization parameter specifies the upper limit of the total amount of storage within which CICS can allocate the individual extended dynamic storage areas (EDSAs) that reside above the 16 MB boundary.

EDSALIM={34M|number}

34M The default EDSA limit is 34 MB (35 651 584 bytes).

number

Specify *number* as a value in the range 10 MB to 2047 MB, in multiples of 1 MB. If the size specified is not a multiple of 1 MB, CICS rounds the value up to the next multiple.

You can specify *number* in bytes (for example, 33 554 432), or as a whole number of kilobytes (for example, 32 768 KB), or a whole number of megabytes (for example, 32 MB).

The maximum value allowed depends on a number of factors, such as:

- The size of the region you have specified on the MVS **REGION** parameter in the CICS job or procedure
- How much storage you require for the CICS internal trace table
- How much private storage you must leave free to satisfy MVS GETMAIN requests for storage above the 16 MB boundary outside the DSAs

From the storage value that you specify on the **EDSALIM** parameter, CICS allocates the following extended dynamic storage areas:

The extended user DSA (EUDSA)

The user-key storage area for all user-key task-lifetime storage above the 16 MB boundary.

The extended read-only DSA (ERDSA)

The key-0 storage area for all reentrant programs and tables above the 16 MB boundary.

The extended shared DSA (ESDSA)

The user-key storage area for any non-reentrant user-key RMODE(ANY) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage above the 16 MB boundary with the SHARED option.

The extended CICS DSA (ECDSA).

The CICS-key storage area for all non-reentrant CICS-key RMODE(ANY) programs, all CICS-key task-lifetime storage above the 16 MB boundary, and CICS control blocks that reside above the 16 MB boundary.

CICS allocates all the DSAs above the 16 MB boundary in multiples of 1 MB.

Note:

1. For information about calculating the amount of storage to specify on the **EDSALIM** parameter, see the *CICS Performance Guide*
2. Dynamic changes to the EDSA limit are cataloged in the local catalog, and override the **DSALIM** parameter, if it is specified in the system initialization table, during all forms of restart-initial, cold, and warm. The cataloged value is not used if:
 - You specify startup values as system initialization parameters overrides (for example, in SYSIN).
 - You re-initialize the CICS catalog data sets.

EJBROLEPRFX

The **EJBROLEPRFX** system initialization parameter specifies a prefix that is used to qualify the security role defined in an enterprise bean's deployment descriptor.

EJBROLEPRFX=*ejbrole-prefix*

The prefix is applied to the security role:

- when a role is defined to an external security manager
- when CICS maps a security role to a RACF user ID
- when an application invokes the `isCallerInRole()` method

For more information about how the **EJBROLEPRFX** parameter is used to qualify security roles for enterprise beans, see *Java Applications in CICS*.

You can specify a prefix of up to 16 characters. The prefix must not contain a period (.) character. If you specify a prefix that contains lower case characters, blanks, or punctuation characters, you must enclose it in apostrophes. If the prefix contains an apostrophe, code two successive apostrophes to represent it.

Restrictions:

1. You can specify the **EJBROLEPRFX** parameter in the SIT, PARM, or SYSIN only.

2. The **EJBRROLEPRFX** parameter is ignored if security role support is not enabled. To enable security role support you must specify **SEC=YES** and **XEJB=YES**.

ENCRYPTION

The **ENCRYPTION** system initialization parameter specifies the cipher suites that CICS uses for secure TCP/IP connections.

ENCRYPTION={STRONG|WEAK|MEDIUM}

When a secure connection is established between a pair of processes, the most secure cipher suite supported by both is used.

- Use **ENCRYPTION=STRONG** when you can tolerate the overhead of using high encryption if the other system requires it.
- Use **ENCRYPTION=WEAK** when you want to use encryption up to 40 bits in length.
- Use **ENCRYPTION=MEDIUM** when you want to use encryption up to 56 bits in length.

For compatibility with previous releases, **ENCRYPTION=NORMAL** is accepted as an equivalent to **ENCRYPTION=MEDIUM**. For more information about cipher suites, see the *CICS RACF Security Guide*.

CICS can use only the cipher suites which are supported by the underlying z/OS operating system.

Possible values for z/OS 1.4 are:

STRONG

Specifies that CICS should use the following cipher suites:

| Cipher suite | Encryption algorithm | Key length | MAC algorithm |
|--------------|----------------------|------------|---------------|
| 01 | No encryption | | MD5 |
| 02 | No encryption | | SHA |
| 03 | RC4 | 40 bits | MD5 |
| 04 | RC4 | 128 bits | MD5 |
| 05 | RC4 | 128 bits | SHA |
| 06 | RC2 | 40 bits | MD5 |
| 09 | DES | 56 bits | SHA |
| 0A | Triple DES | 168 bits | SHA |
| 2F | AES | 128 bits | SHA |
| 35 | AES | 256 bits | SHA |

The terms used in this table are:

MD5 Message Digest algorithm

SHA Secure Hash algorithm

RC2, RC4
Rivest encryption

DES Data Encryption Standard

Triple DES
DES applied three times

AES Advanced Encryption Standard

WEAK

Specifies that CICS should use the following cipher suites:

| Cipher suite | Encryption algorithm | Key length | MAC algorithm |
|--|----------------------|------------|---------------|
| 01 | No encryption | | MD5 |
| 02 | No encryption | | SHA |
| 03 | RC4 | 40 bits | MD5 |
| 06 | RC2 | 40 bits | MD5 |
| The terms used in this table are: MD5 Message Digest algorithm SHA Secure Hash algorithm RC2, RC4 Rivest encryption | | | |

MEDIUM

Specifies that CICS should use the following cipher suites:

| Cipher suite | Encryption algorithm | Key length | MAC algorithm |
|---|----------------------|------------|---------------|
| 01 | No encryption | | MD5 |
| 02 | No encryption | | SHA |
| 03 | RC4 | 40 bits | MD5 |
| 06 | RC2 | 40 bits | MD5 |
| 09 | DES | 56 bits | SHA |
| The terms used in this table are: MD5 Message Digest algorithm SHA Secure Hash algorithm RC2, RC4 Rivest encryption DES Data Encryption Standard | | | |

EODI

The **EODI** system initialization parameter specifies the end-of-data indicator for input from sequential devices.

EODI={E0|xx}

The characters "xx" represent two hexadecimal digits in the range 01 through FF. The default value is X'E0', which represents the standard EBCDIC backslash symbol (\).

ERDSASZE

The **ERDSASZE** system initialization parameter specifies the size of the ERDSA.

ERDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1 MB). If the size specified is not a multiple of 1 MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the **ERDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

ESDSASZE

The **ESDSASZE** system initialization parameter specifies the size of the ESDSA.

ESDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1 MB). If the size specified is not a multiple of 1 MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the **ESDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

ESMEXITS

The **ESMEXITS** system initialization parameter specifies whether installation data is to be passed through the RACROUTE interface to the external security manager (ESM) for use in exits written for the ESM.

ESMEXITS={NOINSTLN|INSTLN}

Valid values are as follows:

NOINSTLN

The INSTLN parameter is not used in RACROUTE macros.

INSTLN

CICS-related and installation-supplied data is passed to the ESM using the INSTLN parameter of the RACROUTE macro. For programming information, including the format of the data passed, see the *CICS Customization Guide*. This data is intended for use in exits written for the ESM.

Restrictions You can specify the ESMEXITS parameter in the SIT only.

EUDSASZE

The **EUDSASZE** system initialization parameter specifies the size of the EUDSA.

EUDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824

bytes in multiples of 1048576 bytes (1MB). If the size specified is not a multiple of 1MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions You can specify the EUDSAZSE parameter in PARM, SYSIN, or CONSOLE only.

FCT

The **FCT** system initialization parameter specifies the suffix of the file control table to be used.

FCT={NO|xx|YES}

This parameter is effective only on a CICS cold or initial start. CICS does not load an FCT on a warm or emergency restart and all file resource definitions are recovered from the global catalog.

For information about coding the macros for this table, see the *CICS Resource Definition Guide*.

You can use a mixture of macro definitions and RDO definitions for files in your CICS region. However, your FCT should contain definitions for only BDAM files to be loaded on a CICS cold start. Other types of files are loaded from their file definitions in RDO groups specified in the **GRPLIST** system initialization parameter. Any definitions in the FCT other than for BDAM files are ignored.

FCQRONLY

The **FCQRONLY** system initialization parameter specifies whether you want CICS to force all CICS API user application programs that are specified as threadsafe to run file control requests under the CICS QR TCB, as if they were specified as quasi-reentrant programs.

FCQRONLY={YES|NO}

Valid values are as follows:

NO CICS honors the CONCURRENCY(THREADSAFE) attribute on program resource definitions, and allows user application programs to run applicable file control request on an open TCB to avoid unnecessary TCB switching.

YES CICS forces all file control requests to run under the CICS QR TCB, as if they were specified as CONCURRENCY(QUASIRENT) programs. With all file requests on the QR TCB, CICS is able to minimize the amount of locking required at the expense of additional TCB switches if requests are run on open TCBs.

Specifying **FCQRONLY=YES** can improve the performance of file-owning regions.

FEPI

The **FEPI** system initialization parameter specifies whether or not you want to use the Front End Programming Interface feature (FEPI).

FEPI={NO|YES}

Valid values are as follows:

NO FEPI support is not required. You should specify NO on this parameter (or allow it to default) if you do not have the feature installed, or if you do not require FEPI support.

YES You require FEPI support, and CICS is to start the CSZI transaction.

For information about what is required to run FEPI see the *CICS Front End Programming Interface User's Guide*.

FLDSEP

The **FLDSEP** system initialization parameter specifies one through four field-separator characters, each of which indicates end of field in the terminal input data.

FLDSEP={ ' ' | 'xxxx' }

The default is four blanks. The field separator allows you to use transaction identifications of less than four characters followed by one of the separator characters. When less than four characters are coded, the parameter is padded with blanks, so that the blank is then a field separator. None of the specified field separator characters should be part of a transaction identification; in particular, the use of alphabetic characters as field separators is not recommended.

The character specified in the **FLDSEP** parameter must not be the same as any character specified in the **FLDSTRT** parameter. This means that it is invalid to allow both parameters to take the default value. **Restrictions**

If you specify **FLDSEP** in the SIT, the characters must be enclosed in single quotation marks.

If you specify **FLDSEP** as a PARM, SYSIN, or CONSOLE parameter, do **not** enclose the characters in quotation marks, and the characters you choose must not include an embedded blank, or any of these characters:

() ' = ,

FLDSTRT

The **FLDSTRT** system initialization parameter specifies a single character to be the field-name-start character for free-form input for built-in functions.

FLDSTRT={ ' ' | 'x' }

The default is a blank. The character specified should not be part of a transaction identification; in particular, the use of alphabetic characters is not recommended.

The character specified in the **FLDSTRT** parameter must not be the same as any character specified in the **FLDSEP** parameter. This means that it is invalid to allow both parameters to take the default value.

Restrictions

If you specify **FLDSTRT** in the SIT, the parameter must be enclosed in single quotation marks.

If you specify **FLDSTRT** as a PARM, SYSIN, or CONSOLE parameter, do **not** enclose the character in quotation marks, and the character you choose must not be a blank or any of the following characters:

() ' = ,

FORCEQR

The **FORCEQR** system initialization parameter specifies whether you want CICS to force all CICS API user application programs that are specified as threadsafe to run under the CICS QR TCB, as if they were specified as quasi-reentrant programs.

FORCEQR={NO|YES}

This parameter applies to all application programs that are restricted to the current CICS programming interfaces (that is, those which specify API(CICSAPI)), and does not apply to any of the following:

- Java programs that are run in a JVM,
- C/C++ programs using XPLINK,
- OPENAPI programs,

none of which can run on the QR TCB.

NO CICS honors the CONCURRENCY(THREADSAFE) attribute on program resource definitions, and allows user application programs to run on an open TCB to avoid unnecessary TCB switching.

YES CICS forces all CICSAPI user application programs specified with the CONCURRENCY(THREADSAFE) attribute to run under the CICS QR TCB, as if they were specified as CONCURRENCY(QUASIRENT) programs

FORCEQR=YES allows you, in a test environment, to run incompletely tested threadsafe application programs that have proved to be non-threadsafe.

The **FORCEQR** parameter applies to all programs defined as threadsafe that are not invoked as task-related user exits, global user exits, or user-replaceable modules.

FSSTAFF

The **FSSTAFF** system initialization parameter prevents transactions initiated by function-shipped **EXEC CICS START** requests being started against incorrect terminals.

FSSTAFF={YES|NO}

Specify this parameter in an application-owning region (AOR). You might need to code the function-shipped START affinity (FSSTAFF) parameter in an AOR if all of the following are true:

1. The AOR is connected to two or more terminal-owning regions (TORs) that use the same, or a similar, set of terminal identifiers.
2. One or more of the TORs issues EXEC CICS START requests for transactions in the AOR.
3. The START requests are associated with terminals.
4. You are using shippable terminals, rather than statically defining remote terminals in the AOR.

Consider the following scenario:

Terminal-owning region TOR1 issues an EXEC CICS START request for transaction TRAR, which is owned by region AOR1. It is to be run against terminal T001. Meanwhile, terminal T001 on region TOR2 has been transaction routing to AOR1; a definition of T001 has been shipped to AOR1 from TOR2. When the START request arrives at AOR1, it is shipped to TOR2, rather than TOR1, for transaction routing from terminal T001.

To prevent this situation, code YES on the FSSTAFF parameter in the AOR.

YES When a START request is received from a terminal-owning region, and a shipped definition for the terminal named on the request is already installed in the AOR, the request is always shipped back to a TOR, for routing, *across the link it was received on*, irrespective of the TOR referenced in the remote terminal definition.

If the TOR to which the START request is returned is **not** the one referenced in the installed remote terminal definition, a definition of the terminal is shipped to the AOR, and the autoinstall user program is called. Your autoinstall user program can then allocate an *alias* termid in the AOR, to avoid a conflict with the previously installed remote definition. For information about writing an autoinstall program to control the installation of shipped definitions, see the *CICS Customization Guide*.

NO When a START request is received from a terminal-owning region, and a shipped definition for the named terminal is already installed in the AOR, the request is shipped to the TOR referenced in the definition, for routing.

Note:

1. FSSTAFF has no effect:
 - On statically-defined (hard-coded) remote terminal definitions in the AOR. If you use these, START requests are always shipped to the TORs referenced in the definitions.
 - On START requests issued in the local region. It affects only START requests shipped from other regions.
 - When coded on intermediate regions in a transaction-routing path. It is effective only when coded on an application-owning region.
2. If the AOR contains no remote definition of a terminal named on a shipped START request, the “terminal not known” global user exits, XICTENF and XALTENF, are called. For details of these exits, see the *CICS Customization Guide*.

FTIMEOUT

The **FTIMEOUT** system initialization parameter specifies a timeout interval for requests made on files that are opened in RLS mode.

FTIMEOUT={30|nn}

The interval is in seconds, from 1 through 4080 (sixty eight minutes) and indicates how long VSAM should wait before terminating a request and returning an exception condition.

The default is 30 seconds.

FTIMEOUT applies to transactions that do not have a deadlock timeout interval active. If the DTIMOUT keyword of the TRANSACTION definition is specified, it is used as the file timeout value for that transaction.

GMTEXT

The **GMTEXT** system initialization parameter specifies whether the default logon message text (WELCOME TO CICS) or your own message text is to be displayed on the screen.

GMTEXT={'DFHZC2312 * WELCOME TO CICS ***'|'text'}**

The message text can be displayed by the CSGM (good morning) transaction

when a terminal is logged on to CICS through VTAM, by the CESN transaction if used to sign on to CICS, or by your own transactions using the **EXEC CICS INQUIRE SYSTEM GMMTEXT** command.

You can use apostrophes to punctuate your message, in addition to using them as message delimiters. However, you must code *two* successive apostrophes to represent a single apostrophe in your text. For example,
GMMTEXT='User''s logon message text.'

The whole message must still be enclosed by a pair of single delimiting apostrophes.

Your message text can be from 1 through 246 characters (bytes), and can extend over two lines by extending the text to column 80 on the first line, and continuing in column 1 of the second line. For example, the following might be used in the SYSIN data set:

```
*           CICS Transaction Server for z/OS, Version 4 Release 1 SYSTEM   *
GMMTEXT='An Information Development CICS Terminal-Owning Region (TOR) - C
ICSIDC. This message is to show the use of continuation lines when creating a GM
TEXT parameter in the SYSIN data set' (for first signon
```

The CSGM transaction displays this as follows (with the time appended to the end of message):

```
An Information Development CICS Terminal-Owning Region (TOR) - C
ICSIDC. This message is to show the use of continuation lines when creating a GM
TEXT parameter in the SYSIN data set 09:56:14
```

The CESN transaction displays this as follows:

```
Signon for CICS Transaction Server for z/OS, Version 4 Release 1 APPLID CICSHTH1
An Information Development CICS Terminal-Owning Region (TOR) - CICSIDC.
This message is to show the use of continuation lines when creating a GMMTEXT
parameter in the SYSIN data set
```

For any transaction other than CESN that displays the text specified by this parameter, you must use a TYPETERM with LOGONMSG(YES) for all terminals requiring the logon message. For information about using TYPETERM, see the *CICS Resource Definition Guide*.

GMTRAN

The **GMTRAN** system initialization parameter specifies the name of a transaction ID.

GMTRAN={CSGM|CESN|transaction-id}

Specify the name of the transaction that is:

1. Initiated by ATI when terminals are logged on to CICS by VTAM, and LOGONMSG(YES) is specified in the TYPETERM definition.
2. Set to be the next transaction initiated by the terminal operator following expiry of the terminal user's TIMEOUT period (specified in the External Security Manager) and either:
 - LOGONMSG(YES) and SIGNOFF(YES)
 - or
 - LOGONMSG(YES), SIGNOFF(LOGOFF) and DISCREQ(NO)

is specified in the TYPETERM definition.

§ initiated when terminals are logged on to CICS by VTAM. Do not specify the name of a remote transaction. The transaction must be capable of being automatically initiated (ATI). The default is the transaction CSGM, that displays the text specified in the GMTEXT parameter. Alternatively, you can specify the CICS signon transaction, CESN, which also displays the text specified in the GMTEXT parameter. The GMTRAN parameter can be used with the LGNMSG parameter to retrieve VTAM logon data.

GNTRAN

The **GNTRAN** system initialization parameter specifies the transaction that you want CICS to invoke when a user's terminal-timeout period expires.

GNTRAN={NO|transaction_id}

Valid values are as follows:

NO The default value, NO, specifies that no special transaction is to be executed when the timeout period expires. Instead, the user is signed off (subject to the SIGNOFF attribute of the TYPETERM resource definition for the terminal, as described below). After the signoff, if the LOGONMSG(YES) option is specified in the TYPETERM resource definition for the terminal, the transaction specified in the **GNTRAN** system initialization parameter is executed.

transaction_id

The name of a timeout transaction to signoff the user at the timed-out terminal. You can specify CESF as the timeout transaction. Specifying your own transaction allows you to specify functions in addition to, or instead of, signoff. For example, your own transaction could issue a prompt for the terminal user's password, and allow the session to continue if the correct password is entered.

The transaction to be used must have been specially written to handle the GNTRAN COMMAREA that is passed to it. Of the CICS-supplied transactions, only CESF has been written to handle the GNTRAN COMMAREA. For more information about writing your own transactions for GNTRAN, see in the *CICS Customization Guide*.

Note: When either the CICS CESF transaction, or your own transaction, attempts to sign off a terminal, the result is subject to the SIGNOFF attribute of the TYPETERM resource definition for the terminal, as follows:

SIGNOFF

Effect

YES The terminal is signed off, but not logged off.

NO The terminal remains signed on and logged on.

LOGOFF

The terminal is both signed off and logged off.

Note: If GNTRAN fails to attach, and SIGNOFF(LOGOFF) has been specified, the terminal which has reached timeout will be signed off and logged off. GNTRAN will not run and will have no effect.

GRNAME

The **GRNAME** system initialization parameter specifies the VTAM generic resource name, as 1 through 8 characters, under which a group of CICS terminal-owning regions in a CICSplex register to VTAM.

GRNAME=name

There is no default for the **GRNAME** parameter. If you do not specify a value for this parameter, CICS does not register itself with the VTAM generic resources function. Do not confuse the term *generic applid* with *generic resource name*. Generic APPLIDs apply only to CICS regions that use XRF. Generic resource names apply only to VTAM generic resource groups.

Note:

1. If you are operating a CICSplex that comprises separate terminal-owning regions and application-owning regions, you should ensure that you define a VTAM generic resource name to the CICS terminal-owning regions only.
2. If you specify the XRF=YES parameter, do not specify a value for the **GRNAME** system initialization parameter. Any value specified is set to blanks.
3. If you specify the XRF=NO parameter, and a value for GRNAME, do not specify a specific applid (name2) for the **APPLID** system initialization parameter. Any specific applid (name2) specified for the **APPLID** parameter is used for the generic applid (name1); that is, the CICS region will be known to VTAM by the value of the specific applid.
4. Generic resource names must be unique within a single network. A generic resource cannot be identical to:
 - A USERVAR
 - An alias name
 - A real LU name

Note: It is the responsibility of the user to see that these rules are kept.

5. The first character of the **GRNAME** parameter value cannot be a number.

For example, a CICS region with the system initialization parameters:

```
APPLID=CICSHTH1
GRNAME=CICSH###
```

would register to VTAM with the applid CICSHTH1 and the generic resource CICSH###. Other LUs in the same sysplex can communicate with the CICS region either through the generic resource or the applid.

However, care should be taken with LU6 connections initiated from this side (such as AUTOCONNECT(YES)) because the bind will now contain the generic resource name and may fail if the partner only knows this region by the applid. Binds initiated from the partner are examined to identify the name by which the partner knows this region (generic resource or applid), thus allowing the appropriate connection to be built. Recommendations on defining connections can be found in the *CICS Intercommunication Guide*.

GRPLIST

The **GRPLIST** system initialization parameter specifies the names (each 1 - 8 characters) of up to four lists of resource definition groups on the CICS system definition (CSD) file.

GRPLIST={DFHLIST |name| (name [,name2] [,name3] [,name4]) }

The resource definitions in all the groups in the specified lists are loaded during initialization when CICS performs a cold start. If a warm or emergency start is performed, the resource definitions are derived from the global catalog, and the **GRPLIST** parameter is ignored.

Each name can be either a real group list name or a generic group list name that incorporates global filename characters (+ and *). If you specify more than one group list (either by specifically coding two or more group list names or by coding a group list name with global filename characters), the later group lists are concatenated onto the first group list. Any duplicate resource definitions in later group lists override those in earlier group lists.

Use the CEDAS command LOCK to protect the lists of resource groups specified on the GRPLIST parameter.

The default is DFHLIST, the CICS-supplied list that specifies the set of resource definitions needed by CICS. If you create your own group list, either add to it the groups specified in DFHLIST (omitting only those for CICS functions that you know you do not need) or specify the DFHLIST name on the **GRPLIST** parameter. Do not code GRPLIST=NO unless you have a group list named NO.

Note:

1. Group lists specified by a generic group list name are concatenated in alphabetic then numeric order. For example, the generic list name CICSHT* would concatenate the group lists CICSHT#1, CICSHTAP, CICSHTSD, and CICSHT3V in that order. If the order of concatenation is important (for example, to ensure that a particular resource definition overrides another), consider coding real group list names.
2. If a group list contains resource definitions that are needed by another group list, the group list containing those definitions must be installed first. For example, if list A has TYPETERM definitions needed for TERMINAL definitions in list B, list A must be installed first. Therefore, you might need to specifically name the prerequisite group on the **GRPLIST** parameter.
3. Take care when using generic group list names, because if a group list on your CSD satisfies the generic name, it will be installed. This means that a group list can be installed more than once; for example, if you specify the real group list name and a generic group list name that it satisfies, or if you specify two generic group list names that the group list name satisfies.
4. To override one or more of the group lists specified on the **GRPLIST** system initialization parameter, you must specify all list names (both real and generic) that you want to use, even if you are not changing the names.

For example, to use the four group lists CICSHT#1, CICSHTAP, CICSHTSD, and CICSHT3V, you could specify either of the following system initialization parameters:

```
GRPLIST=(CICSHT*)  
GRPLIST=(CICSHT#1,CICSHTAP,CICSHT3V,CICSHTSD)
```

In the first example, the group lists are loaded in the order CICSHT#1, CICSHTAP, CICSHTSD, then CICSHT3V. Resource definitions installed from the CICSHT3V group list override any duplicate definitions installed by the other groups.

In the second example, the group lists are loaded in the order specified. Resource definitions installed from the CICSHTSD group list override any duplicate definitions installed by the other groups.

If you specify GRPLIST=(CICSHT#1,CICSAP*,CICSHT3V,CICSHTSD) and you want to replace the list CICSHT3V with the list ANOLST05, specify the override:

```
GRPLIST=(CICSHT#1,CICSAP*,ANOLST05,CICSHTSD)
```

In general, any required resource definitions should appear in one of the group lists specified on the **GRPLIST** system initialization parameter.

For information about resource definitions, groups, lists, and the CSD, see the *CICS Resource Definition Guide*.

GTFTR

The **GTFTR** system initialization parameter specifies whether CICS can use the MVS generalized trace facility (GTF) as a destination for trace data.

GTFTR={OFF|ON}

This parameter controls whether any of the three types of CICS trace entry are written to GTF data sets. The three types are: CICS system trace (see the SYSTR parameter), user trace (see the USERTR parameter), and exception trace entries (which are always made and not controlled by a system initialization parameter).

OFF CICS does not use GTF as a destination for CICS trace data.

ON CICS uses GTF as a destination for CICS trace data. To use the GTF data sets for CICS trace data, you must have started GTF with the USR option, in addition to coding GTFTR=ON.

For information about GTF, see the *OS/390 MVS Diagnosis: Tools and Service Aids* manual, SY28-1985.

HPO

The **HPO** system initialization parameter specifies whether you want to use the VTAM authorized path feature of the high performance option (HPO).

HPO={NO|YES}

If you code YES, the CICS type 6 SVC must be link-edited in your MVS nucleus, and defined to MVS in an SVC Parm statement. If the SVC number is not 215 (the default) you must specify the SVC number on the **SRBSVC** parameter.

For information about installing the CICS type 6 SVC in your MVS system, and about changing the default number, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the HPO parameter in the SIT only.

ICP

The **ICP** system initialization parameter specifies that you want to cold start the interval control program.

ICP=COLD

If COLD is not specified, the ICP start type will be determined by the **START** and **TS** parameter values. See "Defining CICS resource table and module keywords" on page 138 for further information.

ICV

The **ICV** system initialization parameter specifies the region exit time interval in milliseconds.

ICV={1000|number}

The **ICV** system initialization parameter specifies the maximum time in milliseconds that CICS releases control to the operating system when there are no transactions ready to resume processing. This time interval can be any

integer in the range 100 through 3600000 milliseconds (specifying an interval up to 60 minutes). A typical range of operation might be 100 through 2000 milliseconds.

A low value interval can enable much of the CICS nucleus to be retained in dynamic storage, and not be paged-out at times of low terminal activity. This reduces the amount of dynamic storage paging necessary for CICS to process terminal transactions (thus representing a potential reduction in response time), sometimes at the expense of concurrent batch region throughput. Large networks with high terminal activity are inclined to run CICS without a need for this value, except to handle the occasional, but unpredictable, period of inactivity. These networks can usually function with a large interval (10000 to 3600000 milliseconds). Once a task has been initiated, its requests for terminal services and the completion of the services are recognized by the system and this maximum delay interval is overridden.

Small systems, or those with low terminal activity, are subject to paging introduced by other jobs running in competition with CICS. By specifying a low value interval, key portions of the CICS nucleus are referenced more frequently, thus reducing the probability of these pages being paged-out. However, the execution of the logic without performing productive work might be considered wasteful. The need to increase the probability of residency by frequent but unproductive referencing must be weighed against the overhead and response time degradation incurred by allowing the paging to occur. By increasing the interval size, less unproductive work is performed at the expense of performance if paging occurs during the periods of CICS activity. For information about the effect of ICV on performance, see the *CICS Performance Guide*.

Note: The region exit time interval process contains a mechanism to ensure that CICS does not constantly set and cancel timers (thus degrading performance) while attempting to meet its objectives for a low region exit time interval. This mechanism can cause CICS to release control to the operating system for up to 0.5 seconds when the interval has been set at less than 250; and up to 0.25 seconds more than the region exit time interval when the interval has been set greater than 250.

ICVR

The **ICVR** system initialization parameter specifies the default runaway task time interval in milliseconds as a decimal number.

ICVR={5000|number}

You can specify zero, or a number in the range 500 through 2 700 000, in multiples of 500. CICS rounds down values that are not multiples of 500. This is the RUNAWAY interval used by transactions defined with RUNAWAY=SYSTEM. See the *CICS Resource Definition Guide* for further information.

CICS may purge a task if it has not given up control after the RUNAWAY interval for the transaction (or ICVR if the transaction definition specified RUNAWAY=SYSTEM). If you code ICVR=0, runaway task control is inoperative for transactions specifying RUNAWAY=SYSTEM in their transaction definition (that is, tasks do not get purged if they appear to be looping). The ICVR value is independent of the ICV value, and can be less than the ICV value. Note that CICS runaway task detection is based upon task time, that is, the interval is decremented only when the task has control of the processor.

For information about commands that reinitialize the ICVR value, see the *CICS Problem Determination Guide*.

ICVTSD

The **ICVTSD** system initialization parameter specifies the terminal scan delay value.

ICVTSD={500|number}

The terminal scan delay facility determines how quickly CICS deals with some terminal I/O requests made by applications. The range is 0 through 5000 milliseconds, with a default of ICVTSD=500.

There is an overhead in dealing with such requests. By specifying a nonzero value, the overhead may be spread over several transactions. A value close to zero (for example 200) would be adequate.

IIOPLISTENER

The **IIOPLISTENER** system initialization parameter specifies whether the CICS region is to function as an IIOP listener region.

IIOPLISTENER={YES|NO}

Valid values are as follows:

YES The CICS region is an IIOP listener region, or a combined listener and application owning region (AOR).

NO The CICS region is an IIOP application owning region, and the *CICS Resource Definition Guide* installed in the region, that specify PROTOCOL(IIOP), cannot be opened.

This parameter has no effect if the region is not an IIOP listener region or an AOR.

For more information about IIOP listener regions and AORs, see *Java Applications in CICS*

INFOCENTER

The **INFOCENTER** system initialization parameter specifies the server name of where the CICS Information Center is installed and the port number that it uses to run in server mode.

INFOCENTER=servername:portnumber

The port number is specified in the start up script for the information center. The default value is 29127, but you can change it to a suitable number by editing the script file. CICS-supplied transactions with a Web browser interface use the value of this parameter to construct links to topics in the information center.

INFOCENTER=http://server_name:29127

If you do not code this parameter, CICS does not construct links to the information center.

If you add this parameter to the Web User Interface (WUI) CICS startup JCL, a hyperlink labelled **Information Center** is displayed on WUI views and menus. Clicking on this hyperlink opens a new browser window displaying the CICS Information Center home page.

INITPARM

The **INITPARM** system initialization parameter specifies that parameters are to be passed to application programs that use the **ASSIGN INITPARM** command.

INITPARM=(pgmname_1='parmstring_1'[, ,pgmname_n='parmstring_n'])

You can use **INITPARM** to pass parameters to PLTPI programs to be executed in the final stages of system initialization. The area giving access to the parameters is specified by the **ASSIGN INITPARM** command. For programming information about the **ASSIGN INITPARM** command, see the *CICS Application Programming Reference*.

pgmname

The name of a program. This name must be 1 through 8 alphanumeric or national language characters.

parmstring

The parameter string (up to 60 characters enclosed by single quotes) to be passed to the associated program. Any quotes imbedded in the string must be duplicated. For information on coding **INITPARM** in the **SYSIN** data set, see "Rules for coding CICS system initialization parameters in the **SYSIN** data set" on page 307.

You can specify up to 255 **pgmname='parmstring'** sets.

Note: You can specify the **INITPARM** keyword and its parameters more than once, see Sample startup job stream, note 5.

INTTR

The **INTTR** system initialization parameter specifies whether the internal CICS trace destination is to be activated at system initialization.

INTTR={ON|OFF}

This parameter controls whether any of the three types of CICS trace entry are written to the internal trace table. The three types are: CICS system trace (see the **SYSTR** parameter), user trace (see the **USERTR** parameter), and exception trace entries (which are always made and not controlled by a system initialization parameter).

ON Activate main storage trace.

OFF Do not activate main storage trace.

IRCSTRT

The **IRCSTRT** system initialization parameter specifies whether IRC is to be started up at system initialization.

IRCSTRT={NO|YES}

If **IRCSTRT=YES** is not coded, IRC can be initialized by issuing a **SET IRC OPEN** command.

ISC

The **ISC** system initialization parameter specifies whether the CICS programs required for multiregion operation (MRO) and intersystem communication over SNA are to be included.

ISC={NO|YES}

For IPIC interconnectivity (IPIC), specify **ISC=YES** and **TCPIP=YES**.

JESDI

The **JESDI** system initialization parameter specifies, in a SIT for an alternate XRF system, the JES delay interval.

JESDI={30|number} (alternate)

The minimum is 5 seconds. The alternate CICS region has to ensure that the active CICS region has been canceled before it can take over the resources owned by the active.

Note: You must give careful consideration to the values you specify for the parameters ADI and JESDI so that they do not conflict with your installation's policy on PR/SM RESETTIME and the XCF INTERVAL and OPNOTIFY intervals. You should ensure that the sum of the interval you specify for ADI plus JESDI exceeds the interval specified by the XCF INTERVAL and the PR/SM policy interval RESETTIME.

JVMCCSIZE

The **JVMCCSIZE** system initialization parameter specifies the size of the shared class cache on an initial or cold start of CICS.

JVMCCSIZE={24M|number}

The size of the shared class cache can be between 8 MB and 2047 MB. You can specify the number in bytes, or as a whole number of kilobytes followed by the letter K, or as a whole number of megabytes followed by the letter M. The default value is 24 MB (specified as 24M). You can use the CEMT PERFORM CLASSCACHE START or RELOAD command (or the equivalent EXEC CICS command) to change the size of the shared class cache while CICS is running.

Warm or emergency CICS starts use the size specified for the shared class cache during the previous CICS execution. With Version 6, the shared class cache normally persists across warm or emergency starts. If you specify the JVMCCSIZE system initialization parameter as an override, CICS ignores it.

JVMCCSTART

The **JVMCCSTART** system initialization parameter determines the startup behavior for the shared class cache.

JVMCCSTART={AUTO|YES|NO}

AUTO

Autostart is enabled, so the shared class cache starts as soon as a JVM needs it.

YES

On an initial or cold start, or if the shared class cache did not persist, the startup behavior with JVMCCSTART=YES is the same as the behavior with JVMCCSTART=AUTO.

If you need the shared class cache to start up at CICS initialization, you can write an initialization program (PLTPI program) and define it to CICS in a program list table (PLT) to run immediately after CICS initialization is complete. In the program, use the PERFORM JVMPOOL command to manually start one or more JVMs whose profile requires the use of the shared class cache. This makes the shared class cache start up if it does not already exist.

NO Autostart is disabled, so the shared class cache does not start until you issue a CEMT PERFORM CLASSCACHE START command, or the equivalent EXEC CICS command.

Note: You can change the status of autostart while CICS is running, either by using the AUTOSTARTST option on the CEMT PERFORM CLASSCACHE command, or by using the CEMT SET CLASSCACHE command, or the equivalent EXEC CICS commands. Warm or emergency CICS starts use the setting specified during the previous CICS execution. Because the shared class cache normally persists across warm or emergency starts, if you specify the JVMCCSTART system initialization parameter as an override, CICS ignores it. If a shared class cache was active when the system shut down, it normally persists across a warm or emergency start, except in some circumstances such as an IPL of z/OS, and the JVMCCSTART setting has no effect in this situation.

JVMxxxxTRACE

The **JVMxxxxTRACE** system initialization parameters specify the default options for JVM tracing.

JVMxxxxTRACE (JVMLEVEL0TRACE=option, JVMLEVEL1TRACE=option, JVMLEVEL2TRACE=option, JVMUSERTRACE=option)

The *CICS Problem Determination Guide* has information about the JVM trace options that you can set using the JVMLEVEL0TRACE, JVMLEVEL1TRACE, JVMLEVEL2TRACE and JVMUSERTRACE system initialization parameters. There is further information about JVM trace and about problem determination for JVMs in the *IBM Developer Kit and Runtime Environment, Java Technology Edition Diagnostics Guide*, which is available to download from www.ibm.com/developerworks/java/jdk/diagnosis/.

Trace levels 29–32 for the SJ component correspond to JVMLEVEL0TRACE, JVMLEVEL1TRACE, JVMLEVEL2TRACE and JVMUSERTRACE respectively. To activate JVM tracing, specify level numbers 29–32 on the SPCTRSJ or STNTRSJ system initialization parameter, or use the CETR transaction.

Note that JVM trace can produce a large amount of output, so you should normally activate JVM trace for special transactions, rather than turning it on globally for all transactions.

For definitions of the individual JVM tracing parameters, see “JVMLEVEL0TRACE,” “JVMLEVEL1TRACE” on page 196, “JVMLEVEL2TRACE” on page 196, and “JVMUSERTRACE” on page 196.

Restrictions: You can specify the JVMxxxxTRACE parameters in PARM, SYSIN, or CONSOLE only.

JVMLEVEL0TRACE

The **JVMLEVEL0TRACE** system initialization parameter specifies the default option for JVM Level 0 trace, corresponding to trace level 29 of the SJ component.

JVMLEVEL0TRACE={ 'ALL (EXCEPTION)' | 'user override string' }

The default setting for this level of tracing maps to trace point level 0 for JVMs, which is reserved for extraordinary events and errors. Unlike CICS exception trace, which cannot be switched off, the JVM Level 0 trace is normally switched off unless JVM tracing is required. “JVMxxxxTRACE” has more information about these system initialization parameters.

JVMLEVEL1TRACE

The **JVMLEVEL1TRACE** system initialization parameter specifies the default option for JVM Level 1 trace, corresponding to trace level 30 of the SJ component.

JVMLEVEL1TRACE={'ALL(ENTRY,EXIT)' | '*user override string*'}

The default setting for this level of tracing maps to trace point level 1 for JVMs. "JVMxxxxTRACE" on page 195 has more information about these system initialization parameters.

JVMLEVEL2TRACE

The **JVMLEVEL2TRACE** system initialization parameter specifies the default option for JVM Level 2 trace, corresponding to trace level 31 of the SJ component.

JVMLEVEL2TRACE={'ALL' | '*user override string*'}

The default setting for this level of tracing maps to trace point level 2 for JVMs. Note that the JVM trace point levels go up to level 9. "JVMxxxxTRACE" on page 195 has more information about these system initialization parameters.

JVMUSERTRACE

The **JVMUSERTRACE** system initialization parameter specifies the default option for JVM user trace, corresponding to trace level 32 of the SJ component.

JVMUSERTRACE={'NONE' | '*user override string*'}

Use this option for more complex specifications for JVM tracing. "JVMxxxxTRACE" on page 195 has more information about these system initialization parameters.

JVMPROFILEDIR

The **JVMPROFILEDIR** system initialization parameter specifies the name (up to 240 characters long) of a z/OS UNIX directory that contains the JVM profiles for CICS. CICS searches this directory for the profiles it needs to configure JVMs.

JVMPROFILEDIR={*USSHOME value*/**JVMProfiles** | *directory*}

The default value of **JVMPROFILEDIR** is the value of the **USSHOME** system initialization parameter, which specifies the name and path of the root directory for CICS files on z/OS UNIX, followed by the subdirectory **JVMProfiles**. The default value for the **USSHOME** system initialization parameter is `/usr/lpp/cicsts/cicsts41`, so if that default value is used, the default value of **JVMPROFILEDIR** is `/usr/lpp/cicsts/cicsts41/JVMProfiles`.

The **USSHOME** system initialization parameter matches the root directory that you specified for CICS TS 4.1 files on z/OS UNIX when you installed CICS using the DFHISTAR installation job. The directory name that you specify in DFHISTAR is passed to the *uss_path* variable used by the DFHIJVMJ job which creates the sample JVM profiles on z/OS UNIX, and the sample JVM profiles are created in the **JVMProfiles** subdirectory of the root directory. The supplied setting for **JVMPROFILEDIR** therefore points to the **default** directory for the sample JVM profiles. If you chose a different name during CICS installation for this directory, or if you want CICS to load the JVM profiles from a directory other than the samples directory, you must do one of the following:

- Change the value of the **JVMPROFILEDIR** system initialization parameter.
- Link to your JVM profiles from the directory specified by **JVMPROFILEDIR** using UNIX soft links. Use this method to store your JVM profiles in any place in the z/OS UNIX file system.

Note that the supplied sample JVM profiles DFHJVMPR and DFHJVMCD must always be available to CICS. CICS uses these JVM profiles for a number of functions. Both these JVM profiles must therefore either be present in the directory that is specified by **JVMPROFILEDIR**, or linked to using UNIX soft links from that directory.

KEYRING

The **KEYRING** system initialization parameter specifies the fully qualified name of the key ring, within the external security manager's database, that contains the keys and X.509 certificates used by CICS support for the secure sockets layer (SSL) and for Web services security.

KEYRING=*keyring-name*

The maximum length of the **KEYRING** parameter is 47 characters. For more information on creating a key ring file, see in the *CICS RACF Security Guide*

The key ring name is case sensitive.

LGDFINT

The **LGDFINT** system initialization parameter specifies the log defer interval to be used by CICS log manager when determining how long to delay a forced journal write request before invoking the MVS system logger.

LGDFINT={5|*number*}

The value is specified in milliseconds.

5 This is the default. When this parameter was first introduced, the default value was 30 milliseconds, but customer experience has shown that 5 is a more realistic value.

number

number can be any value in the range 0 through 65535. You are recommended to allow **LGDFINT** to assume its default value, 5.

You can modify the log defer interval dynamically using the LOGDEFER option of the **CEMT SET SYSTEM** or **EXEC CICS SET SYSTEM** command. However, you are recommended not to modify this value in a production environment without first performing a system evaluation and performance analysis of any changed value.

If you change the log defer interval value dynamically, the new value is not cataloged. The log defer interval value is taken from the **LGDFINT** system initialization parameter in all types of CICS startup.

When a CICS system has many tasks issuing forced log write requests, these tasks will not be delayed for periods close to the **LGDFINT** parameter value. This is because a forced log write request is normally issued while a log deferral is already being performed for another task. The actual interval might also be affected by the need for tasks to wait across a partition exit.

LGNMSG

The **LGNMSG** system initialization parameter specifies whether VTAM logon data is to be made available to an application program.

LGNMSG={NO|**YES**}

Valid values are as follows:

NO VTAM logon data is not available to an application program.

YES VTAM logon data is available to an application program. The data can be retrieved with an EXEC CICS EXTRACT LOGONMSG command. For programming information about this command, see the *CICS Application Programming Reference*.

You can use this parameter with the **GMTRAN** parameter to retrieve the VTAM logon data at the time a terminal is logged on to CICS by VTAM.

LLACOPY

The LLACOPY system initialization parameter specifies whether CICS is to use the LLACOPY macro or the BLDL macro when locating modules in the DFHRPL or dynamic LIBRARY concatenation.

LLACOPY={YES|NO|NEWCOPY}

Valid values are as follows:

YES CICS always uses the LLACOPY macro when locating modules in the DFHRPL or dynamic LIBRARY concatenation.

NO CICS always uses the BLDL macro when locating modules in the DFHRPL or dynamic LIBRARY concatenation.

NEWCOPY

CICS uses the LLACOPY only when a NEWCOPY or a PHASEIN is being performed. At all other times, CICS uses the BLDL macro when locating modules in the DFHRPL or dynamic LIBRARY concatenation.

Note:

1. If you code LLACOPY=NO or LLACOPY=NEWCOPY you can still benefit from having LLA managed data sets within your DFHRPL or dynamic LIBRARY concatenation. Modules will continue to be loaded from VLF if appropriate.
2. If an LLA managed module has been altered, a BLDL macro may not return the new information and a subsequent load will still return the old copy of the module. To load the new module, an LLACOPY must be issued against that module or a MODIFY LLA,REFRESH command must be issued on a system console.
3. If you set LLACOPY to anything other than NO, ensure that the proper RACF security permissions have been set up first. For more information about this refer to the *CICS RACF Security Guide*.

LOCALCCSID

The LOCALCCSID system initialization parameter specifies the default CCSID for the local region.

LOCALCCSID={037|CCSID}

The CCSID is a value of up to 8 characters. If CCSID value is not specified, the default LOCALCCSID is set to 037. For lists of valid CCSIDs, see:

- *CICS Intercommunication Guide*
- Appendix F of the *z/OS Support for Unicode: Using Conversion Services* manual SA22-7649 .

037 the default value for LOCALCCSID.

CCSID

represents any other valid CCSID value.

LPA

The **LPA** system initialization parameter specifies whether any CICS or user modules can be used from the link pack areas.

LPA={NO|YES}

Valid values are as follows:

- NO** will not load CICS or user modules from the link pack areas.
- YES** CICS or usermodules installed in the LPA or in the ELPA can be used from there, instead of being loaded into the CICS region.

A list of the CICS modules that are read-only, and hence eligible for residence in the link pack areas (LPA or ELPA), are contained in the SMP/E USERMOD supplied on the distribution tape in the CICSTS41.CICS.SDFHSAMP, in a member called DFH\$UMOD. For details of the CICS system initialization parameter PRVMOD that you can use to override LPA=YES for selected modules, see page PRVMOD.

MAXJVMTCBS

The **MAXJVMTCBS** system initialization parameter specifies the maximum number of open TCBs CICS can create in the pool of J8 and J9 mode TCBs for use by Java programs that run in a JVM (the JVM pool).

MAXJVMTCBS={5|number}

You can specify a limit in the range 1 through 999. Within this limit, there are no constraints on how many of the TCBs in the JVM pool are J9 TCBs, and how many are J8 TCBs.

The minimum permitted value is 1, meaning that CICS is always able to create at least 1 open TCB for use by a JVM, of either J8 or J9 mode.

JM TCBs, used for management of the shared class cache, do not count towards the MAXJVMTCBS limit.

“System initialization parameters for open TCBs” on page 134 has more information about managing open TCBs.

MAXOPENTCBS

The **MAXOPENTCBS** system initialization parameter specifies the maximum number, in the range 1 through 2000, of open TCBs CICS can create in the pool of L8 and L9 mode TCBs.

MAXOPENTCBS={12|number}

Within this limit, there are no constraints on how many of the TCBs in the pool are L8 TCBs, and how many are L9 TCBs.

- L9 mode TCBs are used for USERKEY OPENAPI application programs.
- L8 mode TCBs are used:
 - for CICSKEY OPENAPI application programs
 - for OPENAPI task related user exits, for example the CICS-DB2 and WebSphere® MQ Attachment Facility. (Task related user exits always run in CICSKEY.)
 - and by CICS itself, because CICS uses OPENAPI CICSKEY programs which run on L8 TCBs:
 - when accessing doctemplates and HTTP static responses that are stored on z/OS UNIX.
 - when processing WebService requests and parsing XML.

The default is 12 . The minimum permitted value is 1.

Note: If you set the system initialization parameters MAXOPENTCBS=1 and TCPIP=YES, and do not have any TCPIPSEVICES, issuing a CEDA INSTALL List (*listname*) command where List contains a TCPIPSERVICE causes CEDA to wait on DISPATCH for an OPENTCB. This situation means that CICS will not start.

For more information about managing open TCBs, see “System initialization parameters for open TCBs” on page 134.

MAXSOCKETS

The **MAXSOCKETS** system initialization parameter specifies the maximum number of IP sockets that can be managed by the CICS sockets domain.

MAXSOCKETS={65535|*number*}

Set a suitable value that does not exceed the maximum value as defined in the **MAXFILEPROC** parameter in SYS1.PARMLIB member BPXPRMxx. If you specify a value greater than the **MAXFILEPROC** parameter or if you allow CICS to use the default, CICS issues message DFHSO0124, which specifies the value that CICS has used for this parameter. If the CICS region user ID has superuser authority, the **MAXFILEPROC** parameter does not limit the setting of **MAXSOCKETS**.

The maximum number of sockets must be greater than the maximum number of inbound and outbound sockets used by CICS including the number of TCPIPSERVICE resources in service.

MAXSSLTCBS

The **MAXSSLTCBS** system initialization parameter specifies the maximum number of S8 TCBs that can run in the SSL pool.

MAXSSLTCBS={8|*number*}

The default is 8, but you can specify up to 1024 TCBs.

This value must not exceed the **MAXTHREADS** and **MAXTHREADTASKS** parameter values, that are specified in SYS1.PARMLIB member BPXPRMxx.

MAXXPTCBS

The **MAXXPTCBS** system initialization parameter specifies the maximum number, in the range 1 through 999, of open X8 and X9 TCBs that can exist concurrently in the CICS region.

MAXXPTCBS={5|*number*}

X8 and X9 are the TCBs that are used to provide XPLink support. “System initialization parameters for open TCBs” on page 134 has more information about managing open TCBs.

MCT

The **MCT** system initialization parameter specifies the monitoring control table suffix.

MCT={NO|**YES**|*xx*}

If you specify MCT=NO, CICS monitoring builds dynamically a default MCT, ensuring that default monitoring control table entries are always available for use when monitoring is on and a monitoring class (or classes) is active. You can generate an MCT with a single-character suffix only for use by CICS

because single-character suffixes cause an error when the MCT is processed by DFHMNDUP. If you use DFHMNDUP, make sure that you create your MCTs with two-character suffixes.

For information about coding the macros for this table, see the *CICS Operations and Utilities Guide*.

MN

The **MN** system initialization parameter specifies whether monitoring is to be switched on or off at initialization.

MN={OFF|ON}

Use the individual monitoring class system initialization parameters to control which monitoring classes are to be active (see the MNEXC, MNPER, and MNRES parameter descriptions.) The default status is that the CICS monitoring facility is **off**. The monitoring status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Switch off monitoring.

ON Switch on monitoring. However, unless at least one individual class is active, no monitoring records are written.

Note:

1. If the monitoring status is ON, CICS accumulates monitoring data continuously. For each monitoring class that is active, CICS writes the monitoring data to a system management facilities (SMF) data set. If the monitoring status is OFF, CICS does not accumulate or write any monitoring data, even if any of the monitoring classes are active.
2. You can change the monitoring status and the monitoring class settings at any time, as follows:
 - During a warm restart by coding system initialization parameters in PARM, SYSIN, or through the system console.
 - While CICS is running, using:
 - The monitoring facility transaction CEMN.
 - The CEMT SET MONITOR command.
 - The EXEC CICS SET MONITOR command.

When you change the status of monitoring, the change takes effect immediately. If you change the monitoring status from OFF to ON, monitoring starts to accumulate data and write monitoring records to SMF for all tasks that start after the status change is made, for all active monitoring classes. If the status is changed from ON to OFF, monitoring stops writing records immediately and does not accumulate monitoring data for any tasks that start after the status change is made.

3. The monitoring status setting can be manipulated independently of the class settings. This means that, even if the monitoring status is OFF, you can change the monitoring class settings, and the changes take effect for all tasks that are started after the monitoring status is next set to ON.

MNCONV

The **MNCONV** system initialization parameter specifies whether or not conversational tasks are to have separate performance class records produced for each pair of terminal control I/O requests.

MNCONV={NO|YES}

Any clock (including user-defined) that is active at the time such a performance class record is produced is stopped immediately before the record is written. After the record is written, such a clock is reset to zero and restarted. Thus a clock whose activity spans more than one recording interval within the conversational task appears in multiple records, each showing part of the time, and the parts adding up to the total time the clock is active. The high-water-mark fields (which record maximum levels of storage used) are reset to their current values. All other fields are set to X'00', except for the key fields (transid, termid). The monitoring converse status is recorded in the CICS global catalog for use during warm and emergency restarts.

MNEXC

The **MNEXC** system initialization parameter specifies whether the monitoring exception class is to be made active during initialization.

MNEXC={OFF|ON}

The monitoring exception class status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Set the exception monitoring class to “not active”.

ON Set the exception monitoring class to “active”.

For programming information about exception monitoring records, see the *CICS Customization Guide*.

MNFREQ

The **MNFREQ** system initialization parameter specifies the interval for which CICS automatically produces a transaction performance class record for any long-running transaction.

MNFREQ={0|hhmmss}

The monitoring frequency value is recorded in the CICS global catalog for use during warm and emergency restarts. CICS can produce a performance class monitoring record in this way only when the long-running transaction is running on the QR or CO TCBs.

0 No frequency monitoring is active.

hhmmss

The interval for which monitoring produces automatically a transaction performance class record for any long-running transaction. Specify a 1 to 6 digit number in the range 000100–240000. Numbers that are fewer than six digits are padded with leading zeroes.

MNIDN

The **MNIDN** system initialization parameter specifies whether the monitoring identity class is to be made active during CICS initialization.

MNIDN={OFF|ON}

The monitoring identity class status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Set identity monitoring class to not active.

ON Set identity monitoring class to active.

MNPER

The **MNPER** system initialization parameter specifies whether the monitoring performance class is to be made active during CICS initialization.

MNPER={OFF|ON}

The monitoring performance class status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Set the performance monitoring class to “not active”.

ON Set the performance monitoring class to “active”.

For programming information about performance monitoring records, see the *CICS Customization Guide*.

MNRES

The **MNRES** system initialization parameter specifies whether transaction resource monitoring is to be made active during CICS initialization.

MNRES={OFF|ON}

The transaction resource monitoring class status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Set transaction resource monitoring to not active.

ON Set transaction resource monitoring to active.

Transaction resource monitoring applies to CICS file resources when you specify the `FILE=nm` option on the `DFHMCT TYPE=INITIAL` macro.

MNSYNC

The **MNSYNC** system initialization parameter specifies whether you want CICS to produce a transaction performance class record when a transaction takes an implicit or explicit syncpoint (unit-of-work).

MNSYNC={NO|YES}

No action is taken for syncpoint rollbacks. The monitoring syncpoint status is recorded in the CICS global catalog for use during warm and emergency restarts.

MNTIME

The **MNTIME** system initialization parameter specifies whether you want the time stamp fields in the performance class monitoring data to be returned to an application using the **EXEC CICS COLLECT STATISTICS MONITOR(taskno)** command in either GMT or local time.

MNTIME={GMT|LOCAL}

The monitoring time value is recorded in the CICS global catalog for use during warm and emergency restarts.

For programming information on the **EXEC CICS COLLECT STATISTICS** command, see the *CICS System Programming Reference*.

MQCONN

The **MQCONN** system initialization parameter specifies whether you want CICS to start a connection to WebSphere MQ automatically during initialization.

MQCONN={NO|YES}

NO Do not automatically call `DFHMQCOD`, the CICS-WebSphere MQ adapter program, during initialization.

YES

Call the CICS-WebSphere MQ adapter program, DFHMQCOD, automatically during CICS initialization. The **MQCONN** parameter always uses program DFHMQCOD to start the CICS-WebSphere MQ connection. It cannot be customized to use a user-supplied attach program of a different name.

When you specify **MQCONN=YES**, the information that CICS needs to start the connection to WebSphere MQ, such as the name of a WebSphere MQ queue manager or queue-sharing group, is taken from the **MQCONN** resource definition for the CICS region.

An **MQCONN** resource definition must be installed before CICS can start the connection to WebSphere MQ. When you start the connection automatically at CICS initialization, for an initial or cold start, the **MQCONN** resource definition must be present in one of the groups named in the list or lists named by the **GRPLIST** system initialization parameter. For a warm or emergency start of CICS, the **MQCONN** resource definition must have been installed by the end of the previous CICS run.

When you specify **MQCONN=YES**, you do not need to define the CICS-WebSphere MQ adapter program in the CICS post initialization program list table (PLT). For more information about starting and customizing a connection to WebSphere MQ, see *Customizing the CICS-WebSphere MQ adapter*.

MROBTCH

The **MROBTCH** system initialization parameter specifies the number of events that must occur before CICS is posted for dispatch due to the batching mechanism.

MROBTCH={1|number}

The number can be in the range 1 through 255, and the default is 1.

Use this batching mechanism to spread the overhead of dispatching CICS over several tasks. If the value is greater than 1 and CICS is in a system wait, CICS is not posted for dispatch until the specified number of events has occurred. Events include MRO requests from connected systems or DASD I/O and **CHANGE_MODE** processing. For these events, CICS is dispatched as soon as one of the following occurs:

- The current batch fills up (the number of events equals **MROBTCH**)
- An **ICV** interval expires

Therefore, ensure that the time interval you specify in the **ICV** parameter is low enough to prevent undue delay to the system.

If CICS is dispatched for another reason, the current batch is dealt with in that dispatch of CICS.

Note: During periods of low utilization, a value of **MROBTCH** greater than 1 may result in increased transaction response times. Transactions issuing file I/O requests may be delayed due to increased **FCIOWAIT**. For further guidance information about the effect of **MROBTCH** on performance, see the *CICS Performance Guide*.

MROFSE

The **MROFSE** system initialization parameter specifies whether you want to extend the lifetime of the long-running mirror to keep it allocated until the end of the task rather than after a user syncpoint for function shipping applications.

MROFSE={NO|YES}

Valid values are as follows:

NO The lifetime of the MRO long-running mirror is not extended.

YES The mirror task remains available to the application until the end of the application's task. This extended long-running mirror saves the overhead of re-attaching the mirror task following a user syncpoint.

This parameter is ignored for DPL requests (that is a DPL causes the session to be freed at the next syncpoint even if it has been kept for a previous sequence of syncpoints).

It should be used with caution especially if DPL requests with SYNCONRETURN or TRANSID are used. For additional information, see the *CICS Intercommunication Guide*

Do not specify this value in the front-end region when long running tasks might be used to function-ship requests. This because a SEND session is unavailable for allocation to other tasks when unused. Specifying MROFSE=YES could prevent the connection from being released when contact has been lost with the back-end region, until the task terminates or issues a function-shipped request.

MROLRM

The **MROLRM** system initialization parameter specifies whether you want to establish an MRO long-running mirror task.

MROLRM={NO|YES}

Valid values are as follows:

NO The MRO long-running mirror task is not required.

YES The mirror transaction remains available to the application issuing the remote request. This long-running mirror saves the overhead of re-establishing communication with the mirror transaction if the application makes more function shipping requests in this unit of work.

For information about long-running mirror tasks, see the *CICS Intercommunication Guide*.

MSGCASE

The **MSGCASE** system initialization parameter specifies how you want the message domains to display mixed case messages.

MSGCASE={MIXED|UPPER}

Messages handled by the CICS message domain and the CPSM message domain are in mixed case.

MIXED

This is the default in the SIT; all messages displayed by the CICS message domain or the CPSM message domain remain in mixed case.

UPPER

The message domain displays all mixed case messages in uppercase only.

Mixed case output is not displayed correctly on Katakana display terminals and printers. Uppercase English characters appear correctly as uppercase English characters, but lowercase appears as Katakana symbols. If you have any Katakana terminals connected to your CICS region, specify MSGCASE=UPPER.

If you want to use uppercase English for your CICS-WebSphere MQ components, you must set MSGCASE=UPPER, and ensure that ASSIGN NATLANGINUSE returns E (US English).

MSGLVL

The **MSGLVL** system initialization parameter specifies the message level that controls the generation of messages to the console and JES message log.

MSGLVL={1|0}

Valid values are as follows:

- | | |
|----------|--|
| <u>1</u> | All messages are printed or displayed. |
| <u>0</u> | Only critical errors or interactive messages are printed or displayed. |

MXT

The **MXT** system initialization parameter specifies the maximum number, in the range 1 through 999, of **user** tasks CICS allows to exist at any time.

MXT={5|number}

CICS queues requests for tasks above this number but does not action (attach) them until the number of tasks attached drops below the MXT limit.

Note that each active IIOP session requires two tasks.

You should review the region size specified on the REGION parameter for CICS address spaces. The increase in CICS use of virtual storage above the 16MB boundary means that you will probably need to increase the REGION parameter.

The introduction of the transaction isolation facility increases the allocation of some virtual storage above the 16MB boundary for CICS regions that are running with transaction isolation active.

If you are running with transaction isolation active, CICS allocates storage for task-lifetime storage in multiples of 1MB for user-key tasks that run above the 16MB boundary. (1MB is the minimum unit of storage allocation above the line for the EUDSA when transaction isolation is active.) However, although storage is allocated in multiples of 1MB above the 16MB boundary, MVS paging activity affects only the storage that is used (referenced), and unused parts of the 1MB allocation are not paged.

If you are running without transaction isolation, CICS allocates user-key task-lifetime storage above 16MB in multiples of 64KB.

The subspace group facility uses more real storage, as MVS creates for each subspace a page and segment table from real storage. The CICS requirement for real storage varies according to the transaction load at any one time. As a guideline, each task in the system requires 9KB of real storage, and this should be multiplied by the number of concurrent tasks that can be in the system at any one time (governed by the MXT system initialization parameter).

However, automatic DSA sizing removes the need for accurate storage estimates, with CICS dynamically changing the size of DSAs as demand requires.

Note: The MXT value does **not** include CICS system tasks.

NATLANG

The **NATLANG** system initialization parameter specifies the single-character codes for the languages to be supported in this CICS run.

NATLANG=(E,x,y,z,...)

The codes are listed in Table 16.

E English, which is the *system* default (that is, is provided even if you do not specifically code E).

x,y,z,... Specify the appropriate letters for the other supported languages that you require.

For the codes that you specify on this parameter, you must ensure that a DFHMET1x module (where x is the language code) is in a library in the STEPLIB DD concatenation of the CICS startup JCL. (For full language support, you must also provide other DFHMEyyx modules.) For information about using the message editing utility to create your own DFHMEyyx modules, see the *CICS Operations and Utilities Guide*.

English language support is provided, even if you do not specifically code E for English.

The first language code specifies the default language for those elements of CICS enabled to receive National Language Support (NLS) messages, such as some destinations used for CICS messages, and the terminals or users not signed-on with an NLS code. The other language codes are provided to specify the language to be used for messages sent to terminals that are defined with the appropriate language support code. For example, coding NATLANG=(F,G,S) has the same effect as coding NATLANG=(F,G,E,S); that is, in both cases the default NLS language is French (F), and the languages English, German (G), and Spanish (S) are supported. (For such support, you would have to create and install the modules DFHMET1F, DFHMET1G, and DFHMET1S into a library in the STEPLIB DD concatenation of the CICS startup JCL.)

NLS is not available to CICS console messages, which continue to be in English only.

Table 16. Languages and codes supported by CICS

| NATLANG code | NLS code | Language |
|--|----------|----------------------|
| A | ENG | Alternative English |
| Q | ARA | Arabic |
| 1 | BEL | Byelorussian |
| L | BGR | Bulgarian |
| B | PTB | Brazilian Portuguese |
| T (Double-Byte Character Set language) | CHT | Traditional Chinese |
| C (Double-Byte Character Set language) | CHS | Simplified Chinese |
| 2 | CSY | Czech |
| D | DAN | Danish |

Table 16. Languages and codes supported by CICS (continued)

| NATLANG code | NLS code | Language |
|---|----------|---------------------------|
| E | ENU | English |
| G | DEU | German |
| O | ELL | Greek |
| S | ESP | Spanish |
| W | FIN | Finnish |
| F | FRA | French |
| X | HEB | Hebrew |
| 3 | HRV | Croatian |
| 4 | HUN | Hungarian |
| J | ISL | Icelandic |
| I | ITA | Italian |
| K (Double-Byte Character Set language) | JPN | Japanese |
| H (Double-Byte Character Set languages) | KOR | Korean |
| M | MKD | Macedonian |
| 9 | NLD | Dutch |
| N | NOR | Norwegian |
| 5 | PLK | Polish |
| P | PTG | Portuguese |
| 6 | ROM | Romanian |
| R | RUS | Russian |
| Y | SHC | Serbo-Croatian (Cyrillic) |
| 7 | SHL | Serbo-Croatian (Latin) |
| V | SVE | Swedish |
| Z | THA | Thai |
| 8 | TRK | Turkish |
| U | UKR | Ukrainian |

Notes:

- The following language module suffixes are not supported by the message editing utility:
 - E - English master data sets.
 - K - Japanese data sets, where translation is performed by IBM.
 - C - Simplified Chinese data sets, where translation is performed by IBM.
- The NATLANG code is used as the suffix of the message modules for the associated language.

NCPLDFT

The NCPLDFT system initialization parameter specifies the name of the default named counter pool to be used by the CICS region on calls it makes to a named counter server.

NCPLDFT={DFHNC001|name}

If CICS cannot determine, from the named counter options table, the pool name required by an EXEC CICS named counter command, CICS uses the default name specified on the NCPLDFT parameter.

Note: This parameter is relevant to references to a named counter server made through the EXEC CICS API only. It not used by the named counter call interface.

DFHNC001

This is the default name that CICS uses as the named counter pool name if you omit the NCPLDFT system initialization parameter.

name Specifies the 8-character name to be used by CICS as the default pool name in connection with named counter API commands, when the name cannot be resolved by the named counter options table.

NEWSIT

The **NEWSIT** system initialization parameter specifies whether CICS is to load the specified SIT, and enforce the use of all system initialization parameters, modified by any system initialization parameters provided by PARM, SYSIN, or the system console, even in a warm start.

NEWSIT={YES|NO}

Enforcing the use of system initialization parameters in this way overrides any parameters that may have been stored in a warm keypoint at shutdown.

However, there are some exceptions. The following system initialization parameters are always ignored in a warm start, even if they are supplied by PARM, SYSIN, or the console:

- CSDACC
- CSDBUFND
- CSDBUFNI
- CSDDISP
- CSDDSN
- CSDFRLOG
- CSDINTEG
- CSDJID
- CSDLRNO
- CSDRECOV
- CSDRLS
- CSDSTRNO
- FCT
- GRPLIST

In a warm restart, CICS uses the **installed** resource definitions saved in the CICS global catalog at warm shutdown, and therefore the CSD, FCT, and GRPLIST parameters are ignored. (At CICS startup, you can only modify installed resource definitions, including file control table entries, or change to a new FCT, by performing a cold start of CICS with START=COLD.)

For more information about the use of the NEWSIT parameter, see “Controlling start and restart” on page 308.

Restrictions

You can specify the NEWSIT parameter in PARM, SYSIN, or CONSOLE only.

NONRLSRECOV

The **NONRLSRECOV** system initialization parameter specifies whether CICS uses the recovery options of the VSAM catalog or the FILE resource for non-RLS files, including the CSD.

NONRLSRECOV={VSAMCAT | FILEDEF}

Recovery options do not apply to read-only files. Valid values are as follows:

VSAMCAT

By default, CICS uses the recovery options that are specified on the VSAM catalog for non-RLS files. These recovery options include the LOG, LOGSTREAMID, and BWO options. If no recovery options are set, CICS uses the attributes on the FILE resource.

FILEDEF

For non-RLS files, including the CSD, CICS ignores any recovery options on the catalog and uses the values specified in the FILE resource instead. The recovery attributes for the CSD are set by the appropriate system initialization parameters.

OFFSITE

The **OFFSITE** system initialization parameter specifies whether CICS is to restart in off-site recovery mode; that is, a restart is taking place at a remote site.

OFFSITE={NO | YES}

For a successful off-site restart, the log records of the failed CICS region must be available at the remote site. CICS does not provide a facility for shipping log records to a remote backup site, but you can use a suitable vendor product to perform this function. See the relevant product documentation for other procedures you need to follow for a remote site restart.

See the *CICS Recovery and Restart Guide* for more information about remote site recovery.

NO CICS will not perform the special restart processing required for remote site recovery.

YES CICS will perform an off-site restart at a remote site following a disaster at the primary site. CICS performs this special processing for an off-site restart, because some information (for example, a VSAM lock structure) is not available at the remote site.

CICS performs an emergency restart, even if the global catalog indicates that CICS can do a warm start. OFFSITE=YES is valid with START=AUTO only, and CICS initialization is terminated if you specify START=COLD or INITIAL.

Restrictions

You can specify the OFFSITE parameter in PARM, SYSIN, or CONSOLE only.

OPERTIM

The **OPERTIM** system initialization parameter specifies the write-to-operator timeout value, in the range 0 through 86400 seconds (24 hours).

OPERTIM={120 | number}

This is the maximum time in seconds that CICS waits for a reply before returning control to this transaction. You can change the write-to-operator timeout value when issuing messages to the console from an application by

using the timeout option on the **WRITE OPERATOR** command. See the *CICS Application Programming Reference* for details.

OPNDLIM

The OPNDLIM system initialization parameter specifies the open destination and close destination request limit.

OPNDLIM={10|number} (Not required for currently supported releases of VTAM.)

This limit is used to restrict the number of concurrent OPNDSTs and CLSDSTs to prevent VTAM from running out of space in the CICS region. The limit may be any value in the range 0 through 999. When large values are used for OPNDLIM, the value on the EDSALIM system initialization parameter and the value on the MVS REGION parameter may need to be adjusted to ensure that enough operating system storage is available. For information about adjusting these parameters, see the *CICS Performance Guide*.

PARMERR

The PARMERR system initialization parameter specifies what action you want to follow if CICS detects incorrect system initialization parameter overrides during initialization.

PARMERR={INTERACT|IGNORE|ABEND}

When specified as an override, this parameter affects only subsequent system initialization parameter overrides. Errors in earlier system initialization parameter overrides are dealt with according to the PARMERR system initialization parameter value in the SIT.

INTERACT

Enables the operator to communicate with CICS through the console and correct parameter errors.

Note: INTERACT is overridden with IGNORE in the following cases:

- If errors are found in PARM or SYSIN for system initialization parameter overrides that are not allowed to be entered from the console
- In certain circumstances, in response to invalid data when you have been trying to correct a previous invalid system initialization parameter keyword or value

IGNORE

CICS ignores errors, and tries to complete initialization.

ABEND

CICS abends.

PDI

The PDI system initialization parameter specifies the XRF primary delay interval, in seconds, in a SIT for an active CICS region.

PDI={30|decimal-value}

The minimum delay that you can specify is 5 seconds. This is the time that must elapse between the (apparent) loss of the surveillance signal in the alternate CICS region, and any reaction by the active CICS region. The corresponding parameter for the alternate CICS region is ADI. PDI and ADI need not have the same value.

PDIR

The **PDIR** system initialization parameter specifies a suffix for the PDIR list.

PDIR={NO|YES|xx}

A **PDIR** is a list of program specification blocks (PSBs) that define, for DL/I, the use of databases by application programs. A **PDIR** is applicable only if you are using DL/I remote support. See also “The global catalog” on page 308. Specifying a value other than **NO** implies to CICS that remote DLI support is required.

For information about coding the macros for this table, see the the *CICS Resource Definition Guide*.

PGAICTLG

The **PGAICTLG** system initialization parameter specifies whether autoinstalled program definitions should be cataloged.

PGAICTLG={MODIFY|NONE|ALL}

While **CICS** is running, you can set whether autoinstalled programs should be cataloged dynamically, by using either the **EXEC CICS SET SYSTEM** or **CEMT SET SYSTEM** command.

MODIFY

Autoinstalled program definitions are cataloged only if the program definition is modified by a **SET PROGRAM** command subsequent to the autoinstall.

NONE

Autoinstalled program definitions are not cataloged. This gives a faster **CICS** restart (warm and emergency) compared with the **MODIFY** or **ALL** options, because **CICS** does not reinstall definitions from the global catalog. Definitions are autoinstalled on first reference.

ALL

Autoinstalled program definitions are written to the global catalog at the time of the autoinstall, and following any subsequent modification.

PGAEXIT

The **PGAEXIT** system initialization parameter specifies the name of the program autoinstall exit program.

PGAEXIT={DFHPGADX|name}

While **CICS** is running, you can set the name of the program autoinstall exit program dynamically, by using either the **EXEC CICS SET SYSTEM** or **CEMT SET SYSTEM** command.

PGAIPGM

The **PGAIPGM** system initialization parameter specifies the state of the program autoinstall function at initialization.

PGAIPGM={INACTIVE|ACTIVE}

While **CICS** is running, you can set the status of program autoinstall dynamically, by using either the **IBM CICS Explorer™** or **CICSplex SM Web User Interface**.

INACTIVE

The program autoinstall function is disabled.

ACTIVE

The program autoinstall function is enabled.

PGCHAIN

The **PGCHAIN** system initialization parameter specifies the character string that is identified by terminal control as a BMS terminal page-chaining command.

PGCHAIN=character(s)

The character string can be 1 through 7 characters. For more information about the character string, see “PGRET.”

PGCOPY

The **PGCOPY** system initialization parameter specifies the character string that is identified by terminal control as a BMS command to copy output from one terminal to another.

PGCOPY=character(s)

The character string can be 1 through 7 characters. For more information about the character string, see “PGRET.”

PGPURGE

The **PGPURGE** system initialization parameter specifies the character string that is identified by terminal control as a BMS terminal page-purge command.

PGPURGE=character(s)

It can be 1 through 7 characters. For more information about the character string, see “PGRET.”

PGRET

The **PGRET** system initialization parameter specifies the character string that is recognized by terminal control as a BMS terminal page-retrieval command.

PGRET=character(s)

The character string can be 1 through 7 characters.

1. Each character string is unique with respect to the leading characters of every other transaction identification defined in the CSD. A command requested by a single character precludes the use of all other transaction identifications starting with this character.
2. In pseudoconversational mode, each character string is unique with respect to the leading characters of any terminal input message.
3. A field-separator or other suitable delimiter may be specified in each character string to separate this command code from the remainder of the paging command when entered by an operator. For example:

```
PGCHAIN = X/  
PGCOPY = C/  
PGPURGE = T/  
PGRET = P/
```

This reduces the risk of creating a nonunique command. (See Note 1.)

Restrictions

If you specify **PGCHAIN**, **PGCOPY**, **PGPURGE**, or **PGRET** in the **SIT**, the characters you choose must not include any of the following: () ' "

If you specify **PGCHAIN**, **PGCOPY**, **PGPURGE**, or **PGRET** as a **PARM**, **SYSIN**, or console parameter, do not enclose the characters in quotation marks. The characters you choose must not include an embedded blank or any of the following: () ' =

4. PGCHAIN, PGCOPY, PGPURGE, and PGRET are required only if full function BMS is being used. For information about the BMS page retrieval transaction CSPG, see *CICS Supplied Transactions*.
5. CICS always processes a paging command entered by the operator before initiating a transaction invoked by an EXEC CICS RETURN command with the TRANSID option.

PLTPI

The **PLTPI** system initialization parameter specifies a program list table, which contains a list of programs to be run in the final stages of system initialization.

PLTPI={NO|xx|YES}

For information about coding the macros for the program list table, see *CICS Resource Definition Guide*.

For information about writing initialization programs, see the *CICS Customization Guide*. You can use the **INITPARM** system initialization parameter to pass parameters to those programs.

PLTPISEC

The **PLTPISEC** system initialization parameter specifies whether or not you want CICS to perform command security or resource security checking for PLT programs during CICS initialization.

PLTPISEC={NONE|CMDSEC|RESSEC|ALL}

The PLT programs run under the authority of the userid specified on PLTPIUSR, which must be authorized to the appropriate resources defined by PLTPISEC.

NONE

You do not want any security checking on PLT initialization programs.

CMDSEC

You want CICS to perform command security checking only.

RESSEC

You want CICS to perform resource security checking only.

ALL You want CICS to perform both command and resource security checking.

Restrictions You can specify the PLTPISEC parameter in the SIT, PARM, or SYSIN only.

PLTPIUSR

The **PLTPIUSR** system initialization parameter specifies the userid that CICS is to use for security checking for PLT programs that run during CICS initialization.

PLTPIUSR=userid

All PLT programs run under the authority of the specified userid, which must be authorized to all the resources referenced by the programs, as defined by the PLTPISEC parameter.

PLT programs are run under the CICS internal transaction, CPLT. Before the CPLT transaction is attached, CICS performs a surrogate user check against the CICS region userid (the userid under which the CICS region is executing). This is to ensure that the CICS region is authorized as a surrogate for the userid specified on the PLTPIUSR parameter. This ensures that you cannot arbitrarily

specify any PLT userid in any CICS region—each PLT userid must first be authorized to the appropriate CICS region.

If you do not specify the PLTPIUSR parameter, CICS runs PLTPI programs under the authority of the CICS region userid, in which case CICS does not perform a surrogate user check. However, the CICS region userid must be authorized to all the resources referenced by the PLT programs.

Restrictions You can specify the PLTPIUSR parameter in the SIT, PARM, or SYSIN only.

PLTSD

The **PLTSD** system initialization parameter specifies a program list table that contains a list of programs to be executed during system termination

PLTSD={NO|xx|YES}

The default value is NO. See “Defining CICS resource table and module keywords” on page 138).

PRGDLAY

The **PRGDLAY** system initialization parameter specifies the BMS purge delay time interval that is added to the specified delivery time to determine when a message is to be considered undeliverable and therefore purged.

PRGDLAY={0|hhmm}

This time interval is specified in the form *hhmm* (where *hh* represents hours from 00 to 99 and *mm* represents minutes from 00 to 59). If **PRGDLAY** is not coded, or is given a zero value, a message remains eligible for delivery either until it is purged or until temporary storage is cold started.

Note: If you specify **PRGDLAY** as a SIT override, you must still specify a 4-character value (for example 0000).

The **PRGDLAY** facility requires the use of full function BMS. Note also that you must code a **PRGDLAY** value if you want the ERRTERM|ERRTERM(name) parameter on **EXEC CICS ROUTE** commands to be operative.

The **PRGDLAY** value determines the interval between terminal page clean-up operations. A very low value causes the CSPQ transaction to be initiated continuously, and can have a detrimental effect on task-related resources. A zero value stops CSPQ initiating terminal page clean-up. However, this can cause messages to stay in the system forever, resulting in performance problems with long AID queues or lack of temporary storage. The actual purge delay time interval specified is dependent on individual system requirements.

PRINT

The **PRINT** system initialization parameter specifies the method of requesting printout of the contents of a 3270 screen.

PRINT={NO|YES|PA1|PA2|PA3}

Valid values are as follows:

NO Screen copying is not required.

YES Screen copying can be requested by terminal control print requests only.

PA1, PA2, or PA3

Screen copying can be requested by terminal control print request, or by using the PA (program attention) key specified.

The PA key specified by this parameter must not be specified by the TASKREQ option of the RDO TRANSACTION definition or be used for 3270 single keystroke retrieval.

When YES, PA1, PA2, or PA3 is specified, transaction CSPP is initiated which invokes program DFHP3270. The transaction and programs are defined in the CSD group DFHHARDC. In the case of 3270 and LUTYPE2 logical units, the resources defined in CSD group DFHVTAMP are required.

The 3270 print-request facility allows either the application program or the terminal operator to request a printout of data currently displayed on the 3270 display.

If CSPP is invoked to print the screen contents at an associated VTAM printer, the screen size of the printer is chosen according to the screen size defined in the profile for the transaction CSPP. The CICS-supplied definitions use the default screen size. Therefore, if you want DFHP3270 to use the alternate screen size of the printer, you must alter the screen size defined in the profile for the transaction CSPP. For information about defining profiles for transactions, see the *CICS Resource Definition Guide*.

For a VTAM 3270 display without the printer-adapter feature, the PRINT request prints the contents of the display on the first available 3270 printer specified by PRINTER and ALTPRINTER options of the RDO TERMINAL definition. For a printer to be considered available, it must be in service and not currently attached to a task. It is not necessary for the printer to be on the same control unit.

In an MRO environment, the printer must be owned by the same system as the VTAM 3270 display.

For the 3275 with the printer-adapter feature, the PRINT request prints the data currently in the 3275 display buffer on the 3284 Model 3 printer attached to the 3275.

The format of the print operation depends on the size of the display buffer. For a 40-character wide display, the print format is a 40-byte line, and for an 80-character wide display the format is an 80-byte line.

For the 3270 compatibility mode logical unit of the 3790 (if the logical unit has the printer-adapter feature specified), the PRINT request prints the contents of the display on the first printer available to the 3790. The allocation of the printer to be used is under the control of the 3790.

For 3274, 3276, and LUTYPE2 logical units with the printer-adapter feature, the PRINT request prints the contents of the display on the first printer available to the 3270 control unit. The printer to be allocated depends on the printer authorization matrix.

For the 3270 compatibility mode logical unit without the printer-adapter feature, see the preceding paragraph on VTAM 3270 displays without the printer-adapter feature.

PRTYAGE

The **PRTYAGE** system initialization parameter specifies the number of milliseconds to be used in the priority aging algorithm for incrementing the priority of a task.

PRTYAGE={32768|value}

The value can be in the range 0 through 65535, and 32768 is the default.

The priority aging factor is used to increase the effective priority of a task according to the amount of time it is held on a ready queue. The value represents the number of milliseconds that must elapse before the priority of a waiting task can be adjusted upwards by 1. For example, if you code **PRTYAGE**=3000, a task has its priority raised by 1 for every 3000 milliseconds it is held on the ready queue. Thus a high value for **PRTYAGE** results in a task being promoted very slowly up the priority increment range, and a low value enables a task to have its priority incremented quickly.

If you specify a value of 0, the priority aging algorithm is not used (task priorities are not modified by age) and tasks on the ready queue are handled according to the user assigned priority.

PRVMOD

The **PRVMOD** system initialization parameter specifies the names of those modules that are not to be used from the LPA.

PRVMOD={name| (name,name...name)}

The operand is a list of 1-8 character module names. This enables you to use a private version of a CICS nucleus module in the CICS address space, and not a version that might be in the LPA. For information about **PRVMOD**, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the **PRVMOD** parameter in PARM, SYSIN, or CONSOLE only.

PSBCHK

The **PSBCHK** system initialization parameter specifies whether CICS is to perform PSB authorization checks for remote terminal users who use transaction routing to initiate a transaction in this CICS region to access an attached IMS system.

PSBCHK={NO|YES}

Valid values are as follows:

NO The remote link is checked, but no check is made against the remote terminal. This is the default.

YES The remote link is checked, and the remote terminal is also checked if **RESSEC(YES)** is coded in the definition of the transaction in the CSD.

Restrictions You can specify the **PSBCHK** parameter in the SIT, PARM, or SYSIN only.

Note: If you require DL/I security checking, you must specify the **XPSB** system initialization parameter as **XPSB=YES** or **XSPB=name**. For further information about the **XPSB** system initialization parameter, see **XPSB**.

PSDINT

The **PSDINT** system initialization parameter specifies the persistent session delay interval, which states if, and for how long, VTAM holds sessions in a recovery-pending state.

PSDINT={0|hhmmss}

0 If a failure occurs, VTAM sessions are terminated. Zero is the default, and means that persistent sessions support is not exploited.

hhmmss

The time for which VTAM retains sessions if a failure occurs, from 1 second up to the maximum of 23 hours 59 minutes and 59 seconds. Specify a 1 to 6-digit time in hours, minutes, and seconds. If you specify fewer than six digits, CICS pads the value with leading zeros. Thus, a value of 500 is taken as 5 minutes exactly.

You can override this value while CICS is running. Overriding the value changes the action taken by VTAM if a failure occurs. The changed interval is not stored in the CICS global catalog, and therefore is not restored in an emergency restart.

VTAM holds all sessions in a recovery-pending state for up to the interval specified, unless they are unbound through path failure or VTAM operator action, or other-system action in the case of intelligent LUs. The interval you specify must be able to cover the time from a CICS failure to the time when the VTAM ACB is opened by CICS during a subsequent emergency restart.

- If you specify SNPS (the default) or MNPS for the **PSTYPE** system initialization parameter for the CICS region, set a nonzero value for the persistent session delay interval, so that sessions are retained.
- If you specify NOPS (no persistent sessions support) for the **PSTYPE** system initialization parameter, a zero value is required for the persistent session delay interval.
- If you are using XRF, the extended recovery facility, for the CICS region (**XRF=YES** system initialization parameter), you cannot use persistent sessions support as well, and the persistent session delay interval is ignored. The **XRFSTATUS** option in the **INQUIRE SYSTEM** command shows whether or not XRF is used for the CICS region.

When choosing your **PSDINT** value, take account of the types and numbers of sessions involved. You must exercise care when specifying large **PSDINT** values because of the problems such a value might give in some environments, in particular:

- Dial-up sessions, for which real costs might be incurred.
- LU6.2 sessions to other host systems. If these sessions are retained in recovery pending state, the other host systems might experience excessive queuing delays. This point applies to LU6.1 sessions that are retained until restart, when they are unbound.

PSTYPE

The **PSTYPE** system initialization parameter specifies whether CICS uses VTAM single-node persistent sessions (SNPS), multinode persistent sessions (MNPS), or does not use VTAM persistent sessions support (NOPS).

PSTYPE={SNPS|MNPS|NOPS}

The default setting, SNPS (single-node persistent sessions), means that persistent sessions support is available, so that VTAM sessions can be recovered after a CICS failure and restart. MNPS (multinode persistent sessions) means that, in addition to the SNPS support, VTAM sessions can also be recovered after a VTAM or z/OS failure in a sysplex (across LPARs).

For single-node persistent sessions support, you require VTAM V3.4.1 or later, which supports persistent LU-LU sessions. CICS Transaction Server for z/OS, Version 4 Release 1 functions with releases of VTAM earlier than V3.4.1, but in the earlier releases sessions are not retained in a bound state if CICS fails. For multinode persistent sessions support, you require VTAM V4.R4 or later, and

| VTAM must be in a Parallel Sysplex[®] with a coupling facility. The *VTAM Network Implementation Guide* explains the exact VTAM configuration requirements for multinode persistent sessions support.

| If you specify SNPS or MNPS, set a nonzero value for the **PSDINT** system initialization parameter, which specifies the retention time for session information. The default is zero, which means that sessions are not retained.

| If you do not require persistent sessions support, specify NOPS. A CICS region that is used only for development or testing might not require this support. Removing persistent sessions support where it is not required reduces resource consumption, and can enable you to increase the number of CICS regions in an LPAR. If you specify NOPS, a zero value is required for the **PSDINT** system initialization parameter.

PVDELAY

The **PVDELAY** system initialization parameter specifies the persistent verification delay as a value in the range 0 through 10080 minutes (up to 7 days).

PVDELAY={30|number}

PVDELAY defines how long entries can remain in the signed-on-from lists for those connections for which persistent verification is specified in a connection resource definition. If you specify **PVDELAY=0**, entries are deleted immediately after use.

For information about the use of **PVDELAY**, see the *CICS Performance Guide*.

QUIESTIM

The **QUIESTIM** system initialization parameter specifies a timeout value for data set quiesce requests.

QUIESTIM={240|number}

In a busy CICSplex, it is possible for the default timeout to expire before the quiesce request has been processed by all the CICS regions, even though there is nothing wrong. If the quiesce operation is not completed when the timeout period expires, SMS VSAM cancels the quiesce. If you find that timeout is occurring too frequently, increase the timeout value.

Specify the timeout value as a number of seconds. The default value is 240 seconds (4 minutes)

The maximum timeout value you can specify is 3600 (1 hour).

RAMAX

The **RAMAX** system initialization parameter specifies the size in bytes of the I/O area allocated for each RECEIVE ANY issued by CICS, in the range 0 through 32767 bytes.

RAMAX={256|number}

If you are using APPC, do not code a value less than 256; otherwise, the results are unpredictable.

For information about coding this parameter, see com.ibm.cics.ts.performance.doc/topics/dfht34e.dita the *CICS Performance Guide*.

RAPOOL

The **RAPOOL** system initialization parameter specifies the size of the CICS receive any pool.

RAPOOL={50|value1|(value1,value2,FORCE)}

value1 is the number of fixed request parameter lists (RPLs), receive any control elements (RACEs), and receive any input areas (RAIAs) that are to be generated whether or not CICS uses the high performance option (HPO). *value1*, in the range 1 through 999, is also the number that are active in a non-HPO system; *value2*, in the range 0 through 999, is the number that are active in an HPO system. The default for *value1* in the DFHSIT macro is 2. The default for *value2* is calculated from *value1* as follows:

If *value1* = 1, *value2* = 1
If *value1* ≤ 5, *value2* = (*value1* minus 1)
If *value1* ≥ 6 and ≤ 49, *value2* = 5
If *value1* ≥ 50, *value2* is 10 per cent of *value1*

Note: You should code *value1* equal to or greater than *value2*; if you code *value1* less than *value2*, CICS forces *value2* equal to *value1*.

If you omit the RAPOOL parameter altogether, RAPOOL=(50,1) is assumed. CICS maintains n VTAM RECEIVE ANYs, where n is either the RAPOOL “number active” value, or the MXT value minus the number of active tasks, whichever is the smaller. For example, in a non-HPO system:

If RAPOOL=2, MXT=50, active tasks = 45 then RECEIVE ANY = 2
If RAPOOL=10, MXT=50, active tasks = 45 then RECEIVE ANY = 5
If RAPOOL=10, MXT=50, active tasks = 35 then RECEIVE ANY = 10

or in an HPO system:

If RAPOOL=(20,10), MXT=50, active tasks = 45 then RECEIVE ANY = 5

FORCE tells CICS to free up Receive_Any_RPLs if they are stalled. CICS decides that the Receive_Any_RPLs are stalled if all the RA RPLs have been posted but the TCTTE for each one is waiting for a response from a VTAM terminal or session for 10 dispatches of the TCP (CSTP) task.

This typically happens only if a protocol error has occurred, and sessions are waiting for a response; for example, to a BID SHUTD request from CICS.

Each session is unbound, the Receive_Any data is lost and the RA RPL is reissued thus allowing VTAM activity to continue: Message DFHZC4949 is issued for each session affected.

Consider increasing the size of the RAPOOL before resorting to the use of FORCE.

If FORCE is not specified and a Receive_Any stall occurs, DFHZC2118 is written to the console for each session affected.

If FORCE is specified in the SIT, and RAPOOL is supplied as an override, you must again specify FORCE as otherwise it defaults to FORCE not specified.

The number of RECEIVE ANYs needed depends on the expected activity of the system, the average transaction lifetime, and the MAXTASK value specified. For information about coding this parameter, see the *CICS Performance Guide*.

RDSASZE

The **RDSASZE** system initialization parameter specifies the size of the RDSA.

RDSASZE={0K|number}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify number as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256 KB). If the size specified is not a multiple of 256 KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the RDSASZE parameter in PARM, SYSIN, or CONSOLE only.

RENTPGM

The **RENTPGM** system initialization parameter specifies whether you want CICS to allocate the read-only DSAs, RDSA and ERDSA, from read-only key-0 protected storage.

RENTPGM={PROTECT|NOPROTECT}

The permitted values are PROTECT (the default), or NOPROTECT:

PROTECT

CICS obtains the storage for the read-only DSAs from key-0 protected storage.

NOPROTECT

CICS obtains the storage from CICS-key storage, effectively creating two more CICS DSAs (CDSA and ECDSA). This allows programs eligible for the read-only DSAs to be modified by programs that execute in CICS key.

You are recommended to specify RENTPGM=NOPROTECT for development regions only, and to specify RENTPGM=PROTECT for production CICS regions.

RESP

The **RESP** system initialization parameter specifies the type of request that CICS terminal control receives from logical units.

RESP={FME|RRN}

Valid values are as follows:

FME Function management end is the default.

RRN Reached recovery node.

RESSEC

The **RESSEC** system initialization parameter specifies whether you want CICS to honor the RESSEC option specified on a transaction's resource definition.

RESSEC={ASIS|ALWAYS}

Valid values are as follows:

ASIS CICS honors the RESSEC option defined in a transaction's resource definition. CICS calls its resource security checking routine only when RESSEC(YES) is specified in a transaction resource definition. This is

normally a sufficient level of control, because often you will need only to control the ability to execute a transaction.

ALWAYS

CICS overrides the RESSEC option, and always calls its resource security checking routine to issue the appropriate call to the SAF interface.

Use this option only if you need to control or audit all accesses to CICS resources. Using this option can significantly degrade performance.

Restrictions You can specify the RESSEC parameter in the SIT, PARM, or SYSIN only.

RLS

The **RLS** system initialization parameter specifies whether CICS is to support VSAM record-level sharing (RLS).

RLS={NO|YES}

Valid values are as follows:

NO RLS support is not required in this CICS region. Files whose definitions specify RLSACCESS(YES) will fail to open, with an error indicating that RLS access is not supported. You should not specify RLS=NO if you have files that you want to open in RLS access mode (including the CSD).

YES RLS support is required in this CICS region. During initialization, CICS automatically registers with an SMSVSAM control ACB to enable RLS access to files opened with RLSACCESS(YES).

RLSTOLSR

The **RLSTOLSR** system initialization parameter specifies whether CICS is to include files that are to be opened in RLS mode when calculating the number of buffers, strings, and other resources for an LSR pool.

RLSTOLSR={NO|YES}

CICS performs this calculation only when you have not explicitly defined an LSRPOOL resource definition that corresponds to an LSRPOOLID in a file definition. CICS calculates and builds a default LSR pool only when it is opening the first file in LSR mode that references the default pool.

NO CICS is not to include files opened in RLS mode, and which also specify an LSRPOOLID, when it is building default LSR pools. Files defined with RLSACCESS(YES) are ignored when CICS is scanning file entries looking for files that specify an LSR pool it is about to build using default values.

If the LSR pools referenced by LSRPOOLIDs in your file resource definitions are defined explicitly by LSRPOOL resource definitions, you should specify RLSTOLSR=NO.

YES CICS is to include in its calculation, when building default LSR pools, files that specify both RLSACCESS(YES) and an LSRPOOLID.

Note that an LSR pool built including files that are opened in RLS mode is larger than necessary initially. This option is provided to ensure that, if files are subsequently switched to LSR, the LSR pool is

adequate for the extra files. You should specify RLSTOLSR=YES only if both of the following conditions are true:

1. You do not define LSR pools explicitly, relying instead on CICS obtaining a default set of values for you.
2. You have files that are sometimes accessed in RLS mode and sometimes accessed in non-RLS mode (although this is generally not recommended).

The RLSTOLSR parameter is provided to support files that are normally opened in RLS mode, but which may be closed and then switched to LSR mode.

If LSR pools are not defined explicitly using LSRPOOL resource definitions, CICS calculates the resources needed for an LSR pool using default attributes. CICS performs this calculation when opening the first file that specifies an LSR pool that is not explicitly defined. To calculate a default LSR pool, CICS scans all the file entries to count all the files that specify the same LSRPOOLID. The size of an LSR pool built dynamically in this way remains fixed until all files that reference the LSR pool are closed. After all files have been closed, another request to open a file with the same LSRPOOLID causes CICS to recalculate the size.

If you add files to the system *after* the LSR calculation has been performed there may be insufficient storage available to enable CICS to open a file that specifies a default pool. This situation could occur if files are opened initially in RLS mode and later closed and reopened in LSR mode. There are two ways to ensure that enough resources are built into the LSR pool to support subsequent switches of files from RLS to LSR:

1. You can explicitly define LSRPOOL resource definitions that correspond to the LSRPOOLIDs on file definitions, removing the need for CICS to calculate default values.
2. You can specify RLSTOLSR=YES to force CICS to include RLS files when calculating defaults.

RMTRAN

The **RMTRAN** system initialization parameter specifies the name of the transaction that you want an alternate CICS to initiate when logged-on class 1 terminals, which are defined with the attribute RECOVNOTIFY(TRANSACTION) specified, are switched following a takeover.

RMTRAN=({CSGM|name1}[,{CSGM |name2}])

This parameter is applicable only on an alternate CICS region.

If you do not specify a name here, CICS uses the CSGM transaction, the default CICS good morning transaction.

name1 This is the transaction that CICS initiates at terminals that do **not** remain signed-on after the takeover (that is, they are still connected to CICS, but are signed off).

name2 This is the transaction that CICS initiates at terminals that remain signed-on after the takeover. If you specify only name1, CICS uses the CSGM transaction as the default for name2.

If you are using VTAM persistent sessions, the name2 transaction is ignored and the name1 transaction is always initiated.

RRMS

The **RRMS** system initialization parameter specifies whether CICS is to register as a resource manager with recoverable resource management services (RRMS).

RRMS={NO|YES}

Valid values are as follows:

NO You do not require RRMS support.

YES You require RRMS support to enable DPL requests to be coordinated by resource recovery services (RRS).

Note: If you specify RRMS=YES, ensure that the DFHRXSVC module is available during CICS initialization. This module, which provides RRMS authorized services, is supplied in the SDFHLINK library. For information about this link list library, see the *CICS Transaction Server for z/OS Installation Guide*.

RST

The **RST** system initialization parameter specifies a recoverable service table suffix.

RST={NO|xx|YES}

If you are running CICS with XRF=YES, and you are using DBCTL, you must specify an RST if you want XRF support for DBCTL. For information about the use of the RST in a CICS-DBCTL environment with XRF=YES, see the *CICS IMS Database Control Guide*.

For information about coding the macros for this table, see the *CICS Resource Definition Guide*.

RSTSIGNOFF

The **RSTSIGNOFF** system initialization parameter specifies whether all users signed-on to the active CICS region are to remain signed-on following a persistent sessions restart or an XRF takeover.

RSTSIGNOFF={NOFORCE|FORCE}

It applies to the following events:

- A persistent sessions restart, where PSDINT=*value* and PSTYPE=SNPS or MNPS are specified, and the restart follows a CICS abnormal or immediate shutdown.
- A persistent sessions restart, where PSDINT=*value* and PSTYPE=MNPS are specified, and terminal sessions are recovered as a result of a VTAM restart.
- An XRF takeover, where XRF=YES is specified.

NOFORCE

Do not sign off users, unless FORCE is specified on either:

- The RSTSIGNOFF parameter in the TYPETERM definition referenced by the user's terminal definition.
- The XRFSOFF parameter in the CICS segment of the user's RACF profile.

Thus for a user to remain signed on after a persistent sessions restart or an XRF takeover, NOFORCE must be specified as a system initialization parameter, on the TYPETERM definition, and in the CICS segment.

FORCE

Sign off all users regardless of the options specified on:

- The RSTSIGNOFF attribute in the TYPETERM definition referenced by the user's terminal definition.
- The XRFSOFF parameter in the CICS segment of the user's RACF profile.

See the *CICS RACF Security Guide* for information about user profile options in the CICS segment, and see the *CICS Resource Definition Guide* for information about the TYPETERM resource definition.

RSTSIGNTIME

The **RSTSIGNTIME** parameter specifies the time-out delay interval for signon retention during a persistent sessions restart or an XRF takeover.

RSTSIGNTIME={500|decimal-value}

You can specify a 1-to-6 digit time in hours, minutes and seconds, up to the maximum time of 23 hours 59 minutes 59 seconds. If you specify less than six digits, CICS pads the value with leading zeros. Thus a value of 500 is taken as five minutes exactly.

RSTSIGNTIME is counted from the time when CICS failed. Note that the time of failure cannot be determined with complete accuracy.

If you specify NOFORCE on all the appropriate parameters to enable a user to remain signed on, but the persistent sessions restart or XRF takeover takes longer than the specified on the RSTSIGNTIME parameter, CICS ensures users do not remain signed on after the delay period expires.

500 Five minutes is the default value.

time This is the time, in the range 0 through 23 hours 59 minutes 59 seconds, during which CICS permits users to remain signed on during a persistent sessions restart or an XRF takeover. The period is measured as follows:

- For a persistent sessions restart, the period is the time from the CICS failure and the time when the user starts working on the terminal. If the specified time expires before the user starts working on the terminal, users signed on at the time CICS failed are not signed on again after restart.
- For an XRF takeover, the period is the time from when the takeover is initiated to the time at which the alternate CICS has completed takeover and is ready to process user transactions. If the takeover takes longer than the specified period, all users signed on at the time the takeover was initiated are signed off.

A value of 0 means there is no time-out delay, and terminals are not signed on after a persistent sessions restart or XRF takeover, which means that RSTSIGNTIME=0 has the same effect as coding RSTSIGNOFF=FORCE.

When XRF is in use with non-XRF-capable terminals, take into account any AUTCONN delay period when setting the value for RSTSIGNTIME. For example, you may need to increase the time specified on RSTSIGNTIME to allow for the delay up to the start of the CXRE transaction imposed by the AUTCONN parameter; otherwise, terminals could be signed off too early.

RUWAPPOOL

The **RUWAPPOOL** parameter specifies the option for allocating a storage pool the first time a program invoked by Language Environment runs in a task.

RUWAPPOOL={NO|YES}

Valid values are as follows:

- NO** CICS disables the option and provides no RUWA storage pool. Every EXEC CICS LINK to a program that runs under Language Environment results in a GETMAIN for RUWA storage.
- YES** CICS creates a pool of storage the first time a program invoked by Language Environment runs in a task. This provides an available storage pool that reduces the need to GETMAIN and FREEMAIN run-unit work areas (RUWAs) for every EXEC CICS LINK request.

Note: This applies only to application programs running with the Language Environment run-time option ALL31(ON).
RUWAPPOOL=YES has no effect on application programs running with the Language Environment run-time option ALL31(OFF).

SDSASZE

The **SDSASZE** system initialization parameter specifies the size of the SDSA.

SDSASZE={0K|number}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify number as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256 KB). If the size specified is not a multiple of 256 KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the **SDSAZSE** parameter in PARM, SYSIN, or CONSOLE only.

SDTRAN

The **SDTRAN** system initialization parameter specifies the name of the shutdown transaction to be started at the beginning of normal and immediate shutdown.

SDTRAN={CESD|name_of_shutdown_tran|NO}

The shutdown transaction enables CICS to shut down in a controlled manner, within a reasonable period of time. For example, you can use it to purge and backout long-running tasks, while ensuring that as many tasks as possible commit or backout cleanly. For information about the CICS-supplied program, DFHCESD, started by the default shutdown transaction, CESD, and how to use it as the basis for your own transaction, see the *CICS Operations and Utilities Guide*.

Note:

1. The transaction runs under the userid authority of the issuer of the shutdown command.

2. If the program named by the shutdown transaction cannot be loaded, CICS waits indefinitely for all user tasks to complete. *This happens on an immediate, as well as on a normal, shutdown.*

CESD Starts the CICS-supplied program DFHCESD.

name_of_shutdown_transaction

The 1-to 4-character name of your own shutdown transaction.

- NO** No shutdown transaction is to be run. On a normal shutdown, CICS waits indefinitely for all user tasks to complete.

SEC

The **SEC** system initialization parameter specifies what level of external security you want CICS to use.

SEC={YES|NO}

Valid values are as follows:

- YES** You want to use full external security. CICS requires the appropriate level of authorization for the access intent: a minimum of READ permission for read intent, and a minimum of UPDATE permission for update intent.

Note: You must also ensure that the default userid (CICSUSER or another userid specified on the **DFLTUSER** system initialization parameter) has been defined to RACF.

If command security checking is defined for CICS SP-type commands, then specifying SEC=YES means that the appropriate level of authority is checked for; therefore:

- A check for READ authority is made for **INQUIRE** and **COLLECT** commands.
- A check for UPDATE authority is made for **SET**, **PERFORM**, and **DISCARD** commands.

- NO** You do not want CICS to use an external security manager. All users have access to all resources, whether determined by attempts to use them or by the **QUERY SECURITY** command. Users are not allowed to sign on or off.

Note: With MRO bind-time security, even if you specify SEC=NO, the CICS region userid is still sent to the secondary CICS region, and bind-time checking is still carried out in the secondary CICS region. For information about MRO bind-time security, see the *CICS RACF Security Guide*.

Define whether to use RACF for resource level checking by using the **XDCT**, **XFCT**, **XHFS**, **XJCT**, **XPCT**, **XPPT**, **XPSB**, **XRES**, and **XTST** system initialization parameters. Define whether to use RACF for transaction-attach security checking by using the **XTRAN** system initialization parameter. Define whether to use RACF for enterprise bean method authorization checks by using the "XEJB" on page 259 system initialization parameter. Define whether RACF session security can be used when establishing APPC sessions by using the **XAPPC** system initialization parameter.

For information on defining command security checking for CICS SP-type commands, and about CICS security in general, see the *CICS RACF Security Guide*.

For programming information about the use of external security for CICS system commands, see the *CICS System Programming Reference*.

Restrictions You can specify the **SEC** parameter in the SIT, PARM, or SYSIN only.

Note: If you are using preset terminal security and you perform a warm start with SEC=NO and then again with SEC=YES, you must reinstall the terminal definition to preserve the preset user ID that is replaced by the default user ID when security is switched off. See the *CICS RACF Security Guide* for details.

SECPRFX

The **SECPRFX** system initialization parameter specifies whether CICS is to prefix the resource names in any authorization requests to the external security manager.

SECPRFX={NO|YES|**prefix**}

Valid values are as follows:

NO CICS does not use prefixes on any resource names.

YES CICS prefixes all resource names with the CICS region user ID. This is the user ID under which the CICS job runs. It is one of the following:

- If CICS is a batch job, it is the user ID corresponding to the USER parameter of the CICS JOB statement.
- If CICS is a started task, it is the user ID associated with the name of the started procedure in the RACF ICHRIN03 table.
- If CICS is a started job, it is the user ID specified in the user parameter of the STDATA segment of a STARTED general resource class profile.

For more information, see the *CICS RACF Security Guide*.

prefix CICS prefixes all resource names with the string you specify. It can be any string of 1 to 8 upper case alphanumeric characters except NO or YES, and must start with an alphabetic character.

Restrictions You can specify the SECPRFX parameter in the SIT, PARM, or SYSIN only.

The SECPRFX parameter is effective only if you specify YES for the SEC system initialization parameter.

SIT

The **SIT** system initialization parameter specifies the suffix, if any, of the system initialization table that you want CICS to load at the start of initialization.

SIT=xx

If you omit this parameter, CICS loads the unsuffixed table, DFHSIT, which is pregenerated with all the default values. This default SIT (shown in “The default system initialization table” on page 293) is in CICSTS41.CICS.SDFHAUTH, and its source, named DFHSIT\$\$, is in CICSTS41.CICS.SDFHSAMP.

Restrictions You can specify the system initialization parameter anywhere in PARM or SYSIN, or as the **first** parameter entry at the CONSOLE.

SKRxxxx

The **SKRxxxx** system initialization parameter specifies that a single-keystroke-retrieval operation is required.

SKRxxxx='page-retrieval-command'

'xxxx' specifies a key on the 3270 keyboard which, during a page retrieval session, is to be used to represent a page retrieval command. The valid keys you can specify as a system initialization parameter or as an override are PA1 through PA3, and PF1 through PF24. However, if you use full function BMS, you can also define PA25 through PF36. Therefore it is possible to specify up to 39 keys in total.

The 'page-retrieval-command' value represents any valid page retrieval command, and must be enclosed in apostrophes. It is concatenated to the character string coded in the PGRET parameter. The combined length must not exceed 16 characters.

Note: If full function BMS is used, all PA keys and PF keys are interpreted for page retrieval commands, even if some of these keys are not defined.

SNSCOPE

The **SNSCOPE** system initialization parameter specifies whether a userid can be signed on to CICS more than once, within the scope of a single CICS region, a single MVS image, and a sysplex.

SNSCOPE={NONE|CICS|MVSIMAGE|SYSPLEX}

The signon **SCOPE** is enforced with the MVS ENQ macro where there is a limit on the number of outstanding MVS ENQs per address space. If this limit is exceeded, the MVS ENQ is rejected and CICS is unable to detect if the user is already signed on. When this happens, the signon request is rejected with message DFHCE3587. You can use the ISGADMIN macro to set or reset the MVS ENQ limit. For more information, see the *z/OS MVS Authorized Assembler Services Reference*.

NONE

Each user ID can be used to sign on for any number of sessions on any CICS region. This is the compatibility option, providing the same signon scope as in releases of CICS before CICS Transaction Server for z/OS, Version 4 Release 1.

CICS Each user ID can be signed on once only in the same CICS region. A signon request is rejected if the userid is already signed on to the same CICS region. However, the user ID can be used to signon to another CICS region in the same, or another, MVS image.

MVSIMAGE

Each userid can be signed on once only, and to only one of the set of CICS regions in the same MVS image that also specify **SNSCOPE=MVSIMAGE**. A signon request is rejected if the user is already signed on to another CICS region in the same MVS image.

SYSPLEX

Each user ID can be signed on once only, and to only one of the set of CICS regions within an MVS sysplex that also specify **SNSCOPE=SYSPLEX**. A signon is rejected if the user is already signed on to another CICS region in the same MVS sysplex.

The signon scope (if specified) applies to all user IDs signing on by an explicit signon request (for example, by an EXEC CICS SIGNON command or the CESN transaction). **SNSCOPE** is restricted to users signing on at local terminals, or signing on after using the CRTE transaction to connect to another system.

Signon scope specified by SNSCOPE does not apply to:

- Non-terminal users.
- The CICS default userid, specified by the **DFLTUSER** system initialization parameter.
- Preset user IDs, specified in the USERID option of the **DEFINE TERMINAL** command.
- User IDs for remote users, received in attach headers.
- User IDs for link security. For information about which userid is used for link security on a specific connection, see the *CICS RACF Security Guide*.
- The user ID specified on the **PLTPIUSR** system initialization parameter.
- The CICS region user ID.

Restrictions You can specify the **SNSCOPE** parameter in the SIT, PARM, or SYSIN only.

SPCTR

The **SPCTR** system initialization parameter specifies the level of special tracing required for CICS as a whole.

SPCTRxx={ (1,2) | (1[,2] [,3]) | ALL | OFF }

Specifies the level of special tracing for all CICS components used by a transaction, terminal, or both. If you want to set different tracing levels for an individual component of CICS, use the **SPCTRxx** system initialization parameter.

It is possible to select up to 32 levels of tracing using this parameter. However, most CICS components only use levels 1, 2, and 3, and some do not have trace points at all these levels. The exceptions are the SM component (storage manager domain), which also has level 4 tracing; and the SJ component (JVM domain), which also has trace levels 29-32, that are reserved to indicate the JVM trace levels 0, 1, and 2, plus a user-definable JVM trace level. Use the **SPCTRxx** system initialization parameter to set special tracing levels above 3 for these components.

number

The level numbers for the level of special tracing you want for all CICS components.

ALL Enables the special tracing facility for all available levels.

OFF Disables the special tracing facility.

SPCTRxx

The **SPCTRxx** system initialization parameter specifies the level of special tracing for a particular CICS component used by a transaction, terminal, or both.

SPCTRxx={ (1,2) | (1[,2] [,3] [,4] [,29] [,30] [,31] [,32]) | ALL | OFF }

You identify the component by coding a value for *xx* in the keyword. You code one **SPCTRxx** keyword for each component you want to define selectively. For a CICS component being specially traced that does not have its trace level set by **SPCTRxx**, the trace level is that set by **SPCTR** (which, in turn, defaults to (1,2)). The CICS component codes that you can specify for *xx* on the **SPCTRxx** keyword are shown in the following table:

| Code | Component name |
|------|------------------------------|
| AP | Application domain |
| BA | Business application manager |

| Code | Component name |
|-------------|----------------------------------|
| BF* | Built-in function |
| BM* | Basic mapping support |
| BR* | 3270 bridge |
| CP* | Common programming interface |
| DC* | Dump compatibility layer |
| DD | Directory manager domain |
| DH | Document handling domain |
| DI* | Data interchange |
| DM | Domain manager domain |
| DP | Debugging profiles domain |
| DS | Dispatcher domain |
| DU | Dump domain |
| EC* | Event capture and emission |
| EI* | Exec interface |
| EJ | Enterprise Java domain |
| EM | Event manager domain |
| EP | Event processing domain |
| FC* | File control |
| GC | Global catalog domain |
| IC* | Interval control |
| IE | ECI over TCP/IP domain |
| II | IIOP domain |
| IS* | ISC or IRC |
| KC* | Task control |
| KE | Kernel |
| LC | Local catalog domain |
| LD | Loader domain |
| LG | Log manager domain |
| LM | Lock domain |
| ME | Message domain |
| ML | Markup language domain |
| MN | Monitoring domain |
| NQ | Enqueue domain |
| OT | Object transaction domain |
| PA | Parameter domain |
| PC* | Program control |
| PG | Program manager domain |
| PI | Pipeline domain |
| PT | Partner domain |
| RA | Resource manager adapters |
| RI* | Resource manager interface (RMI) |

| Code | Component name |
|--|---------------------------------|
| RL | Resource life-cycle domain |
| RM | Recovery manager domain |
| RS | Region status domain |
| RX | RRS-coordinated EXCI domain |
| RZ | Request streams domain |
| SC* | Storage control |
| SH | Scheduler services domain |
| SJ | JVM domain |
| SM | Storage manager domain |
| SO | Sockets domain |
| ST | Statistics domain |
| SZ* | Front End Programming Interface |
| TC* | Terminal control |
| TD* | Transient data |
| TI | Timer domain |
| TR | Trace domain |
| TS | Temporary storage domain |
| UE* | User exit interface |
| US | User domain |
| WB | Web domain |
| WU | System management RESTful API |
| W2 | Web 2.0 domain |
| XM | Transaction manager domain |
| XS | Security manager domain |
| Note: | |
| 1. Components marked * are subcomponents of the AP domain. The trace entries for these components are produced with a trace point ID of AP <i>nnnn</i> . | |
| 2. For the DS domain function CHANGE_MODE, a trace entry is generated if DS level 2 or 3 tracing is active. | |

number

The level numbers for the level of special tracing you want for the required CICS component. You can use level numbers 1, 2, 3, 4, 29, 30, 31 and 32, depending on the component.

Most CICS components only use levels 1, 2 and 3, and some do not have trace points at all these levels. The exceptions are:

- The SM component (storage manager domain) has level 4 tracing. This level of tracing is intended for IBM field engineering staff.
- The SJ component (JVM domain) has trace levels 29–32, that are reserved to indicate the JVM trace levels 0, 1, and 2, plus a user-definable JVM trace level. You can use the system initialization parameters **JVMLEVEL0TRACE**, **JVMLEVEL1TRACE**, **JVMLEVEL2TRACE** and **JVMUSERTRACE** to specify options for these JVM trace levels, and then activate them using the **SPCTRSJ** system initialization parameter.

When you activate JVM trace, using trace levels 29–32 for the SJ component, the JVM trace appears as CICS trace point SJ 4D02 (when formatted), or SJ 4D01 (if unformatted).

ALL You want all the available levels of special CICS tracing switched on for the specified component.

OFF Switches off all levels of special CICS tracing for the CICS component indicated by xx.

For details of using trace, see the *CICS Problem Determination Guide*.

Restrictions You can specify the **SPCTRxx** parameter in PARM, SYSIN, or CONSOLE only.

SPOOL

The **SPOOL** system initialization parameter specifies whether the system spooling interface is required.

SPOOL={NO|YES}

Valid values are as follows:

NO The system spooling interface is not required.

YES The system spooling interface is required.

The CICS spool interface uses the MVS exit, IEFDOIXT, which is provided in the SYS1.LINKLIB library. For further information about the MVS exit IEFDOIXT, see the current z/OS release information on z/OS MVS Installation Exits.

SRBSVC

The **SRBSVC** system initialization parameter specifies the number that you have assigned to the CICS type 6 SVC.

SRBSVC={215|number}

The default number is 215.

For information on changing the SVC number, see the *CICS Transaction Server for z/OS Installation Guide*. A CICS type 6 SVC with the specified (or default) number must have been link-edited with the system nucleus.

SRT

The **SRT** system initialization parameter specifies the system recovery table suffix.

SRT={1\$|YES|NO|xx}

If SRT=YES is coded, the default DFHSRT1\$ table is used.

Restriction: SRT=YES can only be specified when assembling the SIT table; it cannot be specified as an override parameter.

If SRT=NO is coded, the system recovery program (DFHSRP) does not attempt to recover from a program check or from an operating system abend. However, CICS issues ESPIE macros to intercept program checks to perform cleanup operations before CICS terminates. Therefore, you must provide a SRT if you require recovery from either program checks or abnormal terminations, or both. For information about coding the macros for this table, see the *CICS Resource Definition Guide*

SRVERCP

The **SRVERCP** system initialization parameter specifies the default server code page to be used by the DFHCNV data conversion table but only if the SRVERCP parameter in the DFHCNV macro is set to SYSDEF.

SRVERCP={037|codepage}

The *codepage* is a field of up to 8 characters and can take the values supported by the SRVERCP parameter in the DFHCNV macro. See the *CICS Family: Communicating from CICS on System/390* for the list of valid code pages. The default is 037.

SSLCACHE

The **SSLCACHE** system initialization parameter specifies whether SSL is to use the local or sysplex caching of session ids.

SSLCACHE={CICS|SYSPLEX}

Sysplex caching is only allowed if multiple CICS socket-owning regions accept SSL connections at the same IP address.

SSLDELAY

The **SSLDELAY** system initialization parameter specifies the length of time in seconds for which CICS retains session ids for secure socket connections.

SSLDELAY={600|number}

Session ids are tokens that represent a secure connection between a client and an SSL server.

While the session id is retained by CICS within the SSLDELAY period, CICS can continue to communicate with the client without the significant overhead of an SSL handshake. The value is a number of seconds in the range 0 through 86400.

SSLTCBS

The **SSLTCBS** system initialization parameter is obsolete and is only kept for compatibility.

SSLTCBS={8|number}

If it is specified, it is rejected with a message and MAXSSLTCBS is assumed.

START

The **START** system initialization parameter specifies the type of start for the system initialization program.

START=({AUTO|INITIAL|COLD|STANDBY}[,ALL])

The value specified for START, or the default of AUTO, becomes the default value for each resource.

AUTO

CICS performs a warm, emergency, cold or initial start, according to the status of two control records on the global catalog:

- The recovery manager (RM) control record written by the previous execution of CICS
- The RM autostart override record written by a run of the recovery manager utility program, DFHRMUTL

Note: If the global catalog does *not* contain the RM control record:

- If it contains an RM autostart override record with option AUTOINIT, CICS performs an initial start.
- If it does not contain an RM autostart override record with option AUTOINIT, CICS does not start.

If you code START=AUTO, you must do one of the following:

- Provide the global catalog and system log from the previous execution of CICS. For an emergency restart to be successful, you must also have coded an activity keypoint value (see the “AKPFREQ” on page 150 parameter) on the previous execution of CICS.
- Provide a global catalog against which you have run the DFHRMUTL utility program, specifying SET_AUTO_START=AUTOINIT.

You may choose to leave the START parameter set to AUTO for all types of startup other than XRF standby, and use the DFHRMUTL program to reset the startup mode to COLD or INITIAL when necessary, using SET_AUTO_START=AUTOCOLD or SET_AUTO_START=AUTOINIT, respectively. For information about the DFHRMUTL utility program, see the *CICS Operations and Utilities Guide*.

INITIAL

The status of CICS resource definitions saved in the global catalog at the previous shutdown is ignored, and all resource definitions are reinstalled, either from the CSD or CICS control tables.

You should rarely need to specify START=INITIAL; if you want to reinstall definitions of local resources from the CSD, use START=COLD instead.

Examples of times when an initial start is necessary are:

- When bringing up a new CICS system for the first time.
- After a serious software failure, when the system log has been corrupted.
- If the global catalog is cleared or initialized.
- When you want to run CICS with a dummy system log. (If the system log is defined as a dummy, it is ignored.)

COLD

The status of CICS resource definitions saved in the global catalog at the previous shutdown is ignored, and all resource definitions (except those for the system log) are reinstalled, either from the CSD or CICS control tables.

Resynchronization information in the global catalog relating to remote systems or to RMI-connected resource managers is preserved. The CICS system log is scanned during startup, and information regarding unit of work obligations to remote systems, or to non-CICS resource managers (such as DB2) connected through the RMI, is preserved. (That is, any decisions about the outcome of local UOWs, needed to allow remote systems or RMI resource managers to resynchronize their resources, are preserved.)

Note that, on a cold start, the following are *not* preserved:

- Updates to *local* resources that were not fully committed or backed out during the previous execution, *even if the updates were part of a distributed unit of work.*
- Resynchronization information for remote systems connected by LU6.1 links, or for earlier releases of CICS systems connected by MRO.
- Any program LIBRARY definitions that had been dynamically defined. Only the static DFHRPL concatenation will remain, together with any LIBRARY definitions in the group list specified at startup or installed via BAS at startup.

If you want to reinstall resource definitions from the CSD, use START=COLD rather than START=INITIAL.

STANDBY

Coding START=STANDBY, but only when you have also specified XRF=YES, defines this CICS as the alternate CICS region in an XRF pair. In other words, you **must** specify START=STANDBY for the system that starts off as the alternate. (To start an active CICS region, specify AUTO or COLD, as you would without XRF.)

(option,ALL)

The ALL option is a special option you can use on the START parameter when you supply it as a system initialization parameter at CICS startup; you cannot code it in the SIT. If you specify START=(AUTO,ALL), CICS initializes all resources according to the type of startup that it selects (warm, emergency, initial, or cold). The ALL option overrides any individual settings in other system initialization parameters.

However, if you do not use the ALL option, you can individually cold start those resources that have a COLD operand. For details of resources that have a COLD option, see “Defining CICS resource table and module keywords” on page 138.

Restrictions You can specify START=(option,ALL) in PARM, SYSIN, or CONSOLE only.

For more information about the types of CICS startup, see “Controlling start and restart” on page 308.

STARTER

The **STARTER** system initialization parameter specifies whether the generation of starter system modules (with \$ and # suffixes) is permitted, and various MNOTES are to be suppressed.

STARTER={NO|YES}

This parameter should only be used when service is being performed on starter system modules.

Restrictions You can specify the **STARTER** parameter in the SIT only.

STATEOD

The **STATEOD** system initialization parameter specifies the end-of-day time in the format hhmmss.

STATEOD={0|hhmmss}

The default is 0, which is midnight.

End-of-day time is expressed in local time and must be in the range 00:00:00-23:59:59. That is, the hh value cannot exceed 23, and the mm and ss values can be specified in the range 00 to 59. If you leave out leading zeros, the DFHSIT macro inserts them (for example, 100 becomes 000100—that is, 1 minute 00 seconds past midnight).

This parameter is the equivalent of the ENDOFDAY option on the CEMT and EXEC CICS SET STATISTICS command, which you can use to modify the value set by STATEOD.

STATINT

The **STATINT** system initialization parameter specifies the recording interval for system statistics in the format *hhmmss*.

STATINT={030000|*hhmmss* }

The default is 3 hours.

The interval must be at least one minute and cannot be more than 24 hours. The minutes and seconds part of the value can be specified in the range 00 to 59. If you leave out leading zeros, the DFHSIT macro inserts them. For example, 3000 becomes 003000 (that is, an interval of 30 minutes).

This parameter is the equivalent of the INTERVAL option on the CEMT and **EXEC CICS SET STATISTICS** command, which you can use to modify the value set by STATINT.

STATRCD

The **STATRCD** system initialization parameter specifies the interval statistics recording status at CICS initialization.

STATRCD={OFF|**ON**}

This status is recorded in the CICS global catalog for use during warm and emergency restarts. Statistics collected are written to the SMF data set.

OFF Interval statistics are not collected (no action is taken at the end of an interval).

End-of-day statistics are collected at the logical end of day and on shutdown. Unsolicited statistics are written to SMF as resources are discarded or closed.

ON Interval statistics are collected.

On a cold start of a CICS region, interval statistics are recorded by default at three-hourly intervals. All intervals are timed using the end-of-day time (midnight is the default) as a base starting time (**not** CICS startup time). This means that the default settings give collections at 00.00, 03.00, 06.00, 09.00, and so on, regardless of the time that you start CICS.

On a warm or emergency restart the statistics recording status is restored from the CICS global catalog.

You can change the statistics recording status at any time as follows:

- During a warm or emergency restart by coding the STATRCD system initialization parameter.
- While CICS is running by using the CEMT or EXEC CICS SET STATISTICS command.

Whatever the value of the STATRCD system initialization parameter, you can ask for requested statistics and requested reset statistics to be collected. You can get statistics “on demand” for all, or for specified, resource types by using the CEMT or EXEC CICS PERFORM STATISTICS command. The period covered for statistics requested in this way is from the last reset time (that is, from the beginning of the current interval or from when you last issued a CEMT or EXEC CICS statistics command specifying RESETNOW) up to the time that you issue the PERFORM STATISTICS command.

For information about using these CEMT commands, see *CICS Supplied Transactions*. For programming information about the EXEC CICS PERFORM commands, see the *CICS System Programming Reference*. For information about the statistics utility program DFHSTUP, or recording statistics in the sample program hlq.SAMPLIB, see the *CICS Operations and Utilities Guide*.

STGPROT

The **STGPROT** system initialization parameter specifies whether you want storage protection in the CICS region.

STGPROT={NO|YES}

The permitted values are NO (the default), or YES:

NO If you specify NO, or allow this parameter to default, CICS does not operate any storage protection, and runs in a single storage key as in earlier releases. See Table 21 on page 338 for a summary of how STGPROT=NO affects the storage allocation for the dynamic storage areas.

YES If you specify YES, and if you have the required hardware and software, CICS operates with storage protection, and observes the storage keys and execution keys that you specify in various system and resource definitions. See Table 21 on page 338 for a summary of how STGPROT=YES affects the storage allocation for the dynamic storage areas.

If you do not have the required hardware and software support, CICS issues an information message during initialization, and operates without storage protection.

STGRVCVY

The **STGRVCVY** system initialization parameter specifies whether CICS should try to recover from a storage violation.

STGRVCVY={NO|YES}

Valid values are as follows:

NO CICS does not try to repair any storage violation that it detects.

YES CICS tries to repair any storage violation that it detects.

In both cases, CICS continues unless you have specified in the dump table that CICS should terminate.

In normal operation, CICS sets up four task-lifetime storage subpools for each task. Each element in the subpool starts and ends with a 'check zone' that includes the subpool name. At each freemain, and at end-of-task, CICS checks the check zones and abends the task if either has been overwritten.

Terminal input-output areas (TIOAs) have similar check zones, which are set up with identical values. At each freemain of a TIOA, CICS checks the check zones and abends the task if they are not identical.

If you specify **STGRVCY(YES)**, CICS resets the check zones correctly and the task continues running.

If you specify **STGRVCY(NO)**, CICS abends the task if it is still running. The storage is not reusable and is not returned to the DSA for the remainder of the CICS cycle. If an error is detected when the task ends, no abend is issued. Any sync point that has taken place could save data that is corrupted.

STNTR

The **STNTR** system initialization parameter specifies the level of standard tracing required for CICS as a whole.

STNTR={1 | (1[,2] [,3]) | ALL | OFF}

It is possible to select up to 32 levels of tracing using the **STNTR** system initialization parameter. However, most CICS components only use levels 1, 2 and 3, and some do not have trace points at all these levels. The exceptions are the SM component (storage manager domain), which also has level 4 tracing; and the SJ component (JVM domain), which also has trace levels 29–32, that are reserved to indicate the JVM trace levels 0, 1, and 2, plus a user-definable JVM trace level. You should use the **STNTRxx** system initialization parameter, rather than the **STNTR** system initialization parameter, if you need to set standard tracing levels above 3 for these components.

CAUTION:

Before globally activating tracing levels 3 and ALL, which will set these tracing levels for the storage manager (SM) component and the JVM domain (SJ) component, read the warnings given in the description for the STNTRxx system initialization parameter.

number

Code the level number(s) for the level of standard tracing you want for all CICS components. The options are: 1, (1,2), or (1,2,3). The default, 1, specifies standard tracing for level 1 for all CICS components.

ALL Enables standard tracing for all levels.

OFF Disables standard tracing.

For information about the differences between special and standard CICS tracing, see the *CICS Problem Determination Guide*.

STNTRxx

The **STNTRxx** system initialization parameter specifies the level of standard tracing you require for a particular CICS component.

STNTRxx={1 | (1[,2] [,3] [,4] [,29] [,30] [,31] [,32]) | ALL | OFF}

You identify the component by coding a value for xx in the keyword. You code one **STNTRxx** keyword for each component you want to define selectively. For a CICS component being specially traced that does not have its trace level set by **STNTRxx**, the trace level is that set by **STNTR** (which, in turn, defaults to 1). You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels.

The CICS component codes that you can specify for xx on this **STNTRxx** keyword are shown in the following table:

| Code | Component name |
|-------------|------------------------------|
| AP | Application domain |
| BA | Business application manager |
| BF* | Built-in function |
| BM* | Basic mapping support |
| BR* | 3270 bridge |
| CP* | Common programming interface |
| DC* | Dump compatibility layer |
| DD | Directory manager domain |
| DH | Document handling domain |
| DI* | Data interchange |
| DM | Domain manager domain |
| DP | Debugging profiles domain |
| DS | Dispatcher domain |
| DU | Dump domain |
| EC* | Event capture and emission |
| EI* | Exec interface |
| EJ | Enterprise Java domain |
| EM | Event manager domain |
| EP | Event processing domain |
| FC* | File control |
| GC | Global catalog domain |
| IC* | Interval control |
| IE | ECI over TCP/IP domain |
| II | IIOP domain |
| IS* | ISC or IRC |
| KC* | Task control |
| KE | Kernel |
| LC | Local catalog domain |
| LD | Loader domain |
| LG | Log manager domain |
| LM | Lock domain |
| ME | Message domain |
| ML | Markup language domain |
| MN | Monitoring domain |
| NQ | Enqueue domain |
| OT | Object transaction domain |
| PA | Parameter domain |
| PC* | Program control |
| PG | Program manager domain |
| PI | Pipeline domain |
| PT | Partner domain |

| Code | Component name |
|--|----------------------------------|
| RA | Resource manager adapters |
| RI* | Resource manager interface (RMI) |
| RL | Resource life-cycle domain |
| RM | Recovery manager domain |
| RS | Region status domain |
| RX | RRS-coordinated EXCI domain |
| RZ | Request streams domain |
| SC* | Storage control |
| SH | Scheduler services domain |
| SJ | JVM domain |
| SM | Storage manager domain |
| SO | Sockets domain |
| ST | Statistics domain |
| SZ* | Front End Programming Interface |
| TC* | Terminal control |
| TD* | Transient data |
| TI | Timer domain |
| TR | Trace domain |
| TS | Temporary storage domain |
| UE* | User exit interface |
| US | User domain |
| WB | Web domain |
| WU | System management RESTful API |
| W2 | Web 2.0 domain |
| XM | Transaction manager domain |
| XS | Security manager domain |
| Note: | |
| 1. Components marked * are subcomponents of the AP domain. The trace entries for these components are produced with a trace point ID of AP <i>nnnn</i> . | |
| 2. For the DS domain function CHANGE_MODE, a trace entry is generated if DS level 2 or 3 tracing is active. | |

ALL You want all the available levels of standard tracing switched on for the specified component.

Warning! Selecting ALL for standard tracing for the storage manager (SM) component, or the temporary storage domain (TS), degrades the performance of your CICS region. This is because ALL switches on trace flags that are used by SM domain for field engineering purposes.

Warning! Selecting ALL for standard tracing for the JVM domain (SJ) component is not recommended. JVM trace can produce a large amount of output, so you should normally activate JVM trace for special transactions (using the SPCTRSJ system initialization parameter), rather than turning it on globally for all transactions.

number

The level number(s) for the level of standard tracing you want for the CICS component indicated by xx. Level numbers 1, 2, 3, 4, 29, 30, 31 and 32 can be used, depending on the component.

Most CICS components only use levels 1, 2 and 3, and some do not have trace points at all these levels. The exceptions are:

- The SM component (storage manager domain), which also has level 4 tracing. This level of tracing is intended for IBM field engineering staff.

Warning! Selecting tracing levels 3, 4, or ALL for standard tracing for the storage manager (SM) component, or the temporary storage domain (TS), degrades the performance of your CICS region. This is because options 3 and 4 (and ALL) switch on trace flags that are used by SM domain for field engineering purposes.

SM trace flag 3 deactivates the quickcell mechanism, and SM trace flag 4 forces subpool element chaining on every CICS subpool. Furthermore, once these settings have been activated during system initialization, they cannot be unset, either through a PLTPI program or by using the CETR trace transaction, because they are not used for tracing as such. Thus, a significant performance overhead is incurred if these storage manager trace levels are selected for standard tracing.

See the *CICS Problem Determination Guide* for information about the effects of trace levels 3 and 4.

- The SJ component (JVM domain), which also has trace levels 29–32, that are reserved to indicate the JVM trace levels 0, 1, and 2, plus a user-definable JVM trace level. You can use the system initialization parameters JVMLEVEL0TRACE, JVMLEVEL1TRACE, JVMLEVEL2TRACE and JVMUSERTRACE to specify options for these JVM trace levels, and then activate them using the SPCTRSJ system initialization parameter.

Warning! Selecting tracing levels 29, 30, 31, 32 or ALL for standard tracing for the JVM domain (SJ) component is not recommended. JVM trace can produce a large amount of output, so you should normally activate JVM trace for special transactions (using the SPCTRSJ system initialization parameter), rather than turning it on globally for all transactions.

OFF Switches off all levels of standard CICS tracing for the CICS component indicated by xx.

Restrictions You can specify the STNTRxx parameter in PARM, SYSIN, or CONSOLE only.

SUBTSKS

The **SUBTSKS** system initialization parameter specifies the number of task control blocks (TCBs) you want CICS to use for running tasks in concurrent mode.

SUBTSKS={0|1}

A concurrent mode TCB allows CICS to perform management functions as system subtasks.

CICS always uses at least two TCBs:

1. The quasi-reentrant mode TCB. CICS runs all user applications under this TCB.

2. The resource-owning mode TCB. CICS runs tasks that open and close files under this TCB.

If you specify SUBTSKS=0, CICS runs under these two TCBs.

If you specify SUBTSKS=1, CICS uses an additional TCB, a concurrent mode TCB, to perform system subtasking functions.

SUFFIX

The **SUFFIX** system initialization parameter specifies the last two characters of the name of this system initialization table.

SUFFIX=xx

The first 6 characters of the name of the SIT are fixed as DFHSIT. You can specify the last two characters of the name, using the SUFFIX parameter. Because the SIT does not have a TYPE=INITIAL macro statement like other CICS resource control tables, you specify its SUFFIX on the TYPE=CSECT macro statement.

The suffix allows you to have more than one version of the SIT. Any one or two characters (other than NO and DY) are valid. You select the version of the table to be loaded into the system during system initialization by coding SIT=xx, either in the PARM parameter or the SYSIN data set. (You can, in some circumstances, specify the SIT using the system console, but this is not recommended.)

Restrictions You can specify the **SUFFIX** parameter in the SIT only.

SYDUMAX

The **SYDUMAX** system initialization parameter specifies the limit on the number of system dumps that can be taken per dump table entry.

SYDUMAX={999|number}

If this number is exceeded, subsequent system dumps for that particular entry will be suppressed. The **SYDUMAX** parameter applies for new or added system dump codes. It does not override the limit on the number of system dumps for existing dump table entries.

number

A number in the range 0 through 999. The default, 999, enables an unlimited number of dumps to be taken.

SYSIDNT

The **SYSIDNT** system initialization parameter specifies a 1-4 character name that is known only to your CICS region.

SYSIDNT={CICS|name}

If your CICS region also communicates with other CICS regions, the name you choose for this parameter to identify your local CICS region must not be the same name as an installed CONNECTION resource definition for a remote region.

The value for SYSIDNT, whether specified in the SIT or as an override, can only be updated on a cold start. After a warm start or emergency restart, the value of SYSIDNT is that specified in the last cold start.

For information about the SYSIDNT of a local CICS region, see the *CICS Intercommunication Guide*.

SYSTR

The **SYSTR** system initialization parameter specifies the setting of the master system trace flag.

SYSTR={ON|OFF}

Valid values are as follows:

ON The master trace flag is set, causing CICS to write trace entries of system activity for the individual CICS components. Trace entries are captured and written only for those components for which the trace level is 1 or greater, as specified on the STNTR or STNTRxx system initialization parameters. Entries are written only to those trace destinations that are active.

OFF The master trace flag is unset, and no standard trace entries are captured, overriding any trace levels specified by the STNTR or STNTRxx system initialization parameters.

Note: Setting the master trace flag OFF affects only standard tracing and has no effect on special tracing, which is controlled separately by SPCTR or SPCTRxx trace levels and the CETR transaction.

See the *CICS Problem Determination Guide* for more information about controlling CICS trace.

TAKEOVR

The **TAKEOVR** system initialization parameter specifies the action to be taken by the alternate CICS region, following the apparent loss of the surveillance signal in the active CICS region.

TAKEOVR={MANUAL|AUTO|COMMAND} (**alternate**)

Use this parameter in the SIT for an alternate CICS region. This parameter also specifies the level of operator involvement.

If both active and alternate CICS regions are running under different MVS images in the same sysplex, and an MVS failure occurs in the MVS image of the active CICS region, the TAKEOVR option is overridden.

- If the MVS images are running in a PR/SM environment, CICS XRF takeover to an alternate CICS region on a separate MVS image completes without the need for any operator intervention.
- If the MVS images are not running in a PR/SM environment, the CICS takeover is still initiated automatically, but needs operator intervention to complete, because XCF outputs a WTOR (IXC402D). Sysplex partitioning does not complete until the operator replies to this message, and CICS waits for sysplex partitioning to complete before completing the XRF takeover.

MANUAL

The operator is asked to approve a takeover if the alternate CICS region cannot detect the surveillance signal of the active CICS region.

The alternate CICS region does not ask the operator for approval if the active CICS region signs off abnormally, or if there is an operator or program command for takeover. In these cases, there is no doubt that the alternate CICS region should take over, and manual involvement by the operator would be an unnecessary overhead in the takeover process.

You could use this option, for instance, to ensure manual takeover of a master or coordinator region in MRO.

AUTO

No operator approval, or intervention, is needed for a takeover.

COMMAND

Takeover occurs only when a CEBT PERFORM TAKEOVER command is received by the alternate CICS region. It ensures, for instance, that a dependent alternate CICS region, in MRO, is activated only if it receives the command from the operator, or from a master or coordinator region.

TBEXITS

The **TBEXITS** system initialization parameter specifies the names of your backout exit programs for use during emergency restart backout processing.

TBEXITS=([name1] [,name2] [,name3] [,name4] [,name5] [,name6])

The order in which you code the names is significant. If you do not want to use all the exits, code commas in place of the names you omit. For example:

TBEXITS=(, ,EXITF,EXITV)

The program names for *name1* through *name6* apply to global user exit points as follows:

- *name1* and *name2* are the names of programs to be invoked at the XRCINIT and XRCINPT global user exit points (but note that XRCINIT and XRCINPT are invoked only for user log records).
- *name3* is the name of the program to be invoked at the file control backout failure global user exit point, XFCBFAIL.
- *name4* is the name of the program to be invoked at the file control logical delete global user exit point, XFCLDEL.
- *name5* is the name of the program to be invoked at the file control backout override global user exit point, XFCBOVER.
- *name6* is the name of the program to be invoked at the file control backout override global user exit point, XFCBOUT.

This exit is invoked (if required) during backout of a unit of work, regardless of whether the backout is taking place at emergency restart, or at any other time.

The XFCBFAIL, XFCLDEL, and XFCBOVER global user exit programs are enabled on all types of CICS start if they are named on the TBEXITS system initialization parameter.

If no backout exit programs are required, you can do one of the following:

- Omit the **TBEXITS** system initialization parameter altogether
- Code the parameter as TBEXITS=(, , , , , ,)

TCAM

The **TCAM** system initialization parameter is obsolete and is only kept for compatibility.

TCAM={NO|YES}

If it is specified, it is rejected with a message and TCAM=NO is assumed.

TCP

The **TCP** system initialization parameter specifies whether the pregenerated non-VTAM terminal control program, DFHTCP, is to be included.

TCP={YES|NO}

You must code TCP=YES if you intend using card reader/line printer (sequential) devices.

TCPIP

The **TCPIP** system initialization parameter specifies whether CICS TCP/IP services are to be activated at CICS startup.

TCPIP={NO|YES}

The default is NO, meaning that these services cannot be enabled. If TCPIP is set to YES, the IPIC, HTTP, IIOPI, and ECI over TCP/IP services can process work.

For IPIC, you must specify TCPIP=YES and ISC=YES.

Note: The TCPIP system initialization parameter affects only CICS internal TCP/IP Services defined by TCPIPSERVICE resource definitions. It has nothing to do with the TCP/IP socket interface for CICS feature of TCP/IP for MVS.

TCSACTN

The **TCSACTN** system initialization parameter specifies the required action that CICS terminal control should take if the terminal control shutdown wait threshold expires.

TCSACTN={NONE|UNBIND|FORCE}

For details of the wait threshold, see the **TCSWAIT** system initialization parameter. **TCSACTN** only takes effect when **TCSWAIT** is coded with a value in the range 1 through 99. This is a global default action. On a terminal-by-terminal basis, you can code a DFHZNEP routine to override this action.

NONE

No action is taken. This can be overridden by DFHZNEP.

- To report hung terminals and not attempt to force-close them specify the TCSWAIT=mm (with an appropriate time interval) and TCSACTN=NONE system initialization parameters.
- To attempt to force-close some hung terminals, and only report others, specify the TCSWAIT=mm (with an appropriate time interval) and TCSACTN=NONE system initialization parameters, and code a DFHZNEP routine that selects the required terminals and sets TWAOCN on for them.

UNBIND

CICS terminal control attempts to close the session by issuing a VTAM CLSDST and sending an SNA UNBIND command to the hung terminal. This can be overridden by DFHZNEP.

- To attempt to force-close all hung terminals specify the TCSWAIT=mm (with an appropriate time interval) and TCSACTN=UNBIND system initialization parameters.

FORCE

CICS terminal control attempts to forceclose the CICS VTAM ACB if there are any hung terminals or parallel connection sessions. All CICS VTAM terminals and sessions are released and CICS normal shutdown continues. This parameter will only take effect if all LU Type 6.2 parallel connections, if any, have successfully completed CNOS close processing.

- To attempt to force-close the CICS VTAM ACB if there are any hung terminals, specify the TCSWAIT=mm (with an appropriate time interval) and TCSACTN=FORCE system initialization parameters.

TCSWAIT

The **TCSWAIT** system initialization parameter specifies the required CICS terminal control shutdown wait threshold.

TCSWAIT={4|number|NO|NONE|0}

The wait threshold is the time, during shutdown, that CICS terminal control allows to pass before it considers terminal shutdown to be hung. If all VTAM sessions shutdown and close before the threshold expires then the CICS shutdown process moves on to its next stage, and the terminal control wait threshold then no longer applies. If, however, some of the VTAM sessions do not complete shutdown and close, then CICS takes special action with these sessions. For details of this special action see the description of the TCSACTN system initialization parameter. The wait threshold only applies to VTAM sessions; that is, VTAM terminals and VTAM intersystem connections. The wait time is specified as a number of minutes, in the range 1 through 99. As a special case, TCSWAIT=NO may be specified to indicate that terminal control shutdown is never to be considered hung, no matter how long the shutdown and close process takes. TCSWAIT=NONE and TCSWAIT=0 are alternative synonyms for TCSWAIT=NO, and all three have the same effect (internally they are held as the one value 0 (zero)).

The value that you specify on the TCSWAIT system initialization parameter should be large enough so that under normal circumstances all VTAM terminals and connections shutdown in an orderly fashion. To help choose this value, consider using a value slightly larger than the elapsed time between the following two CICS terminal control shutdown messages:

```
DFHZC2305 Termination of VTAM sessions beginning
DFHZC2316 VTAM ACB is closed
```

TCT

The **TCT** system initialization parameter specifies which terminal control table, if any, is to be loaded.

TCT={NO|xx|YES}

For guidance about coding the macros for this table, see the *CICS Resource Definition Guide*.

If you reassemble the TCT after starting CICS, any changes are applied when you next start CICS, even if it is a warm or emergency startup.

If you have VTAM-connected terminals only, you can specify TCT=NO. If you do this, a dummy TCT called DFHTCTDY, is loaded during system initialization. For more information about DFHTCTDY, see “The dummy TCT, DFHTCTDY” on page 302. If you code TCT=NO, you must specify a CSD group list in the **GRPLIST** parameter.

TCTUAKEY

The **TCTUAKEY** system initialization parameter specifies the storage key for the terminal control table user areas (TCTUAs) if you are operating CICS with storage protection (STGPROT=YES).

TCTUAKEY={USER|CICS}

The permitted values are USER (the default), or CICS:

USER CICS obtains the amount of storage for TCTUAs in user key. This allows a user program executing in any key to modify the TCTUA.

CICS CICS obtains the amount of storage in CICS key. This means that only programs executing in CICS key can modify the TCTUA, and user-key programs have read-only access.

If CICS is running without storage protection, the TCTUAKEY parameter only designates which DSA (User or CICS) the storage comes from. The TCTUAs are accessed in CICS-key whether they are in the UDSA or CDSA.

See “The terminal control table user areas” on page 335 for more information about TCTUAs.

TCTUALOC

The **TCTUALOC** system initialization parameter specifies where terminal user areas (TCTUA) are to be stored.

TCTUALOC={BELOW|ANY}

Valid values are as follows:

BELOW

The TCTUAs are stored below the 16 MB line.

ANY The TCTUAs are stored anywhere in virtual storage. CICS stores TCTUAs above the 16 MB line if possible.

For more information about TCTUAs, see the *CICS Application Programming Guide*.

For details about defining terminals using RDO, see the *CICS Resource Definition Guide*.

TD

The **TD** system initialization parameter specifies the number of VSAM buffers and strings to be used for intrapartition transient data (TD).

TD=({3|decimal-value-1}[,{ 3|decimal-value-2}])

Valid values are as follows:

decimal-value-1

The number of buffers to be allocated for the use of intrapartition transient data. The value must be in the range 1 through 32 767. The default value is 3.

CICS obtains, above the 16 MB line, storage for the TD buffers in units of the page size (4 KB). Because CICS optimizes the use of the storage obtained, TD may allocate more buffers than you specify, depending on the control interval (CI) size you have defined for the intrapartition data set.

For example, if the CI size is 1536, and you specify 3 buffers (the default number), CICS allocates 5 buffers. This is because 2 pages (8192 bytes) are required to obtain sufficient storage for three 1536-byte buffers, a total of only 4608 bytes, which would leave 3584 bytes of spare storage in the second page. In this case, CICS allocates another 2 buffers (3072 bytes) to minimize the amount of unused storage. In this way CICS makes use of storage that would otherwise be unavailable for any other purpose.

decimal-value-2

The number of VSAM strings to be allocated for the use of

intrapartition transient data. The value must be in the range 1 through 255, and must not exceed the value specified in decimal-value-1. The default value is 3.

For example, TD=(8,5) specifies 8 buffers and 5 strings.

The operands of the TD parameter are positional. You must code commas to indicate missing operands if others follow. For example, TD=(,2) specifies the number of strings and allows the number of buffers to default.

TDINTRA

The **TDINTRA** system initialization parameter specifies whether CICS is to initialize with empty intrapartition TD queues.

TDINTRA={NOEMPTY | EMPTY}

Valid values are as follows:

NOEMPTY

CICS recovers all the intrapartition TD queues to the state they were in at the previous termination of CICS, as in a normal emergency restart. The TD queue resource definitions are recovered from the CICS global catalog.

EMPTY

CICS initializes with all the intrapartition TD queues empty. This option must be used when CICS is initializing in remote site recovery mode (OFFSITE=YES).

You can optionally use this option to COLD start your intrapartition TD queues to initialize them as empty.

The option is significant only on warm and emergency restarts—cold starts always initialize with empty queues. Note that the EMPTY option may cause data integrity problems because all indoubt log records associated with logically recoverable TD queues are discarded.

The TD queue resource definitions are recovered from the CICS global catalog.

TDSUBTASK

The **TDSUBTASK** system initialization parameter specifies whether CICS should use the FO TCB to write to an extrapartition transient data queue, where the record format is FIXED and the block format is UNBLOCKED.

TDSUBTASK={OFF | ON}

The default is OFF, so that no TCB switch can occur. This particularly benefits you if you are submitting work to JES using the internal reader (INTRDR).

TRANISO

The **TRANISO** system initialization parameter specifies, together with the **STGPROT** system initialization parameter, whether you want transaction isolation in the CICS region.

TRANISO={NO | YES}

The permitted values are NO (the default), or YES.

NO This is the default. If you specify NO, or allow this parameter to default, CICS operates without transaction isolation, and all storage in

the CICS address space is addressable. If you specify STGPROT=YES and TRANISO=NO, CICS storage protection is active without transaction isolation.

YES Transaction isolation is required. If you specify TRANISO=YES and STGPROT=YES, and you have the required hardware and software, CICS operates with transaction isolation. This ensures that the user-key task-lifetime storage of transactions defined with the ISOLATE(YES) option is isolated from the user-key programs of other transactions.

If you specify TRANISO=YES, but you do not have the required hardware and software or STGPROT=NO is specified, CICS issues an information message during initialization, and operates without transaction isolation.

STGPROT=NO and TRANISO=YES specified in the system initialization table causes an error during assembly (MNOTE 8).

Note: VSAM nonshared resources (NSR) are not supported for transactions that use transaction isolation. You should specify ISOLATE(NO) when you define transactions that access VSAM files using NSR. You can also function ship the file request to a remote region. The DFHMIRS program that carries out the request is defined with an EXECKEY of CICS. A CICS-key program has read and write access to CICS-key and user-key storage of its own task and all other tasks, whether or not transaction isolation is active.

TRAP

The **TRAP** system initialization parameter specifies whether the FE global trap exit is to be activated at system initialization.

TRAP={OFF|ON}

This exit is for diagnostic use under the guidance of service personnel. For background information about this exit, see the *CICS Problem Determination Guide*.

TRDUMAX

The **TRDUMAX** system initialization parameter specifies the limit on the number of transaction dumps that may be taken per Dump Table entry.

TRDUMAX={999|number}

If this number is exceeded, subsequent transaction dumps for that particular entry will be suppressed.

number

A number in the range 0 through 999. The default, 999, enables an unlimited number of dumps to be taken.

TRTABSZ

The **TRTABSZ** system initialization parameter specifies the size in kilobytes of the internal trace table.

TRTABSZ={4096|number-of-kilobytes}

(1 KB = 1024 bytes.) The CICS trace table is allocated in virtual storage above the 16 MB line, and it is allocated **before** the extended CICS-key DSA (ECDSA) and the extended user-key DSA (EUDSA). Ensure that there is sufficient virtual storage for the trace table, the ECDSA, and the EUDSA by specifying a large enough region size on the MVS REGION parameter of your CICS job.

Use caution when setting this parameter to a very high value. There must be enough MVS page storage to satisfy the request as well as the DSA sizes. Use the **DISPLAY ASM MVS** system command to display current information about the status and utilization of all MVS page data sets.

4096 4096 KB is the default size of the trace table.

number

The number of kilobytes of storage to be allocated for the internal trace table, in the range 16 KB through 1048576 KB (1 GB). Subpool 1 is used for the trace table storage, which exists for the duration of the CICS execution. The table is page aligned and occupies a whole number of pages. If the value specified is not a multiple of the page size (4 KB), it is rounded up to the next multiple of 4 KB.

Trace entries are of variable lengths, but the average length is approximately 100 bytes.

Note: To switch on internal tracing, use the “INTTR” on page 193 parameter.

TRTRANSZ

The **TRTRANSZ** system initialization parameter specifies the size in kilobytes of the transaction dump trace table.

TRTRANSZ={16|number-of-kilobytes}

(1 KB = 1024 bytes.)

When a transaction dump is taken, CICS performs an MVS GETMAIN for storage above the 16 MB line for the transaction dump trace table.

16 16 KB is the default size of the transaction dump trace table.

number

The number of kilobytes of storage to be allocated for the transaction dump trace table, in the range 16 KB through 1048576 KB.

TRTRANTY

The **TRTRANTY** system initialization parameter specifies which trace entries should be copied from the internal trace table to the transaction dump trace table.

TRTRANTY={TRAN|ALL}

Valid values are as follows:

TRAN Only the trace entries associated with the transaction that is abending will be copied to the transaction dump trace table.

ALL All of the trace entries from the internal trace table will be copied to the transaction dump trace table. If the internal trace table size is larger than the transaction dump trace table size, the transaction dump trace table could wrap. This results in only the most recent trace entries being written to the transaction dump trace table.

TS

The **TS** system initialization parameter specifies whether you want to cold start temporary storage, as well as the number of VSAM buffers and strings to be used for auxiliary temporary storage.

TS=([COLD] [, {0|3|decimal-value-1 }][, {3|decimal-value-2}])

Valid values are as follows:

COLD

The type of start for the temporary storage facility. COLD forces a cold start regardless of the value of the START parameter. If COLD is omitted, the TS start type is determined by the value of START.

0 No buffers are required; that is, only MAIN temporary storage is required.

decimal-value-1

The number of buffers to be allocated for the use of auxiliary temporary storage. The value must be in the range 3 through 32 767.

decimal-value-2

The number of VSAM strings to be allocated for the use of auxiliary temporary storage. The value must be in the range 1 through 255, and must not exceed the value specified in decimal-value-1. The default value is 3.

For example, TS=(8,5) specifies 8 buffers and 5 strings.

The operands of the TS parameter are positional. You must code commas to indicate missing operands if others follow. For example, TS=(,8) specifies the number of buffers and allows the other operands to default.

TST

The **TST** system initialization parameter specifies the temporary storage table suffix.

TST={NO|YES|xx}

See "Defining CICS resource table and module keywords" on page 138 for information on defining the temporary storage table suffix.

For information about coding the macros for this table, see the *CICS Resource Definition Guide*

UDSASZE

The **UDSASZE** system initialization parameter specifies the size of the UDSA.

UDSASZE={0K|number}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify number as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256KB). If the size specified is not a multiple of 256KB (or 1MB if transaction isolation is active), CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions You can specify the UDSASZE parameter in PARM, SYSIN, or CONSOLE only.

UOWNETQL

The **UOWNETQL** system initialization parameter specifies a qualifier for the NETUOWID for units of work initiated on the local CICS region.

UOWNETQL=user_defined_value

UOWNETQL is required only if VTAM=NO is coded. The specified value is used in the following circumstances:

- CICS is being cold started and VTAM=NO has been specified.
- CICS is being cold started and the VTAM ACB has failed to open.
- CICS is being started with VTAM=NO and the VTAM ACB has not been opened since CICS was last cold started.
- CICS is being started, the VTAM ACB has failed to open, and the VTAM ACB has not been opened since CICS was last cold started.

If any of the above conditions apply and UOWNETQL is not specified, a dummy default UOWNETQL of 9UNKNOWN is used. This dummy UOWNETQL is invalid because the first character is a number. UOWNETQL is given this invalid name to avoid a conflict with any real, valid netid.

If any of the above conditions apply, UOWNETQL, or its default value, is used as the IPIC NETWORKID of this CICS region. It is also used as the default NETWORKID on IPCONN definitions for IPIC connections to other CICS regions.

The value you code can be from 1 to 8 characters long, and must consist of uppercase letters (A through Z), or numbers in the range 0 through 9. The first character must be a letter.

USERTR

The **USERTR** system initialization parameter specifies whether the master user trace flag is to be set on or off.

USERTR={ON|OFF}

If the user trace flag is off, the user trace facility is disabled, and EXEC CICS ENTER TRACENUM commands receive an INVREQ condition if EXCEPTION is not specified. If the program does not handle this condition the transaction will abend AEIP.

For programming information about the user trace facility using EXEC CICS ENTER TRACENUM commands, see the *CICS Application Programming Reference* manual.

USRDELAY

The **USRDELAY** system initialization parameter specifies the maximum time, in the range 0 - 10080 minutes (up to seven days), that an eligible user ID and its associated attributes are to be retained in the user table if the user ID is unused.

USRDELAY={30|number}

An entry in the user table for a user ID that is retained during the delay period can be reused.

A user ID that is eligible for reuse in the **USRDELAY** period must have one of the following properties:

- Received from remote systems
- Specified on the SECURITYNAME attribute in the CONNECTION resource
- Specified on the USERID attribute in the SESSIONS resource
- Specified on the USERID attribute in the definition of an intrapartition transient data queue
- Specified on the USERID option on a START command

- Specified on the USERID attribute for a non-terminal task, such as the alias tasks that are attached for processing HTTP requests

Within the **USRDELAY** period, a user ID in any one of these categories can be reused in one of the other categories, provided that the request for reuse has the same qualifiers. If a user ID is qualified by a different group ID, APPLID, or terminal ID, a retained entry is not reused, except when changing the terminal ID on LU6.2 when the retained entry is used.

If a user ID is unused for more than the **USRDELAY** limit, it is removed from the system, and the message DFHUS0200 is issued. You can suppress this message in an XMEOUT global user exit program. If you specify **USRDELAY=0**, all eligible user IDs are deleted immediately after use, and the message DFHUS0200 is not issued. Do not code **USRDELAY=0** if:

- This CICS region communicates with other CICS regions, and:
- ATTACHSEC=IDENTIFY is specified on the CONNECTION resource for the connections used, and
- The connections used carry high volumes of transaction routing or function shipping activity.

You should specify a value that gives the optimum level of performance for your CICS environment.

If you specify **USRDELAY=0** in the above scenario, CICS drives a full signon for each incoming request (with I/O to RACF) and a full signoff at the end of each transaction. For function shipping, there might be multiple signons and signoffs driven on a data-owning region for one task on an AOR.

If a value, other than 0, is specified for **USRDELAY**, the ability to change the attributes of the user or revoke the user ID becomes more difficult because the user ID and its attributes are retained in the region until the **USRDELAY** value has expired. For example, if you have specified **USRDELAY=30** for a user ID, but that user ID continues to run transactions every 25 minutes, the **USRDELAY** value never expires and any changes made to the user ID never comes into effect.

When running a remote transaction, a user ID remains signed-on to the remote CICS region (after the conversation associated with the first attach request is complete) until the delay specified by **USRDELAY** has elapsed since the last transaction associated with the attach request for the user ID has completed. When this event occurs, the user ID is removed from the remote CICS region.

If you specify a low value for the **USRDELAY** system initialization parameter to ensure that CICS quickly detects changes to RACF profiles, you might want to increase this value if your z/OS system is z/OS 1.11 or above, because from z/OS 1.11, CICS is notified immediately if RACF profile changes occur. The primary impact of a high **USRDELAY** value is that the amount of storage used for RACF control blocks is increased.

USSHOME

The **USSHOME** system initialization parameter specifies the name and path of the root directory for CICS TS 4.1 files on z/OS UNIX.

USSHOME={/usr/lpp/cicsts/cicsts41/ | *directory* | **NONE**}

The value for the **USSHOME** system initialization parameter must match the directory that you specified for CICS TS 4.1 files on z/OS UNIX when you installed CICS using the DFHISTAR installation job. The default value for the

USSHOME system initialization parameter is /usr/lpp/cicsts/cicsts41, which matches the default values for the DFHISTAR installation job.

If you changed any of the **TINDEX**, **PATHPREFIX**, or **USSDIR** parameters in the DFHISTAR installation job, you must specify a value for the **USSHOME** system initialization parameter to match the name and path that you specified for the root directory using those DFHISTAR parameters.

If you specify **USSHOME=NONE** instead of specifying a directory name, CICS does not use any default root directory in the UNIX System Services file system. In this case, some CICS functions that request data from this directory might produce unpredictable results.

VTAM

The **VTAM** system initialization parameter specifies whether the VTAM access method is to be used.

VTAM={YES|NO}

The default is VTAM=YES.

VTPREFIX

The **VTPREFIX** system initialization parameter specifies the first character to be used for the terminal identifiers (termids) of autoinstalled virtual terminals.

VTPREFIX={\|character}

Virtual terminals are used by the External Presentation Interface (EPI) and terminal emulator functions of the CICS Client products.

Termids generated by CICS for autoinstalled Client terminals consist of a 1-character prefix and a 3-character suffix. The default prefix is '\'. The suffix can have the values 'AAA' through '999'. That is, each character in the suffix can have the value 'A' through 'Z' or '0' through '9'. The first suffix generated by CICS has the value 'AAA'. This is followed by 'AAB', 'AAC', ... 'AAZ', 'AA0', 'AA1', and so on, up to '999'.

Each time a Client virtual terminal is autoinstalled, CICS generates a 3-character suffix that it has not recorded as being in use.

By specifying a prefix, you can ensure that the termids of Client terminals autoinstalled on this system are unique in your transaction routing network. This prevents the conflicts that could occur if two or more terminal-owning regions (TORs) ship definitions of Client virtual terminals to the same application-owning region (AOR).

If such a naming conflict does occur—that is, if a Client virtual terminal is shipped to an AOR on which a remote terminal of the same name is already installed—the autoinstall user program is invoked in the AOR. Your user program can resolve the conflict by allocating an alias terminal identifier to the shipped definition. For details of writing an autoinstall user program to install shipped definitions, see the *CICS Customization Guide*. However, you can avoid potential naming conflicts by specifying a different prefix, reserved for virtual terminals, on each TOR on which Client virtual terminals are to be installed.

You must not use the characters + - * < > = { } or blank.

Note:

1. When specifying a prefix, ensure that termids generated by CICS for Client terminals do not conflict with those generated by your autoinstall user program for user terminals, or with the names of any other terminals or connections.
2. Client terminal definitions are not recovered after a restart. Immediately after a restart, no Client terminals are in use, so when CICS generates suffixes it begins again with 'AAA'. This means that CICS does **not** always generate the same termid for any given Client terminal. This in turn means that server applications should not assume that a particular CICS-generated termid always equates to a particular Client terminal.
If your server programs do make this assumption, you can use your autoinstall user program to allocate alias termids, by which the virtual terminals will be known to CICS, in a consistent manner.
3. Clients can override CICS Transaction Server for z/OS-generated termids.

WEBDELAY

The **WEBDELAY** system initialization parameter specifies two Web delay periods.

WEBDELAY=(5|time_out,60|keep_time)

These periods are:

1. A time-out period. The maximum time, in minutes, in the range 1-60, that a transaction started through the Web 3270 bridge interface, is allowed to remain in terminal wait state before it is automatically purged by CICS.
2. The terminal keep time. The time, in minutes, in the range 1-6000, during which state data is kept for a CICS Web 3270 bridge transaction, before CICS performs clean-up.

WRKAREA

The **WRKAREA** system initialization parameter specifies the number of bytes to be allocated to the common work area (CWA).

WRKAREA={512|number}

This area, for use by your installation, is initially set to binary zeros, and is available to all programs. It is not used by CICS. The maximum size for the work area is 3584 bytes.

XAPPC

The **XAPPC** system initialization parameter specifies whether RACF session security can be used when establishing APPC sessions.

XAPPC={NO|YES}

Valid values are as follows:

NO RACF session security cannot be used.

YES RACF session security can be used.

If you specify **BINDSECURITY=YES** for a particular APPC connection, a request to RACF is issued to extract the security profile. If the profile exists, it is used to bind the session.

Note: If you specify **XAPPC=YES**, the external security manager that you use must support the APPCLU general resource class, otherwise CICS fails to initialize.

Restrictions You can specify the XAPPC parameter in the SIT, PARM, or SYSIN only.

XCMD

The **XCMD** system initialization parameter specifies whether you want CICS to perform command security checking, and optionally the RACF resource class name in which you have defined the command security profiles.

XCMD={YES | *name* | **NO**}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to use a CICS command for the specified resource. Such checking is performed every time a transaction tries to use a COLLECT, DISABLE, DISCARD, ENABLE, EXTRACT, INQUIRE, PERFORM, RESYNC, or SET command, or any of the FEPI commands, for a resource.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the CMDSEC(YES) option on the transaction resource definition.

YES CICS calls RACF, using the default class name of CICSCMD prefixed by C or V, to check whether the userid associated with a transaction is authorized to use a CICS command for the specified resource. The resource class name is CCICSCMD and the grouping class name is VCICSCMD.

name CICS calls RACF, using the specified resource class name prefixed by C or V, to verify that the userid associated with a transaction is authorized to use a CICS command for the specified resource. The resource class name is *Cname* and the grouping class name is *Vname*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any command security checks, allowing any user to use commands that would be subject to those checks.

Restrictions: You can specify the **XCMD** parameter in the SIT, PARM, or SYSIN only.

XCFGROUP

The **XCFGROUP** system initialization parameter specifies the name of the cross-system coupling facility (XCF) group to be joined by this region.

XCFGROUP={DFHIR000 | *name*}

The group name must be eight characters long, padded on the right with blanks if necessary. The valid characters are A-Z 0-9 and the national characters \$ # and @. To avoid using the names IBM uses for its XCF groups, do not begin group names with the letters A through C, E through I, or the character string "SYS". Also, do not use the name "UNDESIG", which is reserved for use by the system programmer in your installation.

It is recommended that you use a group name beginning with the letters "DFHIR".

You can specify **XCFGROUP** on the SIT macro or as a SYSIN override. You cannot specify it as a console override.

Each CICS region can join only one XCF group, which happens when it signs on to CICS interregion communication (IRC). The default XCF group is DFHIR000.

XCF groups allow CICS regions in different MVS images within the same sysplex to communicate with each other across multi-region operation (MRO) connections.

Note: Regions in the same MVS image too, can communicate with each other using MRO, but this does not require a coupling facility. The only situation in which CICS regions in the same MVS image cannot communicate via MRO is when they are members of different XCF groups.

For introductory information about XCF/MRO, and instructions on how to set up XCF groups, see the *CICS Intercommunication Guide*.

XDB2

The **XDB2** system initialization parameter specifies whether you want CICS to perform DB2ENTRY security checking.

XDB2={NO|name}

Valid values are as follows:

NO CICS does not perform any DB2 resource security checks.

name CICS calls RACF, using the specified general resource class name, to check whether the userid associated with the CICS DB2 transaction is authorized to access the DB2ENTRY referenced by the transaction.

Unlike the other *Xaaa* system initialization parameters, this DB2 security parameter does not provide a YES option that implies a default CICS resource class name for DB2ENTRY resources. You have to specify your own DB2 resource class name.

XDCT

The **XDCT** system initialization parameter specifies whether you want CICS to perform transient data resource security checking.

XDCT={YES|name|NO}

If you specify YES or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to access the transient data destination. Such checking is performed every time a transaction tries to access a transient data destination.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the transaction resource definition.

YES CICS calls RACF, with the default CICS resource class name of CICSXDCT prefixed by D or E, to verify whether the userid associated with the transaction is authorized to access the specified destination.

The resource class name is DCICSDCT and the grouping class name is ECICSDCT.

name CICS calls RACF, using the specified resource class name, to check whether the userid associated with the transaction is authorized to access the specified destination. The resource class name is *Dname* and the grouping class name is *Ename*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any transient data security checks, allowing any user to access any transient data destination.

Restrictions: You can specify the XDCT parameter in the SIT, PARM, or SYSIN only.

XEJB

The **XEJB** system initialization parameter specifies whether support of security roles is enabled.

XEJB={YES|NO}

Valid values are as follows:

YES CICS support for security roles is enabled:

- When an application invokes a method of an enterprise bean, CICS calls the external security manager to verify that the userid associated with the transaction is defined in at least one of the security roles associated with the method.
- When an application invokes the `isCallerInRole()` method, CICS calls the external security manager to determine whether the userid associated with the transaction is defined in the role specified on the method call.

NO CICS support for security roles is disabled:

- CICS does not perform enterprise bean method level checks, allowing any userid to invoke any enterprise bean method.
- The `isCallerInRole()` method always returns a value of true.

Restrictions:

1. You can specify the XEJB parameter in the SIT, PARM, or SYSIN only.
2. To enable security role support, you must also specify SEC=YES.

XFCT

The **XFCT** system initialization parameter specifies whether you want CICS to perform file resource security checking, and optionally specifies the RACF resource class name in which you have defined the file resource security profiles.

XFCT={YES|name|NO}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to access File Control-managed files. Such checking is performed every time a transaction tries to access a file managed by CICS file control. The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the resource definitions.

Note: You can specify the **XFCT** parameter in the SIT, PARM, or SYSIN only.

YES CICS calls RACF, using the default CICS resource class name of CICSFCT prefixed by F or H, to verify that the userid associated with a transaction is authorized to access files reference by the transaction. The resource class name is FCICSFCT and the grouping class name is HCICSFCT.

name CICS calls RACF, using the specified resource class name, to verify that the userid associated with a transaction is authorized to access files referenced by the transaction. The resource class name is *Fname* and the grouping class name is *Hname*.

The resource class name specified must be 1 through 7 characters.

- NO** CICS does not perform any file resource security checks, allowing any user to access any file.

XHFS

The **XHFS** system initialization parameter specifies whether CICS is to check the transaction user's ability to access files in the z/OS UNIX System Services file system.

XHFS={YES|**NO**}

At present, this checking applies only to the user ID of the Web client when CICS Web support is returning z/OS UNIX file data as the static content identified by a URIMAP definition. The checking is performed only if you have specified YES for the SEC system initialization parameter. However, the RESSEC option on the transaction resource definition does **not** affect this security checking.

Note: You can specify the **XHFS** parameter in the SIT, PARM, or SYSIN only.

YES CICS is to check whether the user identified as the Web client is authorized to access the file identified by the URIMAP that matches the incoming URL. This check is in addition to the check performed by z/OS UNIX System Services against the CICS region user ID. If access to the file is denied for either of these user IDs, the HTTP request is rejected with a 403 (Forbidden) response.

NO CICS is not to check the client user's access to z/OS UNIX files. Note that the CICS region user ID's access to these files is still checked by z/OS UNIX System Services.

XJCT

The **XJCT** system initialization parameter specifies whether you want CICS to perform journal resource security checking.

XJCT={YES|**name**|**NO**}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to access the referenced journal. Such checking is performed every time a transaction tries to access a CICS journal. The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC is active for the resource definitions.

Note: You can specify the **XJCT** parameter in the SIT, PARM, or SYSIN only.

YES CICS calls RACF using the default CICS resource class name of CICSJCT prefixed by a J or K, to check whether the userid associated with a transaction is authorized to access CICS journals referenced by the transaction. The resource class name is JCICSJCT and the grouping class name is KCICSJCT.

name CICS calls RACF, using the specified resource class name prefixed by J or K, to verify that the userid associated with a transaction is authorized to access CICS journals.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any journal resource security checks, allowing any user to access any CICS journal.

XLT

The **XLT** system initialization parameter specifies a suffix for the transaction list table.

XLT={NO|xx|YES}

The table contains a list of transactions that can be attached during the first quiesce stage of system termination. See “Defining CICS resource table and module keywords” on page 138.

YES The default transaction list table, DFHXLT, is used.

xx The transaction list table DFHXLTxx is used.

NO A transaction list table is not used.

For guidance information about coding the macros for this table, see the *CICS Resource Definition Guide*

XPCT

The **XPCT** system initialization parameter specifies whether you want CICS to perform started transaction resource security checking, and optionally specifies the name of the RACF resource class name in which you have defined the started task security profiles.

XPCT={YES|name|NO}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to use started transactions and related EXEC CICS commands. Such checking is performed every time a transaction tries to use a started transaction or one of the EXEC CICS commands: COLLECT STATISTICS TRANSACTION, DISCARD TRANSACTION, INQUIRE TRANSACTION, or SET TRANSACTION. The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the resource definitions.

Note: You can specify the **XPCT** parameter in the SIT, PARM, or SYSIN only.

YES CICS calls RACF using the default CICS resource class name CICSPCT prefixed with A or B, to verify that the userid associated with a transaction is authorized to use started transactions or related EXEC CICS commands.

The resource class name is ACICSPCT and the grouping class name is BCICSPCT.

name CICS calls RACF, using the specified resource class name, to verify that the userid associated with a transaction is authorized to use started transactions or related EXEC CICS commands. The resource class name is ACICSPCT and the grouping class name is BCICSPCT.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any started task resource security checks, allowing any user to use started transactions or related EXEC CICS commands.

XPPT

The **XPPT** system initialization parameter specifies that CICS is to perform application program resource security checks and optionally specifies the RACF resource class name in which you have defined the program resource security profiles.

XPPT={YES|name|NO}

You can specify the **XPPT** parameter in the SIT, PARM, or SYSIN only. Checking is performed every time a transaction tries to invoke another program by using one of the CICS commands: LINK, LOAD, or XCTL.

Note: The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the resource definitions.

YES CICS calls RACF, using the default resource class name prefixed by M or N, to verify that the userid associated with a transaction is authorized to use LINK, LOAD, or XCTL commands to invoke other programs. The resource class name is MCICSPPT and the grouping class name is NCICSPPT.

name CICS calls RACF, with the specified resource class name prefixed by M or N, to verify that the userid associated with a transaction is authorized to use LINK, LOAD, or XCTL commands to invoke other programs. The resource class name is *Mname* and the grouping class name is *Nname*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any application program authority checks, allowing any user to use LINK, LOAD, or XCTL commands to invoke other programs.

XPSB

The **XPSB** system initialization parameter specifies whether you want CICS to perform program specification block (PSB) security checking and optionally specifies the RACF resource class name in which you have defined the PSB security profiles.

XPSB={YES|name|NO}

You can specify the **XPSB** parameter in the SIT, PARM, or SYSIN only. If you specify YES, or a RACF resource class name, CICS calls RACF to check that the userid associated with a transaction is authorized to access PSBs (which describe databases and logical message destinations used by application programs). Such checking is performed every time a transaction tries to access a PSB.

Note:

1. The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the resource definitions.
2. If you require security checking for PSBs to apply to remote users who access this region by means of transaction routing, the system initialization parameter **PSBCHK**=YES must be specified. For more information about this parameter, see "PSBCHK" on page 217.

YES CICS calls RACF, using the default resource class name CICSPSB prefixed by P or Q, to verify that the userid associated with a

transaction is authorized to access PSBs. The resource class name is PCICSPSB and the grouping class name is QCICSPSB.

name CICS calls RACF, using the specified resource class name prefixed by P or Q, to verify that the userid associated with a transaction is authorized to access PSBs. The resource class name is *Pname* and the grouping class name is *Qname*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any PSB resource security checks, allowing any user to access any PSB.

XRES

The **XRES** system initialization parameter specifies whether you want CICS to perform resource security checking for particular CICS resources and optionally specifies the general resource class name in which you have defined the resource security profiles.

XRES={YES | name | NO}

You can specify the **XRES** parameter in the SIT, PARM, or SYSIN only. If you specify YES, or a general resource class name, CICS calls the external security manager to verify that the user ID associated with a transaction is authorized to use the resource. This checking is performed every time a transaction tries to access a resource.

The actual profile name passed to the external security manager is the name of the resource to be checked, prefixed by its resource type; for example, for a document template whose resource definition is named "WELCOME", the profile name passed to the external security manager is DOCTEMPLATE.WELCOME. Even if a command references the document template using its 48-character template name, the shorter name (up to 8 characters) of the DOCTEMPLATE resource definition is always used for security checking.

The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the TRANSACTION resource definition.

YES CICS calls the external security manager, using the default CICS resource class name of RCICSRES, to check whether the user ID associated with a transaction is authorized to use the resource it is trying to access. The resource class name is RCICSRES and the grouping class name is WCICSRES.

name CICS calls the external security manager, using the specified resource class name prefixed by the letter R, to check whether the userid associated with a transaction is authorized to use the resource it is trying to access. The resource class name is *Rname* and the grouping class name is *Wname*. The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any security checks for resources, allowing access to any user.

XRF

The **XRF** system initialization parameter specifies whether XRF support is to be included in the CICS region.

XRF={NO|YES} (active and alternate)

If the CICS region is started with the START=STANDBY system initialization parameter specified, the CICS region is the alternate CICS region. If the CICS region is started with the START=AUTO, START=INITIAL or START=COLD system initialization parameter specified, the CICS region is the active CICS region. The active CICS region signs on as such to the CICS availability manager.

XRFSOFF

The XRFSOFF system initialization parameter is obsolete and is only kept for compatibility.

XRFSOFF={NOFORCE|FORCE}

If it is specified, it is rejected with a message and RSTSIGNOFF is assumed.

XRFSTME

The XRFSTME system initialization parameter is obsolete and is only kept for compatibility.

XRFSTME={5|decimal-value}

If it is specified, it is rejected with a message and RSTSIGNTIME is assumed.

XTRAN

The XTRAN system initialization parameter specifies whether you want CICS to perform transaction-attach security checking and optionally specifies the RACF resource class name in which you have defined the transaction security profiles.

XTRAN={YES|name|NO}

You can specify the XTRAN parameter in the SIT, PARM, or SYSIN only. If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with the transaction is permitted to run the transaction.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter.

YES CICS calls RACF, using the default CICS resource class name of CICSTRN prefixed by T or G, to verify that the userid associated with the transaction is authorized to run the transaction. The resource class name is TCICSTRN and the grouping class name is GCICSTRN.

name CICS calls RACF, using the specified resource class name prefixed by T or G, to verify that the userid associated with the transaction is authorized to run the transaction. The resource class name is *Tname* and the corresponding grouping class name is *Gname*.

The name specified must be 1 through 7 characters.

NO CICS does not perform any transaction-attach security checks, allowing any user to run any transaction.

XTST

The XTST system initialization parameter specifies whether you want CICS to perform temporary storage security checking and optionally specifies the RACF resource class name in which you have defined the temporary storage security profiles.

XTST={YES|name|NO}

You can specify the XTST parameter in the SIT, PARM, or SYSIN only. If you

specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a temporary storage request is authorized to access the referenced temporary storage queue.

Note: The checking is performed only if you have specified YES for the **SEC** system initialization parameter, specified the RESSEC option on the resource definitions, and specified TYPE=SECURITY in the temporary storage table (TST).

YES CICS calls RACF, using the default CICS resource class name of CICSTST prefixed by S or U, to verify that the userid associated with the transaction is authorized to access temporary storage queues referenced by the transaction. The resource class name is SCICSTST and the corresponding grouping class name is UCICSTST.

name CICS calls RACF, using the specified resource class name prefixed by S or U, to verify that the userid associated with a transaction is authorized to access temporary storage queues.

The name specified must be 1 through 7 characters.

NO CICS does not perform any temporary storage security checks, allowing any user to access any temporary storage queue.

XUSER

The **XUSER** system initialization parameter specifies whether CICS is to perform surrogate user checks.

XUSER={YES|NO}

You can specify the **XUSER** parameter in the SIT, PARM, or SYSIN only. Valid values are as follows:

YES CICS is to perform surrogate user checking in all those situations that permit such checks to be made; for example, on EXEC CICS START commands without an associated terminal. Surrogate user security checking is also performed by CICS against user IDs installing or modifying DB2 resource definitions that specify AUTHID or COMAUTHID.

Note: The **XUSER** parameter is also used by CICS to control access to the AUTHTYPE and COMAUTHTYPE attributes on DB2 resource definitions, although not through surrogate user checks. For more information about AUTHTYPE and COMAUTHTYPE attributes, see the *CICS Resource Definition Guide*.

For information about the various circumstances in which CICS performs surrogate user checks, see the *CICS RACF Security Guide*.

NO CICS is not to perform any surrogate user checking.

System initialization parameters that can be defined only with the DFHSIT macro

Some system initialization parameters can be defined in just one way, in a DFHSIT macro.

ESMEXITS

The **ESMEXITS** system initialization parameter specifies whether installation data is to be passed through the RACROUTE interface to the external security manager (ESM) for use in exits written for the ESM.

ESMEXITS={NOINSTLN|INSTLN}

Valid values are as follows:

NOINSTLN

The INSTLN parameter is not used in RACROUTE macros.

INSTLN

CICS-related and installation-supplied data is passed to the ESM using the INSTLN parameter of the RACROUTE macro. For programming information, including the format of the data passed, see the *CICS Customization Guide*. This data is intended for use in exits written for the ESM.

Restrictions You can specify the ESMEXITS parameter in the SIT only.

HPO

The **HPO** system initialization parameter specifies whether you want to use the VTAM authorized path feature of the high performance option (HPO).

HPO={NO|YES}

If you code YES, the CICS type 6 SVC must be link-edited in your MVS nucleus, and defined to MVS in an SVC Parm statement. If the SVC number is not 215 (the default) you must specify the SVC number on the **SRBSVC** parameter.

For information about installing the CICS type 6 SVC in your MVS system, and about changing the default number, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the HPO parameter in the SIT only.

STARTER

The **STARTER** system initialization parameter specifies whether the generation of starter system modules (with \$ and # suffixes) is permitted, and various MNOTES are to be suppressed.

STARTER={NO|YES}

This parameter should only be used when service is being performed on starter system modules.

Restrictions You can specify the **STARTER** parameter in the SIT only.

SUFFIX

The **SUFFIX** system initialization parameter specifies the last two characters of the name of this system initialization table.

SUFFIX=xx

The first 6 characters of the name of the SIT are fixed as DFHSIT. You can specify the last two characters of the name, using the SUFFIX parameter. Because the SIT does not have a TYPE=INITIAL macro statement like other CICS resource control tables, you specify its SUFFIX on the TYPE=CSECT macro statement.

The suffix allows you to have more than one version of the SIT. Any one or two characters (other than NO and DY) are valid. You select the version of the table to be loaded into the system during system initialization by coding SIT=xx, either in the PARM parameter or the SYSIN data set. (You can, in some circumstances, specify the SIT using the system console, but this is not recommended.)

Restrictions You can specify the **SUFFIX** parameter in the SIT only.

System initialization parameters that cannot be defined with the DFHSIT macro

Some system initialization parameters cannot be defined in a DFHSIT macro.

Specify these parameters in one of the following ways:

- In the PARM parameter of the EXEC PGM=DFHSIP statement
- In the SYSIN data set defined in the startup job stream
- Through the system operator's console

CDSASZE

The **CDSASZE** system initialization parameter specifies the size of the CDSA.

CDSASZE={0K|number}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify number as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256KB). If the size specified is not a multiple of 256KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions You can specify the **CDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

CHKSTRM

The **CHKSTRM** parameter specifies that terminal storage-violation checking is to be activated or deactivated.

CHKSTRM={CURRENT|NONE}

Valid values are as follows:

CURRENT

TIOA storage violations are to be checked.

NONE

TIOA storage-violation checking is to be deactivated.

You can also use the CICS-supplied transaction, CSFE, to switch terminal storage-violation checking on and off.

For information about checking for storage violations, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the **CHKSTRM** parameter in PARM, SYSIN, or CONSOLE only.

CHKSTSK

The **CHKSTSK** parameter specifies that task storage-violation checking at startup is to be activated or deactivated.

CHKSTSK={CURRENT|NONE}

Valid values are as follows:

CURRENT

All storage areas on the transaction storage chain for the current task only are to be checked.

NONE

Task storage-violation checking is to be deactivated.

You can also use the CICS-supplied transaction, CSFE, to switch task storage-violation checking on and off.

For information about checking for storage violations, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the **CHKSTSK** parameter in PARM, SYSIN, or CONSOLE only.

ECDSASZE

The **ECDSASZE** system initialization parameter specifies the size of the ECDSA.

ECDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify **number** as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1 MB). If the size specified is not a multiple of 1 MB, CICS rounds the value up to the next multiple.

You can specify **number** in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Note: For the DS domain function CHANGE_MODE, a trace entry will be generated if DS level 2, 3, or ALL tracing is active.

Restrictions You can specify the **ECDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

ERDSASZE

The **ERDSASZE** system initialization parameter specifies the size of the ERDSA.

ERDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify **number** as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1 MB). If the size specified is not a multiple of 1 MB, CICS rounds the value up to the next multiple.

You can specify **number** in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the **ERDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

ESDSASZE

The **ESDSASZE** system initialization parameter specifies the size of the ESDSA.

ESDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1 MB). If the size specified is not a multiple of 1 MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the **ESDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

EUDSASZE

The **EUDSASZE** system initialization parameter specifies the size of the EUDSA.

EUDSASZE={0K|number}

The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1MB). If the size specified is not a multiple of 1MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions You can specify the **EUDSASZE** parameter in PARM, SYSIN, or CONSOLE only.

JVMLEVEL0TRACE

The **JVMLEVEL0TRACE** system initialization parameter specifies the default option for JVM Level 0 trace, corresponding to trace level 29 of the SJ component.

JVMLEVEL0TRACE={'ALL (EXCEPTION)'|'*user override string*'}

The default setting for this level of tracing maps to trace point level 0 for JVMs, which is reserved for extraordinary events and errors. Unlike CICS exception trace, which cannot be switched off, the JVM Level 0 trace is normally switched off unless JVM tracing is required. "JVMxxxxTRACE" on page 195 has more information about these system initialization parameters.

JVMLEVEL1TRACE

The **JVMLEVEL1TRACE** system initialization parameter specifies the default option for JVM Level 1 trace, corresponding to trace level 30 of the SJ component.

JVMLEVEL1TRACE={'ALL(ENTRY,EXIT)' | 'user override string'}

The default setting for this level of tracing maps to trace point level 1 for JVMs. "JVMxxxxTRACE" on page 195 has more information about these system initialization parameters.

JVMLEVEL2TRACE

The **JVMLEVEL2TRACE** system initialization parameter specifies the default option for JVM Level 2 trace, corresponding to trace level 31 of the SJ component.

JVMLEVEL2TRACE={'ALL' | 'user override string'}

The default setting for this level of tracing maps to trace point level 2 for JVMs. Note that the JVM trace point levels go up to level 9. "JVMxxxxTRACE" on page 195 has more information about these system initialization parameters.

JVMUSERTRACE

The **JVMUSERTRACE** system initialization parameter specifies the default option for JVM user trace, corresponding to trace level 32 of the SJ component.

JVMUSERTRACE={'NONE' | 'user override string'}

Use this option for more complex specifications for JVM tracing. "JVMxxxxTRACE" on page 195 has more information about these system initialization parameters.

NEWSIT

The **NEWSIT** system initialization parameter specifies whether CICS is to load the specified SIT, and enforce the use of all system initialization parameters, modified by any system initialization parameters provided by PARM, SYSIN, or the system console, even in a warm start.

NEWSIT={YES | NO}

Enforcing the use of system initialization parameters in this way overrides any parameters that may have been stored in a warm keypoint at shutdown.

However, there are some exceptions. The following system initialization parameters are always ignored in a warm start, even if they are supplied by PARM, SYSIN, or the console:

- CSDACC
- CSDBUFND
- CSDBUFNI
- CSDDISP
- CSDDSN
- CSDFRLOG
- CSDINTEG
- CSDJID
- CSDLRNO
- CSDRECOV
- CSDRLS
- CSDSTRNO
- FCT
- GRPLIST

In a warm restart, CICS uses the **installed** resource definitions saved in the CICS global catalog at warm shutdown, and therefore the CSD, FCT, and

GRPLIST parameters are ignored. (At CICS startup, you can only modify installed resource definitions, including file control table entries, or change to a new FCT, by performing a cold start of CICS with START=COLD.)

For more information about the use of the NEWSIT parameter, see “Controlling start and restart” on page 308.

Restrictions

You can specify the NEWSIT parameter in PARM, SYSIN, or CONSOLE only.

OFFSITE

The **OFFSITE** system initialization parameter specifies whether CICS is to restart in off-site recovery mode; that is, a restart is taking place at a remote site.

OFFSITE={NO|YES}

For a successful off-site restart, the log records of the failed CICS region must be available at the remote site. CICS does not provide a facility for shipping log records to a remote backup site, but you can use a suitable vendor product to perform this function. See the relevant product documentation for other procedures you need to follow for a remote site restart.

See the *CICS Recovery and Restart Guide* for more information about remote site recovery.

NO CICS will not perform the special restart processing required for remote site recovery.

YES CICS will perform an off-site restart at a remote site following a disaster at the primary site. CICS performs this special processing for an off-site restart, because some information (for example, a VSAM lock structure) is not available at the remote site.

CICS performs an emergency restart, even if the global catalog indicates that CICS can do a warm start. OFFSITE=YES is valid with START=AUTO only, and CICS initialization is terminated if you specify START=COLD or INITIAL.

Restrictions

You can specify the OFFSITE parameter in PARM, SYSIN, or CONSOLE only.

PRVMOD

The **PRVMOD** system initialization parameter specifies the names of those modules that are not to be used from the LPA.

PRVMOD={name | (name, name . . . name)}

The operand is a list of 1-8 character module names. This enables you to use a private version of a CICS nucleus module in the CICS address space, and not a version that might be in the LPA. For information about **PRVMOD**, see the *CICS Transaction Server for z/OS Installation Guide*.

Restrictions You can specify the **PRVMOD** parameter in PARM, SYSIN, or CONSOLE only.

RDSASZE

The **RDSASZE** system initialization parameter specifies the size of the RDSA.

RDSASZE={0K|number}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify number as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256 KB). If the size specified is not a multiple of 256 KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the RDSASZE parameter in PARM, SYSIN, or CONSOLE only.

SDSASZE

The **SDSASZE** system initialization parameter specifies the size of the SDSA.

SDSASZE={0K|number}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify number as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256 KB). If the size specified is not a multiple of 256 KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096 KB), or a whole number of megabytes (for example, 4 MB).

Restrictions You can specify the **SDSAZSE** parameter in PARM, SYSIN, or CONSOLE only.

SIT

The **SIT** system initialization parameter specifies the suffix, if any, of the system initialization table that you want CICS to load at the start of initialization.

SIT=xx

If you omit this parameter, CICS loads the unsuffixed table, DFHSIT, which is pregenerated with all the default values. This default SIT (shown in “The default system initialization table” on page 293) is in CICSTS41.CICS.SDFHAUTH, and its source, named DFHSIT\$\$, is in CICSTS41.CICS.SDFHSAMP.

Restrictions You can specify the system initialization parameter anywhere in PARM or SYSIN, or as the **first** parameter entry at the CONSOLE.

SPCTRxx

The **SPCTRxx** system initialization parameter specifies the level of special tracing for a particular CICS component used by a transaction, terminal, or both.

SPCTRxx={ (1,2) | (1 [,2] [,3] [,4] [,29] [,30] [,31] [,32]) | ALL | OFF }

You identify the component by coding a value for *xx* in the keyword. You code one SPCTRxx keyword for each component you want to define selectively. For a CICS component being specially traced that does not have its trace level set by SPCTRxx, the trace level is that set by **SPCTR** (which, in turn, defaults to (1,2)). The CICS component codes that you can specify for *xx* on the SPCTRxx keyword are shown in the following table:

| Code | Component name |
|-------------|------------------------------|
| AP | Application domain |
| BA | Business application manager |
| BF* | Built-in function |
| BM* | Basic mapping support |
| BR* | 3270 bridge |
| CP* | Common programming interface |
| DC* | Dump compatibility layer |
| DD | Directory manager domain |
| DH | Document handling domain |
| DI* | Data interchange |
| DM | Domain manager domain |
| DP | Debugging profiles domain |
| DS | Dispatcher domain |
| DU | Dump domain |
| EC* | Event capture and emission |
| EI* | Exec interface |
| EJ | Enterprise Java domain |
| EM | Event manager domain |
| EP | Event processing domain |
| FC* | File control |
| GC | Global catalog domain |
| IC* | Interval control |
| IE | ECI over TCP/IP domain |
| II | IIOP domain |
| IS* | ISC or IRC |
| KC* | Task control |
| KE | Kernel |
| LC | Local catalog domain |
| LD | Loader domain |
| LG | Log manager domain |
| LM | Lock domain |
| ME | Message domain |
| ML | Markup language domain |
| MN | Monitoring domain |
| NQ | Enqueue domain |
| OT | Object transaction domain |
| PA | Parameter domain |
| PC* | Program control |
| PG | Program manager domain |
| PI | Pipeline domain |
| PT | Partner domain |

| Code | Component name |
|--|----------------------------------|
| RA | Resource manager adapters |
| RI* | Resource manager interface (RMI) |
| RL | Resource life-cycle domain |
| RM | Recovery manager domain |
| RS | Region status domain |
| RX | RRS-coordinated EXCI domain |
| RZ | Request streams domain |
| SC* | Storage control |
| SH | Scheduler services domain |
| SJ | JVM domain |
| SM | Storage manager domain |
| SO | Sockets domain |
| ST | Statistics domain |
| SZ* | Front End Programming Interface |
| TC* | Terminal control |
| TD* | Transient data |
| TI | Timer domain |
| TR | Trace domain |
| TS | Temporary storage domain |
| UE* | User exit interface |
| US | User domain |
| WB | Web domain |
| WU | System management RESTful API |
| W2 | Web 2.0 domain |
| XM | Transaction manager domain |
| XS | Security manager domain |
| Note: | |
| 1. Components marked * are subcomponents of the AP domain. The trace entries for these components are produced with a trace point ID of AP <i>nnnn</i> . | |
| 2. For the DS domain function CHANGE_MODE, a trace entry is generated if DS level 2 or 3 tracing is active. | |

number

The level numbers for the level of special tracing you want for the required CICS component. You can use level numbers 1, 2, 3, 4, 29, 30, 31 and 32, depending on the component.

Most CICS components only use levels 1, 2 and 3, and some do not have trace points at all these levels. The exceptions are:

- The SM component (storage manager domain) has level 4 tracing. This level of tracing is intended for IBM field engineering staff.
- The SJ component (JVM domain) has trace levels 29–32, that are reserved to indicate the JVM trace levels 0, 1, and 2, plus a user-definable JVM trace level. You can use the system initialization parameters **JVMLEVEL0TRACE**, **JVMLEVEL1TRACE**, **JVMLEVEL2TRACE** and

JVMUSERTRACE to specify options for these JVM trace levels, and then activate them using the **SPCTRSJ** system initialization parameter.

When you activate JVM trace, using trace levels 29–32 for the SJ component, the JVM trace appears as CICS trace point SJ 4D02 (when formatted), or SJ 4D01 (if unformatted).

- ALL** You want all the available levels of special CICS tracing switched on for the specified component.
- OFF** Switches off all levels of special CICS tracing for the CICS component indicated by xx.

For details of using trace, see the *CICS Problem Determination Guide*.

Restrictions You can specify the **SPCTRxx** parameter in PARM, SYSIN, or CONSOLE only.

STNTRxx

The **STNTRxx** system initialization parameter specifies the level of standard tracing you require for a particular CICS component.

STNTRxx={1| (1[,2] [,3] [,4] [,29] [,30] [,31] [,32]) |ALL|OFF}

You identify the component by coding a value for xx in the keyword. You code one STNTRxx keyword for each component you want to define selectively. For a CICS component being specially traced that does not have its trace level set by STNTRxx, the trace level is that set by STNTR (which, in turn, defaults to 1). You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels.

The CICS component codes that you can specify for xx on this STNTRxx keyword are shown in the following table:

| Code | Component name |
|------|------------------------------|
| AP | Application domain |
| BA | Business application manager |
| BF* | Built-in function |
| BM* | Basic mapping support |
| BR* | 3270 bridge |
| CP* | Common programming interface |
| DC* | Dump compatibility layer |
| DD | Directory manager domain |
| DH | Document handling domain |
| DI* | Data interchange |
| DM | Domain manager domain |
| DP | Debugging profiles domain |
| DS | Dispatcher domain |
| DU | Dump domain |
| EC* | Event capture and emission |
| EI* | Exec interface |
| EJ | Enterprise Java domain |
| EM | Event manager domain |
| EP | Event processing domain |

| Code | Component name |
|-------------|----------------------------------|
| FC* | File control |
| GC | Global catalog domain |
| IC* | Interval control |
| IE | ECI over TCP/IP domain |
| II | IIOP domain |
| IS* | ISC or IRC |
| KC* | Task control |
| KE | Kernel |
| LC | Local catalog domain |
| LD | Loader domain |
| LG | Log manager domain |
| LM | Lock domain |
| ME | Message domain |
| ML | Markup language domain |
| MN | Monitoring domain |
| NQ | Enqueue domain |
| OT | Object transaction domain |
| PA | Parameter domain |
| PC* | Program control |
| PG | Program manager domain |
| PI | Pipeline domain |
| PT | Partner domain |
| RA | Resource manager adapters |
| RI* | Resource manager interface (RMI) |
| RL | Resource life-cycle domain |
| RM | Recovery manager domain |
| RS | Region status domain |
| RX | RRS-coordinated EXCI domain |
| RZ | Request streams domain |
| SC* | Storage control |
| SH | Scheduler services domain |
| SJ | JVM domain |
| SM | Storage manager domain |
| SO | Sockets domain |
| ST | Statistics domain |
| SZ* | Front End Programming Interface |
| TC* | Terminal control |
| TD* | Transient data |
| TI | Timer domain |
| TR | Trace domain |
| TS | Temporary storage domain |

| Code | Component name |
|--|-------------------------------|
| UE* | User exit interface |
| US | User domain |
| WB | Web domain |
| WU | System management RESTful API |
| W2 | Web 2.0 domain |
| XM | Transaction manager domain |
| XS | Security manager domain |
| Note: | |
| <ol style="list-style-type: none"> Components marked * are subcomponents of the AP domain. The trace entries for these components are produced with a trace point ID of AP <i>nnnn</i>. For the DS domain function CHANGE_MODE, a trace entry is generated if DS level 2 or 3 tracing is active. | |

ALL You want all the available levels of standard tracing switched on for the specified component.

Warning! Selecting ALL for standard tracing for the storage manager (SM) component, or the temporary storage domain (TS), degrades the performance of your CICS region. This is because ALL switches on trace flags that are used by SM domain for field engineering purposes.

Warning! Selecting ALL for standard tracing for the JVM domain (SJ) component is not recommended. JVM trace can produce a large amount of output, so you should normally activate JVM trace for special transactions (using the SPCTRSJ system initialization parameter), rather than turning it on globally for all transactions.

number

The level number(s) for the level of standard tracing you want for the CICS component indicated by xx. Level numbers 1, 2, 3, 4, 29, 30, 31 and 32 can be used, depending on the component.

Most CICS components only use levels 1, 2 and 3, and some do not have trace points at all these levels. The exceptions are:

- The SM component (storage manager domain), which also has level 4 tracing. This level of tracing is intended for IBM field engineering staff.

Warning! Selecting tracing levels 3, 4, or ALL for standard tracing for the storage manager (SM) component, or the temporary storage domain (TS), degrades the performance of your CICS region. This is because options 3 and 4 (and ALL) switch on trace flags that are used by SM domain for field engineering purposes.

SM trace flag 3 deactivates the quickcell mechanism, and SM trace flag 4 forces subpool element chaining on every CICS subpool. Furthermore, once these settings have been activated during system initialization, they cannot be unset, either through a PLTPI program or by using the CETR trace transaction, because they are not used for tracing as such. Thus, a significant performance overhead is incurred if these storage manager trace levels are selected for standard tracing.

See the *CICS Problem Determination Guide* for information about the effects of trace levels 3 and 4.

- The SJ component (JVM domain), which also has trace levels 29–32, that are reserved to indicate the JVM trace levels 0, 1, and 2, plus a user-definable JVM trace level. You can use the system initialization parameters JVMLEVEL0TRACE, JVMLEVEL1TRACE, JVMLEVEL2TRACE and JVMUSERTRACE to specify options for these JVM trace levels, and then activate them using the SPCTRSJ system initialization parameter.

Warning! Selecting tracing levels 29, 30, 31, 32 or ALL for standard tracing for the JVM domain (SJ) component is not recommended. JVM trace can produce a large amount of output, so you should normally activate JVM trace for special transactions (using the SPCTRSJ system initialization parameter), rather than turning it on globally for all transactions.

OFF Switches off all levels of standard CICS tracing for the CICS component indicated by xx.

Restrictions You can specify the STNTRxx parameter in PARM, SYSIN, or CONSOLE only.

UDSASZE

The **UDSASZE** system initialization parameter specifies the size of the UDSA.

UDSASZE={0K|number}

The default size is 0, indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

specify number as an amount of storage in the range 0 to 16777215 bytes in multiples of 262144 bytes (256KB). If the size specified is not a multiple of 256KB (or 1MB if transaction isolation is active), CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions You can specify the UDSASZE parameter in PARM, SYSIN, or CONSOLE only.

System initialization parameters that cannot be defined through the system operator's console

Some system initialization parameters cannot be defined through the system operator's console.

Specify these parameters in one of the following ways:

- With the DFHSIT macro
- In the PARM parameter of the EXEC PGM=DFHSIP statement
- In the SYSIN data set defined in the startup job stream

CMDSEC

The **CMDSEC** parameter specifies whether or not you want CICS to honor the CMDSEC option specified on a transaction's resource definition.

CMDSEC={ASIS|ALWAYS}

Valid values are as follows:

ASIS means that CICS honors the CMDSEC option defined in a transaction's resource definition. CICS calls its command security checking routine only when CMDSEC(YES) is specified in a transaction resource definition.

ALWAYS

CICS overrides the CMDSEC option, and always calls its command security checking routine to issue the appropriate call to the SAF interface.

Note:

1. Specify ALWAYS when you want to control the use of the SPI in all your transactions. Be aware that this might incur additional overhead. The additional overhead is caused by CICS issuing the command security calls on every eligible EXEC CICS command, which are *all* the system programming interface (SPI) commands.
2. If you specify ALWAYS, command checking applies to CICS-supplied transactions such as CESN and CESF. You must authorize all users of CICS-supplied transactions to use the internal CICS resources for the transactions, otherwise you will get unexpected results in CICS-supplied transactions.

Restrictions You can specify the CMDSEC parameter in the SIT, PARM, or SYSIN only.

CONFDATA

The **CONFDATA** parameter specifies whether CICS is to suppress user data that might otherwise appear in CICS trace entries or in dumps.

CONFDATA={SHOW|HIDETC}

This option applies to initial input data received on:

- A VTAM RECEIVE ANY operation
- An MRO connection
- An IPIC connection
- FEPI screens and RPLAREAs

This option also applies to the CICS client use of a Virtual Terminal. Data is traced before and after codepage conversion and is suppressed if HIDETC is used in combination with CONFDATA YES in the transaction.

SHOW

Data suppression is not in effect. User data is traced regardless of the CONFDATA option specified in transaction resource definitions. This option overrides the CONFDATA option in transaction resource definitions.

HIDETC

CICS is to 'hide' user transport data from CICS trace entries. The action taken by CICS is subject to the individual CONFDATA attribute on the transaction resource definition (see Table 15 on page 161).

If you specify CONFDATA=HIDETC, CICS processes VTAM, MRO, IPIC, and FEPI user data as follows:

- **VTAM:** CICS clears the VTAM RAIA containing initial input as soon as it has been processed, and before the target transaction has been identified.

The normal trace entries (FC90 and FC91) are created on completion of the RECEIVE ANY operation with the text "SUPPRESSED DUE TO

CONFDATA=HIDETC IN SIT" replacing all the user data except the first 4 bytes of normal data, or the first 8 bytes of function management headers (FMHs).

CICS then identifies the target transaction for the data. If the transaction definition specifies CONFDATA(NO), CICS traces the user data that it suppressed from the FC90 trace in the trace entry AP FC92. This trace entry is not created if the transaction is defined with CONFDATA(YES).

- **MRO:** CICS does not trace the initial input received on an MRO link.

The normal trace entries (DD16, DD23, and DD25) are created with the text SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT replacing all the user data.

CICS then identifies the target transaction for the data. If the transaction definition specifies CONFDATA(NO), CICS traces the user data that it suppressed from DD16 in the trace entry AP FC92. This special trace entry is not created if the transaction is defined with CONFDATA(YES).

- **IPIC:** Trace points SO 0201 and SO 0202 suppress buffer data with the message "Trace data suppressed because it may contain sensitive data". Subsequent trace point SO 029D (buffer continuation) and buffer data from trace points WB 0700 and WB 0701 is suppressed.

If the transaction definition specifies CONFDATA(NO), IS trace entries are created with the user data, as normal.

If the transaction definition specifies CONFDATA(YES), user data from IS trace points IS 0602, IS 0702, and IS 0906 is replaced with "SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT". Data from IS trace points IS 0603 and IS 0703 is not shown.

- **FEPI:** FEPI screens and RPL data areas (RPLAREAs) areas are suppressed from all FEPI trace points if CONFDATA(YES) is specified in the transaction resource definition. The user data in the FEPI trace points AP 1243, AP 1244, AP 145E, AP 145F, AP 1460, AP 1461, AP 1595, AP 1596, AP 1597, AP 1598, and AP 1599 is replaced with the message SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT. If the transaction definition specifies CONFDATA(NO), the FEPI trace entries are created with the user data as normal.

Mirror transactions: The CICS-supplied mirror transaction definitions are specified with CONFDATA(YES). This ensures that, when you specify CONFDATA=HIDETC as a system initialization parameter, CICS regions running mirror transactions suppress user data as described for VTAM, MRO, and IS data.

Modified data: By waiting until the transaction has been identified to determine the CONFDATA option, VTAM, MRO, or IS data may have been modified (for example, it may have been translated to upper case).

The interaction between the **CONFDATA** system initialization parameter and the CONFDATA attribute on the transaction resource definition is shown in Table 15 on page 161.

Table 17. Effect of CONFDATA system initialization and transaction definition parameters

| CONFDATA on transaction | CONFDATA system initialization parameter | |
|-------------------------|--|-----------------|
| | SHOW | HIDETC |
| NO | Data not suppressed | Data suppressed |
| YES | Data not suppressed | Data suppressed |

You cannot modify the CONFDATA option while CICS is running. You must restart CICS to apply a change.

Restrictions: You can specify the **CONFDATA** parameter in the SIT, PARM, and SYSIN only.

CONFTXT

The **CONFTXT** system initialization parameter specifies whether CICS is to prevent VTAM from tracing user data.

CONFTXT={NO|YES}

Valid values are as follows:

NO CICS does not prevent VTAM from tracing user data.

YES CICS prevents VTAM from tracing user data.

Restrictions You can specify the **CONFTXT** parameter in the SIT, PARM, and SYSIN only.

DFLTUSER

The **PLTPISEC** system initialization parameter specifies the RACF userid of the default user; that is, the user whose security attributes are used to protect CICS resources in the absence of other, more specific, user identification.

DFLTUSER={CICSUSER|userid}

For example, except in the case of terminals defined with preset security, the security attributes of the default user are assigned to terminal users who do not sign on.

The specified userid must be defined to RACF if you are using external security (that is, you have specified the system initialization parameter SEC=YES).

The specified userid is signed on during CICS initialization. If it cannot be signed on, CICS fails to initialize.

Restrictions You can specify the DFLTUSER parameter in the SIT, PARM, or SYSIN only.

PLTPISEC

The **PLTPISEC** system initialization parameter specifies whether or not you want CICS to perform command security or resource security checking for PLT programs during CICS initialization.

PLTPISEC={NONE|CMDSEC|RESSEC|ALL}

The PLT programs run under the authority of the userid specified on PLTPIUSR, which must be authorized to the appropriate resources defined by PLTPISEC.

NONE

You do not want any security checking on PLT initialization programs.

CMDSEC

You want CICS to perform command security checking only.

RESSEC

You want CICS to perform resource security checking only.

ALL You want CICS to perform both command and resource security checking.

Restrictions You can specify the PLTPISEC parameter in the SIT, PARM, or SYSIN only.

PLTPIUSR

The **PLTPIUSR** system initialization parameter specifies the userid that CICS is to use for security checking for PLT programs that run during CICS initialization.

PLTPIUSR=userid

All PLT programs run under the authority of the specified userid, which must be authorized to all the resources referenced by the programs, as defined by the PLTPISEC parameter.

PLT programs are run under the CICS internal transaction, CPLT. Before the CPLT transaction is attached, CICS performs a surrogate user check against the CICS region userid (the userid under which the CICS region is executing). This is to ensure that the CICS region is authorized as a surrogate for the userid specified on the PLTPIUSR parameter. This ensures that you cannot arbitrarily specify any PLT userid in any CICS region—each PLT userid must first be authorized to the appropriate CICS region.

If you do not specify the PLTPIUSR parameter, CICS runs PLTPI programs under the authority of the CICS region userid, in which case CICS does not perform a surrogate user check. However, the CICS region userid must be authorized to all the resources referenced by the PLT programs.

Restrictions You can specify the PLTPIUSR parameter in the SIT, PARM, or SYSIN only.

PSBCHK

The **PSBCHK** system initialization parameter specifies whether CICS is to perform PSB authorization checks for remote terminal users who use transaction routing to initiate a transaction in this CICS region to access an attached IMS system.

PSBCHK={NO|YES}

Valid values are as follows:

NO The remote link is checked, but no check is made against the remote terminal. This is the default.

YES The remote link is checked, and the remote terminal is also checked if RESSEC(YES) is coded in the definition of the transaction in the CSD.

Restrictions You can specify the **PSBCHK** parameter in the SIT, PARM, or SYSIN only.

Note: If you require DL/I security checking, you must specify the XPSB system initialization parameter as XPSB=YES or XSPB=name. For further information about the XPSB system initialization parameter, see XPSB .

RESSEC

The **RESSEC** system initialization parameter specifies whether you want CICS to honor the RESSEC option specified on a transaction's resource definition.

RESSEC={ASIS|ALWAYS}

Valid values are as follows:

ASIS CICS honors the RESSEC option defined in a transaction's resource definition. CICS calls its resource security checking routine only when RESSEC(YES) is specified in a transaction resource definition. This is normally a sufficient level of control, because often you will need only to control the ability to execute a transaction.

ALWAYS

CICS overrides the RESSEC option, and always calls its resource security checking routine to issue the appropriate call to the SAF interface.

Use this option only if you need to control or audit all accesses to CICS resources. Using this option can significantly degrade performance.

Restrictions You can specify the RESSEC parameter in the SIT, PARM, or SYSIN only.

SEC

The **SEC** system initialization parameter specifies what level of external security you want CICS to use.

SEC={YES|NO}

Valid values are as follows:

YES You want to use full external security. CICS requires the appropriate level of authorization for the access intent: a minimum of READ permission for read intent, and a minimum of UPDATE permission for update intent.

Note: You must also ensure that the default userid (CICSUSER or another userid specified on the **DFLTUSER** system initialization parameter) has been defined to RACF.

If command security checking is defined for CICS SP-type commands, then specifying SEC=YES means that the appropriate level of authority is checked for; therefore:

- A check for READ authority is made for **INQUIRE** and **COLLECT** commands.
- A check for UPDATE authority is made for **SET**, **PERFORM**, and **DISCARD** commands.

NO You do not want CICS to use an external security manager. All users have access to all resources, whether determined by attempts to use them or by the **QUERY SECURITY** command. Users are not allowed to sign on or off.

Note: With MRO bind-time security, even if you specify SEC=NO, the CICS region userid is still sent to the secondary CICS region, and bind-time checking is still carried out in the secondary CICS region. For information about MRO bind-time security, see the *CICS RACF Security Guide*.

Define whether to use RACF for resource level checking by using the **XDCT**, **XFCT**, **XHFS**, **XJCT**, **XPCT**, **XPPT**, **XPSB**, **XRES**, and **XTST** system initialization parameters. Define whether to use RACF for transaction-attach security checking by using the **XTRAN** system initialization parameter. Define whether to use RACF for enterprise bean method authorization checks by using the "XEJB" on page 259 system initialization parameter. Define whether RACF session security can be used when establishing APPC sessions by using the **XAPPC** system initialization parameter.

For information on defining command security checking for CICS SP-type commands, and about CICS security in general, see the *CICS RACF Security Guide*.

For programming information about the use of external security for CICS system commands, see the *CICS System Programming Reference*.

Restrictions You can specify the **SEC** parameter in the SIT, PARM, or SYSIN only.

Note: If you are using preset terminal security and you perform a warm start with SEC=NO and then again with SEC=YES, you must reinstall the terminal definition to preserve the preset user ID that is replaced by the default user ID when security is switched off. See the *CICS RACF Security Guide* for details.

SECPRFX

The **SECPRFX** system initialization parameter specifies whether CICS is to prefix the resource names in any authorization requests to the external security manager.

SECPRFX={NO|YES|*prefix*}

Valid values are as follows:

NO CICS does not use prefixes on any resource names.

YES CICS prefixes all resource names with the CICS region user ID. This is the user ID under which the CICS job runs. It is one of the following:

- If CICS is a batch job, it is the user ID corresponding to the USER parameter of the CICS JOB statement.
- If CICS is a started task, it is the user ID associated with the name of the started procedure in the RACF ICHRIN03 table.
- If CICS is a started job, it is the user ID specified in the user parameter of the STDATA segment of a STARTED general resource class profile.

For more information, see the *CICS RACF Security Guide*.

prefix CICS prefixes all resource names with the string you specify. It can be any string of 1 to 8 upper case alphanumeric characters except NO or YES, and must start with an alphabetic character.

Restrictions You can specify the SECPRFX parameter in the SIT, PARM, or SYSIN only.

The SECPRFX parameter is effective only if you specify YES for the SEC system initialization parameter.

SNSCOPE

The **SNSCOPE** system initialization parameter specifies whether a userid can be signed on to CICS more than once, within the scope of a single CICS region, a single MVS image, and a sysplex.

SNSCOPE={NONE|CICS|MVSIMAGE|SYSPLEX}

The signon SCOPE is enforced with the MVS ENQ macro where there is a limit on the number of outstanding MVS ENQs per address space. If this limit is exceeded, the MVS ENQ is rejected and CICS is unable to detect if the user is already signed on. When this happens, the signon request is rejected with message DFHCE3587. You can use the ISGADMIN macro to set or reset the MVS ENQ limit. For more information, see the *z/OS MVS Authorized Assembler Services Reference*.

NONE

Each user ID can be used to sign on for any number of sessions on any CICS region. This is the compatibility option, providing the same signon scope as in releases of CICS before CICS Transaction Server for z/OS, Version 4 Release 1.

CICS Each user ID can be signed on once only in the same CICS region. A signon request is rejected if the userid is already signed on to the same CICS region. However, the user ID can be used to signon to another CICS region in the same, or another, MVS image.

MVSIMAGE

Each userid can be signed on once only, and to only one of the set of CICS regions in the same MVS image that also specify SNSCOPE=MVSIMAGE. A signon request is rejected if the user is already signed on to another CICS region in the same MVS image.

SYSPLEX

Each user ID can be signed on once only, and to only one of the set of CICS regions within an MVS sysplex that also specify SNSCOPE=SYSPLEX. A signon is rejected if the user is already signed on to another CICS region in the same MVS sysplex.

The signon scope (if specified) applies to all user IDs signing on by an explicit signon request (for example, by an EXEC CICS SIGNON command or the CESN transaction). SNSCOPE is restricted to users signing on at local terminals, or signing on after using the CRTE transaction to connect to another system.

Signon scope specified by SNSCOPE does not apply to:

- Non-terminal users.
- The CICS default userid, specified by the **DFLTUSER** system initialization parameter.
- Preset user IDs, specified in the USERID option of the **DEFINE TERMINAL** command.
- User IDs for remote users, received in attach headers.
- User IDs for link security. For information about which userid is used for link security on a specific connection, see the *CICS RACF Security Guide*.
- The user ID specified on the **PLTPIUSR** system initialization parameter.
- The CICS region user ID.

Restrictions You can specify the **SNSCOPE** parameter in the SIT, PARM, or SYSIN only.

XAPPC

The **XAPPC** system initialization parameter specifies whether RACF session security can be used when establishing APPC sessions.

XAPPC={NO|YES}

Valid values are as follows:

NO RACF session security cannot be used.

YES RACF session security can be used.

If you specify BINDSECURITY=YES for a particular APPC connection, a request to RACF is issued to extract the security profile. If the profile exists, it is used to bind the session.

Note: If you specify XAPPC=YES, the external security manager that you use must support the APPCLU general resource class, otherwise CICS fails to initialize.

Restrictions You can specify the XAPPC parameter in the SIT, PARM, or SYSIN only.

XCMD

The **XCMD** system initialization parameter specifies whether you want CICS to perform command security checking, and optionally the RACF resource class name in which you have defined the command security profiles.

XCMD={YES|name|NO}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to use a CICS command for the specified resource. Such checking is performed every time a transaction tries to use a COLLECT, DISABLE, DISCARD, ENABLE, EXTRACT, INQUIRE, PERFORM, RESYNC, or SET command, or any of the FEPI commands, for a resource.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the CMDSEC(YES) option on the transaction resource definition.

YES CICS calls RACF, using the default class name of CICSCMD prefixed by C or V, to check whether the userid associated with a transaction is authorized to use a CICS command for the specified resource. The resource class name is CCICSCMD and the grouping class name is VCICSCMD.

name CICS calls RACF, using the specified resource class name prefixed by C or V, to verify that the userid associated with a transaction is authorized to use a CICS command for the specified resource. The resource class name is *Cname* and the grouping class name is *Vname*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any command security checks, allowing any user to use commands that would be subject to those checks.

Restrictions: You can specify the **XCMD** parameter in the SIT, PARM, or SYSIN only.

XDCT

The **XDCT** system initialization parameter specifies whether you want CICS to perform transient data resource security checking.

XDCT={YES|name|NO}

If you specify YES or a RACF resource class name, CICS calls RACF to verify

that the userid associated with a transaction is authorized to access the transient data destination. Such checking is performed every time a transaction tries to access a transient data destination.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the transaction resource definition.

YES CICS calls RACF, with the default CICS resource class name of CICSDCT prefixed by D or E, to verify whether the userid associated with the transaction is authorized to access the specified destination. The resource class name is DCICSDCT and the grouping class name is ECICSDCT.

name CICS calls RACF, using the specified resource class name, to check whether the userid associated with the transaction is authorized to access the specified destination. The resource class name is *Dname* and the grouping class name is *Ename*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any transient data security checks, allowing any user to access any transient data destination.

Restrictions: You can specify the XDCT parameter in the SIT, PARM, or SYSIN only.

XEJB

The **XEJB** system initialization parameter specifies whether support of security roles is enabled.

XEJB={YES|NO}

Valid values are as follows:

YES CICS support for security roles is enabled:

- When an application invokes a method of an enterprise bean, CICS calls the external security manager to verify that the userid associated with the transaction is defined in at least one of the security roles associated with the method.
- When an application invokes the `isCallerInRole()` method, CICS calls the external security manager to determine whether the userid associated with the transaction is defined in the role specified on the method call.

NO CICS support for security roles is disabled:

- CICS does not perform enterprise bean method level checks, allowing any userid to invoke any enterprise bean method.
- The `isCallerInRole()` method always returns a value of true.

Restrictions:

1. You can specify the XEJB parameter in the SIT, PARM, or SYSIN only.
2. To enable security role support, you must also specify SEC=YES.

XFCT

The **XFCT** system initialization parameter specifies whether you want CICS to perform file resource security checking, and optionally specifies the RACF resource class name in which you have defined the file resource security profiles.

XFACT={YES|name|NO}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to access File Control-managed files. Such checking is performed every time a transaction tries to access a file managed by CICS file control. The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the resource definitions.

Note: You can specify the **XFACT** parameter in the SIT, PARM, or SYSIN only.

YES CICS calls RACF, using the default CICS resource class name of CICSFACT prefixed by F or H, to verify that the userid associated with a transaction is authorized to access files reference by the transaction. The resource class name is FCICSFACT and the grouping class name is HCICSFACT.

name CICS calls RACF, using the specified resource class name, to verify that the userid associated with a transaction is authorized to access files referenced by the transaction. The resource class name is *Fname* and the grouping class name is *Hname*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any file resource security checks, allowing any user to access any file.

XHFS

The **XHFS** system initialization parameter specifies whether CICS is to check the transaction user's ability to access files in the z/OS UNIX System Services file system.

XHFS={YES|NO}

At present, this checking applies only to the user ID of the Web client when CICS Web support is returning z/OS UNIX file data as the static content identified by a URIMAP definition. The checking is performed only if you have specified YES for the SEC system initialization parameter. However, the RESSEC option on the transaction resource definition does **not** affect this security checking.

Note: You can specify the **XHFS** parameter in the SIT, PARM, or SYSIN only.

YES CICS is to check whether the user identified as the Web client is authorized to access the file identified by the URIMAP that matches the incoming URL. This check is in addition to the check performed by z/OS UNIX System Services against the CICS region user ID. If access to the file is denied for either of these user IDs, the HTTP request is rejected with a 403 (Forbidden) response.

NO CICS is not to check the client user's access to z/OS UNIX files. Note that the CICS region user ID's access to these files is still checked by z/OS UNIX System Services.

XJCT

The **XJCT** system initialization parameter specifies whether you want CICS to perform journal resource security checking.

XJCT={YES|name|NO}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to access the

referenced journal. Such checking is performed every time a transaction tries to access a CICS journal. The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC is active for the resource definitions.

Note: You can specify the **XJCT** parameter in the SIT, PARM, or SYSIN only.

YES CICS calls RACF using the default CICS resource class name of CICSJCT prefixed by a J or K, to check whether the userid associated with a transaction is authorized to access CICS journals referenced by the transaction. The resource class name is JCICSJCT and the grouping class name is KCICSJCT.

name CICS calls RACF, using the specified resource class name prefixed by J or K, to verify that the userid associated with a transaction is authorized to access CICS journals.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any journal resource security checks, allowing any user to access any CICS journal.

XPCT

The **XPCT** system initialization parameter specifies whether you want CICS to perform started transaction resource security checking, and optionally specifies the name of the RACF resource class name in which you have defined the started task security profiles.

XPCT={YES|name|NO}

If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a transaction is authorized to use started transactions and related EXEC CICS commands. Such checking is performed every time a transaction tries to use a started transaction or one of the EXEC CICS commands: COLLECT STATISTICS TRANSACTION, DISCARD TRANSACTION, INQUIRE TRANSACTION, or SET TRANSACTION. The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the resource definitions.

Note: You can specify the **XPCT** parameter in the SIT, PARM, or SYSIN only.

YES CICS calls RACF using the default CICS resource class name CICSPCT prefixed with A or B, to verify that the userid associated with a transaction is authorized to use started transactions or related EXEC CICS commands.

The resource class name is ACICSPCT and the grouping class name is BCICSPCT.

name CICS calls RACF, using the specified resource class name, to verify that the userid associated with a transaction is authorized to use started transactions or related EXEC CICS commands. The resource class name is ACICSPCT and the grouping class name is BCICSPCT.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any started task resource security checks, allowing any user to use started transactions or related EXEC CICS commands.

XPPT

The **XPPT** system initialization parameter specifies that CICS is to perform application program resource security checks and optionally specifies the RACF resource class name in which you have defined the program resource security profiles.

XPPT={YES|name|NO}

You can specify the **XPPT** parameter in the SIT, PARM, or SYSIN only. Checking is performed every time a transaction tries to invoke another program by using one of the CICS commands: LINK, LOAD, or XCTL.

Note: The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the resource definitions.

YES CICS calls RACF, using the default resource class name prefixed by M or N, to verify that the userid associated with a transaction is authorized to use LINK, LOAD, or XCTL commands to invoke other programs. The resource class name is MCICSPPT and the grouping class name is NCICSPPT.

name CICS calls RACF, with the specified resource class name prefixed by M or N, to verify that the userid associated with a transaction is authorized to use LINK, LOAD, or XCTL commands to invoke other programs. The resource class name is *Mname* and the grouping class name is *Nname*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any application program authority checks, allowing any user to use LINK, LOAD, or XCTL commands to invoke other programs.

XPSB

The **XPSB** system initialization parameter specifies whether you want CICS to perform program specification block (PSB) security checking and optionally specifies the RACF resource class name in which you have defined the PSB security profiles.

XPSB={YES|name|NO}

You can specify the **XPSB** parameter in the SIT, PARM, or SYSIN only. If you specify YES, or a RACF resource class name, CICS calls RACF to check that the userid associated with a transaction is authorized to access PSBs (which describe databases and logical message destinations used by application programs). Such checking is performed every time a transaction tries to access a PSB.

Note:

1. The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the resource definitions.
2. If you require security checking for PSBs to apply to remote users who access this region by means of transaction routing, the system initialization parameter **PSBCHK**=YES must be specified. For more information about this parameter, see "PSBCHK" on page 217.

YES CICS calls RACF, using the default resource class name CICSPSB prefixed by P or Q, to verify that the userid associated with a

transaction is authorized to access PSBs. The resource class name is PCICSPSB and the grouping class name is QCICSPSB.

name CICS calls RACF, using the specified resource class name prefixed by P or Q, to verify that the userid associated with a transaction is authorized to access PSBs. The resource class name is *Pname* and the grouping class name is *Qname*.

The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any PSB resource security checks, allowing any user to access any PSB.

XRES

The **XRES** system initialization parameter specifies whether you want CICS to perform resource security checking for particular CICS resources and optionally specifies the general resource class name in which you have defined the resource security profiles.

XRES={YES|*name*|**NO**}

You can specify the **XRES** parameter in the SIT, PARM, or SYSIN only. If you specify YES, or a general resource class name, CICS calls the external security manager to verify that the user ID associated with a transaction is authorized to use the resource. This checking is performed every time a transaction tries to access a resource.

The actual profile name passed to the external security manager is the name of the resource to be checked, prefixed by its resource type; for example, for a document template whose resource definition is named "WELCOME", the profile name passed to the external security manager is DOCTEMPLATE.WELCOME. Even if a command references the document template using its 48-character template name, the shorter name (up to 8 characters) of the DOCTEMPLATE resource definition is always used for security checking.

The checking is performed only if you have specified YES for the **SEC** system initialization parameter and specified the RESSEC(YES) option on the TRANSACTION resource definition.

YES CICS calls the external security manager, using the default CICS resource class name of RCICSRES, to check whether the user ID associated with a transaction is authorized to use the resource it is trying to access. The resource class name is RCICSRES and the grouping class name is WCICSRES.

name CICS calls the external security manager, using the specified resource class name prefixed by the letter R, to check whether the userid associated with a transaction is authorized to use the resource it is trying to access. The resource class name is *Rname* and the grouping class name is *Wname*. The resource class name specified must be 1 through 7 characters.

NO CICS does not perform any security checks for resources, allowing access to any user.

XTRAN

The **XTRAN** system initialization parameter specifies whether you want CICS to perform transaction-attach security checking and optionally specifies the RACF resource class name in which you have defined the transaction security profiles.

XTRAN={YES|*name*|**NO**}

You can specify the **XTRAN** parameter in the SIT, PARM, or SYSIN only. If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with the transaction is permitted to run the transaction.

Note: The checking is performed only if you have specified YES for the **SEC** system initialization parameter.

YES CICS calls RACF, using the default CICS resource class name of CICSTRN prefixed by T or G, to verify that the userid associated with the transaction is authorized to run the transaction. The resource class name is TCICSTRN and the grouping class name is GCICSTRN.

name CICS calls RACF, using the specified resource class name prefixed by T or G, to verify that the userid associated with the transaction is authorized to run the transaction. The resource class name is *Tname* and the corresponding grouping class name is *Gname*.

The name specified must be 1 through 7 characters.

NO CICS does not perform any transaction-attach security checks, allowing any user to run any transaction.

XTST

The **XTST** system initialization parameter specifies whether you want CICS to perform temporary storage security checking and optionally specifies the RACF resource class name in which you have defined the temporary storage security profiles.

XTST={YES|*name*|**NO**}

You can specify the **XTST** parameter in the SIT, PARM, or SYSIN only. If you specify YES, or a RACF resource class name, CICS calls RACF to verify that the userid associated with a temporary storage request is authorized to access the referenced temporary storage queue.

Note: The checking is performed only if you have specified YES for the **SEC** system initialization parameter, specified the RESSEC option on the resource definitions, and specified TYPE=SECURITY in the temporary storage table (TST).

YES CICS calls RACF, using the default CICS resource class name of CICSTST prefixed by S or U, to verify that the userid associated with the transaction is authorized to access temporary storage queues referenced by the transaction. The resource class name is SCICSTST and the corresponding grouping class name is UCICSTST.

name CICS calls RACF, using the specified resource class name prefixed by S or U, to verify that the userid associated with a transaction is authorized to access temporary storage queues.

The name specified must be 1 through 7 characters.

NO CICS does not perform any temporary storage security checks, allowing any user to access any temporary storage queue.

XUSER

The **XUSER** system initialization parameter specifies whether CICS is to perform surrogate user checks.

XUSER={YES|NO}

You can specify the **XUSER** parameter in the SIT, PARM, or SYSIN only. Valid values are as follows:

YES CICS is to perform surrogate user checking in all those situations that permit such checks to be made; for example, on EXEC CICS START commands without an associated terminal. Surrogate user security checking is also performed by CICS against user IDs installing or modifying DB2 resource definitions that specify AUTHID or COMAUTHID.

Note: The **XUSER** parameter is also used by CICS to control access to the AUTHTYPE and COMAUTHTYPE attributes on DB2 resource definitions, although not through surrogate user checks. For more information about AUTHTYPE and COMAUTHTYPE attributes, see the *CICS Resource Definition Guide*.

For information about the various circumstances in which CICS performs surrogate user checks, see the *CICS RACF Security Guide*.

NO CICS is not to perform any surrogate user checking.

The default system initialization table

This default SIT is in CICSTS41.CICS.SDFHAUTH, and its source, named DFHSIT\$\$, is in CICSTS41.CICS.SDFHSAMP.

The macro source statements used to assemble the default system initialization table are given in Table 18.

Table 18. DFHSIT, the pregenerated default system initialization table

| Parameter | Default value | Description |
|---------------|--------------------|--|
| ADI | 30 | XRF(B) - Alternate delay interval |
| AIBRIDGE | AUTO | Bridge Autoinstall URM |
| AICONS | NO | No autoinstall for MVS CONSOLES |
| AIEXIT | DFHZATDX | Autoinstall user program name |
| AILDELAY | 0 | Delete delay period for AI terminals |
| AIQMAX | 100 | Maximum no. of terminals queued for AI |
| AIRDELAY | 700 | Restart delay period for AI terminals |
| AKPFREQ | 4000 | Activity keypoint frequency |
| APPLID | DBDCCICS | VTAM APPL identifier |
| AUTCONN | 0 | Autoconnect delay |
| AUTODST | NO | Lang env automatic storage tuning |
| AUTORESETTIME | NO | Time-of-day synchronization |
| AUXTR | OFF | Auxiliary trace option |
| AUXTRSW | NO | Auxiliary trace autoswitch facility |
| BMS | FULL, UNALIGN, DDS | Basic Mapping Support options |
| BRMAXKEEPTIME | 86400 | Bridge Max Keuptime |
| CICSSVC | 216 | The CICS SVC number |
| CILOCK | NO | Don't keep CI lock after read update |

Table 18. DFHSIT, the pregenerated default system initialization table (continued)

| Parameter | Default value | Description |
|-------------|---------------|---|
| CLSDSTP | NOTIFY | Notification for ISSUE PASS command |
| CLT | | The command list table option/suffix |
| CMDPROT | YES | Exec storage command checking |
| CMDSEC | ASIS | API command security checking |
| CONFDATA | SHOW | Show confidential data in dump/trace |
| CONFTXT | NO | Don't prevent VTAM tracing user data |
| CPSMCONN | NO | Do not connect to CPSM |
| CRLPROFILE | | Name of profile that allows CICS to access certificate revocation lists |
| CSDACC | READWRITE | CSD access |
| CSDBKUP | STATIC | Backup type of CSD (STATIC or DYNAMIC) |
| CSDBUFND | | Number of data buffers for the CSD |
| CSDBUFNI | | Number of index buffers for the CSD |
| CSDDISP | | CSD Disposition for dynamic allocation |
| CSDDSN | | CSD data set name for dynamic allocation |
| CSDFRLOG | NO | Journal id. for CSD forward recovery |
| CSDINTEG | UNCOMMITTED | Read integrity = uncommitted |
| CSDJID | NO | Journal id. for CSD auto. journaling |
| CSDLRNO | 1 | The VSAM LSR pool number for the CSD |
| CSDRECOV | NONE | CSD recoverable file option |
| CSDRLS | NO | Use traditional VSAM |
| CSDSTRNO | 6 | CSD Number of strings |
| CWAKEY | USER | CWA storage key |
| DAE | NO | SDUMPS will not be suppressed by DAE |
| DATFORM | MMDDYY | CSA date format |
| DB2CONN | NO | Do not connect to DB2 at CICS startup |
| DBCTLCON | NO | Do not connect to DBCTL at CICS start |
| DEBUGTOOL | NO | No Debugging Tool access |
| DFLTUSER | CICSUSER | Default user |
| DIP | NO | Batch data interchange program |
| DISMACP | YES | Disable macro programs |
| DOCCODEPAGE | 037 | Default host code page |
| DSALIM | 5M | Upper limit of DSA below 16 MB line |
| DSHIPIDL | 020000 | Delete shipped idle time |
| DSHIPINT | 120000 | Delete shipped interval |
| DSRTPGM | NONE | Distributed routing program |
| DTRPGM | DFHDYP | Dynamic routing program |
| DTRTRAN | CRTX | Default dynamic tran routing transid |
| DUMP | YES | Dump option |

Table 18. DFHSIT, the pregenerated default system initialization table (continued)

| Parameter | Default value | Description |
|---------------|--|--|
| DUMPDS | AUTO | CICS dump data set opening option |
| DUMPSW | NO | Dump data set autoswitch option |
| DURETRY | 30 | SDUMP total retry time (in seconds) |
| EDSALIM | 34M | Upper limit of DSA above 16 MB line |
| EJBROLEPRFX | | EJB ROLE PREFIX |
| ENCRYPTION | STRONG | Level of encryption for SSL |
| EODI | E0 | End-of-data indicator for sequential devices |
| ESMEXITS | NOINSTLN | External security manager exits |
| FCT | NO | File control table option/suffix |
| FCQRONLY | YES | Threadsafe FC runs on QR TCB |
| FEPI | NO | Front-End Programming Interface |
| FLDSEP | | End-of-field separator characters |
| FLDSTRT | | Field start character for built in function |
| FORCEQR | NO | Don't force QR for threadsafe programs |
| FSSTAFF | NO | Function-shipped START affinity option |
| FTIMEOUT | 30 | File timeout 30 seconds |
| GMTEXT | 'WELCOME TO CICS' | Good-morning message text |
| GMTRAN | CSGM | Initial transaction |
| GNTAN | NO | Signoff transaction |
| GRNAME | | Generic resource name for CICS TORs |
| GRPLIST | DFHLIST | List name of CSD groups for startup |
| GTFTR | OFF | GTF trace option |
| HPO | NO | VTAM High Performance Option (HPO) |
| ICP | | Interval control program start option |
| ICV | 1000 | Region exit interval (milliseconds) |
| ICVR | 5000 | Runaway task interval (milliseconds) |
| ICVTSD | 500 | Terminal scan delay interval (") |
| INITPARM | | Initialization parms for programs |
| INTTR | ON | CICS internal trace option |
| IRCSTRT | NO | Interregion communication start |
| ISC | NO | Intersystem communication option |
| JESDI | 30 | JES delay interval for XRF alternate |
| JVMCCPROFILE | | This setting is obsolete. If you specify this parameter, an error warning message is produced (MNOTE4) |
| JVMCCSIZE | 24M | Shared Class Cache size |
| JVMCCSTART | AUTO | Start SharedClassCache when needed |
| JVMPROFILEDIR | /usr/lpp/cicsts /cicsts41/JVMProfiles | JVM profile directory |
| KEYRING | | Key ring to be used by SSL support |

Table 18. DFHSIT, the pregenerated default system initialization table (continued)

| Parameter | Default value | Description |
|-------------|---------------|---|
| LGDFINT | 5 | Log defer interval in Log Manager |
| LGNMSG | NO | Extract VTAM logon data |
| LLACOPY | YES | Use MVS LLACOPY support |
| LPA | NO | Use-LPA option for CICS/user modules |
| MAXJVMTCBS | 5 | Maximum number of JVM open TCBs |
| MAXOPENTCBS | 12 | Maximum number of open TCBs |
| MAXSOCKETS | 65535 | Maximum number of IP sockets |
| MAXSSLTCBS | 8 | Limit on number of SSL TCBs |
| MAXXPTCBS | 5 | Limit on number of XP TCBs |
| MCT | NO | Monitoring control table option/suffix |
| MNIDN | OFF | Monitoring identity class option |
| MN | OFF | CICS monitoring option |
| MNCONV | NO | Monitoring converse recording option |
| MNEXC | OFF | Monitoring exception class option |
| MNFREQ | 0 | Monitoring frequency period |
| MNPER | OFF | Monitoring performance class option |
| MNRES | OFF | Monitoring resource class option |
| MNSYNC | NO | Monitoring syncpoint recording option |
| MNTIME | GMT | Monitoring timestamp (GMT/LOCAL) |
| MQCONN | NO | Do not connect to MQ at startup |
| MROBTCH | 1 | Number of MRO requests to batch |
| MROFSE | NO | Extend lifetime of Long-running mirror |
| MROLRM | NO | Long-running mirror task option |
| MSGCASE | MIXED | CICS messages in mixed case |
| MSGLVL | 1 | System console MSG level option |
| MXT | 5 | Maximum number of tasks in CICS |
| NATLANG | E | List of national languages |
| NCPLDFT | DFHNC001 | Named counter default pool name |
| NONRLSRECOV | VSAMCAT | Select location of recovery options for non-RLS files |
| OPERTIM | 120 | Write to operator timeout (seconds) |
| OPNDLIM | 10 | OPNDST/CLSDST request limit |
| PARMERR | INTERACT | System init. parameter errors option |
| PDI | 30 | Primary delay interval - XRF active |
| PDIR | NO | DL/I PSB directory option/suffix |
| PGAICTLG | MODIFY | PG autoinstall catalog state |
| PGAEXIT | DFHPGADX | PG autoinstall exit program |
| PGAIPGM | INACTIVE | PG autoinstall state |
| PGCHAIN | | BMS CHAIN command |
| PGCOPY | | BMS COPY command |

Table 18. DFHSIT, the pregenerated default system initialization table (continued)

| Parameter | Default value | Description |
|-------------|---------------|--|
| PGPURGE | | BMS PURGE command |
| PGRET | | BMS RETURN command |
| PLTPI | NO | Program list table PI option/suffix |
| PLTPISEC | NONE | No PLT security checks on PI programs |
| PLTPIUSR | | PLT PI userid = CICS region userid |
| PLTSD | NO | Program list table SD option/suffix |
| PRGDLAY | 0 | BMS purge delay interval |
| PRINT | NO | Print key option |
| PRTYAGE | 32768 | Dispatcher priority aging value |
| PSBCHK | NO | PSB resource checking required |
| PSDINT | 0 | Persistent session delay interval |
| PSTYPE | SNPS | VTAM single node persistent Sessions |
| PVDELAY | 30 | Timeout value for LUIT table |
| QUIESTIM | 240 | Timeout value for quiesce requests |
| RAMAX | 256 | Maximum I/O area for RECEIVE ANY |
| RAPOOL | 50 | Maximum RECEIVE ANY request parm. lists |
| RENTPGM | PROTECT | Reentrant program write protection |
| RESP | FME | Logical unit response type |
| RESSEC | ASIS | Resource security check |
| RLS | NO | RLS option |
| RLSTOLSR | NO | RLS files in LSRPOOL build calculation |
| RMTRAN | CSGM | XRF alternate recovery transaction |
| RRMS | NO | Recoverable resource management services |
| RST | NO | Recovery service table (XRF-DBCTL) |
| RSTSIGNOFF | NOFORCE | XRF - Re-sign on after takeover |
| RSTSIGNTIME | 500 | XRF - sign off timeout value |
| RUWAPool | NO | Allocating storage pool for Language Environment |
| SDTRAN | CESD | Shutdown transaction |
| SEC | YES | External security manager option |
| SECPRFX | NO | Security prefix |
| SKRPA1 | | SKR PA1 PAGE RETRIEVAL CMD |
| SKRPA2 | | SKR PA2 PAGE RETRIEVAL CMD |
| SKRPA3 | | SKR PA3 PAGE RETRIEVAL CMD |
| SKRPF1 | | SKR PF1 PAGE RETRIEVAL CMD |
| SKRPF2 | | SKR PF2 PAGE RETRIEVAL CMD |
| SKRPF3 | | SKR PF3 PAGE RETRIEVAL CMD |
| SKRPF4 | | SKR PF4 PAGE RETRIEVAL CMD |
| SKRPF5 | | SKR PF5 PAGE RETRIEVAL CMD |

Table 18. DFHSIT, the pregenerated default system initialization table (continued)

| Parameter | Default value | Description |
|-----------|---------------|--------------------------------------|
| SKRPF6 | | SKR PF6 PAGE RETRIEVAL CMD |
| SKRPF7 | | SKR PF7 PAGE RETRIEVAL CMD |
| SKRPF8 | | SKR PF8 PAGE RETRIEVAL CMD |
| SKRPF9 | | SKR PF9 PAGE RETRIEVAL CMD |
| SKRPF10 | | SKR PF10 PAGE RETRIEVAL CMD |
| SKRPF11 | | SKR PF11 PAGE RETRIEVAL CMD |
| SKRPF12 | | SKR PF12 PAGE RETRIEVAL CMD |
| SKRPF13 | | SKR PF13 PAGE RETRIEVAL CMD |
| SKRPF14 | | SKR PF14 PAGE RETRIEVAL CMD |
| SKRPF15 | | SKR PF15 PAGE RETRIEVAL CMD |
| SKRPF16 | | SKR PF16 PAGE RETRIEVAL CMD |
| SKRPF17 | | SKR PF17 PAGE RETRIEVAL CMD |
| SKRPF18 | | SKR PF18 PAGE RETRIEVAL CMD |
| SKRPF19 | | SKR PF19 PAGE RETRIEVAL CMD |
| SKRPF20 | | SKR PF20 PAGE RETRIEVAL CMD |
| SKRPF21 | | SKR PF21 PAGE RETRIEVAL CMD |
| SKRPF22 | | SKR PF22 PAGE RETRIEVAL CMD |
| SKRPF23 | | SKR PF23 PAGE RETRIEVAL CMD |
| SKRPF24 | | SKR PF24 PAGE RETRIEVAL CMD |
| SKRPF25 | | SKR PF25 PAGE RETRIEVAL CMD |
| SKRPF26 | | SKR PF26 PAGE RETRIEVAL CMD |
| SKRPF27 | | SKR PF27 PAGE RETRIEVAL CMD |
| SKRPF28 | | SKR PF28 PAGE RETRIEVAL CMD |
| SKRPF29 | | SKR PF29 PAGE RETRIEVAL CMD |
| SKRPF30 | | SKR PF30 PAGE RETRIEVAL CMD |
| SKRPF31 | | SKR PF31 PAGE RETRIEVAL CMD |
| SKRPF32 | | SKR PF32 PAGE RETRIEVAL CMD |
| SKRPF33 | | SKR PF33 PAGE RETRIEVAL CMD |
| SKRPF34 | | SKR PF34 PAGE RETRIEVAL CMD |
| SKRPF35 | | SKR PF35 PAGE RETRIEVAL CMD |
| SKRPF36 | | SKR PF36 PAGE RETRIEVAL CMD |
| SNSCOPE | NONE | Multiple CICS sessions per userid |
| SPCTR | (1,2) | Level(s) of special tracing required |
| SPOOL | NO | System spooling interface option |
| SRBSVC | 215 | HPO Type 6 SVC number |
| SRT | 1\$ | System recovery table option/suffix |
| SSLCACHE | CICS | SSL session id caching |
| SSLDELAY | 600 | SSL timeout value |
| START | AUTO | CICS system initialization option |

Table 18. DFHSIT, the pregenerated default system initialization table (continued)

| Parameter | Default value | Description |
|-----------|------------------------------|---|
| STARTER | YES | Starter (\$ and &numsign;) suffixes option Note: The default is NO but the parameter must be set to YES here to enable the SIT to assemble correctly. |
| STATEOD | 0 | Statistics end-of-day time |
| STATINT | 030000 | Statistics interval time |
| STATRCD | OFF | statistics recording status |
| STGPROT | NO | Storage protection facility |
| STGRCVY | NO | Storage recovery option |
| STNTR | 1 | Level of standard tracing required |
| SUBTSKS | 0 | Number of concurrent mode TCBs |
| SUFFIX | \$\$ | Suffix of this SIT |
| SYDUMAX | 999 | No of SYSDUMPS to be taken |
| SYSIDNT | CICS | Local system identifier |
| SYSTR | ON | Master system trace flag |
| TAKEOVR | MANUAL | XRF alternate takeover option |
| TBEXITS | | Backout exit programs |
| TCP | YES | Terminal control program option/suffix |
| TCPIP | NO | TCP/IP support |
| TCSACTN | NONE | TC Shutdown action |
| TCSWAIT | 4 | TC Shutdown wait |
| TCT | NO | Terminal control table option/suffix |
| TCTUAKEY | USER | TCT user area storage key |
| TCTUALOC | BELOW | TCT user area below 16MB |
| TD | (3,3) | Transient data buffers and strings |
| TDINTRA | NOEMPTY | Initial state of transient data queues |
| TDSUBTASK | OFF | No TD subtasking |
| TRANISO | NO | Transaction Isolation |
| TRAP | OFF | F.E. global trap exit option |
| TRDUMAX | 999 | No of TRANDUMPS to be taken |
| TRTABSZ | 4096 | Internal trace table size in 1K bytes |
| TRTRANSZ | 16 | Transaction dump trace table size |
| TRTRANTY | TRAN | Transaction dump trace option |
| TS | (3,3) | Temporary storage buffers and strings |
| TST | NO | Temporary storage table option/suffix |
| UOWNETQL | | Qualifier for NETUOWID |
| USERTR | ON | Master user trace flag |
| USRDELAY | 30 | Timeout value for User Dir. Entries |
| USSHOME | /usr/lpp/cicsts/ cicsts41 | The name and path of the root directory for CICS files on z/OS UNIX |

Table 18. DFHSIT, the pregenerated default system initialization table (continued)

| Parameter | Default value | Description |
|-----------|---------------|--|
| VTAM | YES | VTAM access method option |
| VTPREFIX | \ | Client virtual terminal prefix |
| WEBDELAY | (5,60) | Web timer values |
| WRKAREA | 512 | Common work area (CWA) size in bytes |
| XAPPC | NO | RACF class APPCLU required |
| XCFGROUP | DFHIR000 | XCF group to use for MRO communications |
| XCMD | YES | SPI use default name for RACF check |
| XDCT | YES | DCT use default name for RACF check |
| XDB2 | NO | No RACF security profile for DB2 |
| XEJB | YES | EJB security required |
| XFCT | YES | FCT use default name for RACF check |
| XHFS | YES | Security check for z/OS UNIX files |
| XJCT | YES | JCT use default name for RACF check |
| XLT | NO | Transaction list table option/suffix |
| XPCT | YES | PCT use default name for RACF check |
| XPPT | YES | PPT use default name for RACF check |
| XPSB | YES | PSB use default name for RACF check |
| XRES | YES | For resources subject to XRES security, checks use the default name for the RACF check. For a list of resources subject to XRES security checks, see the <i>CICS RACF Security Guide</i> . |
| XRF | NO | Extended recovery feature (XRF) option |
| XTRAN | YES | Transid use default name, RACF check |
| XTST | YES | TST use default name for RACF check |
| XUSER | YES | Surrogate user checking to be done |

Assembling the SIT

When you have coded the DFHSIT macro, you must assemble the SIT.

About this task

Procedure

1. Assemble and link-edit the table into an APF-authorized library, such as CICSTS41.CICS.SDFHAUTH.
2. Include this library in the STEPLIB concatenation of the CICS startup job stream.

Results

If you code a system initialization parameter in your SIT source, and the parameter's keyword is not defined in the CICS-supplied version of the DFHSIT macro, you get an IEV017 warning message from the assembly, as follows:

```
IEV017 ** WARNING ** UNDEFINED KEYWORD PARAM. DEFAULT TO POSITIONAL, INCLUDING KW --
```

What to do next

For information about assembling and link-editing CICS control tables, and an explanation of the syntax notation used to describe CICS macros, see the *CICS Resource Definition Guide*.

Selecting versions of CICS programs and tables

A CICS program is usually made up from a group of related CICS functional modules, one example of which is the terminal control program.

About this task

For most CICS programs you can only have one version, which is supplied with CICS. However, for some CICS programs you can create more than one version; for example, with different service levels. To select a particular version of a program, you can include the load library containing that version in the CICS startup JCL. For the basic mapping support (BMS) suite, however, you can select from different versions, by explicitly selecting the level of function needed.

You can also specify that a program is **not** needed (see “Excluding unwanted programs” for details).

You can use these methods **only** for the programs referred to in this section and in “Excluding unwanted programs,” by coding system initialization parameters.

Using an explicit level of function to select programs

You use an explicit level of function to select the BMS suite of programs.

When you specify your BMS requirement on the system initialization parameter BMS, you can select one of three versions. The BMS level of function is selected by the parameter options MINIMUM, STANDARD, or FULL, from which the system initialization program loads the set of programs you require.

Excluding unwanted programs

There are three ways to exclude programs that are not required.

You can exclude programs by specifying:

1. programname=NO
2. tablename=NO
3. function=NO

Specifying programname=NO

If you code *programnamesystem* initialization parameter=NO as a , (for example, DIP=NO), you exclude the named management program at CICS system initialization.

The programs that you can exclude by coding *programname:=NO* are

- Batch data interchange program (DIP)
- Terminal control program (TCP)

Note: In the case of DIP, you get a dummy version of the management program, which is supplied on the distribution tape with a suffix of **DY**.

Specifying tablename=NO for the programs control table

Not all of the CICS programs have a programname parameter in the SIT.

Ansystem initialization parameter alternative method is to code NO on the for the associated table. This has the same effect as coding NO against a program name parameter, and the associated CICS program is excluded at system initialization, either by loading a dummy program, or by some other technique.

The system recovery table (SRT) can be used in this way, and the associated system recovery program (SRP) will be excluded.

The dummy TCT, DFHTCTDY

There is a special case where you can also specify tablename=NO, but this does not load a dummy terminal control program. You specify TCT=NO when you are using resource definition online, and all your terminal resource definitions are in the CSD.

When you specify TCT=NO, CICS loads a dummy TCT named DFHTCTDY. A pregeneratedCICSTS41.CICS dummy table of this name is supplied in .SDFHLOAD, and the sourceCICSTS41.CICS statements of DFHTCTDY are supplied in .SDFHSAMP. If you specify TCT=NO, a generated table of this name must be available in a library of the DFHRPL concatenation when you start CICS.

The dummy TCT provides **only** the CICS and VTAM control blocks that you need if you are using VTAM terminals and using the CSD for storing terminal definitions. You define your VTAM terminals using the RDO transaction, CEDA, or the DEFINE command of the CSD batch utility program, DFHCSDUP.

Specifying function=NO

If you code function=NO as a system initialization parameter (for example, XRF=NO), you exclude the management program associated with the named function at CICS system initialization.

You can exclude intersystem communication (ISC), the 3270 print-request facility, the system spooling interface, or the extended recovery facility (XRF), in this way.

Chapter 17. Processing system initialization parameters

This chapter describes the CICS system initialization process.

About this task

It covers:

1. A brief introduction to the process of how you supply system initialization parameters, and the role of the CICS parameter manager domain in this process
2. An explanation of how CICS uses the special system initialization keywords
3. A description of the start and restart classes, and how they are controlled

This chapter describes:

- “Supplying system initialization parameters to CICS”
- “Using system initialization control keywords” on page 304
- “Controlling start and restart” on page 308

Supplying system initialization parameters to CICS

The CICS parameter manager domain loads a system initialization table (SIT) at the start of the initialization process.

About this task

You specify the SIT that defines the CICS characteristics appropriate to your needs by coding the suffix of the DFHSITxx load module (where xx is the suffix) on the SIT= system initialization parameter. If you fail to specify the suffix of a SIT, then CICS tries to load an unaffixed module.

You can modify many of the system initialization parameters dynamically at the beginning of CICS initialization by providing system initialization parameters in the startup job stream, or through the system console. There are also some system initialization parameters that you cannot code in the SIT, and can only supply at startup time. You specify system initialization parameters at startup time in any of three ways:

1. In the PARM parameter of the EXEC PGM=DFHSIP statement
2. In the SYSIN data set defined in the startup job stream
3. Through the system operator's console

You can use just one of these methods, or two, or all three. However, CICS processes these three sources of input in strict sequence, as follows:

1. The PARM parameter
2. The SYSIN data set (but only if SYSIN is coded in the PARM parameter; see SYSIN)
3. The console (but only if CONSOLE is coded in either the PARM parameter or in the SYSIN data set; see CONSOLE(CN))

Note: If you supply duplicate system initialization parameters, either through the same or a different medium, CICS takes the last one that it reads. For example, if

you specify MCT=1\$ in the PARM parameter, MCT=2\$ in the SYSIN data set, and finally enter MCT=3\$ through the console, CICS loads DFHMCT3\$.

Using system initialization control keywords

You can use the SYSIN, CONSOLE, and END control keywords at startup to control how CICS is to read system initialization parameters.

About this task

The purpose of these special keywords, and where you can code them, are described as follows:

SYSIN (SI)

This keyword tells CICS to read initialization parameters from the SYSIN data set.

Where to code SYSIN: You can code SYSIN (or SI) only in the PARM parameter of the EXEC PGM=DFHSIP statement. The keyword can appear once only and must be at the end of the PARM parameter. CICS does not read SYSIN until it has finished scanning all of the PARM parameter, or until it reaches a .END before the end of the PARM parameter. (See .END on page END.)

Examples:

```
//stepname EXEC PGM=DFHSIP,PARM='SIT=6$,SYSIN,.END'  
//stepname EXEC PGM=DFHSIP,PARM='SIT=6$,DLI=YES,SYSIN,.END'
```

CONSOLE (CN)

This keyword tells CICS to read initialization parameters from the console. CICS prompts you with message DFHPA1104 when it is ready to read parameters from the console.

Where to code CONSOLE: You can code CONSOLE (or CN) in the PARM parameter of the EXEC PGM=DFHSIP statement or in the SYSIN data set. This keyword can appear either at the end of the PARM parameter or in the SYSIN data set, but code it in one place only.

If you code CONSOLE (or CN) in the PARM parameter, and PARM also contains the SYSIN keyword, CICS does not begin reading parameters from the console until it has finished reading and processing the SYSIN data set.

Similarly, wherever you place the CONSOLE keyword in the SYSIN data set, CICS does not begin reading parameters from the console until it has finished reading and processing the SYSIN data set.

Examples:

```
//stepname EXEC PGM=DFHSIP,PARM='SIT6$,CONSOLE,.END'  
//stepname EXEC PGM=DFHSIP,PARM='CONSOLE,SYSIN,.END'  
//stepname EXEC PGM=DFHSIP,PARM='SIT=6$,CN,SI,.END'
```

If both SYSIN (or SI) and CONSOLE (or CN) appear as keywords of the PARM parameter, the order in which they are coded is irrelevant as long as no other keywords, other than .END, follow them.

.END

The meaning of this keyword varies:

PARM The use of the .END keyword is optional in the PARM parameter. If you omit it, CICS assumes it to be at the end of the PARM parameter. If you code .END in the PARM parameter it can have one of two meanings:

1. If you also code one, or both, of the other control keywords (CONSOLE and/or SYSIN) .END denotes the end of the PARM parameter only.

For example:

```
//stepname EXEC PGM=DFHSIP,PARM='SIT6$,SI,CN,.END'
```

2. If you code .END as the only control keyword in the PARM parameter, it denotes the end of all system initialization parameters, and CICS begins the initialization process. For example:

```
//stepname EXEC PGM=DFHSIP,PARM='SIT=6$, .END'
```

If .END is not the last entry in the PARM parameter, CICS truncates the PARM parameter and the parameters following the .END keyword are lost.

SYSIN

The use of the .END keyword is optional in the SYSIN data set. If you omit it, CICS assumes it to be at the end of SYSIN. If you code .END in the SYSIN data set its meaning depends on your use of the CONSOLE keyword, as follows:

- If you code the CONSOLE control keyword in the PARM parameter or in the SYSIN data set, .END denotes the end of the SYSIN data set only.
- If you do not code the CONSOLE control keyword in the PARM parameter or in the SYSIN data set, .END denotes the end of all CICS system initialization parameters, and CICS begins the initialization process.

If you code .END, and it is not the last entry in the SYSIN data set, or not at the end of a SYSIN record, CICS initialization parameters following the .END are ignored. To avoid accidental loss of initialization parameters, ensure that the .END keyword is on the last record in the SYSIN data set, and that it is the only entry on that line. (However, if you want to remove some system initialization parameters from a particular run of CICS, you could position them after the .END statement just for that run.)

The following example shows the use of .END in a SYSIN data set:

```
//SYSIN DD *
* CICS system initialization parameters
SIT=6$,START=COLD,
* XRF=NO,          ( XRF this run - SIT defines XRF=YES
PDIR=1$,          ( SUFFIX of PSB directory

.END
/*
```

CONSOLE

The meaning of .END through the console depends on whether you are entering new parameters or entering corrections. The two meanings are as follows:

1. If you are keying new parameters in response to message DFHPA1104, .END terminates parameter reading, and CICS starts initialization according to the SIT it has loaded, but modified by any system initialization parameters you have supplied. Until you enter the .END control keyword, CICS continues to prompt you for system initialization parameters.
2. If you have coded PARMERR=INTERACT, and CICS detects a parameter error, either in the keyword or in the value that you

have assigned to it, CICS prompts you to correct the error with message DFHPA1912 or DFHPA1915. If you enter the correct keyword or value, initialization resumes with CICS continuing to process either the PARM parameter, the SYSIN data set, or prompting for more system initialization parameters through the console, depending on where CICS detected the error. If you cannot correct the error, but want CICS to continue with the initialization process, you can enter .END to end the error correction phase.

Processing the PARM parameter

If you omit the **PARM** parameter from the EXEC PGM=DFHSIP statement, CICS assumes that there are no SIT overrides or other initialization parameters and tries to load an unaffixed module named DFHSIT.

As a general rule, the **PARM** parameter must at least specify the suffix of your system initialization table, using the SIT keyword. Alternatively, you can code the special SYSIN keyword as the only **PARM** parameter, and supply the suffix of your SIT and the other system initialization parameters from the SYSIN data set.

CICS scans the PARM string looking for a **SIT** parameter, any of the special control keywords, or any system initialization parameters, and proceeds as follows:

- If CICS finds a **SIT** parameter but no SYSIN keyword, CICS tries to load the SIT as soon as it has finished scanning the **PARM** parameter. Processing any CICS system initialization parameters that are also present in the **PARM** parameter takes place only after the SIT has been loaded.
- If CICS finds a **SIT** parameter and also a SYSIN keyword, CICS does not try to load the SIT until it has also finished scanning the SYSIN data set. In this case, loading the SIT is deferred because there might be other **SIT** parameters coded in the SYSIN data set that override the one in the **PARM** parameter.

Processing any system initialization parameters that are also present in the **PARM** parameter takes place only after the SIT has been loaded.

Rules for coding the PARM parameter

The following rules apply when coding the **PARM** parameter:

- The maximum number of characters you can code is 100, excluding the opening and closing delimiters, which can be either apostrophes or parentheses.
- All CICS system initialization parameters must be separated by a comma, and the separating commas are included in the 100 character limit. Because of this limiting factor, you might prefer to limit the use of the **PARM** parameter to specify the SYSIN control keyword only.

The rules for coding the **PARM** parameter on an EXEC job control statement are described fully in *z/OS MVS JCL Reference*.

Processing the SYSIN data set

CICS scans the SYSIN data set looking for a SIT= parameter and any of the special keywords, as well as system initialization parameters.

If CICS finds a SIT= parameter in SYSIN, it tries to load that SIT, overriding any that was specified in the PARM parameter. If CICS does not find a SIT= parameter in SYSIN, it tries to load any SIT specified in the PARM parameter.

However, if after scanning the PARM parameter and the SYSIN data set CICS has not found a SIT= parameter, CICS does one of the following:

1. If you specified CONSOLE in the PARM parameter or in the SYSIN data set, CICS prompts you with the following message to enter the SIT suffix as the first parameter through the console:

```
rr DFHPA1921 DBDCCICS PLEASE SPECIFY THE REQUIRED SIT SUFFIX, OR
                        SPECIFY 'NONE' (UNSUFFIXED).
```

2. If you did not specify CONSOLE, CICS tries automatically to load an unsuffixed SIT module (DFHSIT). If this load fails, CICS issues message DFHPA1106, requesting a SIT suffix in reply to the message.

Note: CICS does not process any system initialization parameters that are coded in the PARM parameter and the SYSIN data set until after the SIT has been loaded.

Rules for coding CICS system initialization parameters in the SYSIN data set

There are a few rules to observe when coding CICS system initialization parameters in the SYSIN data set.

These are:

- You must use a comma to separate parameters that are on the same line, but the use of a comma at the end of a SYSIN record is optional.
- You can use an asterisk in column 1 to code comments, or to remove temporarily an initialization parameter from a particular execution of CICS.
- You can also add comments after the parameters on a SYSIN line, but they must be preceded by at least one blank character.
- SYSIN is an 80-byte file, and everything that appears in positions 1 through 80 is treated by CICS as input data.
- You can continue, on another line in SYSIN, parameters that have multiple operands if you make the split immediately after a comma. CICS concatenates the operands, omitting the intervening blanks.
- As a general rule, you cannot split an individual operand between lines. However, in the case of the GMTEXT parameter, you can enter the operand over more than one line up to the maximum of 246 characters. The format of this parameter is:

```
GMTEXT='User''s text'
```

You can use apostrophes to punctuate message text, provided that you code *two* successive apostrophes to represent a single apostrophe (as shown in the example above). The apostrophes delimiting the text are mandatory.

- You must take care when coding parameters that use apostrophes, parentheses, or commas as delimiters, because failure to include the correct delimiters is likely to cause unpredictable results.

Processing the console entries

Generally, CICS does not begin to read from the console until it has loaded the SIT and processed any initialization parameters that are coded in the PARM parameter and the SYSIN data set. CICS accepts system initialization parameters from the console until you terminate the input with .END.

You can specify a SIT= parameter only as the first parameter through the console when prompted by message DFHPA1921, at which point CICS tries to load the specified SIT. If you try to specify a SIT= parameter after CICS has loaded the SIT it is rejected as an error.

Rules for coding CICS system initialization parameters at the console

When it is ready to read parameters from the console, CICS displays the DFHPA1104 message.

You can enter as many initialization parameters as you can get on one line of the console, but you must use a comma to separate parameters. CICS continues to prompt for system initialization parameters with displays of message DFHPA1105 until you terminate console input by entering the .END control keyword.

Entering corrections to initialization parameters through the console

If you have coded PARMERR=INTERACT, and CICS detects a parameter error, either in the keyword or in the value you have assigned to it, CICS prompts you to correct the error with message DFHPA1912 or DFHPA1915.

CICS prompts you to enter corrections to any errors it finds in the **PARM** parameter or the **SYSIN** data set after it has loaded the **SIT** and as each error is detected. This means that if there is an **APPLID** parameter following the parameter that is in error, either in the **PARM** parameter or in the **SYSIN** data set, it is the **APPLID** coded in the **SIT** that CICS displays in messages DFHPA1912 and DFHPA1915.

Controlling start and restart

The type of initialization that CICS performs is not only determined by the **START** parameter.

The CICS local and global catalogs also play a major role in the initialization process, together with any system initialization parameters that you provide, either in the **SIT** or at run time by one of the three methods described in “Using system initialization control keywords” on page 304..

The role of the CICS catalogs

CICS uses its catalogs to save information between CICS shutdown and the next restart.

The global catalog

CICS uses the global catalog to save all resource definitions that were installed when CICS shut down.

These are:

- BMS maps sets and partition sets
- Connections and sessions
- Files
- **LIBRARY** concatenations
- Programs
- Terminals and typeterms
- Transactions and transaction profiles
- Transient data queues

The resource definitions that CICS saves at its shutdown might have been installed during a cold start from a list of groups specified by a group list system initialization parameter, or dynamically using RDO when the CICS region is running.

If you run CICS with `START=AUTO` and a warm or emergency restart results, CICS restores all the installed resource definitions as they were at normal CICS shutdown, or at the time of system failure. If a subsequent error occurs during the restore, appropriate messages are displayed. The general rule is that you cannot alter installed resource definitions during a restart except by coding `START=COLD` or `START=INITIAL`. For details of the results of the possible combinations of CICS restart-type and global catalog state, see “The `START` system initialization parameter.”

The CICS domains also use the global catalog to save their domain status between runs. In some cases this information can be overridden during a restart by supplying system initialization parameters. In other cases the domain information saved in the catalog is always used in a restart. For example, the CICS statistics interval time is always restored from the catalog in a warm or emergency restart, because the statistics domain does not have this as a system initialization parameter. To change the interval time, you must use `CEMT` or `EXEC CICS` commands after control is given to CICS. Alternatively, you can enforce system defaults by performing a cold start.

Note: If you have to reinitialize the global catalog for any reason, you must also reinitialize the local catalog.

The local catalog

The CICS domains use the local catalog to save some of their information between CICS runs.

If you delete and redefine the local catalog, you must:

1. Initialize the local catalog with an initial set of domain records.
2. Use the CICS-supplied utility program, `DFHSMUTL`, to re-add records to enable the CICS self-tuning mechanism for storage manager domain subpools. For details of how to do this see the *CICS Operations and Utilities Guide*.
3. Delete and reinitialize the global catalog.

For more information about initializing the local catalog, see “Defining the local catalog” on page 71. Some of the information that is saved in the local catalog can be overridden at CICS system initialization by system initialization parameters, such as CICS transaction dump data set status.

Note: If you need to reinitialize the local catalog for any reason, you must also reinitialize the global catalog.

The `START` system initialization parameter

You can influence the type of startup that CICS performs, by specifying the `START` system initialization parameter, as follows:

`START=AUTO`

If you code `AUTO` as the `START` operand, CICS determines which of four possible types of start to perform by looking for two records which may or may not be present in the global catalog:

- The recovery manager control record

- The recovery manager autostart override record

Depending on whether either or both of these records exist, and their contents, CICS decides which type of start to perform:

1. Initial start

CICS performs an initial start in each of the following cases:

- There is a recovery manager autostart override record that specifies AUTOINIT in the global catalog.
- The control record specifies an initial start. (This can happen if a previous initial start failed.)

If you set CICS to perform an initial start, you should reinitialize the local catalog before bringing up CICS.

2. Cold start

CICS performs a cold start in the following cases:

- The recovery manager control record specifies a cold start. (This can happen if a previous cold start did not complete.)
- There is both a recovery manager control record (which specifies anything other than an initial start) *and* an autostart override record that specifies AUTOCOLD.

Log records for local resources are purged and resource definitions rebuilt from the CSD or CICS control tables. Units of work on other systems are resynchronized with this system, as described under START=COLD.

3. Warm start

If the recovery manager control record indicates that the previous run of CICS terminated normally with a successful warm keypoint, CICS performs a warm restart—*unless* the autostart override record specifies AUTOINIT or AUTOCOLD, in which case an initial or cold start is performed.

For the warm restart to be successful, the local catalog must contain the information saved by the CICS domains during the previous execution.

A warm start restores CICS to the state it was in at the previous shutdown.

You can modify a warm restart by coding the NEWSIT system initialization parameter. This has the effect of enforcing the system initialization parameters coded in the SIT, overriding any cataloged status from the previous CICS shutdown.

The exceptions to this is the system initialization parameter FCT, the CSDxxxxx group (for example CSDACC), and GRPLIST, which are always ignored in a warm restart, even if you specify NEWSIT=YES. Specifying NEWSIT=YES causes, in effect, a partial cold start.

4. Emergency start

If the control record in the global catalog indicates that the previous run of CICS terminated in an immediate or uncontrolled shutdown, CICS performs an emergency restart.

START=AUTO should be the normal mode of operation, with the choice of start being made by CICS automatically. Use the recovery manager utility program, DFHRMUTL, to set overrides.

START=INITIAL

CICS initializes using the resource definitions specified by the system initialization parameters, ignoring any previously installed resource definitions saved in a warm keypoint in the global catalog. This includes all the groups of resources specified by the GRPLIST= system initialization parameter, and those resources specified in CICS control tables.

Note: The global catalog and system log are initialized, and all information in them is lost. Because recovery information for remote systems is not preserved, damage may be done to distributed units of work

You should rarely need to specify START=INITIAL; if you want to reinstall definitions of local resources from the CSD, use START=COLD instead.

Examples of times when an initial start is necessary are:

- When bringing up a new CICS system for the first time.
- After a serious software failure, when the system log has been corrupted.
- If the global catalog is cleared or reinitialized.
- When you want to run CICS with a dummy system log. (If the system log is defined as a dummy, it is ignored.)

If it *is* necessary to perform an initial start of CICS, you can do so in two ways:

- By specifying START=INITIAL.
- By using the recovery manager utility program, DFHRMUTL, to set the autostart override record to AUTOINIT, and specifying START=AUTO. For information about DFHRMUTL, see the *CICS Operations and Utilities Guide*.

START=COLD

CICS initializes using the resource definitions specified by the system initialization parameters, ignoring any previously installed resource definitions saved in a warm keypoint in the global catalog. This includes all the groups of resources specified by the GRPLIST= system initialization parameter, and those resources specified in CICS control tables.

Recovery information relating to remote systems or to RMI-connected resource managers is preserved. The CICS log is scanned during startup, and any information regarding unit of work obligations to remote systems, or to non-CICS resource managers (such as DB2) connected through the RMI, is preserved. (That is, any decisions about the outcome of local UOWs, needed to allow remote systems or RMI resource managers to resynchronize their resources, are preserved.)

Note that, on a cold start, the following are *not* preserved:

- Updates to *local* resources that were not fully committed or backed out during the previous execution of CICS. In particular, although remote systems may resynchronize their units of work successfully, local resources updated in those distributed units of work are not locked and need not be in either a committed or a backed-out state.
- Recovery information for remote systems connected by LU6.1 links, or for earlier releases of CICS systems connected by MRO.
- Any program LIBRARY definitions that had been dynamically defined. Only the static DFHRPL concatenation will remain, together with any LIBRARY definitions in the grouplist specified at startup or installed via BAS at startup.

A start initiated by START=COLD is not entirely without reference to the previous run of a CICS system using the same global catalog. If you want to

perform a fully cold start of CICS, without reference to any previous execution, code START=INITIAL. If you want to reinstall definitions of local resources from the CSD, use START=COLD.

There may be times when it is necessary to perform a cold start of CICS, irrespective of the type of system termination that CICS recorded in the global catalog. You can do this in two ways:

- By specifying START=COLD.
- By using DFHRMUTL to set the autostart override record to AUTOCOLD, and specifying START=AUTO.

START=STANDBY

The STANDBY option is for use only with XRF=YES. START=STANDBY causes the alternate CICS region to initialize only to the point at which it can monitor the active CICS region. Depending on how the active CICS region was shut down, the alternate CICS region completes either a warm or emergency restart, if it needs to take over, as follows:

- If the active CICS region was shut down by a successfully completed CEMT PERFORM SHUTDOWN TAKEOVER command, the alternate CICS region performs a **warm** start.
- If the active CICS region was shut down abnormally, the alternate CICS region performs an **emergency** restart.

When it takes over, the alternate CICS region becomes the active CICS region.

The effect of specifying START=STANDBY on the alternate CICS region is similar to that of the START=AUTO option.

If you code START=STANDBY with XRF=NO, initialization fails with message DFHXA6530, and CICS terminates abnormally with a dump.

For information about operating a CICS region with XRF, see the *CICS Operations and Utilities Guide*.

Table 19 shows how the effect of the START parameter depends on the state of the CICS global catalog and system log.

Note: If the system log is defined as a dummy, it is ignored.

Table 19. Effect of the START= parameter in conjunction with the global catalog and system log

| START parm. | State of global catalog | State of system log | Result at restart |
|-------------|--|--------------------------------|--|
| Any. | Not defined to VSAM. | Any. | JCL error. |
| INITIAL | Defined. | Any. | CICS performs an initial start. The global catalog and system log are initialized. |
| COLD | Defined but contains no recovery manager control record. | Any. | After prompting for confirmation, CICS performs an initial start. The global catalog and system log are initialized. |
| COLD | Contains recovery manager records. | Not defined or dummy or empty. | Message DFHRM0401 is issued. Startup fails. |

Table 19. Effect of the START= parameter in conjunction with the global catalog and system log (continued)

| START parm. | State of global catalog | State of system log | Result at restart |
|-------------|--|-------------------------------------|---|
| COLD | Contains recovery manager records. | Contains records from previous run. | CICS performs a cold start. Recovery records in the system log that relate to changes to local resources are deleted. |
| AUTO | Defined but contains no recovery manager control record and no AUTOINIT autostart override. | Any. | Message DFHRM0137 is issued. Startup fails. |
| AUTO | Contains a recovery manager AUTOINIT autostart override. | Any. | CICS performs an initial start, without prompting. The global catalog and system log are initialized. |
| AUTO | Contains a recovery manager control record that does <i>not</i> indicate an initial start, and an AUTOCOLD autostart override. | Contains records from previous run. | CICS performs a cold start. Recovery records in the system log that relate to changes to local resources are deleted. |
| AUTO | Contains recovery manager records, but no AUTOINIT override. | Not defined or dummy or empty. | Message DFHRM0401 is issued. Startup fails. |
| AUTO | Contains a recovery manager control record indicating an initial start. | Any. | CICS performs an initial start. The global catalog and system log are initialized. |
| AUTO | Contains a recovery manager control record indicating a cold start, and no autostart override. | Contains records from previous run. | CICS performs a cold start. Recovery records in the system log that relate to changes to local resources are deleted. |
| AUTO | Contains a recovery manager control record indicating an emergency start, and no autostart override. | Contains records from previous run. | CICS performs an emergency start. |
| AUTO | Contains a recovery manager control record indicating a warm start, and no autostart override. | Contains records from previous run. | CICS performs a warm start. |

Note:

1. It is important to keep the CICS global and local catalogs in step. If CICS tries to perform a warm or emergency start and finds that the local catalog has been initialized, startup fails. Therefore, only initialize the local catalog at the same time as the global catalog.
2. It is recommended that you always run the DFHRMUTL and DFHCCUTL utilities in the same job. Run DFHRMUTL first and check its return code before running DFHCCUTL. If you do this, the global and local catalogs should never get out of step. For information about running DFHRMUTL and DFHCCUTL, see the *CICS Operations and Utilities Guide*.

Table 20 shows the effect of different types of CICS startup on the CICS trace, monitoring, statistics, and dump domains.

Table 20. Effect of the type of startup on CICS domains

| Domain | State of the CICS catalogs | Warm or emergency start | Initial or cold start |
|------------|---|---|---|
| Trace | Not relevant. | Domain initializes according to the system initialization parameters. | Domain initializes according to the system initialization parameters. |
| Monitoring | The global catalog is newly initialized. | Domain initializes according to the system initialization parameters. | Domain initializes according to the system initialization parameters. |
| Monitoring | The global catalog contains status of monitoring at the previous CICS shutdown. | Domain uses monitoring status from the catalog, but modified by any system initialization override parameters. | Domain initializes according to the system initialization parameters. |
| Statistics | The global catalog is newly initialized. | Domain initializes according to CICS-defined system default values. | Domain initializes according to CICS-defined system default values. |
| Statistics | The global catalog contains status of statistics at CICS shutdown. | Domain uses statistics status from the catalog. | Domain initializes according to CICS-defined system default values. |
| Dump | The global catalog is newly initialized. | Domain initializes the dump table according to CICS-defined system default values. Other dump attributes are set by system initialization parameters. | Domain initializes an empty dump table, and takes CICS-defined default action for all dump requests. Other dump attributes are set by system initialization parameters. |
| Dump | The global catalog contains dump status at CICS shutdown. | Domain reads the dump table and dump status from the catalog. Other dump attributes are modified by any system initialization parameters. | Domain initializes an empty dump table, and takes CICS-defined default action for all dump requests. Other dump attributes are set by system initialization parameters. |

CICS startup and the VTAM session

In a VTAM network, the session between CICS and VTAM is started automatically if VTAM is started before CICS.

If VTAM is not active when you start CICS, you receive the following messages:

```
F vtamname,USERVAR,ID=generic-applid,VALUE=specific-applid
+DFHSI1589D 'applid' VTAM is not currently active.
+DFHSI1572 'applid' Unable to OPEN VTAM ACB - RC=xxxxxxx, ACB CODE=yy.
```

Although the MODIFY NET, USERVAR command is only significant when you are running CICS with XRF, the USERVAR message occurs for both XRF=YES and XRF=NO CICS systems. If you receive messages DFHSI1589D and DFHSI1572, and if the CICS region is not initializing as an alternate CICS region, you can start the

CICS-VTAM session manually when VTAM is eventually started, by means of the CEMT SET VTAM OPEN command from a supported MVS console or a non-VTAM terminal.

If VTAM is active, but CICS still cannot open the VTAM ACB because VTAM does not recognize the CICS APPLID, you receive the following messages:

```
F vtamname,USERVAR,ID=generic-applid,VALUE=specific-applid
+DFHSI1592I 'applid' CICS applid not (yet) active to VTAM.
+DFHSI1572 'applid' Unable to OPEN VTAM ACB - RC=00000008, ACB CODE=5A.
```

This may be caused by an error in the value of APPLID operand, in which case you must correct the error and restart CICS. For information about other causes and actions, see the *CICS Messages and Codes*.

Concurrent initialization of VTAM and XRF alternate CICS regions

An XRF alternate CICS region cannot initialize properly until it has successfully opened the VTAM ACB.

Because VTAM and the alternate CICS region may be initialized concurrently, it is possible that several tries may have to be made to open the VTAM ACB. If VTAM is not active, the following message is written to the system console every 15 seconds:

```
DFHSI1589D 'applid' VTAM is not currently active.
```

If VTAM is active, but CICS cannot open the VTAM ACB, the following messages are written to the system console:

```
+DFHSI1572 'applid' Unable to OPEN VTAM ACB - RC=xxxxxxx, ACB CODE=yy.
DFHSI1590 'applid' XRF alternate cannot proceed without VTAM.
```

CICS abends with a dump (abend code 1590).

The CICS parameter manager domain

In addition to loading the system initialization table at the start of initialization, and reading any other parameters from PARM, SYSIN, or the system console, the parameter manager domain is responsible for the management of the SIT.

With the exception of the application domain (AP) which uses the SIT directly, the parameter manager domain passes system initialization parameters to the other CICS domains on request.

The domain initialization process is as follows:

Query the type of startup

With the exception of the trace domain, each domain asks the parameter manager for the type of startup—initial, cold, or warm. (For this purpose, the parameter manager domain treats an emergency restart as a warm start.)

Startup is initial or cold

If startup is initial or cold, domains do not read their domain records from the catalogs. Instead, they request system initialization parameters from the parameter manager domain. Because it is a cold start, the parameter manager domain returns all system initialization parameters from the SIT, modified by any overrides.

Startup is warm

If startup is warm, domains try to perform a warm start by reading their domain records from the catalogs:

- If they succeed in reading their status records, domains perform a warm start. Where applicable, they also request system initialization parameters from the parameter manager domain. Because it is a warm start, the parameter manager domain returns only those system initialization parameters supplied as overrides by PARM, SYSIN, or the system console.
- If they fail for any reason to read their status records, domains perform a cold start. They do this either by requesting **all** system initialization parameters from the parameter manager domain, or by using system default values if the domain does not have any system initialization parameters.

NEWSIT or new suffix

Although a START=AUTO may resolve to a warm start, parameter manager enforces most system initialization parameters if:

- You specify NEWSIT=YES as a system initialization parameter in PARM, SYSIN, or through the console.
- You specify a different SIT suffix from the previous run of CICS. Parameter manager saves the suffix from each run in the global catalog, and, if it detects a new suffix, it forces the NEWSIT=YES option.

For details of the parameters that are ignored when you specify NEWSIT=YES, see the NEWSIT parameter description on page NEWSIT.

Note: The trace domain is an exception to the above rules in that it always cold starts. Trace does not save its status at CICS shutdown like the other domains, and regardless of the type of startup, it requests **all** of its system initialization parameters from the parameter manager domain.

End of CICS startup

Whichever type of startup is performed, when the DFHSI1517 message is displayed on the operating system console, CICS is ready to process terminal requests.

When the startup process is completed, users are able to enter transactions from any terminals that are connected to CICS. For information about the CICS-supplied transactions, see *CICS Supplied Transactions*.

Chapter 18. Retrieving information about system initialization parameters

You can use the CICSplex SM API to discover information about CICS system initialization parameters and system initialization parameter overrides. System initialization parameter retrieval is supported by the CICSplex SM command-level interface, the CICS management client interface (CMCI), and the Web User Interface (WUI).

When you retrieve parameters you have the following options:

- You can retrieve the current values of the parameters in the SIT including any override values.
- You can retrieve the original SIT values as specified at system startup.
- You can retrieve the values from a single override source.

In common with many other CICSplex SM operations, you can control which CICS regions the retrieval operates on by specifying context and scope.

System initialization parameter retrieval is implemented using the CICSplex SM resource SYSPARM. The SYSPARM resource has two required parameters, PARMSRCE and PARMTYPE, associated with the GET operation. You use these parameters to specify which parameters to retrieve according to their source.

You can implement system initialization parameter discovery in three ways:

- In an API program using the EXEC CPSM GET command operating on the SYSPARM object.
- Using the CMCI GET method operating on the CICSSystemParameter external resource.
- Using the WUI operations view based on the SYSPARM resource table linked from the CICS region view set.

When you use EXEC CPSM GET command or CMCI, you can define a parameter expression using PARMSRCE and PARMTYPE to specify which parameters to retrieve. In the WUI you can use PARMSRCE and PARMTYPE as filters to control the records displayed. Both of these parameters are mandatory. For PARMTYPE, you must specify a value of SIT. For PARMSRCE, specify one of the following options:

COMBINED

Combination of the original system initialization table definitions and any applied parameter overrides

CONSOLE

Override parameters as specified at startup on the system console

JCL Override parameters provided through a JCL EXEC PGM statement

SYSIN

Override parameters from the startup job stream defined in the SYSIN data set

TABLE

The original system initialization table values extracted from the DFHSITxx load module

The parameters and override values are returned in the order of the KEYWORD attribute of the SYSPARM resource table irrespective of their order in the source, for example the SYSIN data set.

System parameter values can exceed 255 bytes. In these cases the parameter value is split into multiple records each containing a maximum of 255 bytes of parameter value. Each record has a unique segment number, which can be used to order the records correctly, a total segment number which contains the number of segments for this parameter value, and a total value length field, which contains the overall length of the segment lengths added together.

You must ensure that the system initialization parameters to be retrieved were valid for the CICS region at CICS startup. The retrieval operation can behave inconsistently if invalid parameter values have been corrected from the console at startup. Some parameters will show the corrected values while others will display their original SIT values. Additionally for some corrected values, the system parameter source location might be returned as the console instead of the original source location.

This CMCI request retrieves the values of the control tables suffixes set to for region REGION in the CICSplex MYPLEX.

```
/CICSSystemManagement/CICSSystemParameter/<MYPLEX>/<REGION>?PARAMETER=PARMSRCE(COMBINED)  
%20PARMTYPE(SIT)&CRITERIA=KEYWORD%3D++T%20AND%20NOT%20KEYWORD%3DMXT
```

Chapter 19. CICS startup

This chapter describes how to start up CICS in the CICS region. Depending on your system environment, you can start the CICS job from a procedure by using the **START** command, or you can submit the CICS startup job stream through the internal reader.

About this task

This chapter gives an example of each of these methods. For an example of a batch job that you can submit through the internal reader, see “A sample CICS startup job” on page 321. “A sample CICS startup procedure” on page 341 gives an example of a cataloged procedure suitable for starting CICS as a started task.

When you run the startup job, you start a process called **CICS system initialization**. This process must finish before you run any transactions. Completion of CICS initialization is shown by the following message at the system console:

```
DFHSI1517 - applid: Control is being given to CICS.
```

CICS initialization involves many activities, some of which are:

- Obtaining the required storage for CICS execution from the private area in the CICS address space, above and below the 16MB line.
- Setting up CICS system parameters for the run, as specified by the system initialization parameters.
- Loading and initializing the CICS domains according to the start option specified by the **START=** system initialization parameter.
- Loading the CICS nucleus with the required CICS modules.
- Installing CICS resource definitions by:
 - Loading, from the CSD, the groups of resources specified by the **GRPLIST=** system initialization parameter
 - Loading the control tables specified by system initialization parameters
- If **XRF=YES** is specified, signing on to the CICS availability manager (CAVM) to check that it is possible to continue initialization and perform the role requested, that is, as an active or alternate CICS region.
- Opening the data sets necessary for initialization, including any needed for backout if the previous run of your CICS region was not shut down normally. Data sets may also be opened for backout even after a successful shutdown, if they had suffered backout failures before the CICS shutdown, because failed backouts are retried at emergency restart.
- Opening BSAM sequential devices as required in the terminal control table (TCT).

Also, if you are operating CICS with CICS recovery options, backout procedures may be used to restore recoverable resources to a logically consistent state. Briefly, backout occurs if you start CICS in one of the following ways:

- With **START=AUTO** and CICS detects that the previous shutdown was immediate or uncontrolled. With **SDTRAN**, an immediate shutdown does not always leave in-flight units of work to be backed out. Also, even if there were

no in-flight UOWs, it is possible (although rare) that there were backout-failed UOWs for which backout will be retried.

- With START=STANDBY and XRF=YES, and a takeover occurs.

For background information about backout, and recovery and restart, see the *CICS Recovery and Restart Guide*.

In the final stages of initialization, a set of programs can be executed as specified in a program list table (PLT). You specify the suffix of the PLT you want by means of the PLTPI parameter in the SIT. Initialization PLT programs run under control of a CICS task in CICS key. Their storage is in CICS-key storage, and protected from overwriting by other transactions.

The PLTPI resource managers start up in the following sequence when specified in the system initialization table:

1. CPSMCONN=CMAS
2. CPSMCONN=LMAS
3. CPSMCONN=WUI
4. DBCTLCON=YES
5. DB2CON=YES
6. MQCON=YES

This sequence only applies to the above system initialization parameters and not for PLTPI programs in general.

For more information about storage protection, see “Storage protection” on page 334. For programming information about writing PLT programs, see the *CICS Customization Guide*.

If you are running CICS with DB2, you can specify the DB2 subsystem ID to be used at PLT startup by the **INITPARM** system initialization parameter, as follows:

```
INITPARM=(DFHD2INI='yyyy')
```

where yyyy is the 4-character DB2 subsystem ID. The value must conform to MVS JCL rules about special characters. You cannot use the ID of a data sharing group of DB2 subsystems on the **INITPARM** system initialization parameter — you must specify the ID of a single DB2 subsystem. If you want to use the **INITPARM** system initialization parameter to specify a DB2 subsystem, leave blank both the DB2GROUPLD and the DB2ID in the installed DB2CONN definition. An ID specified in either of these attributes of the DB2CONN definition overrides an ID specified on the **INITPARM** system initialization parameter.

Using the sample startup job stream

You can use the sample job control statements to initialize a CICS region.

About this task

The sample job stream shown in “A sample CICS startup job” on page 321 specifies START=AUTO and XRF=YES to start an active CICS region. The notes that follow the sample explain which statements are unique to XRF and can therefore be omitted if you want to run with XRF=NO.

The sample startup job stream is based on the system initialization parameters contained in the CICS-supplied sample table, DFHSIT6\$.

For more information about the DD statements in this job stream that are needed by CICS and IMS, see the appropriate section in Part 1, “Defining data sets,” on page 1.

JCL similar to that in “A sample CICS startup job” is supplied as a sample startup procedure, DFHSTART, in the CICSTS41.CICS.SDFHINST library. You can use the DFHSTART procedure to start the CIC regions, or as a basis for your own startup procedures. For information about the DFHSTART procedure, see the *CICS Transaction Server for z/OS Installation Guide*.

A sample CICS startup job

You can use the sample CICS startup job to help you determine and adjust values needed for CICS startup parameters.

The EXEC statement contains the **REGION** parameter to define the size of CICS region. In this example, the value is set to 240, requesting MVS to allocate to the job all 16 MB of private storage below the 16 MB line and an extended region size of 240 MB.

```

/*
/** 1 The JOB statement
//CICSRUN JOB accounting info,name,CLASS=A,
// MSGCLASS=A,MSGLEVEL=(1,1),NOTIFY=userid
/**
/** 2 The JOBPARM statement
/*JOBPARM SYSAFF=sysid
/*
//*****
//***** EXECUTE CICS *****
//*****
/**
/** 3 The EXEC CICS=DFHSIP statement
//CICS EXEC PGM=DFHSIP,REGION=240M,
/** 4 System initialization parameters specified on PARM parameter
// PARM=('SIT=6$',
// 'DSALIM=6M,EDSALIM=120M',
// 'RENTPGM=PROTECT,STGPROT=YES',
// 'START=AUTO,SI')
/**
/** 5 System initialization parameters specified on the SYSIN data set
//SYSIN DD *
GRPLIST=(DFHLIST,userlist1,userlist2),
LPA=YES,
APPLID=CICSHTH1,
DFLTUSER=CICSUSER, The default user ID
MXT=30, Maximum number of user tasks is 30
INITPARM=(DFHDBCON='01',DFHD2INI=('MYDB')),
Pass DFSPZP01 suffix to DBCTL connect program
Connect to DB2 subsystem MYDB
ISC=YES, Include intersystem communication program
IRCSTRT=YES, Start interregion communication
.END
/*
/** 6 The STEPLIB library
//STEPLIB DD DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR
// DD DSN=CICSTS41.CICS.SDFJAUTH,DISP=SHR
// DD DSN=CEE.SCEERUN2,DISP=SHR
// DD DSN=CEE.SCEERUN,DISP=SHR
/**
/** 7 The CICS library (DFHRPL) concatenation
//DFHRPL DD DSN=CICSTS41.CICS.SDFHLOAD,DISP=SHR
/** Your application library
// DD DSN=your.prog.library,DISP=SHR
/** Your CICS control tables library
// DD DSN=your.table.library,DISP=SHR
/** The Language Environment run-time data sets
// DD DSN=CEE.SCEECICS,DISP=SHR
// DD DSN=CEE.SCEERUN2,DISP=SHR
// DD DSN=CEE.SCEERUN,DISP=SHR
/** The Debug Tool runtime library
// DD DSN=EQA.SEQAMOD,DISP=SHR
/** 7a The DB2 load library
// DD DSN=SYS2.DB2.SDSNLOAD,DISP=SHR
/** 8 Auxiliary temporary storage data sets
//DFHTEMP DD DSN=CICSTS41.CICS.CNTL.CICSHTH1.DFHTEMP,DISP=SHR
/**
/** 9 Intrapartition data sets
//DFHINTRA DD DSN=CICSTS41.CICS.CNTL.CICSHTH1.DFHINTRA,DISP=SHR
/**
/** 10 The auxiliary trace data sets
//DFHAUXT DD DSN=CICSTS41.CICS.CICSHTH1.DFHAUXT,DISP=SHR
//DFHBUXT DD DSN=CICSTS41.CICS.CICSHTH1.DFHBUXT,DISP=SHR
/**

```

Figure 31. CICS startup job stream


```

//* 11 Extrapartition data sets
//DFHCXRF DD SYSOUT=*
//LOGUSR DD SYSOUT=*,DCB=(DSORG=PS,RECFM=V,BLKSIZE=136)
//MSGUSR DD SYSOUT=*,DCB=(DSORG=PS,RECFM=V,BLKSIZE=140)
//COUT DD SYSOUT=*,DCB=(DSORG=PS,RECFM=V,BLKSIZE=137)
//CEEMSG DD SYSOUT=A
//CEEOUT DD SYSOUT=A
//*
//* 12 The CICS local catalog data set
//DFHLCD DD DSN=CICSTS41.CICS.CICSHTH1.DFHLCD,DISP=SHR
//* 13 The CICS global catalog data set
//DFHGCD DD DSN=CICSTS41.CICS.CICSHTH1.DFHGCD,DISP=SHR,
// AMP=('BUFND=33,BUFNI=32,BUFSP=1114112')
//*
//* 14 The CAVM data sets - XRF
//DFHXMSG DD DSN=CICSTS41.CICS.CNTL.CICSHTH1.DFHXRMSG,DISP=SHR
//DFHXCTL DD DSN=CICSTS41.CICS.CNTL.CICSHTH1.DFHXRCTL,DISP=SHR
//*
//* 15 The transient data destination data set (CXRF)
//DFHCXRF DD SYSOUT=A
//*
//* 16 The CICS transaction dump data sets
//DFHDMPA DD DSN=CICSTS41.CICS.CICSHTH1.DFHDMPA,DISP=SHR
//DFHDMPB DD DSN=CICSTS41.CICS.CICSHTH1.DFHDMPB,DISP=SHR
//*
//* 17 MVS system dump data sets
//SYSABEND DD SYSOUT=*
//SYSMDUMP DD DSN=SYS1.SYSMDP00,VOL=SER=valid,SPACE=(CYL,(1,1)),
// DISP=OLD,UNIT=3380
//* SYSUDUMP DD SYSOUT=*
//*
//* 17a Default stdout file for C-language system services used by CICS
//SYSPRINT DD SYSOUT=*
//* 18 The CICS system definition data set
//DFHCSD DD DSN=CICSTS41.CICS.DFHCS,DISP=SHR
//*
//* 19 The CICS BTS local request queue data set
//DFHLRQ DD DSN=CICSTS41.CICS.DFHLRQ,DISP=SHR
//*
//* 20 The EJB directory and object store data sets
//DFHEJDIR DD DSN=CICSTS41.CICS.DFHEJDIR,DISP=SHR
//DFHEJOS DD DSN=CICSTS41.CICS.DFHEJOS,DISP=SHR

```

```

/** 21 The CMAC data file
//DFHMACD DD DSN=CICSTS41.CICS.DFHMACD,DISP=SHR
/**
/** 22 The CDBM group command file
//DFHDBFK DD DSN=CICSTS41.CICS.DFHDBFK,DISP=SHR
/**
/** 23 FILEA & other permanently allocated data sets
/** (The FILEA DD statement below overrides the CSD definition in
/** group DFHMROFD)
//FILEA DD DISP=SHR,
// DSN=CICSTS41.CICS.CICSHTH1.FILEA
/**
//APPLICA DD DSN=CICSTS41.CICS.CICSHTH1.APPLICA,DISP=SHR
//APPLICB DD DSN=CICSTS41.CICS.CICSHTH1.APPLICB,DISP=SHR
/**
/**
/** 24
//PRINTER DD SYSOUT=A,DCB=BLKSIZE=125,OUTLIM=0
//CARDIN DD *

\user transactions input from sequential terminal \

\CESF GOODNIGHT\
/*

```

1. The JOB statement specifies the accounting information that you want to use for this run of CICS. For example:

```

//CICSRUN JOB 24116475,userid,MSGCLASS=A,MSGLEVEL=(1,1),
// CLASS=A,NOTIFY=userid

```

2. The JES JOBPARM statement is included to identify the MVS image on which this CICS (the active CICS region) is to run. If the alternate CICS region is to run on a different MVS image, the job stream for the alternate CICS region specifies the system identifier of the other MVS image.
3. DFHSIP is the program that starts CICS initialization. CICS does not support more than one EXEC PGM=DFHSIP job step in the same MVS job.

You determine how much of the allocated private storage you want for the CICS dynamic storage areas, and how much CICS is to leave for demands on operating system storage, by setting values for the **DSALIM** and **EDSALIM** system initialization parameters. After obtaining the amount of space required for the DSAs from the total defined by the REGION parameter, the remaining storage is available to meet demands for operating system storage.

In our sample job stream, these system initialization parameters are specified in the **PARM** parameter ; See the next point.

For more details about the **REGION** parameter and CICS storage, see “Storage requirements for a CICS region” on page 333.

If you are running CICS with RACF support, see the *CICS RACF Security Guide* for information about RACF-related parameters.

4. You can use the **PARM** parameter of the EXEC statement to specify system initialization parameters as shown.

The information passed by the **PARM** parameter is limited to 100 characters. This limit includes all commas, but excludes the apostrophes delimiting the PARM strings, and excludes the opening and closing parentheses delimiting the **PARM** parameter. (Internal parentheses enclosing system initialization operands *are* included.) If 100 characters are not sufficient for the system initialization parameters that you want to provide at startup, indicate continuation by ending the PARM field with the SYSIN or CONSOLE control keywords (or SI or CN for short). If you specify SYSIN, system initialization parameters are read from the SYSIN data set; if you specify CONSOLE, CICS

prompts you to enter parameters through the console. However, if all of your runtime system initialization parameters are in the **PARM** parameter, you can end the PARM field without any control keywords, or by the END control keyword.

In our example, DFHSIT6\$ is the SIT selected, and CICS system initialization uses the values in that table, modified by the system initialization parameters supplied in the PARM field and the SYSIN data set. For this example, the following system initialization parameters are provided in the PARM parameter:

RENTPGM

Storage for the read-only DSAs, RDSA and ERDSA, is obtained from key-0, non-fetch protected storage by using the default PROTECT option on the **RENTPGM** system initialization parameter. You are recommended to specify RENTPGM=NOPROTECT for development CICS regions and RENTPGM=PROTECT for production CICS regions. For more information about the **RENTPGM** parameter, see “RENTPGM” on page 221.

STGPROT

Storage protection is obtained for this run of CICS by specifying YES on the **STGPROT** system initialization parameter. Before using this parameter, check with the *Program Directory for CICS Transaction Server for z/OS* that you have the required hardware and software. See “STGPROT” on page 238 for details about the **STGPROT** parameter itself.

START

START=AUTO is normally the type of start that you select for a production CICS system, allowing CICS to determine the class of start to be performed (warm, emergency, or cold).

If you are running CICS with XRF, START=AUTO causes CICS to initialize as an active CICS region. To request initialization of CICS as an alternate CICS region, specify XRF=YES and START=STANDBY.

For information about the types of CICS startup and shutdown in an XRF environment, see the *CICS Operations and Utilities Guide*.

SI The PARM statement ends with SI (short for SYSIN), to tell CICS to continue reading overrides from the SYSIN data set.

5.

You can include the SYSIN data set inline as part of the job stream. System initialization parameters entered in the SYSIN data set replace any, for the same keyword, that were entered in the **PARM** parameter. If you include the same parameter more than once, the *last* value read is the value used for initialization except for INITPARM. If you specify the INITPARM keyword and its parameters more than once, each one is accepted by CICS, for example:

```
* The following INITPARM parameters are for DBCTL and a user program
INITPARM=(DFHDBCON='XX,DBCON2',userprog='a,b,c')
* The following INITPARM parameter is for DB2
INITPARM=(DFHD2INI= 'DBA2')
```

Unless you explicitly code the system initialization control keyword **CONSOLE**, CICS stops reading system initialization parameters when it reaches the end of SYSIN or a .END control keyword.

In the sample job, **CONSOLE** is not coded in either PARM or SYSIN. The .END control keyword is the last entry in SYSIN, so CICS does not prompt

through the console for further system initialization parameters. After reading the SYSIN data set, CICS loads the specified SIT, applies any system initialization parameters supplied in the PARM field and the SYSIN data set, and begins the initialization process.

The SYSIN data set in our example includes several system initialization parameters:

GRPLIST

The group list defined in DFHSIT6\$ is DFHLIST, the IBM-defined list that is generated when you initialize the CSD using the DFHCSDUP INITIALIZE command. DFHLIST contains only the standard IBM-defined resource definitions required by CICS. One of the group lists specified on the **GRPLIST** system initialization parameter must contain those resource definitions required by CICS. You can do this either by including the resource definitions in one of your own group lists that you specify, or by specifying the DFHLIST explicitly, as shown. Your own group lists (userlist1 and userlist2 as shown) must contain all the resource definitions generated by your installation for your applications that are required for this CICS run. In addition, your group lists must contain any definitions required for IBM program products that you are using, such as COBOL or DB2.

LPA The SIT specifies that modules are not to be used from the link pack area; LPA=YES in SYSIN specifies that modules are to be used from the LPA in this run.

APPLID

The APPLID for this CICS region is CICSHTH1. If you want this CICS region to use VTAM for terminal access or ISC communication, define this APPLID to VTAM.

If you want this CICS region to use XRF, you must define both a generic and specific APPLID; for example, by:

```
APPLID=(CICSID,CICSHTH1),
```

CICSID is the generic APPLID of this CICS region, and CICSHTH1 is the specific APPLID of the active CICS region. With XRF=YES, the active and alternate CICS regions share the same generic APPLID, but have different specific APPLIDs.

The specific APPLID can be useful for naming those data sets that are unique; for example, dump data sets. Where required, you can use it as the second-level qualifier to distinguish the data sets of the active and alternate CICS regions.

CICSSVC

245 is the CICS type 3 SVC number installed in the LPA, and defined to MVS in an SVC Parm statement. For more information about the **CICSSVC** parameter, see "CICSSVC" on page 157.

For guidance information about installing the CICS SVC in the LPA, and defining it to MVS, see the *CICS Transaction Server for z/OS Installation Guide*.

DFLTUSER

CICSUSER is the default user ID specified to RACF. During startup, CICS tries to sign on the default user ID. If it cannot be signed on (for example, if not defined), CICS issues a message and stops CICS initialization. After the valid default user ID is signed on, its security attributes are used for all CICS terminal users who do not sign on

with the CESN transaction. If the default user ID is defined to RACF with a CICS segment, the operator attributes in that segment are also used for users who do not sign on.

MXT The maximum number of user tasks is limited to 30 for this run. For information about the tasks that are included in the **MXT** parameter, see “MXT” on page 206.

INITPARM

Passes parameters to programs. In this example, the DBCTL suffix (01) of DFSPZPxx is passed to DFHDBCON, and CICS is told to connect to the DB2 subsystem MYDB. You cannot use the ID of a data sharing group of DB2 subsystems on the **INITPARM** system initialization parameter; you must specify the ID of a single DB2 subsystem. Note that if you want to use the **INITPARM** system initialization parameter to specify a DB2 subsystem, leave blank both the DB2GROUPID and the DB2ID in the installed DB2CONN definition. An ID specified in either of these attributes of the DB2CONN definition overrides an ID specified on the **INITPARM** system initialization parameter.

ISC Include the intersystem communication program, DFHISP, in order to use interregion communication (IRC).

IRCSTRT

This CICS is running with MRO, and interregion communication is started during initialization.

6. STEPLIB is the DDNAME of the library containing the modules loaded by the operating system. DFHSIP, which is loaded from STEPLIB, must receive control in an authorized state, so each partitioned data set (library) concatenated in STEPLIB must be individually APF-authorized. In this sample job stream, the CICS authorized library is CICSTS41.CICS.SDFHAUTH. CICSTS41.CICS.SDFJAUTH is also included, which is required for Java support, and must also be APF-authorized. If you do not require Java support, do not include this library.

Placing user-written modules into an APF-authorized library might violate your security and integrity rules. The CICSTS41.CICS.SDFHAUTH library (and the CICSTS41.CICS.SDFJAUTH library, if used) must not be included in the MVS link list, so that you can protect the libraries and thereby ensure that only approved users can run the DFHSIP module. When you define your STEPLIB DD statement, remember that all other libraries concatenated with the CICSTS41.CICS.SDFHAUTH library or the CICSTS41.CICS.SDFJAUTH library must also be APF-authorized; for example, IMS.RESLIB.

APF-authorization is required because, if any of the libraries in a STEPLIB concatenation are not authorized, MVS regards all of them as unauthorized. If there is a non-authorized library in the STEPLIB concatenation, CICS fails to start up, and issues message DFHKE0101 to indicate that the DFHSIP module is not in an APF-authorized library. For information about authorizing access to CICS data sets, see the *CICS RACF Security Guide*.

The pregenerated DFHSIP module, which has been link-edited with the authorized attribute (SETCODE AC(1)), is supplied in CICSTS41.CICS.SDFHAUTH.

You must put the Language Environment runtime libraries, CEE.SCEERUN and CEE.SCEERUN2, in the STEPLIB concatenation, or the MVS link list, to run COBOL, PL/I, C and C++, and Java programs that run in a JVM, under Language Environment. The order of the SCEERUN and SCEERUN2 libraries in relation to each other is important. SCEERUN2 must appear in the

concatenation before SCEERUN. Like SDFHAUTH, SCEERUN and SCEERUN2 must be an APF-authorized library.

The CICS DB2 attachment facility has to load the DB2 program request handler, DSNAPRH. You can either place the DB2 library DSNxxx.SDSNLOAD library in the MVS link list or add it to the STEPLIB concatenation of the CICS job.

7. DFHRPL is the DD name of the library that contains modules loaded by CICS. Protect individually the partitioned data sets constituting this library to prevent unapproved or accidental modification of their contents. The DFHRPL concatenation must include the library containing your CICS application programs, shown in this example as `your.prog.library`, and your CICS control tables, shown in this example as `your.table.library`.

In this sample job stream, the CICS-supplied library is `CICSTS41.CICS.SDFHLOAD`, which must be included in the CICS DFHRPL library concatenation. The `CICSTS41.CICS.SDFHLOAD` library contains only programs that run in problem state and must not be authorized.

Language Environment runtime library requirements

You are recommended to use the services of Language Environment, which provides a common runtime environment for IBM implementations of assembler and those high-level languages (HLLs) supported by CICS, namely COBOL, PL/I, C, and C++.

To use the Language Environment runtime libraries to support all your program languages, add the Language Environment runtime libraries SCEERUN2 and SCEERUN to the DFHRPL DD concatenation. SCEERUN2 must appear ahead of SCEERUN in the concatenation. If you are running COBOL programs, also add Language Environment runtime library SCEECICS to DFHRPL. The SCEECICS library must be concatenated before the SCEERUN library. Remove any libraries that contain runtime routines from earlier versions of COBOL, PL/I, and C/C++ from the DFHRPL DD concatenation.

Debug Tool library

If you are using Debug Tool to debug CICS applications, include the Debug Tool library SEQAMOD in DFHRPL. For information about using Debug Tool with CICS, see the *CICS Application Programming Guide*.

7a DB2 requirements

Generally, you do not have to include any DB2 libraries in the DFHRPL DD statement. If you do require DB2 libraries in the DFHRPL concatenation for an application, they must be placed after the CICS libraries; for example, add SDSNLOAD in the DFHRPL to support those applications that issue dynamic calls to the DB2 message handling module, DSNTIAR, or the later DSNTIA1, both of which are shipped in SDSNLOAD. DSNTIA1 is loaded by applications programs that include the DB2 application stub DSNTIAC, which issues an **EXEC CICS LOAD** command for program DSNTIAC.

8. Define this data set if you want to save data to use later. The temporary storage queues used, identified by symbolic names, exist until explicitly deleted. Even after the originating task is deleted, temporary data can be accessed by other tasks, through references to the symbolic name under which it is stored.

For details of how to define these data sets, and for information about space calculations, see Chapter 2, "Setting up the temporary storage data set," on page 15.

If you are using temporary storage data sharing, ensure that you start the temporary storage server before it is required by the CICS regions.

For more information about the temporary storage server and temporary storage data sharing, see Chapter 22, “Setting up and running a temporary storage server,” on page 349

9. The transient data intrapartition data set is used for queuing messages and data in the CICS region. For information about how to define these data sets, and about space calculations, see “Defining the intrapartition data set” on page 21.
10. Define one or both of these sequential data sets, if you want to use auxiliary trace. If you define automatic switching for your auxiliary trace data sets, define both data sets. If you define only one data set, its DD name must be DFHAUXT.

For details of how to define these data sets, see Chapter 7, “Setting up and using auxiliary trace data sets,” on page 77.

The auxiliary trace data sets in this job stream are unique to the active CICS region, and as such are identified in our example by using the specific applid of the active CICS region (CICSHTH1) as a second-level qualifier. If you are using auxiliary trace with XRF=YES, the alternate CICS region also needs its own trace data sets. These can be identified by the specific applid of the alternate CICS region; for example, CICSHTH2.

If you allocate and catalog the auxiliary trace data sets on disk as shown in Figure 14 on page 79, you can define them to CICS in the startup job stream using the following DD statements:

```
//DFHAUXT DD DSN=CICSTS41.CICS.applid.DFHAUXT,DCB=BUFNO=n,DISP=SHR
//DFHBUXT DD DSN=CICSTS41.CICS.applid.DFHBUXT,DCB=BUFNO=n,DISP=SHR
```

If you specify BUFNO greater than 1, you can reduce I/O involved in writing auxiliary trace records. A value between 4 and 10 can greatly reduce I/O when running with auxiliary trace on.

11. LOGA and CSSL are examples of extrapartition transient data queues.
 - LOGA defines a user data set used by the CICS sample programs.
 - CSSL defines the data set used by a number of CICS services.
 - CCSO is used as an output queue only when you are running C/370™ application programs.
 - CESO is used as an error queue only when you are running application programs under Language Environment.

Sample definitions of the queues used by CICS are supplied in group DFHDCTG. DFHDCTG is unlocked, so you can alter the definitions before installation.

12. The CICS local catalog is used by the CICS domains to save some of their information between CICS runs, and to preserve this information across a cold start. The local catalog is not shared by any other CICS system. If you are running CICS with XRF, define a unique local catalog for the active CICS region and another for the alternate CICS region. For details of how to create and initialize a CICS local catalog, see Chapter 6, “Setting up the catalog data sets,” on page 65.
13. There is only one global catalog, which is passively shared by the active and alternate CICS regions.

For details of how to create and initialize a CICS global catalog, see Chapter 6, “Setting up the catalog data sets,” on page 65.

This sample job illustrates the use of the AMP parameter on the DD statement. Specifying this parameter, with its buffer subparameters, can help to improve restart and shutdown time. This example is based on the recommended DEFINE CLUSTER statements shown in Figure 11 on page 67

and the associated notes given under 4 in “Defining the global catalog” on page 65. The values given are the minimum values suitable for these parameters and must not be reduced.

14. The CAVM data sets are required when you are running CICS with XRF. They are actively shared by the active and the alternate CICS regions. For details of how to create and initialize the CICS availability data sets, see Chapter 9, “Defining the CICS availability manager data sets,” on page 87.
15. This transient data destination is used by CICS as the target for messages sent to any transient data destination before CICS has completed intrapartition transient data initialization. It is particularly necessary in an XRF environment for use in an alternate CICS region before takeover has occurred, during the period when transient data initialization is suspended. For more information about the DFHCXRF data set, see “The DFHCXRF data set” on page 24.
16. CICS records transaction dumps on a sequential data set or a pair of sequential data sets, tape or disk. The data sets must be defined with the DD names DFHDMPA and DFHDMPB, but if you define only one data set, its DD name must be DFHDMPA. CICS always attempts to open at least one transaction dump data set during initialization.

For details about how to define CICS transaction dump data sets and how they are used, see Chapter 8, “Defining dump data sets,” on page 81.

The transaction dump data sets in this job stream are unique to the active CICS region, and as such are identified by using the specific applid of the active CICS region (DBDCCIC1) as a second-level qualifier. The alternate CICS region needs its own transaction dump data sets, and these can be identified by using the specific applid of the alternate CICS region (DBDCCIC2) as a second-level qualifier.

17. Use a SYSABEND, SYSMDUMP, or SYSUDUMP DD statement to direct MVS to produce a dump.

In the sample job stream, the SYSABEND DD statement directs a formatted dump to a printer, and the SYSMDUMP DD statement saves an unformatted dump to the SYS1.SYSMDP00 data set on disk.

MVS produces the requested dump if either of the following is true:

- CICS ends abnormally.
- CICS starts to end abnormally, but system recovery procedures enable CICS to end normally.

The dump DD statements for requesting dumps are as follows:

SYSABEND DD statement

Produces a dump of user and system areas; this dump contains all the areas dumped in a SYSUDUMP plus the local system queue area (LSQA), including subpools 229 and 230, and the I/O system (IOS) control blocks for the failing task. The dump is formatted, so that it can be printed directly.

SYSMDUMP DD statement

Produces a dump of the system areas and the program address space. The dump is unformatted and machine-readable; to be used, it must be printed by the interactive problem control system (IPCS).

Note: The SYSMDUMP DD statement must specify a magnetic tape unit or a direct access device.

To write more than one SYSMDUMP dump in the same data set on tape, specify the following:

- DSNNAME=SYS1.SYSMDPxx where xx is 00 through FF. SYSMDPxx is a preallocated data set that you must initialize with an end-of-file (EOF) mark on the first record.
- DISP=SHR.

You can ask MVS to write additional dumps only if you offload any previous dump and write an EOF mark at the beginning of the SYS1.SYSMDPxx data set. To accomplish this, your MVS installation must install an exit routine for message IEA993. For information on this installation exit routine, see *z/OS MVS Installation Exits*.

SYSUDUMP DD statement

Produces a dump of user areas. The dump is formatted, so that it can be printed directly.

The dump contents are as described only when you use the IBM-supplied defaults for the dumps. The contents of these dumps can be set during MVS system initialization and can be changed for an individual dump in the ABEND macro instruction, in a CHNGDUMP command, and by a SLIP command. For details, see the *z/OS MVS Initialization and Tuning Guide*.

Dumps are optional; use a dump DD statement only when you want to produce a dump.

For information about how defining these MVS system dump data sets, and about printing dumps from them, see the *z/OS MVS JCL Reference*. For information about how to interpret dumps, see *z/OS MVS Diagnosis: Tools and Service Aids*.

17a Default stdout file for C-language system services used by CICS

The SYSPRINT statement is opened as the stdout file by C-language system service routines that are used by CICS (such as system SSL). If it is omitted, multiple sysout files might be dynamically allocated by the operating system instead.

18. The system definition file (CSD) is required by CICS to hold some resource definitions.

You might want to provide job control DD statements for the CSD. If you do, the CSD data set is allocated at the time of CICS job step initiation, and *remains allocated for the duration of the CICS job step*.

On the other hand, you might prefer to use dynamic allocation of the CSD. For dynamic allocation, do *not* specify a DD statement for the CSD. Specify the data set name (DSNAME) and the data set disposition (DISP) either in a SET FILE command or in the System Initialization (as parameters CSDDSN and CSDDISP). CICS uses the DSNAME and DISP to allocate the file as part of OPEN processing.

For information about creating and initializing the CSD, see Chapter 5, “Setting up the CICS system definition data set,” on page 41.

If you are running CICS with XRF, particularly in a multi-MVS environment, special considerations apply to CSD sharing; these considerations are discussed in “Sharing the CSD in non-RLS mode” on page 47.

19. DFHLRQ is a file-control-managed VSAM key-sequenced data set (KSDS), used by CICS business transaction services (BTS). The IBM-supplied file resource definition for DFHLRQ is supplied in CSD group DFHCBTS, which is automatically included in DFHLIST group list when you initialize or upgrade your CSD. The file definition specifies that it is to be opened on first reference, which occurs at the end of CICS initialization when it is opened by BTS. CICS issues warning messages DFHFC0951 and DFHSH0109 if the data set is not found. Although CICS continues running without this data set, you

are recommended to define the DFHLRQ data set, even if you are not using BTS. For information about DFHLRQ, see *CICS Business Transaction Services*.

20. DFHEJDIR is a VSAM key-sequenced data set (KSDS) that contains a request streams directory, which must be shared by all the regions, listeners and AORs, in a logical EJB server. Request streams are used in the distributed routing of method requests for enterprise beans and CORBA stateless objects. DFHEJOS is a VSAM key-sequenced data set (KSDS) that contains details of stateful session beans that have been passivated. It must be shared by all the AORs in the logical EJB server.

For information about defining the EJB directory and object store, see Example JCL to define an EJB object store data set and Example JCL to define an EJB directory data set.

21. DFHCMACD is a VSAM key-sequenced data set (KSDS) that is used by the CMAC transaction to provide online descriptions of CICS messages and codes. Before its first use, it must be defined and loaded as a KSDS data set. Chapter 12, "Defining the CMAC messages data set," on page 117 describes DFHCMACD in greater detail.
22. DFHDBFK is a VSAM key-sequenced data set (KSDS) that is used by the CDBM transaction to store Group commands. Before its first use, it must be defined as a KSDS data set. The DFHDBFK DD statement is required only if you intend to use the command storage functions of the CDBM transaction. Defining the CDBM GROUP command data set describes DFHDBFK in greater detail.
23. You might want to provide job control DD statements for those user files that are defined in the CSD (if you are using RDO) or in a file control table (for BDAM files only). If you do, the data sets are allocated at the time of CICS job step initiation, and *remain allocated for the duration of the CICS job step*. FILEA, the distributed library containing the data for the sample application programs, is included in this startup job stream as an example of direct allocation by job control statement.

Alternatively, you might prefer to take advantage of the CICS dynamic allocation of files. For dynamic allocation, do not specify a DD statement for the CSD. CICS then uses the full data set name as specified in the **DSNAME** parameter of the FILE resource definition (up to 44 characters), together with the **DISP** parameter, to allocate the file as part of OPEN processing. This form of dynamic allocation applies equally to files that are defined to be opened explicitly, and those that are to be opened on first reference by an application. For more information about file opening, see Chapter 10, "Defining user files," on page 93. For information about the parameters that you can code on FILE resource definitions, see the *CICS Resource Definition Guide*.

The card reader/line printer (CRLP) simulated terminals shown in the sample job stream are defined in the sample TCT (not used in this startup job). See the copy member DFH\$TCTS, in CICSTS41.CICS.SDFHSAMP, for the source statements that you need for such devices.

24. For sequential devices, the last entry in the input stream can be CESF GOODNIGHT\ to provide a logical close, and quiesce the device. However, if you close a device in this way, the receive-only status is recorded in the warm keypoint at CICS shutdown. So the terminal is still in RECEIVE status in a subsequent warm start, and CICS does not then read the input file.

Note the end-of-data character (the "\ " symbol) at the end of each line of the sample.

Storage requirements for a CICS region

This section describes considerations about CICS storage requirements.

When you submit your CICS job, an MVS region is allocated for the execution of CICS. You determine the overall size of the region by coding the REGION parameter, either on the JOB card or on the EXEC PGM=DFHSIP statement. If you specify the REGION parameter on the JOB statement, each step of the job executes in the requested amount of space. If you specify the REGION parameter on the EXEC statements in a job, each step executes in its own amount of space. Use the EXEC statement REGION parameters when different steps need greatly different amounts of space; for example, when using extra job steps to print auxiliary trace data sets after CICS has shut down (as in the DFHIVPOL installation verification procedure).

The available address space allocated above and below the 16MB line is determined by the value you code on the REGION parameter, but subject to any limits imposed by the IEFUSI exit routine. For details of using the IEFUSI exit routine, see the *OS/390 MVS Installation Exits* manual.

The transaction isolation facility increases the allocation of some virtual storage above the 16MB boundary for CICS regions that are running with transaction isolation active.

If you are running with transaction isolation active, CICS allocates storage for task-lifetime storage in multiples of 1MB for user-key tasks that run above the 16MB boundary. (1MB is the minimum unit of storage allocation above the line for the EUDSA when transaction isolation is active.) However, although storage is allocated in multiples of 1MB above the 16MB boundary, MVS paging activity affects only the storage that is used (referenced), and unused parts of the 1MB allocation are not paged.

If you are running without transaction isolation, CICS allocates user-key task-lifetime storage above 16MB in multiples of 64KB.

The subspace group facility uses more real storage, as MVS creates for each subspace a page and segment table from real storage. The CICS requirement for real storage varies according to the transaction load at any one time. As a guideline, each task in the system requires 9KB of real storage, and this should be multiplied by the number of concurrent tasks that can be in the system at any one time (governed by the MXT system initialization parameter).

However, automatic DSA sizing removes the need for accurate storage estimates, with CICS dynamically changing the size of DSAs as demand requires.

For details of how MVS allocates the storage requested by your REGION parameter, see the *OS/390 MVS JCL Reference* manual. For ease of reference, examples of possible size ranges are given here.

REGION=0K or 0MB

MVS gives the job all the available private storage below and above the 16MB line. The resulting size of the region below and above 16MB is unpredictable.

REGION=>0 and ≤16M

MVS establishes the specified value as the size of the private area below the 16MB line, and a default extended region size of 32MB. If the region size specified is not available, the job step terminates abnormally.

The amount of private storage available below the line varies from installation to installation, and possibly from IPL to IPL, because of the installation-dependent parameters you use to generate and IPL your MVS system. Typically, the amount of common storage required by MVS is 7MB or more, leaving you with a potential private storage area of less than 9MB.

REGION=>16M and ≤32M

MVS gives the job all the storage available below 16MB, the size of which is unpredictable, and a default extended region size of 32MB.

REGION >32M and ≤2047M

MVS gives the job all the storage available below 16MB, the size of which is unpredictable, and the extended region size as specified. If the region size specified is not available above 16 megabytes, the job step terminates abnormally.

When calculating the size of your CICS region, allow for the following areas of private storage in the CICS region, above and below the 16MB line:

- Storage that CICS reserves at the start of CICS initialization for exclusive use by the CICS kernel.
- Storage that the CICS storage manager reserves for the dynamic storage areas, which you specify on the system initialization parameters, DSALIM and EDSALIM.
- Storage remaining in the private area, after the kernel and DSA requirements have been allocated, to meet additional CICS storage demands from the operating system, for control blocks and buffers for the various access methods.
- If you are using CICS data tables, the amount of storage left after deducting the CDSA requirements must be enough for the records you want to include in the data tables.

For information about CICS storage requirements, and the effect on CICS performance, see the *CICS Performance Guide*. For guidance information about virtual storage management in an MVS environment, see the *OS/390 MVS Initialization and Tuning Guide*.

Paging requirements should be reviewed when you are defining large STG_SIZE values.

Storage protection

CICS uses the storage protection facilities available in the operating system to prevent CICS code and control blocks from being overwritten accidentally by your user application programs.

This is done by allocating separate storage areas (with separate storage keys) for your user application programs, and for CICS code and control blocks. Access to a storage area is not permitted unless the access key matches the key for that storage area.

The storage allocated for CICS code and control blocks is known as **CICS-key** storage, and the storage allocated for your user application programs is known as **user-key** storage. In addition to CICS-key and user-key storage, CICS can also use **key-0 storage** for separate dynamic storage areas below and above the 16MB boundary called the **read-only** DSAs (RDSA and ERDSA). The ERDSA is used for eligible re-entrant CICS and user application programs link-edited with the RENT and RMODE(ANY) attributes. The RDSA is used for eligible re-entrant CICS and

user application programs link-edited with the RENT and RMODE(24) attributes. The allocation of key-0 storage for the read-only DSAs is from the same storage limit as the other DSAs, as specified by the DSALIM and EDSALIM system initialization parameters.

Use of the storage protection facilities are optional. You can enable them by coding options on new storage protection system initialization parameters. Between them, these new parameters enable you to define or control:

- The storage key for the common work area (CWAKEY)
- The storage key for the terminal control table user areas (TCTUAKEY)
- A storage protection global option (STGPROT)
- A read-only program storage key option (RENTPGM)
- A transaction isolation option (TRANISO)

To help you get started, CICS provides DFHSIT\$\$, a default system initialization table. This default table is supplied in the CICSTS41.CICS.SDFHSAMP library in source form, and you can modify this to suit your own requirements. When assembled and link-edited, DFHSIT\$\$ becomes the unsuffixed DFHSIT, which is supplied in pregenerated form in CICSTS41.CICS.SDFHAUTH.

The common work area

The common work area (CWA) is an area of storage within your CICS region that any user application can access.

You determine the size of this work area by means of the WRKAREA system initialization parameter, which allows you to specify sizes up to 3584 bytes. If you omit the WRKAREA parameter, CICS allocates a 512-byte CWA by default. You specify the storage key for the CWA on the CWAKEY parameter.

Because this work area is available to all transactions in a CICS region, you should ensure that the storage key is appropriate to the use of the CWA by all transactions. If there is only one transaction that runs in user key, and which requires **write** access, you must specify user-key storage for the CWA, otherwise it will fail with a storage protection exception (an ASRA abend). CICS obtains user-key storage for the CWA by default, and you must review the use of this storage by all programs before you decide to change it to CICS key.

It is possible that you might want to protect the CWA from being overwritten by applications that should not have write access. In this case, provided all the transactions that legitimately require write access to the CWA run in CICS key, you can specify CICS-key storage for the CWA.

See page CWAKEY for details of how to specify the CWAKEY parameter.

The terminal control table user areas

A terminal control user area (TCTUA) is an optional storage area associated with a terminal control table terminal entry (TCTTE), and is available for application program use.

For VTAM terminals, you specify that you want a TCTUA by means of the USERAREALEN attribute on the TYPETERM resource definition. The USERAREALEN attribute determines the TCTUA sizes for all terminals that reference the TYPETERM resource definition.

For sequential terminals, definitions are added to the terminal control table (TCT), and sizes are defined by means of the **TCTUAL** parameter on the DFHTCT TYPE=TERMINAL and TYPE=LINE entries. For information about the TCTUAL parameter, see the *CICS Resource Definition Guide*.

You specify the storage key for the TCTUAs globally for a CICS region by the **TCTUAKEY** system initialization parameter. By default, CICS obtains user-key storage for all TCTUAs.

You must review the use of TCTUAs in your CICS regions, and specify CICS key only for TCTUAs when you are sure that this is justified. If you specify CICS-key storage for TCTUAs, no user-key applications can write to any TCT user areas.

See “TCTUAKEY” on page 247 for details of how to specify the **TCTUAKEY** parameter.

The storage protection global option

You can control whether your CICS region uses storage protection by specifying the STGPROT parameter. By default, CICS does not use storage protection, and all applications run in the same key as CICS, as in earlier releases.

The default option is suitable for pure terminal-owning regions (TORs) and data-owning regions (DORs) that do not execute user transactions. If you want storage protection in a CICS region, you must specify this on the STGPROT parameter.

See “STGPROT” on page 238 for details of how to specify the STGPROT parameter.

Transaction isolation

CICS transaction isolation builds on CICS storage protection, enabling user transactions to be protected from one another. You can specify transaction isolation globally for a CICS region on the TRANISO (and STGPROT) system initialization parameter.

In addition to being able to specify the storage and execution key individually for each user transaction, you can specify that CICS is to isolate a transaction's user-key task-lifetime storage to provide transaction-to-transaction protection. You do this by using the ISOLATE option of the TRANSACTION resource definition.

For an overview of transaction isolation, and CICS' use of MVS subspaces, see the *CICS Performance Guide*.

The read-only storage override option

CICS obtains storage for the read-only DSAs (RDSA and ERDSA) from MVS read-only storage.

CICS loader automatically loads eligible modules into the RDSA and ERDSA; that is, if they are link-edited with the RENT attribute, and for the ERDSA with RMODE(ANY). If you do not want such modules to be loaded into read-only storage (perhaps because you are using a development aid package that sets break points in your application programs) you can override the selection of read-only storage for the RDSA and ERDSA by specifying NOPROTECT on the RENTPGM system initialization parameter.

Note: When you specify RENTPGM=NOPROTECT, CICS still allocates separate read-only DSAs, but obtains CICS key-storage for the RDSA and ERDSA instead of read-only storage.

You are recommended to specify RENTPGM=NOPROTECT for development regions only, and to specify RENTPGM=PROTECT for production CICS regions. See “RENTPGM” on page 221 for details of how to specify the RENTPGM parameter.

The dynamic storage areas and associated storage cushions

The type of storage for the read-only DSAs is controlled by the RENTPGM system initialization parameter.

CICS supports a number of separate dynamic storage areas (DSAs). These dynamic storage areas reside either below the 16MB boundary, between the 16MB and 2GB boundaries or above the 2GB boundary. Each of the eight DSAs below the 2GB boundary has its own “storage cushion”. CICS always allocates eight separate DSAs below the 2GB boundary, even if you are running without storage protection (either because the necessary hardware or MVS support is not available, or because you switch off storage protection by means of the STGPROT parameter). If you are running with STGPROT=NO, CICS allocates the DSAs that are controlled by the STGPROT parameter from CICS-key storage (the same storage key as in earlier releases). However, because the CICS and user DSAs are physically separate, it makes the overwriting of CICS storage less likely, even if the DSAs are all in the same storage key.

GCDSA is the dynamic storage area that resides above the 2GB boundary. Storage is handled differently above the 2GB boundary, and is covered in the topic Dynamic storage above the 2GB boundary.

The eight DSAs below the 2GB boundary are as follows:

CDSA

The CICS DSA, allocated below the 16MB boundary, always from CICS-key storage.

RDSA

The read-only DSA, allocated below the 16MB boundary from either read-only storage or CICS-key storage depending on the RENTPGM parameter.

SDSA

The shared DSA, allocated below the 16MB boundary from either user-key or CICS-key storage depending on the STGPROT parameter.

UDSA

The user DSA, allocated below the 16MB boundary from either user-key or CICS-key storage depending on the STGPROT parameter.

ECDSA

The extended CICS-key DSA, allocated above the 16MB boundary, always from CICS-key storage.

ERDSA

The extended read-only DSA, allocated above the 16MB boundary, either from read-only storage or CICS-key storage, depending on the RENTPGM parameter.

ESDSA

The extended shared DSA, allocated above the 16MB boundary from either user-key or CICS-key storage depending on the STGPROT parameter.

EUDSA

The extended user DSA, allocated above the 16MB boundary, from either user-key or CICS-key storage depending on the STGPROT parameter.

Table 21 shows the type of storage allocated according to the system initialization parameters specified.

You specify the overall limits within which CICS can allocate the DSAs by the DSALIM and EDSALIM system initialization parameters (for the DSAs below and above the 16MB boundary respectively). Within these limits, CICS dynamically controls the sizes of the individual DSAs and their associated cushions. Also, you can vary these overall limits dynamically, by using either the CEMT SET SYSTEM command or an EXEC CICS SET SYSTEM command.

Table 21. Controlling the storage key for the dynamic storage areas

| Dynamic storage area | STGPROT= NO | STGPROT= YES | RENTPGM= PROTECT | RENTPGM= NOPROTECT |
|----------------------|------------------|------------------|------------------|--------------------|
| CDSA | CICS key | CICS key | N/A ¹ | N/A ¹ |
| RDSA | N/A ² | N/A ² | Read-only key-0 | CICS key |
| SDSA | CICS key | User key | N/A ¹ | N/A ¹ |
| UDSA | CICS key | User key | N/A ¹ | N/A ¹ |
| ECDSA | CICS key | CICS key | N/A ¹ | N/A ¹ |
| ESDSA | CICS key | User key | N/A ¹ | N/A ¹ |
| ERDSA | N/A ² | N/A ² | Read-only key-0 | CICS key |
| EUDSA | CICS key | User key | N/A ¹ | N/A ¹ |
| GCDSA | N/A ³ | N/A ³ | N/A ³ | N/A ³ |

Note: 1. Not applicable. The RENTPGM option has no effect on these DSAs, for which the storage key is determined only by the STGPROT parameter.
Note: 2. Not applicable. The STGPROT option has no effect on the read-only DSAs, for which the storage key is determined only by the RENTPGM parameter.
Note: 3. Not applicable. Dynamic storage areas above the 2GB boundary are not allocated from key storage.

The storage cushions

CICS dynamically tunes the size of the DSA storage cushions as necessary, within the limits set by the DSALIM and EDSALIM system initialization parameters. However, if the amount of storage available for the storage cushions becomes too small, an SOS condition can still occur.

CICS reserves amounts of storage in the dynamic storage areas (DSAs) below the 2GB boundary (below-the-bar) for use when processing storage stress conditions. Each reserved area, which consists of contiguous virtual storage, is called a "storage cushion." A storage stress condition occurs when CICS cannot satisfy a GETMAIN request, or can satisfy it only by using some of the cushion storage even when all programs that are eligible for deletion, and not in use, have been deleted. This may lead to a short-on-storage condition, if CICS cannot rectify the stress condition.

Reserved areas are not managed by CICS for storage above the 2GB boundary.

Effects:

In a storage stress condition, the cushion mechanism can avert a storage deadlock condition.

This prevents CICS taking on additional work by stopping most of the soliciting for new input messages. For information on the effects of stress conditions, see the *CICS Performance Guide*.

CICS sets the storage stress condition if:

- After any successful GETMAIN, the number of unallocated dynamic storage pages remaining is less than the cushion size. This is shown in the storage statistics as “Times cushion released”.
- CICS cannot satisfy an unconditional GETMAIN because there is no contiguous area large enough for it. This is shown in the storage statistics as “Times request suspended”.

When a storage stress situation exists, the loader domain attempts to alleviate it by releasing the main storage for programs with no current user. If this fails, a short-on-storage condition is indicated, and a message is issued at the console.

While the SOS condition is set, acquisition of new input message areas is prevented, and all ATTACH requests from CICS system modules are deferred.

Optimizing dynamic storage areas

You can help CICS to optimize its use of the dynamic storage areas and their storage cushions.

It is recommended that you do the following:

- Avoid using large GETMAIN requests.
The storage cushion is a contiguous block of storage of fixed size, and therefore might be able to satisfy a request for a large contiguous block of storage.
- Minimize the number of resident programs.

If storage cushion releases occur frequently, you must find out why. Reduce the maximum number of user tasks (using the **MXT** system initialization parameter) to reduce the number of tasks using main storage. You might have to alter your application programs so that they do not issue large GETMAINS.

Only transactions defined as SPURGE(YES) and with a DTIMOUT value can be purged during an SOS condition if they have been waiting for storage for longer than the DTIMOUT value. If such transactions are too few and if storage becomes totally deadlocked, the system can stall.

How implemented

CICS allocates the initial size of the storage cushions for the DSAs from the overall storage limits defined by the DSALIM and EDSALIM system initialization parameters. CICS dynamically tunes the sizes of the DSAs and their storage cushions within these limits.

For descriptions of the DSALIM and EDSALIM system initialization parameters, see page DSALIM and EDSALIM respectively.

You can change the overall storage limits while CICS is running by means of a CEMT SET SYSTEM command or an EXEC CICS SET command.

How monitored

Storage stress conditions are notified in the storage statistics (“Times cushion released” and “Times request suspended”). A storage stress condition may not cause an SOS condition; CICS may be able to alleviate the condition. However, storage stress conditions are costly, and should be avoided.

The SOS condition is notified in the dynamic storage area statistics (“times went short on storage”), and is made apparent to the terminal user by external effects such as ceasing of polling and transaction initiation, and prolonged response times. In addition, a message is displayed on the operating system console when the short-on-storage (SOS) indication is detected. The SOS message, DFHSM0131 or DFHSM0133, indicates that:

- The amount of free space in a dynamic storage area is less than needed, and the associated DSA cannot be enlarged further (because the DSA limit has been reached).
- There are currently suspended GETMAIN requests waiting for large enough areas of contiguous storage to become available.

If there is insufficient storage in the relevant DSA limit to satisfy a GETMAIN request, the request is either queued (for unconditional requests) or a return code is given (for conditional requests).

Coding conventions for DSA limits

You can specify the size of the DSA limits as a number of bytes, a number of kilobytes, or a number of megabytes.

Use the letter K or M as a suffix to indicate whether the value represents a whole number of kilobytes, or a number of megabytes. For example, 2 MB can be coded as either 2048K or 2M. (1 KB = 1024 bytes; 1 MB = 1024 KB = 1048576 bytes.)

If the value you specify is not a multiple of 256 KB for DSALIM, or 1 MB for EDSALIM, CICS rounds up the value to the next multiple.

You cannot specify fractions of megabytes: you must code sizes in bytes or kilobytes. Some examples are shown in Table 22:

Table 22. Examples of DSA limit values in bytes, kilobytes and megabytes

| Coded as: | | | | | |
|-----------|---------|---------|---------|---------|---------|
| bytes | 2097152 | 3145788 | 3670016 | 4194304 | 4718592 |
| kilobytes | 2048K | 3072K | 3584K | 4096K | 4608K |
| megabytes | 2M | 3M | - | 4M | - |

For information about estimating the size of the dynamic storage areas, see the *CICS Performance Guide*.

The sample statistics program, DFH0STAT

You can use the statistics sample program, DFH0STAT, to help you determine and adjust values needed for CICS storage parameters, for example the size of the DSALIM and EDSALIM system initialization parameters.

The program produces a report showing critical system parameters from the CICS Dispatcher, an analysis of the CICS Storage Manager and Loader statistics, and an overview of the MVS storage in use. The program demonstrates the use of EXEC

CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of your CICS. You can use the sample program as provided or modify it to suit your needs.

For more information on the statistics sample program, DFH0STAT, see the *CICS Performance Guide*.

A sample CICS startup procedure

As an alternative to submitting a batch job to the MVS internal reader, you can use the MVS START command to start CICS as a started task. Using this method, your startup job stream must be coded according to the rules for coding procedures, and the procedure must be installed in an MVS procedure library.

About this task

If you intend to start CICS with the **START** command you must either:

- Give the MVS started task procedure a name different from the subsystem name in IEFSSNaa (default 'CICS'), or
- Issue the start command with the parameter SUB=JES2 or SUB=JES3 as appropriate.

You can use the following form of the **MVS START** command to start a job from the console:

```
S|START procname[.identifier] [,SUB=subsystemname] [,keyword=option  
[,keyword=option] . . .]
```

where:

procname

The name of the cataloged procedure that defines the job to be started.

identifier

The name you choose to identify the task.

SUB=subsystemname

The name of the subsystem that is to select the job for processing. If you omit this parameter, the primary job entry subsystem is used.

keyword=option

Any appropriate keyword to override the corresponding parameter in the procedure. You can use this parameter to override symbolic parameters defined in the cataloged procedure.

For guidance information about the complete syntax of the **START** command, and all the keywords and options you can use, see the *OS/390 MVS System Commands* manual.

To start CICS, you only need to code **procname.identifier,keyword(s)=option**.

For example:

```
START DFHSTART.CICSA,SIP=T,REGNAME1=IDA,REGNAM2=IDA
```

In this example of the **MVS START** command:

- DFHSTART is the name of the CICS-supplied cataloged startup procedure.
- CICS is being started with a task ID of CICSA.

- SIP is the suffix of the DFH\$SIPx member in the SYSIN data set, CICSTS41.CICS.SYSIN, to be used by this CICS region.
- REGNAM1 and REGNAM2 are qualifiers added to the CICS system data sets specified in the procedure (for example, CICSTS41.CICS.CICSHTH1.DFHTEMP) to identify uniquely the data sets for this CICS region. REGNAM1 is set to the same value as REGNAM2 for an XRF active CICS region or an MRO region.

For information about the DFHSTART procedure, see the the *CICS Transaction Server for z/OS Installation Guide*.

If you are running CICS with RACF, you must associate the cataloged procedure name with a suitably authorized RACF user through the RACF table, ICHRIN03. For information about this association, see the *CICS RACF Security Guide*.

Chapter 20. Preparing CICS for using debugging tools

Before application programmers can use certain debugging tools with CICS, you will need to perform some system definition tasks. This chapter describes those tasks.

It contains the following topics:

- “Preparing your CICS region for debugging”

Preparing your CICS region for debugging

Before application programmers can use certain debugging tools with CICS, you must configure your CICS region accordingly.

About this task

You can prepare your CICS region for debugging application programs using the following tools:

- Debug Tool, for compiled language application programs (programs written in COBOL, PL/I, C, C++), and for Language Environment-enabled Assembler subroutines).
- Remote debugging tools (for compiled language application programs, Language Environment-enabled Assembler Subroutines, and Java programs). Note that for compiled language programs, and Assembler subroutines, Debug Tool is used as the debugging server.

This topic does not apply to other debugging tools, such as the CICS Execution Diagnostic Facility (CEDF).

Since you will need to restart the region, it is advisable to plan which regions will be used for debugging applications.

- It is unlikely that application programs will be debugged in your production regions, especially if the applications are well established and known to be reliable; it is more likely that debugging will take place in regions which are used to develop and test new applications.

To prepare your CICS region for debugging:

Procedure

1. If you plan to use the region for debugging compiled language programs, include the Debug Tool library SEQAMOD in the DFHRPL concatenation in your CICS startup JCL. For more information, see “Using the sample startup job stream” on page 320.
2. Create the *debugging profile data sets*. For more information, see Chapter 15, “Setting up the debugging profiles data sets,” on page 125.
3. Include the resource definition for the debugging profile file in a resource definition list that is named in the “GRPLIST” on page 188 system initialization parameter.
4. Optionally, specify the following value for the “DEBUGTOOL” on page 170 system initialization parameter:
DEBUGTOOL=YES

If you do not specify `DEBUGTOOL=YES`, you can enable the region for debugging when it is running:

- To enable the region for debugging from a program, use the `EXEC CICS SET SYSTEM DEBUGTOOL` command
- To enable the region for debugging from the master terminal transaction, use the `CEMT SET SYSTEM DEBUGTOOL` command

Enabling the region for debugging when it is running is recommended for regions which are not normally used for debugging. When debugging is complete, you can disable the region for debugging, using the same commands.

5. Define and install Debug Tool's resource definitions. They are located in member `EQACCS`D in Debug Tool's `SEQASAMP` data set. For more information, see the *Debug Tool for z/OS and OS/390 User's Guide*.

Part 3. Initializing CICS data sharing servers

To initialize CICS data sharing servers, you must set up AXM system services and define the appropriate data sharing server.

CICS data sharing servers support:

- Temporary storage data sharing
- Coupling facility data tables
- Named counters

Users implementing CICSplex SM sysplex optimised workload management will require a region status server to be configured. This server is a bespoke type of CFDT server into which CICS regions broadcast generic system status data which is subsequently interrogated by CICSplex SM for making dynamic routing decisions. The table structure managed by this server may not be modified, adjusted, or reconfigured for user application utilization.

- Chapter 21, "Defining and starting AXM system services," on page 347 describes how to define and start AXM system services
- Chapter 22, "Setting up and running a temporary storage server," on page 349 describes how to set up and run a CICS temporary storage data sharing server
- Chapter 23, "Setting up and running a coupling facility data table server," on page 363 describes how to set up and run a CICS coupling facility data table server
- Chapter 26, "Coupling facility server operations," on page 415 gives an overview of the operations which are common to all three CICS coupling facility servers.
- Chapter 25, "Setting up and running a named counter server," on page 397 describes how to set up and run a CICS named counter server.

Chapter 21. Defining and starting AXM system services

The authorized cross-memory (AXM) server environment provides the runtime environment for CICS data sharing server regions. You must define and start AXM system services to run any of the CICS data sharing servers.

About this task

To establish AXM cross-memory connections for an MVS image, you must define an MVS subsystem called AXM. You can choose to create the subsystem statically or dynamically.

Procedure

- To define the MVS subsystem statically, add an entry with the required parameters to the IEFSSNxx member of SYS1.PARMLIB:

```
SUBSYS SUBNAME(AXM) INITRTN(AXMSI)
```

Defining AXM in the IEFSSNxx member ensures that AXM system services automatically become available when you IPL MVS.

The AXM subsystem initialization routine, AXMSI, sets up the appropriate definitions from the master scheduler region. Note that AXM uses the subsystem definition only as a means of scheduling AXM initialization in the master scheduler address space. The MVS subsystem interface for AXM is not activated or used.

- To define the MVS subsystem dynamically, enter the following command:

```
SETSSI ADD,SUBNAME=AXM,INITRTN=AXMSI
```

If initialization of the AXM subsystem fails for any reason, for example, because of an error in the command or because AXMSI is not in a linklist library, MVS does not allow another attempt because the subsystem is then already defined. In this case, use a different subsystem name, such as AXM1, because AXM does not rely on a specific subsystem name. If you start AXM successfully the first time, further attempts are ignored.

What to do next

When the AXM subsystem has started successfully, you can create a data sharing server.

Chapter 22. Setting up and running a temporary storage server

CICS transactions running in an AOR access temporary storage pools through a temporary storage server that supports a named pool. In each MVS image in the sysplex, you must set up one temporary storage server for each pool that is defined in the coupling facility.

About this task

Procedure

1. Define and start up the temporary storage job, for a shared temporary storage pool to run in an mvs batch region.
2. When the temporary storage region is running, you can issue commands to control the queue server for the pool. You can also upload and reload queue pools.

Related tasks

“Defining temporary storage pools for temporary storage data sharing” on page 17
Using TS data sharing means replacing main or auxiliary storage for your TS queues with one or more TS pools, where the scope and function of each TS pool is similar to a QOR.

Overview of the temporary storage data sharing server

CICS transactions running in an AOR access TS pools through a TS data sharing server that supports a named pool. In each MVS image in the sysplex, you need one TS server for each pool defined in a coupling facility which can be accessed from that MVS image. All TS pool access is performed by cross-memory calls to the TS server for the named pool.

An AOR can access more than one TS server concurrently. This multiserver access is required if you create multiple pools, because each TS server provides access to only one pool of TS queues.

CICS maps temporary storage requests to a TS server using the POOLNAME attribute of a TSMODEL resource definition, if one exists. Alternatively, if you are using a temporary storage table (TST) instead of TSMODEL resource definitions, CICS maps temporary storage requests to a TS server by means of the SYSIDNT option on the TST TYPE=SHARED macro.

The methods for specifying a TS pool make it easy to migrate queues from a QOR to a TS data sharing pool. You can use the TS global user exit, XTSEREQ, to modify the SYSID on a TS request so that it references a TS data sharing pool.

Figure 32 on page 350 illustrates a parallel sysplex with three CICS AORs linked to the temporary storage server address space(s).

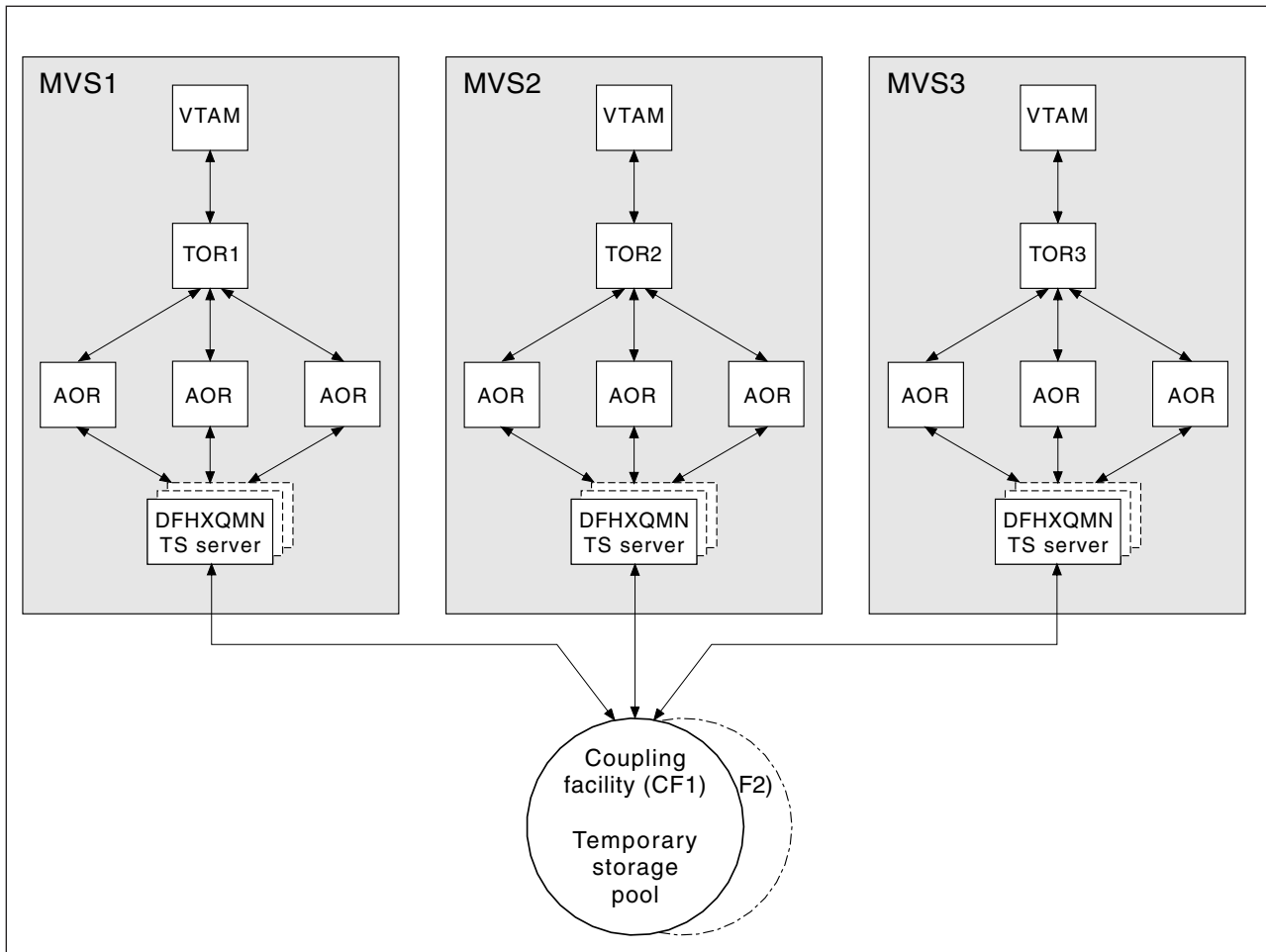


Figure 32. Conceptual view of a Parallel Sysplex with TS data sharing.

Security

The server must be authorized to access the coupling facility list structure in which the temporary storage pool is defined; XES checks this. The server must also be authorized to act as a temporary storage server; AXM checks this. For information on how to define the necessary authorizations see the *CICS RACF Security Guide*.

Defining TS server regions

A shared TS pool consists of an XES list structure, which is accessed through a cross-memory queue server region. You start a shared TS pool in an MVS image by starting up a queue server region for that pool.

About this task

To start up the TS server region for a shared TS pool, you can use either a batch job or a started task. The job or task must invoke the queue server region program, DFHXQMN, which is in an APF-authorized library.

Procedure

1. Specify the DFHXQMN program either in a SYSIN data set defined in the JCL, or in the **PARM** parameter on the EXEC statement.

2. Specify the mandatory and optional startup parameters for the DFHXQMN program.
 - a. You must specify a SYSPRINT DD statement for the print file.
 - b. You must specify a SYSIN DD statement for the server parameters.
 - c. You must specify the TS pool name.
 - d. It is recommended that you specify TIME=NOLIMIT. The server task remains in a wait during most normal processing, because server processing is performed under the TCB of the client CICS region. If you omit this parameter, your server job could fail with abend S522 (wait limit exceeded), depending on the JWT value specified in the SMFPRMxx member of SYS1.PARMLIB.
 - e. Specify additional parameters as required. For example, you might want to control the maximum number of queues that are to be supported in the pool and the number of buffers the server is to allocate.

Sample startup job for a TS server

Figure 33 shows an example of the JCL you might use to start a TS server.

```
//PRODTSQ1 JOB ...
//TSSERVER EXEC PGM=DFHXQMN,REGION=64M,TIME=NOLIMIT Start TS data sharing server
//STEPLIB DD DSN=CICSxxx.SDFHAUTH,DISP=SHR Authorized library
//SYSPRINT DD SYSOUT=* Messages and statistics
//SYSIN DD *
POOLNAME=PRODTSQ1 Pool name
MAXQUEUES=5000 Allow up to 5000 large queues
BUFFERS=1000 1000 buffers (32K each, 32M total)
/*
```

Figure 33. Sample startup job for a TS queue server

Queue server REGION parameter

The queue server REGION parameter JCL needs to specify at least enough virtual storage for the specified number of buffers plus the storage used to process queue requests.

Each buffer occupies a little more than 32K bytes, and each connected CICS region can have up to ten queue requests active at a time, each using 5K to 10K bytes, so to be safe the REGION size should allow at least 32K per buffer and 100K for each connected CICS region, plus a margin of about 10% for other storage areas.

During server initialization, the server acquires all of the available storage above the 16M line, as determined by the REGION size, then releases 5% of it for use by operating system services. It also acquires 5% of the free storage below the line for use in routines which require 24-bit addressable storage, for example sequential file read and write routines.

After server initialization, AXM page allocation services are used to manage server region storage. Server statistics indicate how much storage is allocated and used within the storage areas above and below the 16M line, which are called AXMPGANY and AXMPGLOW in the statistics.

If a task in the server region or a cross-memory request runs out of storage, this is likely to result in AXM terminating that task or request using a simulated abend

with system completion code 80A to indicate a GETMAIN failure. Although the server can usually continue processing other requests in this case, running out of storage in a critical routine can cause the server to terminate, so it is best to ensure that the REGION size is large enough to eliminate the risk.

TS queue server parameters

Parameters are specified in the form KEYWORD=value. You can optionally specify keywords in mixed case to improve readability.

If you specify more than one parameter in the PARM field, or on the same SYSIN input line, the parameters must be separated by commas. Any text following one or more spaces is taken as a descriptive comment. Any parameter line starting with an asterisk or a space is assumed to be a whole line comment.

You can enter some parameter keywords in more than one form, such as an abbreviation. The standard form of each keyword is generally the longest form of the first word shown.

The main parameters used are listed on the server print file during start-up.

The following parameters are all valid as initialization parameters (in the SYSIN file, or the PARM field), and some can be modified by the server SET command.

You can display any parameter with the server DISPLAY command. Display the values of all parameters using DISPLAY ALLPARMS.

Specify the following keywords to give combined lists of information:

- PARMS for main parameter values
- STATS for all available statistics
- INIT to select parameters and statistics whose values are usually displayed when initialization is complete

Primary parameters

These parameters are usually specified for all servers:

POOLNAME=pool_name

specifies the name, of 1 to 8 characters, of the queue pool used to form the server name and the name of the coupling facility list structure DFHXQLS_poolname. This parameter is valid only at initialization, and must always be specified.

This keyword can also be coded as **POOL**.

BUFFERS={100|number}

specifies the number of queue buffers to allocate for the server address space.

A queue index buffer holds a queue index entry plus up to 32K of queue data (for a small queue). When a READ or WRITE request completes the queue index information is retained in the buffer. This can avoid the need to reread the queue index if the same queue is referenced from the same MVS image before the buffer has been reused. If no buffer is available at the time of a request, the request is made to wait until one becomes free.

The number of buffers should preferably be at least ten for each CICS region that can connect to the server in this MVS image. This avoids the risk of buffer waits. Additional buffers may be used to reduce the number of coupling facility accesses by keeping recently used queue index entries in storage. In

particular, if the current version of a queue index entry is in storage at the time a queue item is read, the request requires only one coupling facility access instead of two. If the current version of a queue index entry is in storage when a second or subsequent item is written to the same queue, the request requires only one coupling facility access instead of three.

It is not worth defining extra buffers beyond the point where this might cause MVS paging, as it is more efficient to reread the index entry than to page in the buffer from auxiliary storage. This parameter is valid only at initialization.

The valid range is from 1 to 999999.

This keyword can also be coded as **BUF**.

FUNCTION={SERVER|UNLOAD|RELOAD}

Information about this parameter is given in "Unloading and reloading queue pools" on page 360.

STATSOPTIONS={NONE|SMF|PRINT|BOTH}

specifies the statistics options that determine whether interval statistics are produced and whether statistics are sent to SMF, the print file, or both.

This keyword can also be coded as **STATSOPT**.

ENDOFDAY={00:00|hhmm}

specifies the time when end of day statistics are to be collected and reset. If statistics options specify NONE, end of day statistics are written to the print file. The valid range is from 00:00 to 24:00.

This keyword can also be coded as **EOD**.

STATSINTERVAL={3:00|hhmm}

specifies the statistics interval, within the range of 1 minute to 24 hours. It is ignored if STATSOPTIONS=NONE.

The valid range is from 00:01 to 24:00 (although it may be specified in seconds).

This keyword can also be coded as **STATSINT**.

Automatic restart manager (ARM) parameters

During server initialization, the server unconditionally registers with ARM except when the server program is invoked with either the UNLOAD or the RELOAD functions. The server will not start if the registration fails.

Use the following parameters to override default processing for the automatic restart manager:

ARMELEMENTNAME=elementname

specifies the automatic restart manager element name, up to 16 characters, to identify the server to ARM for automatic restart purposes. The permitted characters for the element name are A to Z 0-9 \$ # @ and the underscore symbol (_).

The default identifier is of the form DFHXQ nm _poolname, where XQ represents the server type, nm is the &SYSC clone value for the system (which can be either one or two characters), and poolname is the name of the pool served by the server.

This parameter is only valid at server initialization.

This keyword can be abbreviated to **ARMELEMENT** or **ARMELEMENTNAME**.

ARMELEMENTTYPE=*elementtype*

specifies the automatic restart manager element type, up to 8 characters for use in ARM policies as a means of classifying similar elements. The permitted characters for the element type are A to Z 0-9 \$ # and @.

The default element type is SYSCICSS.

This parameter is only valid at server initialization.

This keyword can be abbreviated to **ARMELEMENTTYPE**.

List structure parameters

The following parameters specify list structure attributes and are used only for initial allocation of the pool list structure, which occurs the first time a server is started for the pool:

POOLSIZE={0|number_of_bytes{K|M|G}}

specifies the maximum amount of storage to be allocated for the list structure, expressed as kilobytes in the form *nK*, or megabytes in the form *nM*, or gigabytes in the form *nG*.

This takes effect when the list structure is being created with a specified value of less than that specified for the list structure in the CFRM policy.

The default value 0 specifies that no maximum limit is to be applied other than that specified in the CFRM policy. A non-zero value is generally rounded up by MVS to the next multiple of 256K.

The valid range is from 0 to 16777215M.

MAXQUEUES={1000|number}

specifies the maximum number of data lists to be reserved when the structure is allocated, which determines the maximum number of large queues that can be stored in the structure. Note that this parameter also determines how many list headers are defined when the structure is created. Although you should take care to specify a large enough number to handle large queues, specifying an excessively large number will use up an unnecessary amount of coupling facility storage for preallocated list headers.

This number cannot be changed without reallocating the structure. Therefore, if the structure is being allocated at less than its maximum size, the value here should be based on the maximum possible size of the structure rather than its initial size.

The valid range is from 1 to 999999.

This keyword can also be coded as **MAXQ**.

For information about defining list structures, see “Defining temporary storage pools for temporary storage data sharing” on page 17.

Debug trace parameters

These parameters are used only for intensive debug tracing.

Note that using these options in a production environment may significantly impact performance and cause the print file to grow very rapidly, using up spool space.

Trace messages from cross-memory requests may be lost if they are generated faster than the trace print subtask can print them. In such cases, the trace indicates only how many messages were lost.

TRACECF={OFF|ON}

specifies the coupling facility interface debug trace options, OFF or ON. This option produces trace messages on the print file indicating the main parameters to the coupling facility request interface and the result from the IXLIST macro.

This keyword can also be coded as **CFTR** or **CFTRACE**.

TRACERQ={OFF|ON}

specifies the queue request debug trace options, OFF or ON. This option produces trace messages on the print file indicating the main parameters on entry to the shared queue request or shared queue inquire interface and the results on exit.

This keyword can also be coded as **RQTR** or **RQTRACE**.

Tuning parameters

The following parameters are provided for tuning purposes. They are normally allowed to assume their default values:

ELEMENTSIZE={256|number}

specifies the element size for structure space, which must be a power of 2. For current coupling facility implementations there is no known reason to specify other than the default value of 256.

This parameter is valid only at server initialization and is used only when the structure is first allocated. The valid range is 256 to 4096.

This keyword can also be coded as **ELEMSIZE**.

ELEMENTRATIO={1|number}

specifies the element side of the entry/element ratio when structure is first allocated. This determines the proportion of the structure space initially set aside for data elements.

The ideal value for this ratio results from the average size of data for each entry being divided by the element size. However, the server automatically adjusts the ratio according to the actual entry and element usage.

This parameter is valid only at server initialization, and is used only when the structure is first allocated.

The valid range is from 1 to 255.

This keyword can also be coded as **ELEMRAIO**.

ENTRYRATIO={1|number}

specifies the entry side of the entry/element ratio when the structure is first allocated. It determines the proportion of structure space initially to be set aside for list entry controls.

It is not essential to specify this parameter because the server automatically adjusts the ratio based on actual usage to improve space utilization if necessary.

This parameter is valid only at server initialization and is used only when the structure is first allocated.

The valid range is from 1 to 255.

LASTUSEDINTERVAL={00:10|hhmm}

specifies how often the last used time for large queues is to be updated.

For small queues, the last used time is updated on every reference. For large queues, updating the last used time requires an extra coupling facility access, so that it is done only if the queue has not previously been accessed within this interval of the current time. This means that the last used time interval returned by INQUIRE can be greater than the true value by an amount up to the value specified on this parameter. As the main purpose of the last used time specification is to determine whether the queue is obsolete, an interval of a few minutes should be sufficient.

The valid range is from 00:00 to 24:00 (although it may be specified in seconds).

This keyword can also be coded as **LASTUSEDINT**.

SMALLQUEUEITEMS={9999|number}

specifies the maximum number of items that can be stored in the small queue format in the queue index entry data area. This parameter can force a queue to be converted to the large queue format if it has a large number of small items. It can be more efficient to write the items separately than to rewrite the whole small queue data area each time.

The valid range is from 1 to 32767.

SMALLQUEUE SIZE={32K|number}

specifies the maximum data size for a small queue including the two-byte length prefix on each data item. Any queue exceeding the maximum size, when writing the second or subsequent item to a queue, is converted to the large queue format.

This parameter can force queues to be converted to the large queue format at a smaller size than 32K. This is to prevent large amounts of data being written to the small queue format. Performance improvements can result on systems where asynchronous coupling facility processing causes contention for hardware resources. On most systems, however, it is probably more efficient to defer conversion until the maximum size of 32K is reached.

The valid range is from 4096 to 32768.

Warning parameters

These parameters modify the threshold at which warning messages and automatic ALTER actions occur when the structure becomes nearly full:

ELEMENTWARN={80|number}

specifies the percentage of elements in use at which warnings and automatic ALTER actions should be first triggered.

The valid range is from 1 to 100.

This keyword can also be coded as **ELEMWARN**.

ELEMENTWARNINC={5|number}

specifies the percentage increase (or decrease) of elements in use before the next warning should be triggered (reduced to 1 when next increase would otherwise reach 100). Additional messages are issued as the number of elements in use changes. The messages stop when the number of elements in use falls at least by this percentage below the initial warning level.

The valid range is from 1 to 100.

This keyword can also be coded as **ELEMWARNINC**.

ENTRYWARN={80|number}

specifies the percentage of entries in use at which warnings and automatic ALTER actions should be first triggered.

The valid range is from 1 to 100.

ENTRYWARNINC={5|number}

specifies the percentage increase (or decrease) of entries in use before next warning should be triggered (reduced to 1 when next increase would otherwise reach 100). Additional messages are issued as the number of elements change. The messages stop when the number of entries in use falls at by least the specified percentage below the initial warning level.

The valid range is from 1 to 100.

Automatic ALTER parameters

Define the following parameters to modify the conditions under which the server attempts an automatic ALTER action when the structure becomes nearly full.

For details of the queue server automatic ALTER process, see “Queue server automatic ALTER processing.”

ALTERELEMMIN={100|number}

specifies the minimum number of excess elements that must be present for an automatic ALTER to be issued to convert those elements to entries.

The valid range is from 1 to 999999999.

ALTERELEMPC={1|number}

specifies the minimum percentage of excess elements that must be present for an automatic ALTER to be issued to increase the proportion of entries.

The valid range is from 0 to 100.

ALTERENTRYMIN={100|number}

specifies the minimum number of excess entries that must be present for an automatic ALTER to be issued to convert those entries to elements.

The valid range is from 0 to 999999999.

ALTERENTRYPC={1|number}

specifies the minimum percentage of excess entries which must be present for an automatic ALTER to be issued to increase the proportion of elements.

The valid range is from 0 to 100.

ALTERMININTERVAL={00:10|hhmm}

specifies the minimum time interval to be left between automatic ALTER attempts when the structure is nearly full (above the element or entry warning level).

The valid range is from 00:00 to 24:00.

This keyword can also be coded as **ALTERMININT**.

Queue server automatic ALTER processing

The queue server monitors the total number of elements and entries in use in the structure, using information returned by the coupling facility on every request.

When the numbers in use exceed the specified thresholds, a warning message, DFHXQ0411 or DFHXQ0412, is issued, and is repeated each time the number in use increases beyond further thresholds.

Each time the warning is issued, the server tests whether an automatic ALTER for the entry to element ratio should be performed. The test is done by calculating how many excess elements or entries will be left when the other runs out completely. This is based on the ratio between the current numbers of elements and entries in use.

An IXLALTER request is issued to alter the entry to element ratio to the actual current ratio between the number of entries and elements in use if:

- The number of excess elements or entries exceeds the number specified in the ALTERELEMMIN or ALTERENTRYMIN parameter, and
- The same number expressed as a percentage of the total exceeds the value specified in the ALTERELEMPC or ALTERENTRYPC parameter

Only one ALTER request may be active at a time for a given structure. If the ALTER process is started by one server, the ALTER of another server will be rejected. However, the system automatically notifies all servers when the ALTER completes, giving the new numbers of elements and entries so that each server can update its own status information.

A further ALTER attempt is suppressed until at least the minimum ALTER interval (specified by the ALTERMININTERVAL parameter) has elapsed, if:

- Any form of ALTER has already been used recently (by any server or by an operator SETXCF ALTER command), and
- The structure space usage has remained above warning levels since the previous attempt.

Shared TS queue server commands

Commands can be issued to control a queue server using the MVS MODIFY (F) command specifying the job or started task name of the server region.

The general form of a queue server command is as follows:

```
F server,cmd parameter,parameter...  comments
```

The MVS STOP command is equivalent to issuing the server command STOP using the MVS MODIFY command.

The queue server supports the following commands:

SET keyword=value

changes one or more server parameter values. This applies to all parameters other than those indicated as being for initialization only. The command can be abbreviated to T, as for the MVS SET command.

DISPLAY keyword

displays one or more parameter values, or statistics summary information, on the console. The valid parameter keywords for DISPLAY and PRINT are described later in this section. The command can be abbreviated to D, as for the MVS DISPLAY command.

PRINT keyword

produces the same output as DISPLAY but only on the print file.

STOP

terminates the server, waiting for any active connections to terminate first, and preventing any new connections.

The command can be abbreviated to **P**, as for the MVS **STOP** command.

CANCEL {RESTART={NO|YES}}

Terminate the server immediately. You can specify whether automatic restart should be requested.

For information about **CANCEL RESTART** see “The **CANCEL** command options” on page 360.

The server also responds to XES events such as an operator **SETXCF** command to alter the structure size. If the server can no longer access the coupling facility, it automatically issues a server **CANCEL** command to close itself down immediately.

DISPLAY and PRINT keywords

DISPLAY or **PRINT** can display the values of any system initialization parameters.

These keywords can also display the following additional information:

ARMREGISTERED

Shows whether ARM registration was successful (YES or NO).

CONNECTIONS

List of the job names and APPLIDs for the regions currently connected to this server. You can also code this keyword as **CONN**.

Statistics summaries

BUFSTATS

Queue index buffer pool statistics.

You can also code this keyword as **BUFST**.

CFSTATS

Coupling facility interface I/O and response statistics.

You can also code this keyword as **CFST** or **STATSCF**.

POOLSTATS

Usage statistics for the pool list structure as a whole.

You can also code this keyword as **POOLST**.

STORAGESTATS

Main storage allocation statistics for the server address space.

You can also code this keyword as **STGST**, **STGSTATS**, or **STORAGEST**.

Keywords representing combined lists of information

PARAMETERS

Main parameter values.

You can also code this keyword as **PARM**, **PARMS**, or **PARAM**.

ALLPARAMETERS

All parameter values.

You can also code this keyword as **ALLPARMS**.

STATISTICS

All available statistics.

You can also code this keyword as **STAT** or **STATS**.

INITIALIZED

Selected parameters and statistics whose values are usually displayed when initialization is complete.

You can also code this keyword as **INIT**.

ARM

Display all ARM-related parameter values:

- ARMELEMENTNAME
- ARMELEMENTTYPE
- ARMREGISTERED

You can also code this keyword as **ARMSTATUS**.

The CANCEL command options

You can use the CANCEL command to request an automatic restart.

Specify the following parameter:

RESTART={NO|YES}

Terminate the server immediately, specifying whether or not automatic restart should be requested. The default is RESTART=NO.

If the server encounters an unrecoverable problem with the coupling facility connection, consisting either of lost connectivity or a structure failure, it cancels itself using the CANCEL RESTART=YES command. This terminates the existing connection and shuts down the server. A new instance of the server job is then started.

A server can also be restarted explicitly using either the server command CANCEL RESTART=YES or the MVS command CANCEL jobname,ARMRESTART.

You can also enter RESTART on its own for RESTART=YES, NORESTART for RESTART=NO.

Unloading and reloading queue pools

The contents of a queue pool can be unloaded to a sequential file and subsequently reloaded by running the server program using the FUNCTION parameter.

About this task

This can be used to preserve the queue pool across planned coupling facility maintenance, or to move the queue pool to a different coupling facility. The server does not support automatic structure REBUILD, but the unload and reload process is more flexible. The reloaded queue pool does not need the same name as the original pool.

FUNCTION={UNLOAD|RELOAD}

requests the server to perform the special functions UNLOAD or RELOAD. When the unload or reload processing has completed (normally or abnormally) the server program terminates.

If this parameter is omitted, the server program initializes the cross-memory queue server environment.

When UNLOAD or RELOAD is specified, the server program requires exclusive use of the list structure. If the structure is currently being used by a normal server, the attempt to unload or reload is rejected. Similarly, if a normal server attempts to start up while an unload or reload function is in progress, the attempt is rejected because shared access to the structure is not available.

All normal server parameters can be specified on UNLOAD and RELOAD, but many of these, such as the number of queue buffers, are ignored because they do not apply to unload or reload processing.

The UNLOAD function requires a DD statement for file name DFHXQUL describing the sequential data set to which the queue pool is to be unloaded. The format of the unloaded file is:

```
RECFM=F,LRECL=4096,BLKSIZE=4096.
```

An upper limit for the total size of the data set in bytes can be estimated from the pool usage statistics produced by the server. The total data size in bytes is obtained by multiplying the number of elements in use by the element size (usually 256), and for each queue there is also some control information which typically occupies fewer than 100 bytes per queue. The size is normally smaller than this because unused space in data elements is not included in the unloaded file. See Figure 34 for an example of UNLOAD JCL.

```
//UNLDTSQ1 JOB ...
//TSUNLOAD EXEC PGM=DFHXQMN          CICS TS queue server program
//STEPLIB DD DSN=CICSxxx.SDFHAUTH,DISP=SHR Authorized library
//SYSPRINT DD SYSOUT=*              Options, messages and statistics
//DFHXQUL DD DSN=TSQ1.UNLOADED.QPOOL, Unloaded queue pool
//      DISP=(NEW,CATLG),
//      SPACE=(4096,(10000,1000))    Estimated size in 4K blocks
//SYSIN DD *
FUNCTION=UNLOAD                      Function to be performed is UNLOAD
POOLNAME=PRODTSQ1                    Pool name
/*
```

Figure 34. Example of UNLOAD JCL

The RELOAD function requires a DD statement for file name DFHXQRL describing the sequential data set from which the queue pool is to be reloaded. The structure is allocated if necessary during reloading, in which case the same server parameters may be used to control structure attributes as for normal server execution. The RELOAD process bypasses any queues that are already found in the queue pool, because, for example, the structure was too small and the reload job had to be restarted after using ALTER to increase the size.

Note that when a pool is nearly full (with less than about 5% free entries and elements) there is no guarantee that it can be unloaded and reloaded into a structure of exactly the same size. The amount of space available is affected by the current ratio of entries to elements, which can only be controlled approximately by the automatic ALTER process.

If the structure reaches the warning level during reloading, the automatic ALTER process attempts to adjust the entry to element ratio, and the reload process will automatically wait for the ALTER to complete if no space is available while an ALTER is still in progress.

If RELOAD fails because it runs out of space, the resulting messages include the numbers of queues reloaded and blocks read up to the time of the failure. Compare these values with those in the messages from the original UNLOAD to determine how many more queues and how much more data remained to be loaded. See Figure 35 for an example of RELOAD JCL.

```
//RELDTSQ1 JOB ...
//TSRELOAD EXEC PGM=DFHXQMN          CICS TS queue server program
//STEPLIB DD   DSN=CICSxxx.SDFHAUTH,DISP=SHR  Authorized library
//SYSPRINT DD  SYSOUT=*                  Options, messages and statistics
//DFHXQRL DD   DSN=TSQ1.UNLOADED.QPOOL,DISP=OLD Unloaded queue pool
//SYSIN DD *
FUNCTION=RELOAD                          Function to be performed is RELOAD
POOLNAME=PRODTSQ1                         Pool name
POOLSIZE=50M                              Increased pool size
MAXQUEUES=10000                           Increased number of big queues
/*
```

Figure 35. Example of RELOAD JCL

Chapter 23. Setting up and running a coupling facility data table server

CICS coupling facility data tables provide rapid sharing of working data within a sysplex, with update integrity. The data tables are held in coupling facility structures, with access through a named server. You must set up one server for each pool in an MVS image.

Before you begin

Before you can start a server for named coupling facility data table pool, define the coupling facility structure to be used for the pool. See “Defining a coupling facility data table pool” on page 108 for information about defining a coupling facility list structure.

About this task

You can put related groups of coupling facility data tables in separate pools; for example, you might want to have one pool for production and another for test. A pool is defined as a list structure in the coupling facility resource management (CRFM) policy. The pool name is used to form the server name with the prefix DFHCF and is specified in the startup JCL for the server.

Procedure

1. Define and start a coupling facility data table server job, to run in an mvs batch region.
2. When the server is running you can perform various operating tasks:
 - You can control the server region using MVS commands.
 - You can delete or empty the pool for the coupling facility data tables.
 - You can unload and reload the contents of a pool to and from a sequential data set.

Overview of a coupling facility data table server

CICS coupling facility data tables is designed to provide rapid sharing of working data within a sysplex, with update integrity.

The data is held in a table that is similar in many ways to a shared user-maintained data table, and the API used to store and retrieve the data is based on the file control API used for user-maintained data tables.

Figure 36 on page 364 illustrates a Parallel Sysplex with three CICS AORs linked to the coupling facility data table servers.

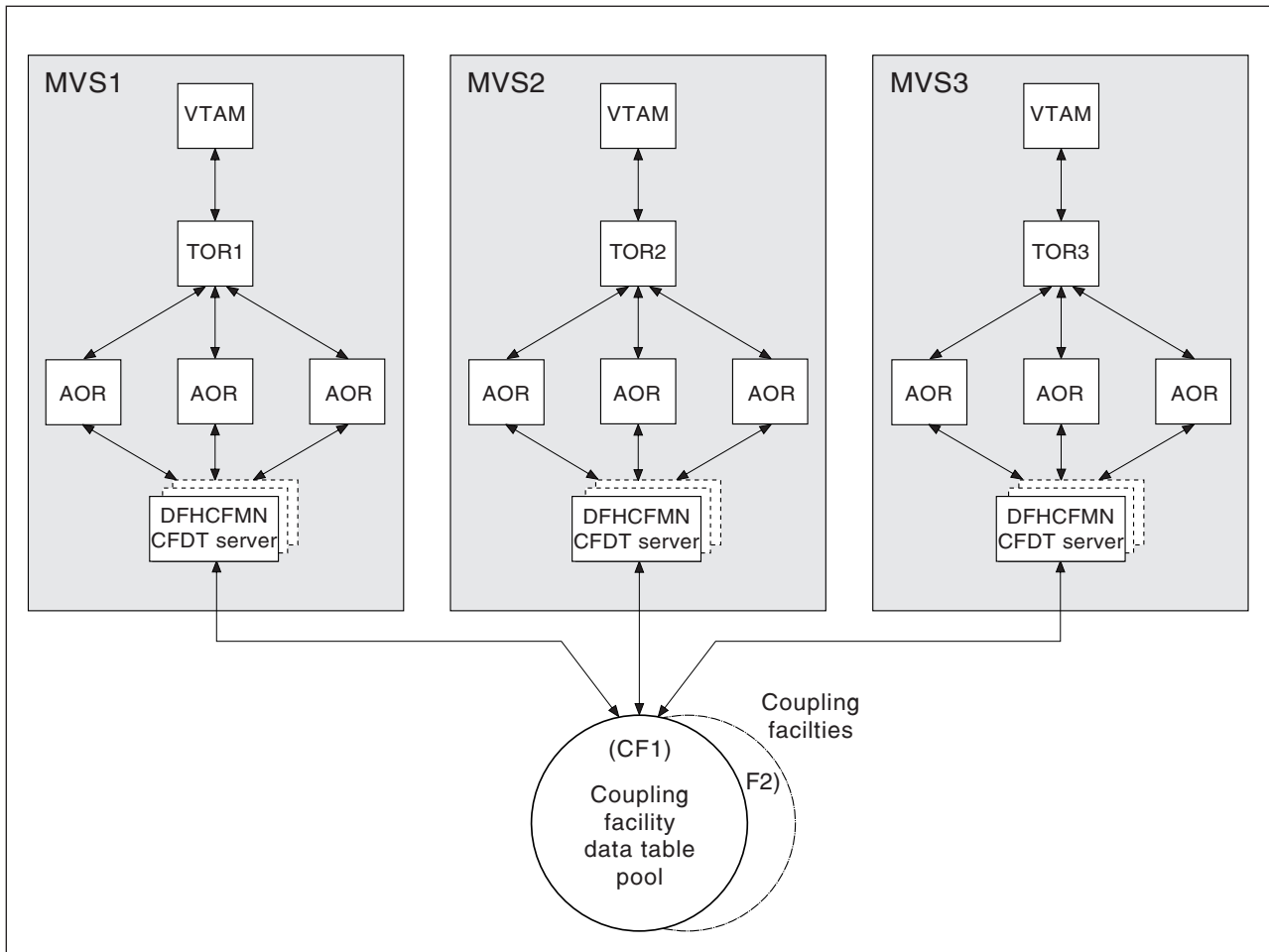


Figure 36. Conceptual view of a Parallel Sysplex with coupling facility data table servers

Defining and starting a coupling facility data table server region

You activate a coupling facility data table pool in an MVS image by starting up a coupling facility data table server region for that pool.

About this task

You can start the server as a started task, started job, or as a batch job. The job or task must invoke the coupling facility data table server region program, DFHCFMN, from an APF-authorized library. DFHCFMN is in the CICS authorized library, CICSTS41.CICS.SDFHAUTH.

Procedure

1. Specify the DFHCFMN program either in a SYSIN data set defined in the JCL, or in the **PARM** parameter on the EXEC statement.
2. Specify the mandatory and optional startup parameters for the DFHCFMN program. If you specify a startup parameter in both the SYSIN data set and the **PARM** parameter, the **PARM** value overrides the SYSIN value because the **MVS START** command can override the **PARM** value.
 - a. You must specify a SYSPRINT DD statement for the print file.

- b. You must specify a SYSIN DD statement for the server parameters.
- c. You must specify the TS pool name.
- d. It is recommended that you specify the **REGION** parameter. This parameter ensures that the coupling facility data table server region has enough storage to process the maximum number of data table requests that can run concurrently.
- e. It is recommended that you specify TIME=NOLIMIT. The server task remains in a wait during most normal processing, because server processing is performed under the TCB of the client CICS region. If you omit this parameter, your server job could fail with abend S522 (wait limit exceeded), depending on the JWT value specified in the SMFPRMxx member of SYS1.PARMLIB.
- f. Specify additional parameters as required. For example, you might want to control the maximum number of queues that are to be supported in the pool and the number of buffers the server is to allocate.

Results

Tip: The easiest way to ensure that all pool-related parameters are consistent across MVS images is to use the same SYSIN parameter data set, or an identical copy of it, for all servers accessing the same pool, and to specify in the PARM field any parameters that vary between servers.

Coupling facility data table server JCL example

```
//PRODCFD1 JOB    ...
//CFSERVER EXEC  PGM=DFHCFMN,REGION=40M,TIME=NOLIMIT    CICS CFDT Server
//STEPLIB  DD    DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR    Authorized library
//SYSPRINT DD    SYSOUT=*                               Messages and statistics
//SYSIN    DD    *
POOLNAME=PRODCFD1                                     Pool name
MAXTABLES=100                                         Allow up to 100 tables
/*
```

Figure 37. Sample JCL to start a CFDT server address space

Coupling facility data table server parameters

Parameters are specified in the form **KEYWORD=value**, where keywords can optionally be specified in mixed case to improve readability.

If you specify more than one parameter in the PARM field or on the same SYSIN input line, the parameters must be separated by a comma. Any text following one or more spaces is taken as a descriptive comment. Any parameter line which starts with an asterisk or a space is assumed to be a whole line comment.

You can enter some parameter keywords in more than one form, such as in abbreviated or truncated form.

The main parameters are listed on the server print file during start-up.

The parameters are all valid as initialization parameters (in the SYSIN file or PARM field), and some can also be modified by the SET command.

REGION parameter

The JCL **REGION** parameter ensures that the coupling facility data table server region has enough storage to process the maximum number of data table requests that can run concurrently.

The number of coupling facility data table requests that each connected CICS region can have active at a time is limited to about 10. Each request requires about 40 KB, therefore the **REGION** size should specify at least 400 KB for each connected CICS region, plus a margin of about 10% for other storage areas. Thus, for a server supporting up to five CICS regions, specify **REGION=2200K**.

During server initialization, the server acquires all the available storage above 16 MB, as determined by the **REGION** parameter, then releases 5% of it for use by operating system services. It also acquires 5% of the free storage below 16MB for use in routines that require 24-bit addressable storage, for example sequential file read and write routines.

After initialization, the server uses AXM page allocation services to manage its storage. Server statistics indicate how much storage is allocated and used within the storage areas above and below 16 MB, which are called AXMPGANY and AXMPGLOW in the statistics.

If a task in the server region or a cross-memory request runs out of storage, this is likely to result in AXM terminating that task or request using a simulated abend with system completion code 80A to indicate a GETMAIN failure. Although the server can usually continue processing other requests in this case, running out of storage in a critical routine can cause the server to terminate. Therefore, it is best to ensure that the **REGION** size is large enough to eliminate this risk.

Pool name parameter

This parameter, **POOLNAME**, is always required:

POOLNAME=*name*

specifies the 8-character name of the coupling facility data table pool. This is appended by the server to the prefix DFHCF to create its own server name, as in DFHCF.*poolname*, and also to the prefix DFHCFLS_ to create the name of the coupling facility list structure, as in DFHCFLS_*poolname*.

This parameter is valid only at server initialization, and must always be specified.

This keyword can be abbreviated to **POOL**.

Security parameters

You can use these parameters to specify whether you want to use the optional security mechanism that the server provides, to check that CICS regions are authorized to open a coupling facility data table. They also allow you to override standard processing for this optional security.

SECURITY={YES|NO}

specifies whether individual coupling facility data table security checks are required.

YES You want the server to perform a security check against each CICS region that attempts to open a coupling facility data table. Access is controlled through profiles defined in the general resource class named on the **SECURITYCLASS** parameter.

This requires an external security manager, such as RACF, that supports the FASTAUTH function in cross-memory mode.

NO You do not want the server to perform this extra security check when opening a coupling facility data table.

This is the only security check performed by the server that is optional. The other file security checks are always performed by the server, as described in the *CICS RACF Security Guide*.

This parameter is valid only at server initialization.

This keyword can be abbreviated to **SEC**.

SECURITYCLASS={FCICSFCT|class}

specifies the name of the RACF general resource class that the server is to use for security checks on coupling facility data table access by CICS regions. The name can be up to 8 characters, and is the name of the class in which the CFDT resource profile and its access list are defined.

This parameter is valid only at server initialization.

This keyword can be abbreviated to **SECCLASS**.

SECURITYPREFIX={NO|YES}

specifies whether the resource name that is passed to RACF for the coupling facility data table security check, when SECURITY=YES is specified, should be prefixed with the server region user ID.

Note: For this security check, the resource name used by the server is the either the name specified on the TABLENAME attribute of the CICS file resource definition, or the FILE name if TABLENAME is not specified.

YES The server prefixes the resource name with the server region user ID (the default) or an alternative prefix specified on the **SECURITYPREFIXID** parameter.

NO The server passes to RACF only the 8-character resource name, without any prefix.

This parameter is valid only at server initialization.

This keyword can be abbreviated to **SECPREFIX** or **SECPRF**.

SECURITYPREFIXID=identifier

specifies an alternative prefix that the server is to use for security checks on coupling facility data table access by CICS regions, instead of the server region user ID. The prefix can be up to 8 characters, and should correspond to the prefix used to define profile names of CFDTs to RACF. This parameter is effective only if you specify SECURITYPREFIX=YES.

This parameter is valid only at server initialization.

This keyword can be abbreviated to **SECPREFIXID**.

Statistics parameters

Use the following parameters to specify server statistics options:

ENDOFDAY={00:00|hh:mm}

specifies the time of day, in hours and minutes, when the server is to collect and reset end-of-day statistics.

Note: If the STATSOPTIONS parameter specifies NONE, the server still writes end-of-day statistics to the print file.

The valid range of times is from 00:00 to 24:00.

This keyword can be abbreviated to **EOD**.

STATSINTERVAL={03:00|*hh:mm*}

specifies the statistics collection interval, in the range 1 minute to 24 hours. This parameter is ignored if the **STATSOPTIONS** parameter specifies **NONE**.

The time interval can range from 00:01 to 24:00.

This keyword can be abbreviated to **STATSINT**.

STATSOPTIONS={NONE|**SMF**|**PRINT**|**BOTH**}

specifies whether the server is to produce interval statistics, and the destination for the statistics it produces.

NONE

The server does not produce any interval statistics.

SMF The server produces interval statistics and writes them to the current SMF data set only.

PRINT

The server produces interval statistics and writes them to the server's print file only.

BOTH The server produces interval statistics and writes them to the current SMF data set and to the server's print file.

This keyword can be abbreviated to **STATSOPT**.

Automatic restart manager (ARM) parameters

During server initialization, the server unconditionally registers with ARM except when the server program is invoked with either the **UNLOAD** or the **RELOAD** functions. The server will not start if the registration fails.

Use the following parameters to override default processing for the automatic restart manager:

ARMELEMENTNAME=*elementname*

specifies the automatic restart manager element name, up to 16 characters, to identify the server to ARM for automatic restart purposes. The permitted characters for the element name are A to Z 0-9 \$ # @ and the underscore symbol (_).

The default identifier is of the form **DFHCFnn_poolname**, where **CF** represents the server type, **nn** is the **&SYSCClone** value for the system (which can be either one or two characters), and **poolname** is the name of the pool served by the server.

This parameter is only valid at server initialization.

This keyword can be abbreviated to **ARMELEMENT** or **ARMELEMNAME**.

ARMELEMENTTYPE=*elementtype*

specifies the automatic restart manager element type, up to 8 characters for use in ARM policies as a means of classifying similar elements. The permitted characters for the element type are A to Z 0-9 \$ # and @.

The default element type is **SYSCICSS**.

This parameter is only valid at server initialization.

This keyword can be abbreviated to **ARMELEMENTTYPE**.

List structure parameters

These parameters specify list structure attributes. They are used only for the initial allocation of resources when the pool list structure is being created, which occurs the first time you start a server for a pool.

MAXTABLES={1000|*number*}

specifies the maximum number of data lists to be reserved when the structure is allocated. The number of data lists determines the maximum number of tables that can be stored in the structure. Note that this parameter also determines how many list headers are defined when the structure is created. Although you should take care to specify a large enough number, specifying an excessively large number will use up an unnecessary amount of coupling facility storage for preallocated list headers.

You cannot change this number without reallocating the structure, which means first deleting the existing structure (see “Deleting or emptying coupling facility data table pools” on page 381). If the structure is being allocated at less than its maximum size, specify a number for the maximum number of tables based on the maximum size of the structure, rather than its initial allocation size.

This parameter is valid only at server initialization and is used only when the structure is first allocated. The valid range is from 1 to 999999.

This keyword can be abbreviated to **MAXT**.

Note: In addition to the data lists defined by **MAXTABLES** there are control lists used internally by CICS so the total number of lists is greater than the number specified in **MAXTABLES**.

POOLSIZE={0|*number*}

specifies the initial amount of coupling facility storage to be allocated for the pool list structure, expressed as kilobytes in the form *number* K, megabytes in the form *number* M or gigabytes in the form *number* G.

0 The special value 0 means that the server is to obtain an initial allocation using the parameters specified in the CFRM policy. If the CFRM policy specifies an **INITSIZE** value for the structure, this determines the initial allocation, otherwise the maximum **SIZE** value is allocated.

number

A non-zero value specifies an initial amount of storage to be allocated, overriding the **INITSIZE** parameter in the CFRM policy. This value is generally rounded up by MVS to the next multiple of 256K. The valid range of values is 1K to 16777215M, but should not be greater than the value specified on the **SIZE** parameter.

It is generally preferable to omit this parameter, and specify the structure size using the **INITSIZE** parameter in the CFRM policy. The **POOLSIZE** option can, however, be useful if the structure is being reallocated or reloaded, and the CFRM policy has not been updated to reflect the required size.

Note: If the value is greater than the value specified on the CFRM **SIZE** parameter, the server **POOLSIZE** parameter is ignored and the initial allocation is based on the parameters specified in the CFRM policy.

This parameter is valid only at server initialization and is only used when the structure is first allocated.

Debug trace parameters

These parameters are provided only for intensive debug tracing.

Using these options in a production environment could have a significant impact on performance and cause the print file to grow very rapidly, using up spool space.

Trace messages from cross-memory requests can be lost if they are generated faster than the trace print subtask can print them. In this event, the trace only indicates how many messages were lost.

CFTRACE={OFF|ON}

specifies the coupling facility interface debug trace option.

OFF Coupling facility interface debug trace is disabled.

ON Coupling facility interface debug trace produces trace messages on the print file, indicating the main parameters to the coupling facility request interface, and the result from the IXLLIST macro.

This keyword can also be specified as **TRACECF**.

RQTRACE={OFF|ON}

specifies the table request debug trace option.

OFF Table request debug trace is disabled.

ON Table request debug trace produces trace messages on the print file, indicating the main parameters on entry to each cross-memory request, and the results on exit.

This keyword can also be specified as **TRACERQ=**.

Tuning parameters

These parameters are provided for tuning purposes, but normally you can omit these and let the server assume their default values.

ELEMENTRATIO={1|number}

specifies the element part of the entry-to-element ratio when the structure is first allocated. This determines what proportion of the structure space is initially set aside for data elements. (For information about list structures and entry-to-element ratios, see the *OS/390 MVS Programming: Sysplex Services Guide*, GC28-1771.)

The valid range is from 1 to 255.

Divide the average size of data per entry by the element size to obtain the optimum value for this ratio. However, if the structure becomes short of space and altering the ratio could improve space utilization, the server automatically adjusts the ratio according to the actual entry and element usage.

This parameter is valid only at server initialization and is used only when the structure is first allocated.

This keyword can be abbreviated to **ELEMNRATIO**.

ELEMENTSIZE={256|size}

specifies the data element size for the list structure, which must be a power of 2. The valid range is from 256 to 4096.

For current coupling facility implementations there is no known reason to use any value other than the default value of 256.

This parameter is valid only at server initialization and is only used when the structure is first allocated.

This keyword can be abbreviated to **ELEMSIZE**.

ENTRYRATIO={1|number}

specifies the entry part of the entry-to-element ratio when the structure is first allocated. This determines what proportion of the structure space is initially set aside for list entry controls. (For information about list structures and entry-to-element ratios, see the *OS/390 MVS Programming: Sysplex Services Guide*, GC28-1771.)

It is not essential to specify this parameter because the server automatically adjusts the ratio based on actual usage to improve space utilization if necessary.

This parameter is valid only at server initialization and is used only when the structure is first allocated. The valid range is from 1 to 255.

Lock wait parameters

Use these parameters to modify the time intervals for the server lock wait retry mechanism.

This is provided mainly as a “wake-up” mechanism to ensure that requests waiting for a record lock do not wait indefinitely in certain rare cases where a system failure or structure full condition could cause a lock release notification message to be lost. In addition, this mechanism also ensures that if a CICS task is purged while it has a request waiting for a lock, the waiting request in the server is woken up as soon as the lock wait retry interval expires. The request process then finds that the CICS task is no longer waiting for it, therefore it terminates rather than continuing to wait and hold on to server resources until the lock becomes free.

There are two times involved: a scan time interval and a wait time. The server starts its lock scan interval timing when the first request is made to wait.

This mechanism has very little effect on normal processing and the default lock wait retry parameter values are designed to suit the majority of installations.

LOCKSCANINTERVAL={5|number}

specifies the time interval after which requests waiting for record locks are scanned to check for lock wait timeout.

This affects the overall duration of the lock wait timeout, because a request that starts waiting for a lock during a given scan interval is timed as if from the start of the interval. The lock scan interval should be less than the lock wait interval, and ideally should be such that the lock wait interval is an exact multiple of the lock scan interval.

You can specify this value as a number of seconds or in the time format *hh:mm:ss*.

The valid range is from 1 (1 second) to 24:00 (24 hours).

This keyword can be abbreviated to **LOCKSCANINT**.

LOCKWAITINTERVAL={10|number}

specifies the maximum time that a request should wait for a record lock before being reinvoked to retry. The actual time a request waits depends on how far into a scan interval it started its wait. For example, in a server using the scan

and wait default intervals, if 4 seconds have already elapsed when a request starts to wait, the maximum time it can wait is 6 seconds. When the server checks the timeout value for the request, it assumes that the request has been waiting for the full scan period. Thus, for the default values, a request is reinvoked to retry after waiting between five and ten seconds.

This value can either be specified as a number of seconds or in time interval format *hh:mm:ss*.

The valid range is from 1 (1 second) to 24:00 (24 hours).

This keyword can be abbreviated to **LOCKWAITINT**.

Warning parameters

Use these parameters to modify the thresholds at which warning messages are issued, and an automatic structure alter occurs, when the structure becomes nearly full.

ELEMENTWARN={80|number}

specifies the percentage of list structure elements in use at which warning messages and an automatic structure alter should be first triggered.

The valid range is from 1 to 100 per cent.

This keyword can be abbreviated to **ELEMWARN**.

ELEMENTWARNINC={5|number}

specifies the percentage increase (or decrease) of elements in use before the next warning message should be triggered (reduced to 1 when the next increase would otherwise reach 100). After the first warning, additional messages are issued as the number of elements increases and decreases. These messages stop when the number of elements in use has fallen at least this percentage below the initial warning level.

The valid range is from 1 to 100 per cent.

This keyword can be abbreviated to **ELEMWARNINC**.

ENTRYWARN={80|number}

specifies the percentage of list structure entries in use at which warning messages and an automatic structure alter action should be first triggered.

The valid range is from 1 to 100 per cent.

ENTRYWARNINC={5|number}

specifies the percentage increase (or decrease) of entries in use before the next warning message should be triggered (reduced to 1 when the next increase would otherwise reach 100). After the first warning, additional messages are issued as the number of elements increases and decreases. These messages stop when the number of entries in use has fallen at least this percentage below the initial warning level.

The valid range is from 1 to 100 per cent.

Automatic structure alter parameters

Use these parameters to modify the conditions under which the server attempts an automatic alter when the structure becomes nearly full.

For information about the structure alter process, see “Coupling facility data table server automatic structure alter” on page 375.

ALTERELEMMIN={100|number}

specifies the minimum number of excess elements that must be present to cause the server to issue an automatic alter to convert those elements to entries.

The valid range is from 0 to 999999999.

ALTERELEMPC={5|number}

specifies the minimum percentage of excess elements that must be present to cause the server to issue an automatic alter to increase the proportion of entries.

The valid range is from 0 to 100 per cent.

ALTERENTRYMIN={100|number}

specifies the minimum number of excess entries that must be present to cause the server to issue an automatic alter to convert those entries to elements.

The valid range is from 0 to 999999999.

ALTERENTRYPC={5|number}

specifies the minimum percentage of excess entries that must be present to cause the server to issue an automatic alter to increase the proportion of elements.

The valid range is from 0 to 100 per cent.

ALTERMININTERVAL={00:10|hh:mm}

specifies the minimum time interval to be left between automatic structure alter attempts when the structure is nearly full (above the element or entry warning level). This is to prevent excessive numbers of alter attempts to try to optimize space when the structure is nearly full.

The valid range is from 00:00 to 24:00.

This keyword can be abbreviated to **ALTERMININT**.

Reserved space parameters

Use these parameters to control the amount of structure space that is reserved for rewrites and server internal operations (such as tracking units of work and notifying other servers when a lock is released).

For information about the effect of these parameters, see “Avoiding structure full conditions” on page 374.

ELEMENTRESERVEMIN={300|number}

specifies the minimum number of list structure data elements (normally 256 bytes each) to be reserved for rewrites and server internal operations.

The valid range is from 0 to 999999999.

This keyword can be abbreviated to **ELEMRESERVEMIN**.

ELEMENTRESERVEPC={5|number}

specifies the percentage of list structure data elements to be reserved for rewrites and internal server use. If this percentage evaluates to less than the minimum number specified on the **ELEMENTRESERVEMIN** parameter, the minimum number is used instead.

The valid range is from 0 to 100.

This keyword can be abbreviated to **ELEMRESERVEPC** or **ELEMRESPC**.

ENTRYRESERVEMIN={100|number}

specifies the minimum number of list structure entries to be reserved for rewrites and server internal operations.

The valid range is from 0 to 999999999.

This keyword can be abbreviated to **ENTRYRESMIN**.

ENTRYRESERVEPC={5|number}

specifies the percentage of list structure elements to be reserved for rewrites and internal server use. If this percentage evaluates to less than the minimum number specified on the **ENTRYRESERVEMIN** parameter, the minimum number is used instead.

The valid range is from 0 to 100.

This keyword can be abbreviated to **ENTRYRESPC**.

Avoiding structure full conditions

If the coupling facility data table structure is allowed to become completely full, this not only prevents the addition of new records or tables, but can also have a significant impact on performance and application function.

In particular, rewrite requests can be rejected even when the size of the new data is less than or equal to the original, and server internal operations can fail, causing internal time-outs and retries.

The parameters **ELEMENTRESERVEMIN**, **ELEMENTRESERVEPC**, **ENTRYRESERVEMIN** and **ENTRYRESERVEPC** are provided to reduce the risk of the structure becoming totally full, by reserving a number of entries and elements, which can only be used for operations that normally only need extra space temporarily, such as rewrites or unit of work control operations. If a server is asked to write a new record or create a new table when the number of entries or elements remaining in the structure (as returned by each coupling facility access request) is less than or equal to the specified reserve level, the request is rejected with an indication that no space is available. Before rejecting the request, the server issues a dummy read request in order to find out the latest usage levels for the structure, in case more space has recently become available.

Using the reserved space parameters means that, even if the structure fills up very rapidly (for example, because a table is being loaded that is too large for the available space), enough space should remain to allow rewrites of existing records and allow internal communication between servers to continue normally.

Note that this mechanism cannot prevent the structure from eventually becoming totally full, as recoverable rewrites are allowed to use the reserved space temporarily, and rewrites that increase the data length will gradually use up the reserved elements. If action is not taken to prevent the structure from becoming totally full, the following effects can occur:

- An attempt to close a table or change the table status could encounter a temporary structure full condition. In this case, the attempt is retried indefinitely, because it must be completed in order to preserve table integrity (the only alternative being to terminate the server). The retry process normally succeeds quickly, but there is a theoretical case where this can cause a loop until another server causes temporarily unavailable resources to be released.
- Rewrites with the same (or smaller) data size for a table using the contention update model are retried indefinitely if they initially fail because of a structure

full condition. This is done to protect the application against having to handle this unexpected form of failure. Again, the retry should normally succeed quickly, but there is a theoretical possibility that this could loop for a while.

- Rewrites for a table using the locking or recoverable update model could be rejected with a structure full condition even if the data size is not increased. No retry is attempted in this case.
- Units of work can be backed out because the server is unable to create unit of work control entries for commit processing.
- There may not be sufficient structure space to send lock release messages, in which case waiting tasks are not woken up immediately but continue to wait for up to the time-out interval specified on the LOCKWAITINTERVAL parameter before finding out that the lock has been released.

Coupling facility data table server automatic structure alter

The coupling facility data table server monitors the total number of elements and entries in use in the structure, using information returned by the coupling facility on every request.

When the numbers in use exceed the specified warning thresholds, the server issues a warning message, and this warning message is repeated each time the number in use increases beyond further thresholds.

Each time the server issues a warning, it also tests whether an automatic structure alter for the entry-to-element ratio should be issued. If any form of alter has already been issued recently (by any server or through an operator SETXCF ALTER command) and the structure space usage has remained above warning levels since the previous attempt, any further structure alter attempt is suppressed until at least the minimum interval (specified through the ALTERMININTERVAL parameter) has elapsed.

The server checks whether an automatic structure alter should be issued, by calculating how many excess elements or entries will be left when the other runs out completely, based on the ratio between the current numbers of elements and entries in use. If the number of excess elements or entries exceeds the number specified in the ALTERELEMMIN or ALTERENTRYMIN parameter, and the same number expressed as a percentage of the total exceeds the value specified in the ALTERELEMPC or ALTERENTRYPC parameter, an IXLALTER request is issued to alter the entry to element ratio to the actual current ratio between the number of entries and elements in use.

Only one alter request can be active at a time for a given structure. This means a server may well find that another server has already started the structure alter process, in which case its own alter is rejected. However, the system automatically notifies all servers when the structure alter is completed, giving the new numbers of elements and entries so that each server can update its own status information.

Controlling coupling facility data table server regions

You can issue commands to control a coupling facility data table server, using the **MVS MODIFY (F)** command to specify the job or started task name of the server region, followed by the server command.

About this task

The general form of a coupling facility data table server command, using the short form F, is as follows:

```
F job_name,command parameters... comments
```

You can use the following commands to control the coupling facility data table server region.

Procedure

- To modify the server initialization parameters, use the **SET** command:

```
SET keyword=operand[,keyword=operand,...]
```

The command can be abbreviated to **T**, as for the **MVS SET** command. See “The SET command options” for details.

- To display the values of one or more parameter values or statistics summary information on the console, use the **DISPLAY** command:

```
DISPLAY keyword[=operand][,keyword[=operand],...]
```

The valid keywords for **DISPLAY** are all the initialization parameters, plus an additional set described under “DISPLAY and PRINT command options” on page 378.

The command can be abbreviated to **D**, as for the **MVS DISPLAY** command.

- To print the output that the **DISPLAY** command produces, use the **PRINT** command:

```
PRINT keyword[=operand][,keyword[=operand],...]
```

The command produces the same output as **DISPLAY**, supporting the same keywords, but on the print file only.

- To delete a table, use the **DELETE TABLE=name** command. The table must not be in use for this command to succeed. You can abbreviate the command to **DEL**.
- To stop the server normally, use either the **STOP** command or the **MVS STOP** command. The server waits for any active connections to end first and prevents any new connections while it is waiting. The command can be abbreviated to **P**; for example **P jobname**.
- To request an automatic restart, use the **CANCEL** command. This command terminates the server immediately. You can specify whether the server will automatically restart. For information about **CANCEL RESTART** see “The CANCEL command options” on page 360.
- The server also responds to XES events such as an operator **SETXCF** command to alter the structure size. If the server can no longer access the coupling facility, it automatically issues a server **CANCEL** command to close itself down immediately.

The SET command options

You can use the SET command to modify groups of server initialization parameters.

These system initialization parameter groups are:

- The statistics parameters
- The debug trace parameters
- The lock wait parameters
- The warning parameters

- The automatic ALTER parameters.

See “Coupling facility data table server parameters” on page 365 for details of these keywords.

The following **SET** keywords are used to modify the server's recovery status of an inactive CICS region that had unresolved units of work when it last terminated:

RESTARTED=*applid*

Establish a temporary recoverable connection for the given APPLID. This resolves any units of work that were in commit or backout processing when the region last terminated, and indicates whether there are any remaining indoubt units of work.

This keyword can be abbreviated to **RESTART** or **REST**.

COMMITTED={*applid*|*applid.uowid*}

Establish a temporary recoverable connection for the specified APPLID and commit all indoubt units of work, or, if *uowid* is also specified, commit that specific unit of work.

This command should be used **only** when it is not possible to restart the original CICS region to resolve the work normally, because it can result in inconsistency between coupling facility data table resources and other CICS resources updated by the same unit of work.

This keyword can be abbreviated to **COMMIT** or **COMM**.

BACKEDOUT={*applid*|*applid.uowid*}

Establish a temporary recoverable connection for the specified APPLID and back out all indoubt units of work, or, if *uowid* is also specified, back out that specific unit of work.

This command should be used *only* when it is not possible to restart the original CICS region to resolve the work normally, because it can result in inconsistency between coupling facility data table resources and other CICS resources updated by the same unit of work.

This keyword can be abbreviated to **BACKOUT** or **BACK**.

Use the following **SET** parameters to modify options relating to a specific table:

TABLE=*name*

specifies the table to which the following table-related parameters in the same command are to be applied. This parameter is required before any table-related parameters.

MAXRECS=*number*

Modify the maximum number of records that can be stored in the table specified by the preceding **TABLE** parameter.

If the maximum number is set to a value less than the current number of records in the table, no new records can be stored until records have been deleted to reduce the current number to within the new maximum limit. For a recoverable table, this also means that records cannot be updated, because the recoverable update process adds a new record on the rewrite operation then deletes the original record when the transaction completes.

This keyword can also be specified as **MAXNUMRECS**.

AVAILABLE={**YES**|**NO**}

Specify whether the table named by the preceding **TABLE** parameter is

available for new OPEN requests. If the table is made unavailable, a CICS region that subsequently issues an OPEN request for the table receives a response indicating that it is unavailable, but regions that currently have the table open are not affected. Even when a table is marked as unavailable, a server can implicitly open it on behalf of a CICS region to allow recoverable work to be resolved during restart processing.

This keyword can be abbreviated to **AVAIL**.

Examples of the SET command: The following example changes the statistics options:

```
SET STATSOPT=BOTH,EOD=21:00,STATSINT=06:00
```

The following example modifies the maximum number of records allowed in the specified table:

```
SET TABLE=PAYECFT1,MAXRECS=200000
```

DISPLAY and PRINT command options

You can use the DISPLAY (and PRINT) commands to display the values of any initialization parameters plus some additional information.

Some of the parameters that provide additional information support generic names. You specify generic names using the following wildcard characters:

- An * (asterisk symbol). Use this anywhere in the parameter value to represent from 0 to 8 characters of any value. For example, CICSH* to represent all the CICS APPLIDs in a CICSplex identified by the letter H.
- A % (per cent symbol). Use this anywhere in the parameter value to represent only one character of any value. For example, CICS%T* to represent all the TOR APPLIDs in all CICSplexes.

The parameters supported by the DISPLAY and PRINT commands are as follows:

APPLIDS

Display the APPLID and MVS system name for every CICS region that currently has a recoverable connection to the pool. This command returns information not only for the server to which the MODIFY command is issued, but for all other servers connected to the same pool.

This keyword can be abbreviated to **APPLID**, **APPLS** or **APPL**.

APPLID={*applid*|*generic*}

Display the APPLID and MVS system name for each region that currently has a recoverable connection to the server's pool, and whose APPLID matches *applid* or *generic*. This command returns information not only for the server to which the MODIFY command is issued, but for all other servers connected to the same pool.

applid Use this for a specific APPLID, which should match only one region in the sysplex.

generic Use a suitable generic value when you want to obtain information about several regions.

If *applid* or *generic* is not specified, the server treats this as equivalent to the command DISPLAY APPLIDS.

This keyword can also be specified as **APPLIDS**, **APPLS** or **APPL**.

ARMREGISTERED

Shows whether ARM registration was successful (YES or NO).

CONNECTIONS

Display the jobnames and applids of the regions currently connected to the server to which the command is issued.

This keyword can be abbreviated to **CONN**.

TABLES

Display the names of all tables currently allocated in the pool.

TABLE={name|generic_name}

Display information about the attributes and status of a specific table, or of a set of tables whose names match the generic name.

If no table name is specified, this is treated as equivalent to **DISPLAY TABLES**.

TABLEUSERS

Display the CICS APPLIDs of the regions that are currently using each of the tables currently defined in the pool.

This keyword can be abbreviated to **TABLEU**.

TABLEUSERS={name|generic_name}

Display the CICS APPLIDs of the regions that are currently using the specified table, or using each of the set of tables whose names match the generic name.

If no table name is specified, this is treated as equivalent to **DISPLAY TABLEUSERS**.

This keyword can be abbreviated to **TABLEU**

UOWIDS

Display the applids of all regions that currently have unresolved recoverable units of work, together with the number of units of work that are currently in doubt, or are in the process of being committed or backed out.

This keyword can be abbreviated to **UOWS**.

UOWIDS={applid|generic_applid}v{applid.*|generic_applid.*}

Display, for the specified regions if they currently have unresolved recoverable units of work, information about those units of work. The information returned depends on the form of operand used.

applid|generic_applid

This form of operand displays the number of units of work that are currently in doubt, or are in the process of being committed or backed out.

If you specify *applid*, the server displays UOW information for a specific APPLID, which should correspond to only one region in the sysplex.

If you specify *generic_applid* the server displays UOW information for all the APPLIDs that match the generic APPLID specified.

applid.|generic_applid.**

This form of operand displays:

- The state and local UOWID of each individual unit of work, followed by
- A summary of the number of units of work that are currently in doubt, or are in the process of being committed or backed out.

If you specify *applid.**, the server displays the UOW information for a specific APPLID, which should correspond to only one region in the sysplex.

If you specify *generic_applid.**, the server displays UOW information for all the APPLIDs that match the generic APPLID specified.

This keyword can be abbreviated to **UOWS**.

UOWID=*applid.uowid*

Display the state of an individual unresolved unit of work, identified by its applid and local unit of work ID (UOWID). Enter the local UOWID as 16 hexadecimal digits.

This keyword can be abbreviated to **UOW**.

DISPLAY and PRINT options for statistics summaries

Use the following parameters to display or print statistics:

CFSTATS

Display statistics for coupling facility interface accesses and responses from the server.

This keyword can also be specified as **CFST** or **STATSCF**.

POOLSTATS

Display usage statistics for the pool list structure as a whole. This is based on information returned by coupling facility access requests, therefore it is only as current as the most recent request made through the server to which the command is issued.

This keyword can be abbreviated to **POOLST**.

TABLESTATS

Display statistics for requests, processed by the server to which the command is issued, for each table plus a summary of all requests processed, including those that are not table-specific, such as unit of work control.

Note that only tables with a non-zero number of requests since the start of the current statistics interval are shown.

This keyword can also be specified as **TABLEST**.

TABLESTATS={*name***|***generic_name***}**

Display request statistics for the specified table or tables.

name A specific table name in the pool accessed by the server. Returns statistics for this table only.

generic_name

A generic name that you can use to obtain statistics about a number of tables. Returns statistics for any table name that matches the generic name.

This keyword can be abbreviated to **TABLEST**.

STORAGESTATS

Display main storage allocation statistics for the server address space.

This keyword can be abbreviated to **STORAGEST** or **STGST**.

DISPLAY and PRINT options for combined lists of information

These keywords represent combined lists of information:

PARAMETERS

Display the main parameter values. These are POOLNAME, SECURITY, SECURITYPREFIX, statistics options, and list structure options.

This keyword can be abbreviated to **PARM** or **PARMS**.

ALLPARAMETERS

Display all parameter values.

This keyword can be abbreviated to **ALLPARMS**.

STATISTICS

Display all available statistics.

This keyword can be abbreviated to **STAT** or **STATS**.

INITIALIZED

Display the parameters and statistics that are usually displayed when initialization is complete. This is equivalent to PARM, POOLSTATS, STGSTATS.

This keyword can be abbreviated to **INIT**.

ARM

Display all ARM-related parameter values:

- ARMELEMENTNAME
- ARMELEMENTTYPE
- ARMREGISTERED

This keyword can be coded as **ARMSTATUS**.

The CANCEL command options

You can use the CANCEL command to request an automatic restart.

Specify the following parameter:

RESTART={NO|YES}

Terminate the server immediately, specifying whether or not automatic restart should be requested. The default is RESTART=NO.

If the server encounters an unrecoverable problem with the coupling facility connection, consisting either of lost connectivity or a structure failure, it cancels itself using the CANCEL RESTART=YES command. This terminates the existing connection and shuts down the server. A new instance of the server job is then started.

A server can also be restarted explicitly using either the server command CANCEL RESTART=YES or the MVS command CANCEL jobname,ARMRESTART.

You can also enter RESTART on its own for RESTART=YES, NORESTART for RESTART=NO.

Deleting or emptying coupling facility data table pools

You can delete a coupling facility data table pool using the MVS SETXCF command to delete its coupling facility list structure.

About this task

For example:

```
SETXCF FORCE,STRUCTURE,STRNAME=DFHCFLS_poolname
```

You can delete a structure only when there are no servers connected to the pool, otherwise MVS rejects the command.

When you attempt to start a server for a pool that has been deleted (or attempt to reload the pool), it is allocated as a new structure. The newly allocated structure uses size and location attributes specified by the currently active CFRM policy, and other values determined by the server initialization parameters (in particular, MAXTABLES).

Unloading and reloading coupling facility data table pools

You can unload, and reload, the complete contents of a coupling facility data table pool to and from a sequential data set by invoking the server program with the **FUNCTION** parameter, using the UNLOAD and RELOAD options.

About this task

You can use this function, for example, to do the following:

- Preserve the coupling facility data table pool during planned coupling facility maintenance, or
- Move the pool to a different coupling facility.
- Increase the size of the pool's list structure.

If the maximum number of tables specified in the original pool was too small, or the pool has reached its maximum size and needs to be expanded further, unload the pool, then delete the structure so that the reload process can reallocate it with more space.

Alternatively, you can use the system-managed rebuild facility to dynamically move a structure to another coupling facility connected to the same system. This allows servers to remain active, but temporarily suspends processing while the rebuild is in progress. For more information, see "System-managed list structure rebuild" on page 423.

FUNCTION={UNLOAD|RELOAD}

Specify the function for which the server is being initialized.

UNLOAD

Unload the entire contents of the coupling facility data table pool specified on the POOLNAME parameter to a sequential data set. When the unload processing has completed (normally or abnormally) the server program terminates.

The UNLOAD function requires a DD statement for DDNAME DFHCFUL describing the sequential data set to which the table pool is to be unloaded. The format of the unloaded data set is:

```
RECFM=F  
LRECL=4096  
BLKSIZE=4096
```

You can obtain an estimate of the upper limit for the total size of the data set, in bytes, from the pool usage statistics produced by the server:

- From the statistics, multiply the number of elements in use by the element size (usually 256) to get a total number of bytes for the data size, although the space needed to unload the data is normally much less, because unused space in a data element is not unloaded.

- Add some space for the record keys, calculated using a two-byte prefix plus the keylength for each record, plus about 100 bytes per table for table control information. Thus, the maximum you should need for keys and control information is:

$$(18 \text{ bytes} \times \text{number of entries}) + (100 \text{ bytes} \times \text{number of tables})$$

RELOAD

Reload, into the coupling facility data table pool named on the POOLNAME parameter, a previously unloaded coupling facility data table pool.

You can reload a pool into a pool with a different name—it does not have to keep the same name as the original pool. When the reload processing has completed (normally or abnormally) the server program terminates.

The RELOAD function requires a DD statement for DDNAME DFHCFRL, describing the sequential data set from which the table pool is to be reloaded.

The structure is allocated, if necessary, during reloading, in which case you can use the same server parameters to control structure attributes as for normal server startup. The reload process bypasses any tables or units of work that are already found in the pool (for example, because the structure was too small and the reload job had to be restarted after using ALTER to increase the structure size).

Note: If the unloaded pool structure was altered dynamically at any time after initial allocation (by using the SETXCF command to increase the size), ensure that the increased size is allocated for the reloaded pool. The recommended way is to update the INITSIZE parameter for the structure in the current CFRM policy whenever you alter the structure size, and to activate the updated policy using the SETXCF START ,POLICY command. Alternatively, you can specify the required pool size in the POOLSIZE parameter in the reload JCL, but note that this does not override the CFRM INITSIZE parameter if it is exactly equal to the maximum pool size.

Note: If you omit the FUNCTION parameter, the server program initializes a coupling facility data table server address space.

For the UNLOAD and RELOAD function, the server program requires exclusive use of the list structure. If the structure is currently being used by a normal server, the unload or reload attempt is rejected. Similarly, if a normal server attempts to start up while an unload or reload job is in progress, the attempt fails because shared access to the structure is not available.

You can specify all normal server parameters when unloading or reloading, but some of these (for example, security-related parameters) are ignored because they do not apply to unload or reload processing.

Note that when a pool is nearly full (with less than about 5% free entries and elements) there is no guarantee that it can be unloaded and reloaded into a structure of exactly the same size. This is because the amount of space available is affected by the current ratio of entries to elements, which is controlled only approximately by the automatic ALTER process. If the structure reaches the warning level during reloading, the automatic ALTER process attempts to adjust

the entry to element ratio. The reload process automatically waits for the ALTER to complete if reloading runs out of space while an ALTER is still in progress.

If reloading fails because it runs out of space, the resulting messages include the numbers of tables reloaded and blocks read up to the time of the failure. You can compare these values with those in the messages from the original unload job, to determine how many more tables and how much more data remains to be loaded. If a table had been partially reloaded before running out of space, it is deleted so that the whole table is reloaded again if the reload is retried later. If reloading is interrupted for any other reason than running out of space, for example by an MVS system failure, reloading can still be restarted using the partially reloaded structure, but in that case the structure space occupied by any partially reloaded table will be unavailable, so it is normally better to delete the structure (using the MVS **SETXCF FORCE** command) and start reloading again with a newly allocated structure.

```
//UNLDCFD1 JOB ...
//DTUNLOAD EXEC PGM=DFHCFMN          CICS CF data table server program
//STEPLIB DD DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR Authorized library
//SYSPRINT DD SYSOUT=*                Options, messages and statistics
//DFHCFUL DD DSN=CFDT1.UNLOADED.POOL, Unloaded data table pool
//      DISP=(NEW,CATLG),
//      SPACE=(4096,(10000,1000)) Estimated size in 4K blocks
//SYSIN DD *
FUNCTION=UNLOAD                      Function to be performed is UNLOAD
POOLNAME=PRODCFD1                   Pool name
/*
```

Figure 38. Unload JCL example

```
//RELD CFD1 JOB ...
//DTRELOAD EXEC PGM=DFHCFMN          CICS CF data table server program
//STEPLIB DD DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR Authorized library
//SYSPRINT DD SYSOUT=*                Options, messages and statistics
//DFHCFRL DD DSN=CFDT1.UNLOADED.POOL,DISP=OLD Unloaded table pool
//SYSIN DD *
FUNCTION=RELOAD                      Function to be performed is RELOAD
POOLNAME=PRODCFD1                   Pool name
POOLSIZE=50M                        Increased pool size
MAXTABLES=500                       Increased max number of tables
/*
```

Figure 39. Reload JCL example

Chapter 24. Setting up and running a region status server

Using a CICS region status server, you can rapidly share CICS region status data in a sysplex to support optimized workload management. A region status server services only region status requests, rather than region status and user application requests.

About this task

CICS region status data is broadcast to the sysplex using a data table named after the hosting CICSplex for the region. Each region in the CICSplex is described by a single record in the CICSplex data table. The data tables are held in coupling facility structures, with access controlled by a coupling facility data table (CFDT) server. You must set up one CFDT server for each pool in an MVS image.

You can put related groups of region status tables in separate pools; for example, you might want to have one pool for production and another for test. A pool is defined as a list structure in the coupling facility resource management (CRFM) policy. The pool name is used to form the server name with the prefix DFHCF and is specified in the startup JCL for the server.

Note: You must use different pool names for your region status server if you have two separate CICSplexes of the same name in the sysplex.

To set up and manage a region status server, follow these steps:

Procedure

1. Optional: For the best performance, define a list structure for a region status server pool. For more information, see “Defining a list structure for a region status server” on page 386.

You do not have to perform this step if you have an existing CFDT pool that you can use for storing your CICSplex data tables. However, the throughput of your optimized workloads might be impeded by any user application activity to the specified pool name, and any application throughput to the pool might be affected by the sysplex optimized workloads.

Note: You must use a different pool name from the pool name that you use for the RS server if you already have a CFDT with the same name as your CICSplex.

2. Define and start a region status server job, to run in an MVS batch region. For more information, see “Defining and starting a region status server region” on page 387.

What to do next

After you have successfully started your region status server, you can issue commands to manage the region status server and delete it if required. For more information, see “Controlling region status servers” on page 388 and “Deleting region status server pools” on page 394.

Related information

Optimized dynamic workload routing implementation

Defining a list structure for a region status server

The region status server pool is defined in the list structure for a coupling facility data table. You define the list structure in a coupling facility resource manager (CFRM) policy.

About this task

You must allocate storage in the coupling facility to store CICS status.

CICS records the status of a CICS region in a coupling facility data table named after the CICSplex to which the region belongs. That table must belong to a CFDT pool that is named in the CICSplex definition for that CICSplex. The default name is DFHRSTAT. In each z/OS image, there must be a region status server for each region status pool that will serve the CICS regions belonging to that CICSplex. A CICSplex data table contains one region status record for each region in that CICSplex.

Define the structure in the current coupling facility resource management (CFRM) policy using the IXCMIAPU utility. For an example of this utility, see member IXCCFRMP in the SYS1.SAMPLIB library. An example of a policy statement for a region status server pool is shown Figure 40 on page 387.

Procedure

1. Specify the name of the list structure. The name is formed by adding the prefix DFHCFLS_ to your chosen pool name, giving DFHCFLS_*poolname*. The default pool name as implemented by CPSM is DFHRSTAT.

You define and modify CICSplexes using the EYUSTARTCPLEXDEF view set. Using the CPLEXDEF detail view, you can modify the coupling facility (CF) tuning parameters for the region status (RS) server, which provide sysplex optimized workload routing.

Note: You can also modify the default region status (RS) pool name that will be used by all regions in the CICSplex. When you do not to use the default name DFHRSTAT, you must change the name before starting any other regions in the CICSplex. CPSM will not prevent you from changing the pool name while the CICSplex is active. If you make a change while the CICSplex is active, all CMAS and MAS regions in the CICSplex must be restarted as soon as possible. Failure to do so can result in inconsistent data in the CPSM WLM views and WLM optimization is deactivated until all the regions in the CICSplex are restarted.

2. Specify the size of the list structure. You can allocate an initial and maximum size using the **INITSIZE** and **SIZE** parameters in the CFRM policy definition. For an accurate estimate of storage requirements, use the IBM CFSizer tool available at <http://www.ibm.com/systems/support/z/cfsizer/>.

A region status record is approximately 40 bytes long.

For example, if PLEX1 contains 100 regions, PLEX2 contains 300 regions, and PLEX3 contains 1000 regions, the required structures are as follows:

- Poolname = DFHRSTAT, Table name = PLEX1, 100 regions x 40 bytes = 4 000 bytes total

- Poolname = DFHRSTAT, Table name = PLEX2, 300 regions x 40 bytes = 12 000 bytes total
 - Poolname = DFHRSTAT, Table name = PLEX3, 1000 regions x 40 bytes = 40 000 bytes total
3. Specify the preference list of coupling facilities in which the policy can be stored.
 4. When you have updated the CFRM new policy with the new structure definition, activate the policy using the MVS command:
SETXCF START,POLICY,POLNAME=*polycname*,TYPE=CFRM. Where *polycname* is the CFRM policy being st

Note that defining the CFRM policy statements for a list structure does not create the list structure. The structure is created the first time an attempt is made to connect to it, which occurs when the first coupling facility data table (CFDT) server that refers to the corresponding pool is started.

Example

```
STRUCTURE NAME(DFHCFLS_DFHRSTAT)
  SIZE(1000)
  INITSIZE(500)
  PREFLIST(FACIL01,FACIL02)
```

Figure 40. Example definition of a list structure for region status servers

Defining and starting a region status server region

When you start a region status server, you activate a pool in an MVS image for that server.

Before you begin

Before you start a region status server region, you must define the region status server structure to be used for the pool. For information about defining a region status server list structure, see “Defining a list structure for a region status server” on page 386.

About this task

You can start the server as a started task, started job, or as a batch job. This task explains how to start a region status server job, to run in an MVS batch region. The job or task must start the region status server program, DFHCFMN, from the CICS authorized library, CICSTS41.CICS.SDFHAUTH.

Procedure

1. Specify the DFHCFMN program either in a SYSIN data set defined in the JCL, or in the **PARM** parameter on the EXEC statement.
2. Specify the mandatory and optional startup parameters for the DFHCFMN program. If you specify a startup parameter in both the SYSIN data set and the **PARM** parameter, the **PARM** value overrides the SYSIN value because the **MVS START** command can override the **PARM** value.
 - a. You must specify a SYSPRINT DD statement for the print file.
 - b. You must specify a SYSIN DD statement for the server parameters.

Tip: To ensure that all pool-related parameters are consistent across MVS images, you must use the same SYSIN parameter data set, or an identical copy of it, for all servers accessing the same pool, and to specify in the PARM field any parameters that vary between servers.

- c. You must specify the region status pool name.
- d. You can specify the **REGION** parameter. This parameter ensures that the coupling facility data table server region has enough storage to process the maximum number of data table requests that can run concurrently.
- e. You can specify **TIME=NOLIMIT**. The server task remains in a wait, during most normal processing, because server processing is performed under the TCB of the client CICS region. If you omit this parameter, your server job might fail with abend S522 (wait limit exceeded), depending on the JWT value specified in the SMFPRMxx member of SYS1.PARMLIB.
- f. Specify additional parameters as required. For example, you might want to control the maximum number of queues that are to be supported in the pool and the number of buffers that the server is to allocate.

Results

The region status server is running, ready to receive and broadcast region status data to the CICS regions connected to it. The CICS regions connect through the poolname specified in the CICSplex definition.

Region status server JCL example

```
//PRODRSS1 JOB ...
//RSSERVER EXEC PGM=DFHCFMN,REGION=40M,TIME=NOLIMIT CICS CFDT Server for RS
//STEPLIB DD DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR Authorized library
//SYSPRINT DD SYSOUT=* Messages and statistics
//SYSIN DD *
POOLNAME=DFHRSTAT Pool name
MAXTABLES=100 Allow up to 100 tables
/*
```

Figure 41. Sample JCL to start a region status server address space

Controlling region status servers

You can issue commands to control a region status server, using the MVS **MODIFY (F)** command to specify the job or started task name of the server region, followed by the server command.

About this task

The general form of an MVS modify command, using the short form F, is as follows:

```
F job_name,command parameters... comments
```

You use the MODIFY command to pass information to a job or started task. In this task, you use the following commands to control the region status servers.

Procedure

- To modify the server initialization parameters, use the MVS **SET** command:
`SET keyword=operand[,keyword=operand,...]`

The **SET** command can be abbreviated to **T**, as for the MVS **SET** command. See “The SET command options” on page 376 for details.

- To display the values of one or more parameter values or statistics summary information on the console, use the **DISPLAY** command:

```
DISPLAY keyword[=operand][,keyword[=operand,]...]
```

The valid keywords for **DISPLAY** are all the initialization parameters, plus an additional set described under “DISPLAY and PRINT command options” on page 378.

The **DISPLAY** command can be abbreviated to **D**, as for the MVS **DISPLAY** command.

- To print the output that the **DISPLAY** command produces, use the MVS **PRINT** command:

```
PRINT keyword[=operand][,keyword[=operand,]...]
```

The **PRINT** command produces the same output as **DISPLAY**, supporting the same keywords, but on the print file only.

- To delete a table, use the **DELETE TABLE=*name*** command. The table must not be in use for this command to succeed. You can abbreviate the command to **DEL**.
- To stop the server normally, use the **STOP** command. The server waits for any active connections to end first, and prevents any new connections while it is waiting. You can abbreviate the command to **P**. You can also use the MVS **STOP** command, which is equivalent to issuing the server STOP command through the **MVS MODIFY** command. The syntax of the **STOP** command is:

```
STOP|P [jobname.]identifier[,A=asid]
```

- To terminate the server immediately, use the **CANCEL** command. You can also specify whether the server automatically restarts with the **RESTART** option. For information about **CANCEL RESTART** see “The CANCEL command options” on page 360.
- The server also responds to Cross System Extended Services (XES) events such as an operator **SETXCF** command to alter the structure size. If the server can no longer access the coupling facility, it automatically issues a server **CANCEL** command to close itself down immediately.

The SET command options

You can use the SET command to modify groups of server initialization parameters.

These system initialization parameter groups are:

- The statistics parameters
- The debug trace parameters
- The lock wait parameters
- The warning parameters
- The automatic ALTER parameters.

See “Coupling facility data table server parameters” on page 365 for details of these keywords.

The following **SET** keywords are used to modify the server's recovery status of an inactive CICS region that had unresolved units of work when it last terminated:

RESTARTED=*applid*

Establish a temporary recoverable connection for the given APPLID. This resolves any units of work that were in commit or backout processing when the region last terminated, and indicates whether there are any remaining indoubt units of work.

This keyword can be abbreviated to **RESTART** or **REST**.

COMMITTED={*applid*|*applid.uowid*}

Establish a temporary recoverable connection for the specified APPLID and commit all indoubt units of work, or, if *uowid* is also specified, commit that specific unit of work.

This command should be used **only** when it is not possible to restart the original CICS region to resolve the work normally, because it can result in inconsistency between coupling facility data table resources and other CICS resources updated by the same unit of work.

This keyword can be abbreviated to **COMMIT** or **COMM**.

BACKEDOUT={*applid*|*applid.uowid*}

Establish a temporary recoverable connection for the specified APPLID and back out all indoubt units of work, or, if *uowid* is also specified, back out that specific unit of work.

This command should be used *only* when it is not possible to restart the original CICS region to resolve the work normally, because it can result in inconsistency between coupling facility data table resources and other CICS resources updated by the same unit of work.

This keyword can be abbreviated to **BACKOUT** or **BACK**.

Use the following **SET** parameters to modify options relating to a specific table:

TABLE=*name*

specifies the table to which the following table-related parameters in the same command are to be applied. This parameter is required before any table-related parameters.

MAXRECS=*number*

Modify the maximum number of records that can be stored in the table specified by the preceding **TABLE** parameter.

If the maximum number is set to a value less than the current number of records in the table, no new records can be stored until records have been deleted to reduce the current number to within the new maximum limit. For a recoverable table, this also means that records cannot be updated, because the recoverable update process adds a new record on the rewrite operation then deletes the original record when the transaction completes.

This keyword can also be specified as **MAXNUMRECS**.

AVAILABLE={**YES**|**NO**}

Specify whether the table named by the preceding **TABLE** parameter is available for new OPEN requests. If the table is made unavailable, a CICS region that subsequently issues an OPEN request for the table receives a response indicating that it is unavailable, but regions that currently have the table open are not affected. Even when a table is marked as unavailable, a server can implicitly open it on behalf of a CICS region to allow recoverable work to be resolved during restart processing.

This keyword can be abbreviated to **AVAIL**.

Examples of the SET command: The following example changes the statistics options:

```
SET STATSOPT=BOTH,EOD=21:00,STATSINT=06:00
```

The following example modifies the maximum number of records allowed in the specified table:

```
SET TABLE=PAYECFT1,MAXRECS=200000
```

DISPLAY and PRINT command options

You can use the DISPLAY (and PRINT) commands to display the values of any initialization parameters plus some additional information.

Some of the parameters that provide additional information support generic names. You specify generic names using the following wildcard characters:

- An * (asterisk symbol). Use this anywhere in the parameter value to represent from 0 to 8 characters of any value. For example, CICSH* to represent all the CICS APPLIDs in a CICSplex identified by the letter H.
- A % (per cent symbol). Use this anywhere in the parameter value to represent only one character of any value. For example, CICS%T* to represent all the TOR APPLIDs in all CICSplexes.

The parameters supported by the DISPLAY and PRINT commands are as follows:

APPLIDS

Display the APPLID and MVS system name for every CICS region that currently has a recoverable connection to the pool. This command returns information not only for the server to which the MODIFY command is issued, but for all other servers connected to the same pool.

This keyword can be abbreviated to **APPLID**, **APPLS** or **APPL**.

APPLID={*applid*|*generic*}

Display the APPLID and MVS system name for each region that currently has a recoverable connection to the server's pool, and whose APPLID matches *applid* or *generic*. This command returns information not only for the server to which the MODIFY command is issued, but for all other servers connected to the same pool.

applid Use this for a specific APPLID, which should match only one region in the sysplex.

generic Use a suitable generic value when you want to obtain information about several regions.

If *applid* or *generic* is not specified, the server treats this as equivalent to the command DISPLAY APPLIDS.

This keyword can also be specified as **APPLIDS**, **APPLS** or **APPL**.

ARMREGISTERED

Shows whether ARM registration was successful (YES or NO).

CONNECTIONS

Display the jobnames and applids of the regions currently connected to the server to which the command is issued.

This keyword can be abbreviated to **CONN**.

TABLES

Display the names of all tables currently allocated in the pool.

TABLE={name|generic_name}

Display information about the attributes and status of a specific table, or of a set of tables whose names match the generic name.

If no table name is specified, this is treated as equivalent to DISPLAY TABLES.

TABLEUSERS

Display the CICS APPLIDs of the regions that are currently using each of the tables currently defined in the pool.

This keyword can be abbreviated to **TABLEU**.

TABLEUSERS={name|generic_name}

Display the CICS APPLIDs of the regions that are currently using the specified table, or using each of the set of tables whose names match the generic name.

If no table name is specified, this is treated as equivalent to DISPLAY TABLEUSERS.

This keyword can be abbreviated to **TABLEU**

UOWIDS

Display the applids of all regions that currently have unresolved recoverable units of work, together with the number of units of work that are currently in doubt, or are in the process of being committed or backed out.

This keyword can be abbreviated to **UOWS**.

UOWIDS={applid|generic_applid}v{applid.*|generic_applid.*}

Display, for the specified regions if they currently have unresolved recoverable units of work, information about those units of work. The information returned depends on the form of operand used.

applid|generic_applid

This form of operand displays the number of units of work that are currently in doubt, or are in the process of being committed or backed out.

If you specify *applid*, the server displays UOW information for a specific APPLID, which should correspond to only one region in the sysplex.

If you specify *generic_applid* the server displays UOW information for all the APPLIDs that match the generic APPLID specified.

applid.|generic_applid.**

This form of operand displays:

- The state and local UOWID of each individual unit of work, followed by
- A summary of the number of units of work that are currently in doubt, or are in the process of being committed or backed out.

If you specify *applid.**, the server displays the UOW information for a specific APPLID, which should correspond to only one region in the sysplex.

If you specify *generic_applid.**, the server displays UOW information for all the APPLIDs that match the generic APPLID specified.

This keyword can be abbreviated to **UOWS**.

| **UOWID**=*applid.uowid*

| Display the state of an individual unresolved unit of work, identified by its
| applid and local unit of work ID (UOWID). Enter the local UOWID as 16
| hexadecimal digits.

| This keyword can be abbreviated to **UOW**.

| **DISPLAY and PRINT options for statistics summaries**

| Use the following parameters to display or print statistics:

| **CFSTATS**

| Display statistics for coupling facility interface accesses and responses from the
| server.

| This keyword can also be specified as **CFST** or **STATSCF**.

| **POOLSTATS**

| Display usage statistics for the pool list structure as a whole. This is based on
| information returned by coupling facility access requests, therefore it is only as
| current as the most recent request made through the server to which the
| command is issued.

| This keyword can be abbreviated to **POOLST**.

| **TABLESTATS**

| Display statistics for requests, processed by the server to which the command
| is issued, for each table plus a summary of all requests processed, including
| those that are not table-specific, such as unit of work control.

| Note that only tables with a non-zero number of requests since the start of the
| current statistics interval are shown.

| This keyword can also be specified as **TABLEST**.

| **TABLESTATS**={*name* | *generic_name*}

| Display request statistics for the specified table or tables.

| *name* A specific table name in the pool accessed by the server. Returns
| statistics for this table only.

| *generic_name*

| A generic name that you can use to obtain statistics about a number of
| tables. Returns statistics for any table name that matches the generic
| name.

| This keyword can be abbreviated to **TABLEST**.

| **STORAGESTATS**

| Display main storage allocation statistics for the server address space.

| This keyword can be abbreviated to **STORAGEST** or **STGST**.

| **DISPLAY and PRINT options for combined lists of information**

| These keywords represent combined lists of information:

| **PARAMETERS**

| Display the main parameter values. These are **POOLNAME**, **SECURITY**,
| **SECURITYPREFIX**, statistics options, and list structure options.

| This keyword can be abbreviated to **PARM** or **PARMS**.

| **ALLPARAMETERS**

| Display all parameter values.

| This keyword can be abbreviated to **ALLPARMS**.

STATISTICS

Display all available statistics.

This keyword can be abbreviated to **STAT** or **STATS**.

INITIALIZED

Display the parameters and statistics that are usually displayed when initialization is complete. This is equivalent to **PARM**, **POOLSTATS**, **STGSTATS**.

This keyword can be abbreviated to **INIT**.

ARM

Display all ARM-related parameter values:

- ARMELEMENTNAME
- ARMELEMENTTYPE
- ARMREGISTERED

This keyword can be coded as **ARMSTATUS**.

The CANCEL command options

You can use the **CANCEL** command to request an automatic restart.

Specify the following parameter:

RESTART={NO|YES}

Terminate the server immediately, specifying whether or not automatic restart should be requested. The default is **RESTART=NO**.

If the server encounters an unrecoverable problem with the coupling facility connection, consisting either of lost connectivity or a structure failure, it cancels itself using the **CANCEL RESTART=YES** command. This terminates the existing connection and shuts down the server. A new instance of the server job is then started.

A server can also be restarted explicitly using either the server command **CANCEL RESTART=YES** or the MVS command **CANCEL jobname,ARMRESTART**.

You can also enter **RESTART** on its own for **RESTART=YES**, **NORESTART** for **RESTART=NO**.

Deleting region status server pools

You can delete a region status server pool by deleting its coupling facility list structure. You might do this for a service upgrade, or when a clean sysplex restart is required.

Before you begin

You can delete a structure only when no servers are connected to the pool; otherwise, MVS rejects the command.

About this task

For example:

```
SETXCF FORCE,STRUCTURE,STRNAME=DFHCFLS_poolname
```

You can verify that the pool has been successfully deleted by issuing the **XCF** command shown here:

| D XCF STRUCTURE,STRNAME=DFHCFLS_*poolname*

| Note that if you delete a region status server structure while CICS regions and
| workload are running, you disable CICSplex SM WLM optimized functions.

| **What to do next**

| When you attempt to start a server for a pool that has been deleted (or attempt to
| reload the pool), it is allocated as a new structure. The newly allocated structure
| uses size and location attributes specified by the currently active CFRM policy and
| other values determined by the server initialization parameters (in particular,
| **MAXTABLES**).

Chapter 25. Setting up and running a named counter server

CICS provides an efficient way of generating unique sequence numbers for use by applications in a Parallel Sysplex environment; for example, to allocate a unique number for orders or invoices. A named counter server maintains each sequence of numbers as a named counter.

About this task

To set up and manage a named counter server, you must follow these steps:

Procedure

1. Define a named counter options table.
2. Define a list structure.
3. Define and start a named counter server job, to run in an mvs batch region.
4. Manage named counter server regions.
5. Change the size of named counter pools.
6. Delete or empty named counter pools.
7. Unload and reload named counter pools.
8. Dump named counter pool list structures.

What to do next

These steps are explained in more detail in the following section.

Named counter server overview

Each time a number is assigned, the corresponding named counter is automatically incremented so that the next request gets the next number in sequence. Named counters are the Sysplex equivalent of COUNTER in the Common System Area (CSA) of a single region CICS system.

A named counter server provides a full set of functions to define and use named counters. Each named counter consists of:

- A 16-byte name
- A current value
- A minimum value
- A maximum value.

The values are internally stored as 8-byte (double word) binary numbers, but the user interface allows them to be treated as any length from 1 to 8 bytes, typically 4 bytes.

Named counters are stored in a pool of named counters, where each pool is a small coupling facility list structure, with keys but no data. The pool name forms part of the list structure name. Each named counter is stored as a list structure entry keyed on the specified name, and each request for the next value requires only a single coupling facility access.

Warning: The counters are lost if the coupling facility fails. See the *Application Programming Guide* for details of recovery techniques.

Named counter structures and servers

Within each MVS image, there must be one named counter server for each named counter pool accessed by CICS regions and batch jobs in the MVS image.

Named counter pools are defined as a list structure in the coupling facility resource management (CFRM) policy. The pool name, which is used to form the server name with the prefix DFHNC, is specified in the startup JCL for the server.

Figure 42 illustrates a parallel sysplex with three CICS AORs linked to named counter servers.

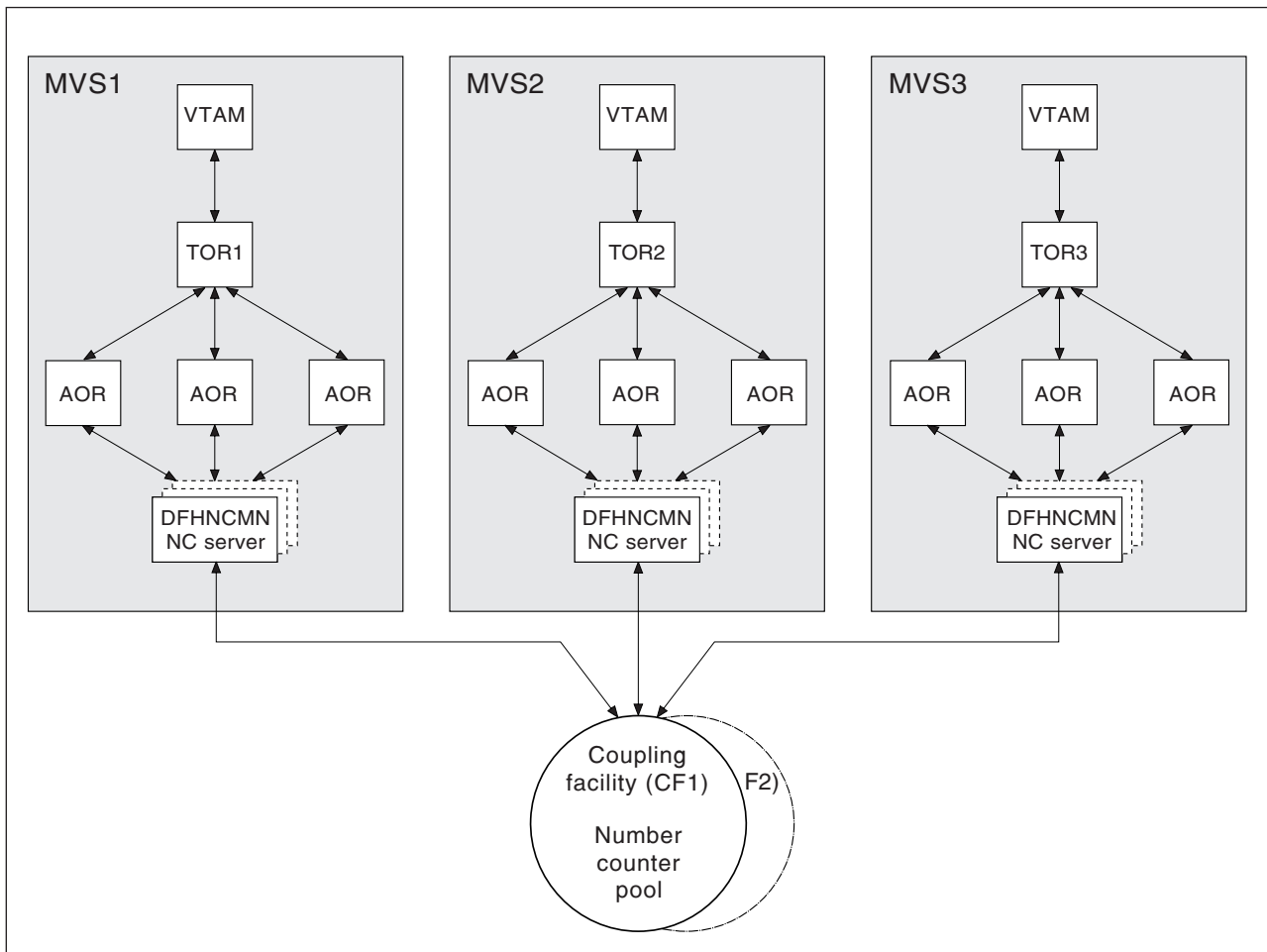


Figure 42. Conceptual view of a parallel sysplex with named counter servers

Application program access to named counters

CICS provides a command level API for the named counter facility. To reference a named counter, an application program can specify either the actual name of the pool in which the named counter is stored, or it can specify a dummy pool selection parameter. The dummy pool value is mapped to the actual pool name by the **POOL** parameter that is specified in the options table, DFHNCOPT.

Specifying a dummy pool makes it easy to use a different pool (for example, to isolate test pools from production pools) without having to change the pool selection parameter in the application program.

To vary the pool used by a CICS region, either load a different copy of the options table from STEPLIB, or use a common options table where the pool name selection is conditional on the job name and CICS APPLID, in addition to the pool name selection parameter. The options table also supports invocation of a user-specified program to select the appropriate pool given the pool selection parameter.

Security

The server must be authorized to access the coupling facility list structure in which the named counter pool is defined. The server must also be authorized to act as a named counter server.

For information on how to define the necessary authorizations see the *CICS RACF Security Guide*.

Note: You cannot control access to individual named counters.

Defining a named counter options table

The named counter callable interface determines the actual pool name in response to a DFHNCTR call by referring to the DFHNCOPT options table.

About this task

When a pool selector value is encountered for the first time, the pool name is determined via the options table. The name is then saved and used for all subsequent requests for the same pool selector from the same TCB. This continues for the life of the TCB or until the NC_FINISH function is used specifying that pool selector value. CICS supplies a default DFHNCOPT in source form, which you can customize and generate using the DFHNCO macro. A typical use of the options table is to enable production and test regions to use a different counter pool without needing to change the pool name in application programs.

To avoid the need to maintain multiple versions of the options table, you can use table entries to select pools based not only on the pool selection parameter specified on the DFHNCTR call, but also on the job name and APPLID of the CICS region. You can also specify the name of a user exit program to be called to make the pool selection.

Define an options table using one or more invocations of the DFHNCO macro. Each invocation generates an options table entry that defines the pool name or user exit program to be used whenever any selection conditions specified on the entry satisfy an application program request. The first entry automatically generates the table header, including the CSECT statement. Follow the last entry with an END statement specifying the table module entry point, DFHNCOPT.

The options table parameters

The DFHNCOPT options table parameters are illustrated in the following figure.

```
DFHNCO [POOLSEL={ (generic_values) | * }, ]
        [JOBNAME={ (generic_values) | * }, ]
        [APPLID={ (generic_values) | * }, ]
        {POOL={YES|NO|name} | CALL=programname}

        Terminate the last DFHNCO entry with the
        following END statement:

END     DFHNCOPT
```

Figure 43. DFHNCOPT options table

The POOLSEL, JOBNAME, and APPLID parameters specify optional selection conditions to determine whether the entry applies to the current request. You can specify each of these operands as

- A single generic name
- A list of names in parentheses, the list containing two or more generic names, each name separated by a comma.

Each name comprises the characters that can appear on the appropriate parameter, plus the wild-card characters * to match any sequence of zero or more non-blank characters, and % to match any single non-blank character. When multiple generic name are specified, the selection condition is satisfied if any one of them matches. A blank pool selector value can be matched using a null POOLSEL operand, for example POOLSEL= or POOLSEL=().

POOLSEL={ (generic1, generic2, ..., ...) v* }

Specifies that this options table entry applies only when the pool selection parameter specified by the application program matches one of the generic names specified on this parameter.

Specifying POOLSEL=, or POOLSEL=() is the equivalent of specifying 8 blanks.

If you omit the POOLSEL keyword, it defaults to *.

JOBNAME={ (generic1, generic2, ..., ...) v* }

Specifies that this options table entry applies only when the caller's job name matches one of the generic names specified on this parameter.

If you omit the JOBNAME keyword, it defaults to *.

APPLID={ (generic1, generic2, ..., ...) v* }

Specifies that this options table entry applies only when the caller's CICS APPLID matches one of the generic names specified on this parameter.

If you omit the APPLID keyword, it defaults to *.

POOL={YES vNO vname}

Specifies the pool name to be used. This parameter is mutually exclusive with the CALL parameter. The options are:

YES specifies that the server is to use the pool selection parameter specified by the application program as the actual pool name. An all-blank pool selection parameter means the server is to use the default pool name. For the call interface, the default name is DFHNC001. For the EXEC CICS API, the default name is specified by the NCPLDFT system initialization parameter.

NO specifies that the server is not to use any pool and is to reject the request with an error.

name specifies the actual pool name that the server is to use. If *name* is omitted, this indicates that the default pool is to be used. (For the CALL interface, the default pool is always DFHNC001, but for the EXEC CICS interface you can specify the default pool using the NCPLDFT system initialization parameter.)

CALL=programname

specifies the name of a user exit program to be called to determine the actual pool name to be used. This parameter is mutually exclusive with the POOL parameter.

The program named can be link-edited with the options table, which generates a weak external reference (WXTRN), or it can be loaded dynamically the first time it is used. The program is called using standard MVS linkage in AMODE 31, with a standard save area and parameter list pointing to four fields, in the following order:

- The 8-byte actual pool name result field
- The 8-byte pool selection parameter.
- The 8-byte job name
- The 8-byte APPLID if running under CICS, otherwise blanks

The end-of-list bit is set in the last parameter address.

The program should be reentrant and should be linked with RMODE ANY, so that it (and the option table if linked with the program) can be loaded above the line. Temporary working storage can be acquired and released using MVS GETMAIN and FREEMAIN. As this program is only called when a new pool selection value is used, the use of GETMAIN and FREEMAIN should not affect performance.

The exit program cannot use any CICS services. If it is used in a CICS region, it must avoid using any MVS services which could result in a long wait, as it will normally be executed under the CICS quasi-reentrant (QR) TCB.

The user exit program indicates its result by setting one of the following return codes in register 15:

- 0** Use the pool name that is successfully set, in the first field of the parameter list, by the user exit program.
- 4** The program cannot determine the pool name on this invocation. Continue options table processing at the next entry, as for the case where selection conditions were not met.
- 8** Reject the request (as if POOL=NO was specified).

The default options table, supplied in CICSTS41.CICS.SDFHLINK, contains the following entries:

```
DFHNCO POOLSEL=DFHNC*,POOL=YES
DFHNCO POOL=
END DFHNCOPT
```

With the default options table in use, any pool selector parameter that specifies a string beginning with DFHNC is taken to be an actual pool name, indicated by POOL=YES in the table entry. Any other value, including a value of all spaces, is assigned the default pool name, indicated by the POOL= table entry without a POOLSEL parameter.

The source for this default table is supplied in CICSTS41.CICS.SDFHSAMP.

Making an options table available to CICS

To ensure that your CICS region can load the named counter options table, install the link-edited table into a CICS authorized library in STEPLIB. Alternatively you can install the table in a suitable library in the LINK list.

Defining a list structure

Define one or more coupling facility list structures for the named counter facility, each list structure representing a pool of named counters. Each named counter pool is accessed through a cross-memory server region.

Before you begin

A coupling facility structure contains not only stored data but also the information needed to manage and access that data, in a similar way to a key-sequenced data set. The amount of internal control information depends on the level of functionality and performance of the coupling facility control code for the current CFLEVEL, and might increase for a higher CFLEVEL. Ensure that you consider storage requirements for your list structures. Read “List structure storage considerations” on page 403 for more information.

About this task

Define the structure in the current coupling facility resource management (CFRM) policy using the utility IXCMIAPU. For an example of this utility, see member IXCCFRMP in the SYS1.SAMPLIB library. An example of a policy statement for a named counter pool is shown in Figure 44 on page 403.

Procedure

1. Specify the name of the list structure. The name is formed by adding the prefix DFHNCLS_ to your chosen pool name, giving DFHNCLS_ *poolname*.
2. Specify the size of the list structure. You can allocate an initial and maximum size using the INITSIZE and SIZE parameters in the CFRM policy definition. For an accurate estimate of storage requirements, use the IBM CFSizer tool at <http://www.ibm.com/systems/support/z/cfsizer/>.
3. Specify the preference list of coupling facilities in which the policy can be stored
4. When you have updated the CFRM new policy with the new structure definition, activate the policy using the MVS command:

```
SETXCF START,POLICY,POLNAME=policyname,TYPE=CFRM.
```

Note that defining the CFRM policy statements for a list structure does not create the list structure. The structure is created the first time an attempt is made to connect to it, which occurs when the first named counter server that refers to the corresponding pool is started.

Example

Your definition could look as follows:


```
STRUCTURE  NAME(DFHNCLS_PRODNC1)
           SIZE(512)
           INITSIZE(256)
           PREFLIST(FACIL01,FACIL02)
```

Figure 44. Example definition of a coupling facility list structure for named counters

Note: Before you attempt to start the named counter server make sure you have defined and started the authorized cross-memory (AXM) server environment (see Chapter 21, “Defining and starting AXM system services,” on page 347).

List structure storage considerations

All structure sizes are rounded up to the next multiple of 256KB at allocation time.

Provided that space is available in the coupling facility, you can use the MVS SETXCF command to increase the structure size dynamically from its initial size towards its maximum size, making the new space available immediately to any currently active servers. If too much space is allocated, you can reduce the structure size to free up coupling facility storage for other purposes. However, this could take some time if the coupling facility has to move existing data out of the storage that is being freed. Note that if the size is altered in this way, you should also update the INITSIZE parameter in the policy to reflect the new size, so that the structure will not revert to its original size if it is subsequently recreated or reloaded.

A coupling facility structure contains not only stored data but also the information needed to manage and access that data, in a similar way to a key-sequenced data set. The amount of internal control information depends on the level of functionality and performance of the coupling facility control code for the current CFLEVEL, and might increase for a higher CFLEVEL. For more information see “Coupling facility storage management” on page 416. The space required for a named counter pool depends on the number of different named counters you need, but the minimum size should be enough for most needs. A minimum-size structure of 256KB can hold hundreds of named counters (as of CFLEVEL=14).

For an accurate estimate of storage requirements for a list structure, use the IBM CFSizer tool at <http://www.ibm.com/systems/support/z/cfsizer/>. The CFSizer tool is a web-based application that returns structure sizes based on the latest level of the coupling facility. If you enter the number of counters you require, the tool calculates the size of a structure that would be sufficiently large to contain at least the number of counters you specified. However, for practical operation, a reasonable proportion of free space must be available, not only to minimize the risk of the structure becoming full but also in order to avoid you receiving low space warning messages. You should aim to utilize no more than approximately 75% of the structure size. When you estimate the maximum number of counters that you require, increase that number by a third to include the free space in the calculation.

Defining and starting a named counter server region

You activate a named counter pool in an MVS image by starting up a named counter server region for that pool.

About this task

You can start the server as a started task, started job, or as a batch job. The job or task must invoke the named counter server region program, DFHNCMN and must

run from an APF-authorized library. DFHNCMN is in the CICS authorized library, CICSTS41.CICS.SDFHAUTH.

Procedure

1. Specify the DFHNCMN program either in a SYSIN data set defined in the JCL, or in the **PARM** parameter on the EXEC statement.
2. Specify the mandatory and optional startup parameters for the DFHCFMN program. If you specify a startup parameter in both the SYSIN data set and the **PARM** parameter, the **PARM** value overrides the SYSIN value because the **MVS START** command can override the **PARM** value.
 - a. You must specify a SYSPRINT DD statement for the print file.
 - b. You must specify a SYSIN DD statement for the server parameters.
 - c. You must specify the TS pool name.
 - d. It is recommended that you specify the **REGION** parameter. This parameter ensures that the coupling facility data table server region has enough storage to process the maximum number of data table requests that can run concurrently.
 - e. It is recommended that you specify TIME=NOLIMIT. The server task remains in a wait during most normal processing, because server processing is performed under the TCB of the client CICS region. If you omit this parameter, your server job could fail with abend S522 (wait limit exceeded), depending on the JWT value specified in the SMFPRMxx member of SYS1.PARMLIB.
 - f. Specify additional parameters as required. For example, you might want to control the maximum number of queues that are to be supported in the pool and the number of buffers the server is to allocate.

Results

Tip: The easiest way to ensure that all pool-related parameters are consistent across MVS images is to use the same SYSIN parameter data set, or an identical copy of it, for all servers accessing the same pool, and to specify in the PARM field any parameters that vary between servers.

Example

```
//MVS $n$ NC1 JOB ...
//NCSEVER EXEC PGM=DFHNCMN,REGION=32M,TIME=NOLIMIT   named counter server
//STEPLIB DD DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR     Authorized library
//SYSPRINT DD SYSOUT=*                               Messages and statistics
//SYSIN DD *
POOLNAME=MVS $n$ NC1                                   Pool name
/*
```

Figure 45. Sample JCL to start a named counter server address space

Named counter server parameters

Parameters are specified in the form KEYWORD=*value*, where keywords can optionally be specified in mixed case to improve readability.

If you specify more than one parameter in the PARM field or on the same SYSIN input line, the parameters must be separated by a comma. Any text following one or more spaces is taken as a descriptive comment. Any parameter line which starts with an asterisk or a space is assumed to be a whole line comment.

You can enter some parameter keywords in more than one form, such as in abbreviated or truncated form.

The main parameters are listed on the server print file during startup.

named counter server REGION parameter

Use the JCL REGION parameter to ensure that the named counter server region has enough storage to process the maximum number of named counter requests that can be executing concurrently.

The named counter server typically uses less than one megabyte of storage above 16MB and less than 20KB below 16MB.

During server initialization, the server acquires all the available storage above 16MB, as determined by the REGION parameter, then releases 5% of it for use by operating system services. It also acquires 5% of the free storage below 16MB for use in routines that require 24-bit addressable storage.

After initialization, the server uses AXM page allocation services to manage its storage. Server statistics indicate how much storage is allocated and used within the storage areas above and below 16MB, which are called AXMPGANY and AXMPGLOW in the statistics.

If a task in the server region or a cross-memory request runs out of storage, this is likely to result in AXM terminating that task or request using a simulated abend with system completion code 80A to indicate a GETMAIN failure. Although the server can usually continue processing other requests in this case, running out of storage in a critical routine can cause the server to terminate. Therefore, it is best to ensure that the REGION size is large enough to eliminate this risk.

Pool name parameter

This parameter, POOLNAME, is always required:

POOLNAME=*name*

specifies the 8-character name of the named counter pool. This is appended by the server to the prefix DFHNC to create its own server name, as in DFHNC.*poolname*, and also to the prefix DFHNCLS_ to create the name of the coupling facility list structure, as in DFHNCLS_*poolname*.

This parameter is valid only at server initialization, and must always be specified.

This keyword can be abbreviated to **POOL**.

Statistics parameters

Use the following parameters to specify server statistics options:

ENDOFDAY={00:00|hh:mm}

specifies the time of day, in hours and minutes, when the server is to collect and reset end-of-day statistics.

Note: If the STATSOPTIONS parameter specifies NONE, the server still writes end-of-day statistics to the print file.

The valid range of times is from 00:00 to 24:00.

This keyword can be abbreviated to **EOD**.

STATSINTERVAL={03:00|*hh:mm*}

specifies the statistics collection interval, in the range 1 minute to 24 hours. This parameter is ignored if the STATSOPTIONS parameter specifies NONE.

The time interval can range from 00:01 to 24:00.

This keyword can be abbreviated to **STATSINT**.

STATSOPTIONS={NONE|**SMF**|**PRINT**|**BOTH**}

specifies whether the server is to produce interval statistics, and the destination for the statistics it produces.

NONE

The server does not produce any interval statistics.

SMF The server produces interval statistics and writes them to the current SMF data set only.

PRINT

The server produces interval statistics and writes them to the server's print file only.

BOTH The server produces interval statistics and writes them to the current SMF data set and to the server's print file.

This keyword can be abbreviated to **STATSOPT**.

Automatic restart manager (ARM) parameters

During server initialization, the server unconditionally registers with ARM except when the server program is invoked with either the UNLOAD or the RELOAD functions. The server will not start if the registration fails.

Use the following parameters to override default processing for the automatic restart manager:

ARMELEMENTNAME=*elementname*

specifies the automatic restart manager element name, up to 16 characters, to identify the server to ARM for automatic restart purposes. The permitted characters for the element name are A to Z 0-9 \$ # @ and the underscore symbol (_).

The default identifier is of the form DFHNC*nn*_*poolname*, where NC represents the server type, *nn* is the &SYSCClone value for the system (which can be either one or two characters), and *poolname* is the name of the pool served by the server.

This parameter is only valid at server initialization.

This keyword can be abbreviated to **ARMELEMENT** or **ARMELEMNAME**.

ARMELEMENTTYPE=*elementtype*

specifies the automatic restart manager element type, up to 8 characters for use in ARM policies as a means of classifying similar elements. The permitted characters for the element type are A to Z 0-9 \$ # and @.

The default element type is SYSCICSS.

This parameter is only valid at server initialization.

This keyword can be abbreviated to **ARMELEMENTTYPE**.

List structure parameter

These parameters specify list structure attributes. They are used only for the initial allocation of resources when the pool list structure is being created, which occurs the first time you start a server for a pool.

POOLSIZE={0|*number*}

specifies the initial amount of coupling facility storage to be allocated for the pool list structure, expressed as kilobytes in the form *number* K, megabytes in the form *number* M or gigabytes in the form *number* G.

0 The special value 0 means that the server is to obtain an initial allocation using the parameters specified in the CFRM policy. If the CFRM policy specifies an INITSIZE value for the structure, this determines the initial allocation. Otherwise the maximum SIZE value is allocated.

number

A non-zero value specifies an initial amount of storage to be allocated, overriding the **INITSIZE** parameter in the CFRM policy. This value is generally rounded up by MVS to the next multiple of 256K. The valid range of values is 1K to 16777215M, but should not be greater than the value specified on the **SIZE** parameter.

It is generally preferable to omit this parameter, and specify the structure size using the **INITSIZE** parameter in the CFRM policy. The POOLSIZE option can, however, be useful if the structure is being reallocated or reloaded, and the CFRM policy has not been updated to reflect the required size.

Note: If the value is greater than the value specified on the CFRM **SIZE** parameter, the server **POOLSIZE** parameter is ignored and the initial allocation is based on the parameters specified in the CFRM policy.

This parameter is valid only at server initialization and is only used when the structure is first allocated.

Debug trace parameters

These parameters are provided only for intensive debug tracing.

Using these options in a production environment could have a significant impact on performance and cause the print file to grow very rapidly, using up spool space.

Trace messages from cross-memory requests can be lost if they are generated faster than the trace print subtask can print them. In this event, the trace only indicates how many messages were lost.

CFTRACE={OFF|**ON**}

specifies the coupling facility interface debug trace option.

OFF Coupling facility interface debug trace is disabled.

ON Coupling facility interface debug trace produces trace messages on the print file, indicating the main parameters to the coupling facility request interface, and the result from the IXLLIST macro.

This keyword can also be specified as **TRACECF**.

RQTRACE={OFF|**ON**}

specifies the request debug trace option.

- OFF** Request debug trace is disabled.
- ON** Request debug trace produces trace messages on the print file, indicating the main parameters on entry to each cross-memory request, and the results on exit.

This keyword can also be specified as **TRACERQ=**.

Warning parameters

Use these parameters to modify the thresholds at which warning messages are issued when the structure becomes nearly full.

ENTRYWARN={80|number}

specifies the percentage of list structure entries in use at which warning messages should be first triggered.

The valid range is from 1 to 100 per cent.

ENTRYWARNINC={5|number}

specifies the percentage increase (or decrease) of entries in use before the next warning message should be triggered (reduced to 1 when the next increase would otherwise reach 100). After the first warning, additional messages are issued as the number of elements increases and decreases. These messages stop when the number of entries in use has fallen at least this percentage below the initial warning level.

The valid range is from 1 to 100 per cent.

Controlling named counter server regions

You can issue commands to control a named counter server, using the **MVS MODIFY (F)** command to specify the job or started task name of the server region, followed by the server command.

About this task

The general form of a named counter server command, using the short form **F**, is as follows:

```
F server_job_name,command
parameters... comments
```

SET keyword=operand[,keyword=operand,...]

Change one or more server parameter values.

Procedure

- To change one or more server parameter values, use the **SET** command. You can abbreviate the command to **T**, as for the **MVS SET** command. See “The SET command options” on page 409 for details.
- To display one or more parameter values or statistics summary information on the console, use the **DISPLAY** command. The syntax is as follows:

```
DISPLAY keyword[=operand][,keyword[=operand],...]
```

The valid keywords for **DISPLAY** are all the initialization parameters, plus an additional set described under “DISPLAY and PRINT command options” on page 409. You can abbreviate the command to **D**, as for the **MVS DISPLAY** command.

- To print the parameter values, use the **PRINT** command. This command produces the same output as the **DISPLAY** command, supporting the same keywords, but on the print file only.

- To stop the server normally, use the **STOP** command. The server waits for any active connections to end first, and prevents any new connections while it is waiting. You can abbreviate the command to **P**. You can also use the **MVS STOP** command, which is equivalent to issuing the server STOP command through the **MVS MODIFY** command. The syntax of the **STOP** command is:
`STOP|P [jobname.]identifier[,A=asid]`
- To terminate the server immediately, use the **CANCEL** command. You can also specify whether the server automatically restarts with the **RESTART** option. For information about **CANCEL RESTART** see “The **CANCEL** command options” on page 360.
- The server also responds to XES events such as an operator **SETXCF** command to alter the structure size. If the server can no longer access the coupling facility, it automatically issues a server **CANCEL** command to close itself down immediately.

The SET command options

You can use the **SET** command to modify groups of server initialization parameters.

The groups of server initialization parameters are as follows:

- The statistics parameters
- The debug trace parameters
- The warning parameters

See “Named counter server parameters” on page 404 for details of these keywords.

Examples of the SET command: The following example changes the statistics options:

```
SET STATSOPT=BOTH,EOD=21:00,STATSINT=06:00
```

DISPLAY and PRINT command options

You can use the **DISPLAY** and **PRINT** commands to display the values of any initialization parameters plus some additional information.

The parameters supported by the **DISPLAY** and **PRINT** commands are as follows:

ARMREGISTERED

Shows whether ARM registration was successful (YES or NO).

CONNECTIONS

Display the job names and APPLIDs of the regions currently connected to the server to which the command is issued.

This keyword can be abbreviated to **CONN**.

COUNTERS

Display the names of all the named counters currently allocated in a pool.

COUNTERS={name|generic_name}

Display the details of a specific named counter, or set of named counters whose names match the generic name. Generic names are specified using the wildcard characters * (asterisk symbol) and % (per cent symbol).

If no named counter is specified, this is treated as equivalent to **DISPLAY COUNTERS**.

This keyword can be abbreviated to **COUNTER**.

DISPLAY and PRINT options for statistics summaries

Use the following parameters to display or print statistics:

CFSTATS

Display statistics for coupling facility interface accesses and responses from the server.

This keyword can also be specified as **CFST** or **STATSCF**.

POOLSTATS

Display usage statistics for the pool list structure as a whole. This is based on information returned by coupling facility access requests, therefore it is only as current as the most recent request made through the server to which the command is issued.

This keyword can be abbreviated to **POOLST**.

STORAGESTATS

Display main storage allocation statistics for the server address space.

This keyword can be abbreviated to **STORAGEST** or **STGST**.

DISPLAY and PRINT options for combined lists of information

These keywords represent combined lists of information:

PARAMETERS

Display the main parameter values:

- POOLNAME
- STATSOPT
- ENDOFDAY
- STATSINTERVAL
- POOLSIZE

This keyword can be abbreviated to **PARM** or **PARMS**.

ALLPARAMETERS

Display all parameter values, which are those listed for PARAMETERS, as well as the following:

- CFTRACE
- RQTRACE
- ENTRYWARN
- ENTRYWARNINC

This keyword can be abbreviated to **ALLPARMS**.

STATISTICS

Display all available statistics. This keyword can be abbreviated to **STAT** or **STATS**.

INITIALIZED

Display the parameters and statistics that are usually displayed when initialization is complete, which are those listed for PARAMETERS, as well as the following:

- POOLSTATS
- STGSTATS

This keyword can be abbreviated to **INIT**.

ARM

Display all ARM-related parameter values:

- ARMELEMENTNAME
- ARMELEMENTTYPE
- ARMREGISTERED

This keyword can be coded as **ARMSTATUS**.

The CANCEL command options

You can use the CANCEL command to request an automatic restart.

Specify the following parameter:

RESTART={NO|YES}

Terminate the server immediately, specifying whether or not automatic restart should be requested. The default is RESTART=NO.

If the server encounters an unrecoverable problem with the coupling facility connection, consisting either of lost connectivity or a structure failure, it cancels itself using the CANCEL RESTART=YES command. This terminates the existing connection and shuts down the server. A new instance of the server job is then started.

A server can also be restarted explicitly using either the server command CANCEL RESTART=YES or the MVS command CANCEL jobname,ARMRESTART.

You can also enter RESTART on its own for RESTART=YES, NORESTART for RESTART=NO.

Deleting or emptying named counter pools

You can delete a named counter pool using the MVS **SETXCF** command to delete its coupling facility list structure.

About this task

For example:

```
SETXCF FORCE,STRUCTURE,STRNAME=DFHNCLS_poolname
```

You can delete a structure only when there are no servers connected to the pool, otherwise MVS rejects the command.

When you attempt to start a server for a pool that has been deleted (or attempt to reload the pool), it is allocated as a new structure. The newly allocated structure uses size and location attributes specified by the currently active CFRM policy.

Changing the size of named counter pools

If the structure is becoming full, and the current pool size is less than the maximum, you can use the SETXCF START,ALTER command to increase the pool size.

About this task

For example:

```
SETXCF START,ALTER,STRNAME=DFHNCLS_poolname,SIZE=size
```

SIZE is expressed in kilobytes.

Unloading and reloading named counter pools

You can unload, and reload, the complete contents of a named counter pool to and from a sequential data set by invoking the server program with the **FUNCTION** parameter, using the UNLOAD and RELOAD options.

About this task

You can use this function, for example, to:

- Preserve the named counter pool during planned coupling facility maintenance, or
- Move the pool to a different coupling facility.

FUNCTION={UNLOAD|RELOAD}

Specify the function for which the server is being initialized.

UNLOAD

Unload the entire contents of the named counter pool specified on the POOLNAME parameter to a sequential data set. When the unload processing has completed (normally or abnormally) the server program terminates.

The UNLOAD function requires a DD statement for DDNAME DFHNCUL describing the sequential data set to which the table pool is to be unloaded. The format of the unloaded data set is:

```
RECFM=F  
LRECL=4096  
BLKSIZE=4096
```

RELOAD

Reload, into the named counter pool named on the POOLNAME parameter, a previously unloaded named counter pool.

The RELOAD function requires a DD statement for DDNAME DFHNCRL, describing the sequential data set from which the table pool is to be reloaded.

The structure is allocated, if necessary, during reloading, in which case you can use the same server parameters to control structure attributes as for normal server startup. The reload process bypasses named counters that are already found in the pool (for example, because the structure was too small and the reload job had to be restarted after using ALTER to increase the structure size).

Note: If the unloaded pool structure was altered dynamically at any time after initial allocation (by using the SETXCF command to increase the size), ensure that the increased size is allocated for the reloaded pool. The recommended way is to update the INITSIZE parameter for the structure in the current CFRM policy whenever you alter the structure size, and to activate the updated policy using the SETXCF START,POLICY command. Alternatively, you can specify the required pool size in the POOLSIZE parameter in the reload JCL.

Note: If you omit the FUNCTION parameter, the server program initializes a named counter server address space.

For the UNLOAD and RELOAD function, the server program requires exclusive use of the list structure. If the structure is currently being used by a normal server, the unload or reload attempt is rejected. Similarly, if a normal server attempts to start up while an unload or reload job is in progress, the attempt fails because shared access to the structure is not available.

You can specify all normal server parameters when unloading or reloading, but some of these (for example, statistics-related parameters) are ignored because they do not apply to unload or reload processing.

If reloading fails because it runs out of space, the resulting messages include the numbers of named counters reloaded and blocks read up to the time of the failure. You can compare these values with those in the messages from the original unload job, to determine how many more named counters remain to be loaded.

Unload JCL example

The JCL in the following example shows you how to unload a named counter pool.

```
//UNLDNCD1 JOB ...
//NCUNLOAD EXEC PGM=DFHNCMN          CICS named counter server program
//STEPLIB DD DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR Authorized library
//SYSPRINT DD SYSOUT=*              Options, messages and statistics
//DFHNCUL DD DSN=NC1.UNLOADED.POOL,  Unloaded named counter pool
//          DISP=(NEW,CATLG),
//          SPACE=(4096,(10000,1000)) Estimated size in 4K blocks
//SYSIN DD *
FUNCTION=UNLOAD                      Function to be performed is UNLOAD
POOLNAME=PRODNC1                    Pool name
/*
```

Figure 46. Unload JCL example

Reload JCL example

The following JCL example shows you how to reload a named counter pool.

```
//RELDNCD1 JOB ...
//NCRELOAD EXEC PGM=DFHNCMN          CICS named counter server program
//STEPLIB DD DSN=CICSTS41.CICS.SDFHAUTH,DISP=SHR Authorized library
//SYSPRINT DD SYSOUT=*              Options, messages and statistics
//DFHNCRL DD DSN=NC1.UNLOADED.POOL,DISP=OLD Unloaded pool
//SYSIN DD *
FUNCTION=RELOAD                      Function to be performed is RELOAD
POOLNAME=PRODNC1                    Pool name
/*
```

Figure 47. Reload JCL example

Dumping named counter pool list structures

You can use the MVS **DUMP** command to obtain a dump of the coupling facility list structure for a named counter pool.

About this task

For information on dumping and formatting a list structure, see the *CICS Problem Determination Guide*.

Chapter 26. Coupling facility server operations

The operations that you can perform on all three CICS coupling facility servers, for temporary storage, coupling facility data tables and named counters, are similar and the following information describes all three unless otherwise indicated.

Monitoring coupling facility server messages

The server issues various messages during execution, some of which might indicate that serious problems are developing, for example that the coupling facility structure is becoming full.

It is important to understand the types of messages that the server issues and to ensure that system status messages are monitored for possible problems.

Server messages

Messages that are issued by the server code itself start with the five-letter server prefix (DFHXQ, DFHCF or DFHNC).

These messages fall into the following groups:

- Operator console system status messages, which are issued by WTO (Write To Operator) with routing codes 2 (operator information) and 11 (programmer information), and descriptor code 4 (system status).

These messages provide information about important status changes, primarily about the beginning and end of server initialization and the beginning and end of server termination, and about problems.

Any server message that is issued in this way during normal running indicates a potentially serious problem, and should not be ignored. If this process is automated, a simple rule is to ignore the specific list of status messages for normal initialization and termination, and to treat any other server message as a warning.

These messages can be issued either from the server address space or from a client address space. If the server code that requests the message is running in cross-memory mode, the message is passed back to a routine that issues the WTO in primary mode in the client address space. This avoids restrictions which apply to WTO messages that are issued in cross-memory mode. For example, cross-memory mode WTO messages do not appear in any job log.

- Command responses, which are issued by WTO with routing codes 2 and 11 and descriptor code 5 (immediate command response).

These messages contain responses to server commands that are issued via the system MODIFY or STOP command, for example to display statistics.

- Job log messages:

These messages typically contain diagnostic information, for example details of an abend or an attempted security violation. They are written to the job log, by WTO with routing code 11 (programmer information).

- Trace and statistics messages:

These messages are written only to the server SYSPRINT file.

All messages that are issued by the server code, whether they are running under the server address space or in cross-memory mode from the client address space, are also copied to the server SYSPRINT file.

AXM messages

The AXM environment code issues operator messages from the server and client address spaces and from the master address space during AXM system services initialization.

These messages are issued using WTO with routing codes 2 (operator information) and 11 (programmer information, not used when running in the master address space), and descriptor code 4 (system status). AXM message numbers are of the form "AXMxxnnns", where the first five characters "AXMxx" are the name of the module issuing the message, "nnnn" is the numeric part of the message number, and "s" is the suffix letter. The suffix letter is "I" if the message is a routine informational message, or is omitted if the message indicates an error. The suffix letter can be used by automation tools to distinguish routine informational messages from error situations.

AXM messages that are issued from the server environment (via AXM run-time environment routines linked with the server load module), are copied to the server SYSPRINT file as well. AXM also writes informational messages to the SYSPRINT file. These contain information such as initialization information and closedown statistics for storage management, and the main procedure entry point of the server module for diagnostic purposes.

Coupling facility storage management

The type of coupling facility structure that CICS uses for its temporary storage data sharing pools, coupling facility data table pools and named counter pools is a keyed list structure.

This contains an array of numbered lists, each of which is like a keyed file, containing entries with 16-byte keys. Each entry has a fixed prefix area containing its key and other control information that includes an "adjunct area" of 64 bytes for program use, followed by a chain of up to 128 256-byte data elements. This allows a maximum data size of 32K bytes. For named counters, only the prefix area is used, with no data elements, and there is only one list. Storage for entry prefixes, known as entry controls, and data elements is allocated from two storage pools within the structure, which are shared between all lists in the structure.

The storage in a list structure can be divided into two basic types: fixed controls and variable controls. Storage for fixed controls is preallocated at a fixed size for the life of the structure. It contains structure control information, including data buffer space, as well as an array of list headers, up to the maximum number of lists defined when the structure was created. For CICS, it consists of a small number of lists used for CICS internal purposes, and the value specified for the parameters **MAXQUEUES** or **MAXTABLES**.

Variable controls are partitioned into storage areas for entry controls, one per entry, and for data elements. You can dynamically adjust this partitioning by altering the ratio of entries to elements, converting storage for one type to the other type. CICS automatically issues a request to alter the ratio when one type of variable storage is running out but there is enough storage of the other type. You can dynamically alter the total size of the storage for variable controls by changing the size of the

structure. This must be within the range of sizes that are defined for the structure in the active CFRM policy, which are set at the time that the structure is created. You can change the structure size by using the system operator command SETXCF ALTER,SIZE. Alternatively the operating system can change the structure size, depending on the automatic alter options that are specified in the CFRM policy when the structure is allocated.

The fixed controls include enough internal control areas to manage the maximum number of elements and entries that can exist in the structure, given the maximum structure size and the range of possible ratios of entries to elements, and an array of list headers to handle the maximum number of lists that can exist in the structure. As neither the maximum size nor the maximum number of lists can be increased without reallocating the structure, this means that you must be careful to specify large enough values when the structure is first allocated. However, if you specify a relatively large maximum size or number of lists, a large amount of storage is preallocated for fixed controls; so the initial size of the structure needed to store a given amount of data will be significantly larger than it would be with less generous allowance for expansion.

Within the CICS pool structures, each queue item, data table record or named counter normally occupies one entry in the structure, together with the appropriate number of data elements. Additional entries are used by CICS for internal control purposes to maintain the index of currently defined queues or data tables, as well as tracking which lists are currently in use and which previously used lists are now free and available for reuse. Note that even after all queues or data tables in a pool have been deleted, some entries in the control lists might remain in use.

The amount of storage required for internal control information depends on the level of functionality and performance of the coupling facility control code for the current CFLEVEL, and may increase for a higher CFLEVEL. The easiest way to calculate storage requirements for the CICS pool structures is to use the web-based IBM CFSizer tool at <http://www.ibm.com/systems/support/z/cfsizer/>. This tool prompts for parameters that describe the amount of information to be stored in the structure in terms of the corresponding CICS resources, converts this information to numbers of lists, entries and elements and then communicates with a coupling facility at a current CFLEVEL to determine the amount of storage required to store the information with the specified amount of free space and room for expansion. For more information about calculating storage requirements, see “Defining temporary storage pools for temporary storage data sharing” on page 17.

Managing the pool structure

It is important to watch out for signs that the pool structure is becoming full, as this might have a serious impact on all the applications that are using the pool.

Monitoring pool structure usage levels

Use the DISPLAY POOLSTAT server command to display the current usage level of the pool structure.

The DISPLAY POOLSTAT command produces messages DFHXQ0432I, DFHF0432I or DFHNC0432I. The most important information in these messages is the maximum percentage of lists, entries, and elements, used during the current statistics interval. For the named counter server, message DFHNC0432I only shows the number of entries because there is always one list and no elements.

Operator messages reporting on pool structure usage

Messages to the operator, for example DFHXQ0411I and DFHXQ0412I, are issued when threshold levels are reached for the numbers of entries or elements used.

Further messages are issued if the pool becomes full. You can set up automated operations processes to watch for these messages and alert your operators to the situation, or take corrective action, if necessary.

You can use the MVS operator command SETXCF ALTER,START with an increased SIZE option, to expand the structure, if the structure has not yet reached the maximum size that is defined in the CFRM policy.

Use of CFRM automatic ALTER to increase pool structure size

The coupling facility resource management (CFRM) policy can specify the keyword ALLOWAUTOALT(YES).

This allows the operating system to issue an ALTER command automatically when the structure is near to becoming full, to increase its size or to adjust the ratio of elements to entries. The threshold at which this happens is specified by the FULLTHRESHOLD keyword in the policy. The default threshold is 80%, which is the same as the default threshold at which the server itself issues an automatic ALTER command to optimize the ratio of entries to elements. The server's automatic ALTER process is more sophisticated, because it takes into account the peak usage of the structure within the current statistics interval, rather than just the current usage. Therefore it is best to ensure that the server's automatic ALTER process is triggered first, by making sure that the percentage values of the server's ENTRYWARN and ELEMENTWARN parameters are at least 5 percent less than the percentage value of the CFRM FULLTHRESHOLD keyword.

Using system-managed rebuild to increase pool structure size

If the structure has not many entries or elements left, but it has already reached its maximum size, you can still use system-managed rebuild to expand it dynamically without closing down the servers, if all the systems using the structure are at a level which supports this function.

First update the CFRM policy to increase the size to the required value, and then activate the updated policy using SETXCF START,POLICY. After this, you can rebuild the structure. The rebuild allocates a new instance of the structure that uses the updated policy, copies across the existing data, and then discards the old instance.

Increasing the number of data lists

If the number of data lists specified via the MAXQUEUES or MAXTABLES server parameter is too small, an attempt to allocate a new data list will fail with message DFHXQ0443 or message DFHCF0443.

There is no way to increase the number of lists without deleting and recreating the structure, which necessitates closing down all of the servers temporarily. You cannot use system-managed rebuild to increase the number of data lists because it copies this number from the existing structure.

You can preserve existing data by using the server program to unload it to a sequential file, and then using SETXCF FORCE to delete the existing structure. You

can then use the server program again to reload the data, and allocate a new structure with the appropriate MAXQUEUES or MAXTABLES parameter.

Deleting or emptying the pool structure

If the pool structure is no longer required, or all of the data in the structure is to be discarded, you can delete a pool by closing down all the servers for that pool, and then using the SETXCF FORCE command to delete the structure.

If the server is subsequently started again for the same structure name, an empty structure will be created using the information in the active CFRM policy and the server initialization parameters.

Server connection management

A client CICS region establishes a cross-memory connection to a server the first time a request refers to the pool for that server. A single connection is established to each server, regardless of the number of tables, queues or counters that are accessed in the pool.

For coupling facility data table and temporary storage queue servers, a multi-threaded asynchronous connection is established. This connection allows requests to be overlapped up to a fixed maximum number of concurrent requests. Requests that exceed this maximum number are queued within the CICS region until a request thread becomes available.

For named counter servers, requests are processed synchronously using a single-threaded interface, and only one request can be active at a time for a given client region.

Terminating server connections

Each connection has a client side and a server side. If either side is terminated separately, the connection is no longer usable, although the other side might not be terminated until some time later.

A connection normally remains active until CICS is closed down. The connection is closed automatically as part of the resource management termination processing for the CICS quasi-reentrant TCB. During resource management termination, the connection termination routine on the client side issues a cross-memory call to the server to terminate the connection on the server side as well.

If you want to close down a server after all the CICS regions that are using it have terminated, for example when you are preparing to close down the system, first quiesce the server and then terminate it using the server STOP command (or the equivalent MVS STOP command). Use of the STOP command prevents any new CICS regions from connecting to the server and terminates the server as soon as all the CICS regions that are using it have been terminated.

If you need to close down a server immediately, while CICS regions are still connected to it, use the server CANCEL command, because the normal server STOP command, (or the MVS STOP command) is not effective until the connections have been terminated. You can also use the MVS CANCEL command to terminate the server, but this prevents the server from going through normal closedown processing.

Note that CICS cannot terminate the client side of the connection automatically. CICS has no way of knowing that the server wants to close down, because although the connection allows CICS to make cross-memory calls to the server, it does not provide any means for the server to notify CICS of asynchronous events. At present, CICS does not provide any means of terminating a connection on demand except in the case of the named counter server CALL interface which provides a FINISH function for this purpose, but this function is primarily for batch use.

If you terminate the server with CANCEL, the server side of each connection is terminated immediately, but the client side is not affected. The next request from CICS to use the original connection fails and CICS tries to establish a new connection instead. This succeeds if the server has been restarted. However, CICS does not close the old connection explicitly, so when it eventually terminates, messages are produced not only for the termination of any active connection, but also for the termination of any previous connections to the same server.

Failed server connections

If CICS terminates abruptly, without going through the normal resource manager termination processing for the quasi-reentrant TCB, for example because of a FORCE command or system completion code 40D, an end-of-memory resource manager routine cleans up the client side of the connection.

Because this routine does not run in the CICS region itself, it cannot use the cross-memory connection to notify the server. Therefore, if CICS terminates without carrying out the normal resource manager termination processing for the quasi-reentrant TCB, the server side of the connection might remain active. For coupling facility data tables this might cause problems because the server does not allow the original CICS region to resynchronize after restart if its APPLID is already in use.

From time to time the server checks the client status for each connection, and cleans up the server side of the connection if the client has gone away. This check is done once a minute, and it is also triggered each time a new connection is to be established. Therefore, any connections which have failed are normally cleaned up before the new connection attempts to resynchronize. Because the clean-up processing that terminates the server side of a connection is asynchronous, and might take one or two seconds, the clean-up processing might not be finished in time for the resynchronization from the original CICS region to succeed immediately, but if the resynchronization does not succeed on the first attempt, it should succeed when it is next retried.

Restarting a server

All three types of CICS data-sharing server (temporary storage, coupling facility data tables, and named counters) support automatic restart using the services of the Automatic Restart Manager (ARM).

The servers also have the ability to wait during start-up, using an Event Notification Facility (ENF) exit, for the coupling facility structure to become available if the initial connection attempt fails.

The server normally registers at start-up with ARM, so that it will be restarted automatically if it fails, subject to any rules in the installation ARM policy. If ARM

registration fails for any reason except for ARM being unavailable, the server cannot be started. If ARM is unavailable, the server starts normally but has to be restarted manually if it fails.

The servers recognize the ARM return code that indicates that the ARM couple data set has not been formatted for the current MVS system, as being equivalent to ARM being unavailable.

A server does not start if registration fails with return code 8 or above.

When a server starts up, if it is unable to connect to its structure because of some environmental error such as a structure failure, it automatically waits for the structure to become available, using the Event Notification Facility (ENF) to watch for events relating to its structure. This wait occurs before the cross-memory interface is enabled, so the server is not visible to client regions at this time and will appear to be unavailable. While it is waiting, the server can be cancelled using the MVS CANCEL command if it is no longer required.

If the server is running normally, but the coupling facility interface reports a loss of connectivity or a structure failure, the server immediately terminates itself. This disconnects it from the coupling facility, and terminates the server side of any current cross-memory connections from client regions. The server will normally be restarted immediately by the ARM, but will continue to be unavailable to client regions until the coupling facility structure is available again (possibly as a new empty instance of the structure).

An abrupt coupling facility failure such as a power failure may result in a loss of connectivity indication even though the structure has failed, because the operating system cannot determine the state of the structure in that case. This could prevent a new structure from being allocated until the operating system can determine the status of the existing structure, for example after the failed coupling facility has been successfully restarted. If it is certain that the old structure has been lost, but the system has not yet recognized the fact, the operator may be able to save some time by issuing the SETXCF FORCE command to delete the old structure, allowing the system to go ahead and create a new instance of the same structure in a different coupling facility.

You can find more information about automatic restart of coupling facility servers in the *CICS Recovery and Restart Guide*

Chapter 27. CICS server support for system-managed processes

The three servers, temporary storage data sharing, coupling facility data tables, and named counters, support system-managed processes for coupling facility list structures.

These system-managed processes are:

- “System-managed list structure rebuild”
- “System-managed list structure duplexing” on page 425

System-managed list structure rebuild

System-managed rebuild allows z/OS to manage the moving of coupling facility structures used for shared temporary storage, coupling facility data tables, and named counter server pools, without recycling the CICS system using them.

Before this, you could move a structure by using server functions to UNLOAD to a sequential data set and then RELOAD elsewhere from the data set, but the switch caused errors in CICS such that restart was recommended, which in some situations was an unacceptable outage.

System-managed rebuild rebuilds the contents of a coupling facility list structure to a new location. The only impact when rebuild occurs is that requests are temporarily suspended within MVS during the few seconds or tens of seconds needed for rebuild processing. The system-managed rebuild process rebuilds only from one structure to another. Therefore, it is not applicable when the original structure has been lost or damaged. It is very useful for planned maintenance and is used for recovery purposes where connectivity to the structure has been lost from one or more systems but at least one system still has access to the original structure. For a loss of connectivity, the system does not initiate a system-managed rebuild automatically, regardless of connectivity options in the policy, but a rebuild can be requested using an operator command.

Any pending requests are made to wait during system-managed rebuild. Apart from the time delay, the application should not notice anything unusual when a system-managed rebuild occurs. For more information about system-managed rebuild, see *z/OS MVS Programming: Sysplex Services Guide*.

CICS supports the MVS system-managed rebuild facility, provided that the SUSPEND=FAIL option of the IXLCONN macro is available. The CICS servers detect automatically whether this support is available.

- If SUSPEND=FAIL support is available, a server connects to its list structure specifying the ALLOWAUTO=YES option of the IXLCONN macro.
- If SUSPEND=FAIL is not available, a server connects with ALLOWAUTO=NO, and the system-managed rebuild facility is not used.

When a server is active, you can use the MVS operator DISPLAY XCF,STR command to display the connection details in message IXC360I.

For example, to look at the connection details for structure DFHXQLS_PRODTSQ1, use the following command:

```
Display XCF,STR,STRNAME=DFHXQLS_PRODTSQ1,CONNAME=ALL
```

To look at the connection details for coupling facility data tables structure DFHCFLS_DTPOOL1, use the following command:

```
Display XCF,STR,STRNAME=DFHCFLS_DTPOOL1,CONNAME=ALL
```

To look at the connection details for the named counter structure DFHNCLS_PRODNC1, use the following command:

```
Display XCF,STR,STRNAME=DFHNCLS_PRODNC1,CONNAME=ALL
```

Using the above command, the resulting IXC360I message output contains the following lines (under the CONNECTION information section), indicating that system-managed rebuild is enabled for the connection:

```
ALLOW AUTO      : YES
SUSPEND         : FAIL
```

TS data sharing and CFDT servers

For temporary storage data sharing and coupling facility data tables, any pending requests are made to wait during system-managed rebuild. Apart from the time delay, the application should not notice anything unusual when a system-managed rebuild occurs.

Timeout considerations

The wait-time for a system-managed rebuild is expected to vary from a few seconds for a small structure to a few tens of seconds for a large structure. This means that transactions can be purged by DTIMOUT, or by an operator command, while waiting on the rebuild.

To minimize the risk of unnecessary problems caused by timeouts during syncpoint processing, the CICS wait exit for CFDT unit of work control functions specify that the wait is not purgeable.

Note: Making the wait for CFDT unit of work control functions non-purgeable does not apply to CICS TS 1.3 regions connected to a later level of the CFDT server that supports system-managed rebuild. In this case, if a transaction makes any recoverable changes to a coupling facility data table before being purged, it will probably go into a wait again during syncpoint processing. If this happens, it should be left waiting until the rebuild completes, because any further attempt to purge it could result in the UOW being shunted as having failed during syncpoint. If the UOW is shunted in these circumstances, it will require manual intervention to retry syncpoint processing and complete the UOW when the structure becomes available again.

Named counter server

Requests issued using the CALL interface to the named counter server are not made to wait during rebuild, but instead the server returns a new environment error return code with value 311, under the NC_ENVIRONMENT_ERROR category.

An EXEC interface request during rebuild waits using a timer wait and retry loop which can be interrupted by an operator purge command but is not eligible for DTIMOUT, since the wait is definitely known not to be caused by any form of deadlock.

Compatibility considerations

The named counter client region interface module DFHNCIF normally resides in a linklist library and should be at the highest level of CICS TS in use within an MVS image.

If a CICS region is running a CICS TS 2.2 or later level of DFHNCIF (from the link list) but the EXEC CICS interface is from CICS TS 2.1 or earlier, the named counter server does not wait and retry during rebuild, but instead returns INVREQ with RESP2 code 311.

If a program running in a region that is using a CICS TS 2.1 or earlier DFHNCIF issues a request to a CICS 2.2 or later level of the server during a rebuild, CICS returns error code 301, UNKNOWN_ERROR. For the EXEC interface, this is mapped to an INVREQ with RESP2 code 301.

System-managed list structure duplexing

System-managed coupling facility duplexing provides a general-purpose, hardware-assisted mechanism for duplexing coupling facility structure data.

It is a robust mechanism that provides recovery from failures such as loss of a single structure or coupling facility, or from loss of connectivity, by a rapid switch to the other structure in a duplex pair.

Unlike system-managed rebuild, which can rebuild only from one structure to another, and is not applicable when the original structure has been lost or damaged, system-managed duplexing creates and maintains a duplexed copy of a structure in advance of any failure. It is largely transparent to the users of the structure, and is available for both planned and unplanned outages of a coupling facility or structure. For example, until CICS TS 2.2, coupling facility data tables and temporary data sharing structures have contained mostly “scratch-pad” information that is not critical to recover. With duplexing, you can use the coupling facilities to store more critical information.

Transactions that are accessing a duplexed structure might experience delays, which could result in the DTIMOUT threshold being reached, when:

- A structure is quiesced by MVS while duplexing is being established.
- A structure is quiesced as a result of an operator command to stop or start duplexing.
- A structure is switched to the secondary structure or back to the primary structure.

A newly-started data sharing server is not allowed to connect to a structure during any phase of a duplexing rebuild until the duplex established phase is reached.

System-managed duplexing is built on the system-managed rebuild function. However, whereas system-managed rebuild requires OS/390 with APAR OW39892, system-managed duplexing requires z/OS Version 1 Release 2 with an enabling APAR. You also need to specify the DUPLEX option, in the CFRM policy information for the structure, as either DUPLEX(ENABLED) or DUPLEX(ALLOWED). If you specify DUPLEX(ENABLED), MVS will initiate and attempt to maintain a user-managed duplexing operation for the structure. If you specify DUPLEX(ALLOWED), an operator can start duplexing, using the SETXCF

START,REBUILD,DUPLEX command. You also need to ensure that coupling facilities that are taking part in duplexing operations communicate with one another through a “peer link”.

For more information about system-managed duplexing, see *OS/390 MVS Programming: Sysplex Services Guide*, GC28-1771, and *OS/390 MVS Setting Up a Sysplex*, GC28-1779.

Part 4. Appendixes

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply in 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 publication 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.

Licensees of this program who want 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 United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN.

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 document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other product and service names might be trademarks of IBM or other companies.

Bibliography

CICS books for CICS Transaction Server for z/OS

General

CICS Transaction Server for z/OS Program Directory, GI13-0536
CICS Transaction Server for z/OS What's New, GC34-6994
CICS Transaction Server for z/OS Upgrading from CICS TS Version 2.3, GC34-6996
CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.1, GC34-6997
CICS Transaction Server for z/OS Upgrading from CICS TS Version 3.2, GC34-6998
CICS Transaction Server for z/OS Installation Guide, GC34-6995

Access to CICS

CICS Internet Guide, SC34-7021
CICS Web Services Guide, SC34-7020

Administration

CICS System Definition Guide, SC34-6999
CICS Customization Guide, SC34-7001
CICS Resource Definition Guide, SC34-7000
CICS Operations and Utilities Guide, SC34-7002
CICS RACF Security Guide, SC34-7003
CICS Supplied Transactions, SC34-7004

Programming

CICS Application Programming Guide, SC34-7022
CICS Application Programming Reference, SC34-7023
CICS System Programming Reference, SC34-7024
CICS Front End Programming Interface User's Guide, SC34-7027
CICS C++ OO Class Libraries, SC34-7026
CICS Distributed Transaction Programming Guide, SC34-7028
CICS Business Transaction Services, SC34-7029
Java Applications in CICS, SC34-7025

Diagnosis

CICS Problem Determination Guide, GC34-7034
CICS Performance Guide, SC34-7033
CICS Messages and Codes, SC34-7035
CICS Diagnosis Reference, GC34-7038
CICS Recovery and Restart Guide, SC34-7012
CICS Data Areas, GC34-7014
CICS Trace Entries, SC34-7013
CICS Supplementary Data Areas, GC34-7015
CICS Debugging Tools Interfaces Reference, GC34-7039

Communication

CICS Intercommunication Guide, SC34-7018
CICS External Interfaces Guide, SC34-7019

Databases

CICS DB2 Guide, SC34-7011
CICS IMS Database Control Guide, SC34-7016

CICSplex SM books for CICS Transaction Server for z/OS

General

CICSplex SM Concepts and Planning, SC34-7044
CICSplex SM Web User Interface Guide, SC34-7045

Administration and Management

CICSplex SM Administration, SC34-7005
CICSplex SM Operations Views Reference, SC34-7006
CICSplex SM Monitor Views Reference, SC34-7007
CICSplex SM Managing Workloads, SC34-7008
CICSplex SM Managing Resource Usage, SC34-7009
CICSplex SM Managing Business Applications, SC34-7010

Programming

CICSplex SM Application Programming Guide, SC34-7030
CICSplex SM Application Programming Reference, SC34-7031

Diagnosis

CICSplex SM Resource Tables Reference, SC34-7032
CICSplex SM Messages and Codes, GC34-7035
CICSplex SM Problem Determination, GC34-7037

Other CICS publications

The following publications contain further information about CICS, but are not provided as part of CICS Transaction Server for z/OS, Version 4 Release 1.

Designing and Programming CICS Applications, SR23-9692
CICS Application Migration Aid Guide, SC33-0768
CICS Family: API Structure, SC33-1007
CICS Family: Client/Server Programming, SC33-1435
CICS Family: Interproduct Communication, SC34-6853
CICS Family: Communicating from CICS on System/390, SC34-6854
CICS Transaction Gateway for z/OS Administration, SC34-5528
CICS Family: General Information, GC33-0155
CICS 4.1 Sample Applications Guide, SC33-1173
CICS/ESA 3.3 XRF Guide, SC33-0661

Other IBM publications

The following publications contain information about related IBM products.

DB2

DB2 for OS/390 and z/OS Administration Guide, SC26-9931
DB2 for OS/390 and z/OS Application Programming and SQL Guide, SC26-9933

MVS

z/OS MVS JCL Reference, SA22-7597
z/OS MVS Initialization and Tuning Reference, SA22-7592
z/OS MVS Installation Exits, SA22-7593
OS/390 MVS Conversion Notebook, GC28-1747

OS/390 MVS System Commands, GC28-1781
OS/390 MVS Diagnosis: Tools and Service Aids, SY28-1085
OS/390 TSO/E System Programming Command Reference, SC28-1972
z/OS MVS Setting Up a Sysplex, SA22-7625
OS/390 MVS Programming: Sysplex Services Guide, GC28-1771
OS/390 MVS Programming: Sysplex Services Reference, GC28-1772
OS/390 MVS IPCS User's Guide, GC28-1756
OS/390 Parallel Sysplex Application Migration, GC28-1863.

Java

IBM Developer Kit and Runtime Environment, Java 2 Technology Edition, Version 1.4.2 Diagnostics Guide, SC34-6358

Accessibility

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

You can perform most tasks required to set up, run, and maintain your CICS system in one of these ways:

- using a 3270 emulator logged on to CICS
- using a 3270 emulator logged on to TSO
- using a 3270 emulator as an MVS system console

IBM Personal Communications provides 3270 emulation with accessibility features for people with disabilities. You can use this product to provide the accessibility features you need in your CICS system.

Index

Special characters

, CICS auxiliary trace utility program 80
, dump utility program 84
.STEPLIB, CICS load library 327

A

active delay interval for XRF 211
activity keypoint frequency (AKPFREQ) 151
ADI, system initialization parameter 148
ADYN, dynamic allocation transaction 102
AIBRIDGE, system initialization parameter 148
AICONS, system initialization parameter 148
AIXIT, system initialization parameter 149
AILDELAY, system initialization parameter 149
AIQMAX, system initialization parameter 150
AIRDELAY, system initialization parameter 150
AKPFREQ, system initialization parameter 151
alternate delay interval for XRF 148
AMP parameter for specification of the GCD 329
API improvements 317
APPL statement, VTAM VBUILD application identifier 151
APPLID, system initialization parameter 151
ARMREGISTERED, named counter server 409
AUTCONN, system initialization parameter 152
authorized libraries 327
AUTHTYPE DB2 parameter 265, 293
AUTODST, system initialization parameter 153
automatic dynamic storage tuning
 AUTODST 153
automatic start 234, 309
AUTORESETTIME, system initialization parameter 153
autostart override record 66
auxiliary storage trace 153
auxiliary temporary storage data set 15, 328
 control interval size 16
 job control statement for CICS execution 16
 job control statements to define 15
 number of control intervals 17
 space considerations 16
auxiliary trace data sets 77

auxiliary trace data sets (*continued*)
 job control statements for CICS execution 79
 job control statements to allocate 78
 space calculations 79
auxiliary trace utility program, 80
AUXTR, system initialization parameter 153
AUXTRSW, system initialization parameter 154

B

backout exit programs 245
backout of resources at emergency restart 245
batching requests 204
BDAM data sets
 creating and loading 99
 opening and closing 102
BMS (basic mapping support)
 BMS system initialization parameter 154
 page-chaining command character string 213
 page-copying command character string 213
 page-purging command character string 213
 page-retrieval command character string 213
 PGCHAIN, BMS CHAIN command 213
 PGCOPY, BMS COPY command 213
 PGPURGE, BMS PURGE command 213
 PGRET, BMS RETRIEVAL command 213
 PRGDLAY, BMS PURGE DELAY command 215
 purge delay time interval 215
 selecting versions of BMS 301
 versions of BMS 155
BMS, system initialization parameter 154
BRMAXKEEPTIME, system initialization parameter 155
BSAM devices
 DD statements for 324
BTS data set, DFHLRQ 331
buffers and strings, VSAM 248, 251
BWO (backup while open)
 data facility data set services (DFDSS) 13
 data facility hierarchical storage manager (DFHSM) 13
 disabling activity keypointing 12
 introduction 11
 restrictions 12
 storage management facilities 12

BWO (backup while open) (*continued*)
 storage management subsystem (SMS) 13
 XRF considerations 12

C

cataloged procedures
 starting CICS as a started task 341
catalogs 308
CAVM (CICS availability manager)
 CAVM control data set 9
 CAVM message data set 9
 DD statements in CICS startup job 88
 DFHXRMMSG, XRF data set 88
 I/O error handling 92
 required data sets 87
 security of XRF data sets 88
 surveillance signal in the control data set 89
CCSO transient data destination 329
CDSA (CICS-key DSA) 337
CDSASZE, system initialization parameter 156, 267
CEDA transaction
 recovery and backup 56
 sharing a CSD between more than one CICS region 41
CEDB transaction 42
CEDC transaction 42
CEMT master terminal transaction 84
CESO transient data destination 329
CETR, trace control transaction 78
CHKSTRM, system initialization parameter 156, 267
CHKSTSK, system initialization parameter 156, 268
CICS journal utility program (DFHJUP) 39
CICS server support for system-managed processes 423
CICS Web interface
 WEBDELAY system initialization parameter 256
CICS-key storage 334
CICSplex SM 317
CICSSVC, system initialization parameter 157
CILOCK, system initialization parameter 157
class, monitoring 201
CLINTCP, system initialization parameter 157
close destination request limit 211
CLSDST destination request limit 211
CLSDSTP, system initialization parameter 158
CLT (command list table) 158
CLT, system initialization parameter 158
CMAC support, messages data set 117

- CMDPROT, system initialization parameter 158
 - CMDSEC, system initialization parameter 159, 278
 - COLD option
 - system initialization parameter BMS 154
 - system initialization parameter ICP 190
 - system initialization parameter START 235
 - system initialization parameter TS 252
 - COMAUGHTYPE DB2 parameter 265, 293
 - command list table (CLT) 158
 - command logs, RDO 59
 - commands
 - CEDA command synpoint criteria 59
 - LIST ALL OBJECTS 45
 - RDO CEDA INSTALL 65
 - REPRO, to run IDCAMS 95
 - common work area (CWA) 256
 - common work area storage key
 - system initialization parameter 169
 - CONFDATA, system initialization parameter 160, 279
 - CONFTXT, system initialization parameter 161, 281
 - consoles
 - entering system initialization parameters 307
 - control data set 89
 - coupling facility
 - managing storage 416
 - coupling facility data tables
 - defining resources for 106
 - starting a server 363
 - coupling facility list structures
 - for coupling facility data tables 106
 - CPSMCONN, system initialization parameter 161
 - cross-system coupling facility (XCF)
 - coding the XCFGROUP system initialization parameter 257
 - CSD (CICS system definition file)
 - CSDSTRNO 168
 - DD statements in CICS startup job 61
 - defining 41
 - definitions for the Japanese language feature 62
 - dynamic allocation 62
 - emergency restart and backout 58
 - job control statements for CICS execution 61
 - job to define and initialize 43
 - making the CSD available to CICS 61
 - recovery and backup 56
 - sharing the CSD 47
 - space for data set 42
 - CSDACC, system initialization parameter 163
 - CSDBKUP, system initialization parameter 163
 - CSDBUFND, system initialization parameter 164
 - CSDBUFNI, system initialization parameter 164
 - CSDDISP, system initialization parameter 164
 - CSDDSN, system initialization parameter 165
 - CSDFRLOG, system initialization parameter 165
 - CSDINTEG, system initialization parameter 165
 - CSDJID, system initialization parameter 166
 - CSDSLNRNO, system initialization parameter 167
 - CSDRECOV, system initialization parameter 167
 - CSDSTRNO, system initialization parameter 168
 - CSFU, CICS file utility transaction 103, 105
 - CSSL transient data destination 329
 - cushion storage 338
 - CWA (common work area) 256
 - CWAKEY, storage key for the CWA 335
 - CWAKEY, system initialization parameter 169
 - CXRF queue 24
 - CXRF transient data queue 24
 - DD statements, DISP operand 25
 - DISP operand of DD statements 25
- ## D
- DAE, system initialization parameter 169
 - data facility data set services 13
 - data facility hierarchical storage manager (DFHSM)
 - backup while open 11
 - data interchange program (DIP) 171
 - data sets
 - actively shared 9
 - allocation 10
 - allocation and dispositions (XRF) 8
 - auxiliary temporary storage 15
 - auxiliary trace 77
 - BDAM 99
 - catalog data sets 65, 71
 - CAVM control data set 9
 - CAVM message data set 9
 - CDBM SUPPORT data set 113
 - created by DFHCMACI job 7
 - created by DFHCOMDS job 7
 - created by DFHDEFDS job 7
 - CSD 41
 - debugging profiles 125
 - creating 125
 - defining 125
 - defining user files 100
 - defining, transient data (extrapartition) 24
 - defining, transient data (intrapartition) 21
 - DFHAUXT, auxiliary trace 10
 - DFHBUXT, auxiliary trace 10
 - data sets (*continued*)
 - DFHDMPx, dump 10
 - DFHLCD, CICS local catalog 10
 - DFHXRCTL, XRF control data set 89
 - DISP option 10
 - dump 81, 175, 176
 - dynamic allocation using CEMT 102
 - for EJB 119
 - GTF data sets 8
 - integrity on shared DASD 10
 - messages data set 117
 - MVS system data sets used by CICS 7, 8
 - passively shared 9
 - SDUMP data sets 8
 - SMF data sets 8
 - transient data (extrapartition) 21
 - transient data (intrapartition) 21
 - unique 10
 - user data sets
 - BDAM 99
 - closing 103
 - defining to CICS 100
 - loading VSAM data sets 95
 - opening 102
 - VSAM 94
 - VSAM bases and paths 94
 - XRF considerations 8
 - XRF control data sets 89
 - data tables
 - closing 105
 - loading 105
 - opening 105
 - overview 104
 - types of 105
 - XRF considerations 106
 - database recovery control (DBRC)
 - system initialization parameter, DLDBRC 152
 - use of generic applid 152
 - date format 169
 - DATFORM, system initialization parameter 169
 - DB2 load library
 - requirement for DSNTIAR and DSNTIA1 328
 - DB2 resource security
 - XUSER system initialization parameter
 - AUHTYPE 265, 293
 - COMAUGHTYPE 265, 293
 - DB2, RCT suffix option of CICS startup 320
 - DB2CONN, system initialization parameter 170
 - DBCTLCON, system initialization parameter 170
 - DBRC (database recovery control)
 - system initialization parameter, DLDBRC 152
 - use of generic applid 152
 - DCT (destination control table)
 - specifying security checking of DCT entries 258, 286
 - DDS option of system initialization parameter BMS 154
 - deadlock timeout 339

- debugging
 - preparing for 343
- debugging profiles data sets
 - creating 125
 - defining 125
 - as remote files 129
 - as VSAM non-RLS files 128
 - as VSAM RLS files 127
- DEBUGTOOL, system initialization parameter 170
- defining debugging profiles data sets
 - as remote files 129
 - as VSAM non-RLS files 128
 - as VSAM RLS files 127
- delay interval, primary, for XRF 211
- delay intervals
 - active delay for XRF 211
 - alternate delay for XRF 148
 - JES for XRF 194
 - reconnection for XRF 152
- delay, persistent verification 219
- destination control table (DCT)
 - specifying security checking of DCT entries 258, 286
- destination request limit, open and close 211
- DEVTYPE macro (MVS) 83
- DFH\$TDWT (transient data write-to-terminal sample program) 21
- DFH0STAT, statistics sample program 340
- DFH99, sample DYNALLOC utility program 102
- DFHAUXT auxiliary trace data set 77
- DFHBUXT auxiliary trace data set 77
- DFHCCUTL, local catalog initialization utility program 73
- DFHCOMP1, CSD resource definition group 52
- DFHCOMP2, CSD resource definition group 52
- DFHCOMP3, CSD resource definition group 52
- DFHCSDUP
 - definitions for the Japanese language feature 62
- DFHCSVC, the CICS type 3 SVC 157
- DFHCXRF data set, transient data extrapartition 24
 - DD statements for 25
 - in active CICS regions 24
 - in alternate CICS regions 25
- DFHDBFK data set 115
 - job control statements for CICS execution 115
 - job control statements to define and load 114
- DFHDCTG, group of transient data definitions 60
- DFHDPFMB
 - debugging profiles data set creating 125
 - defining
 - as remote files 129
 - as VSAM non-RLS files 128
 - as VSAM RLS files 127
- DFHDPFMP
 - debugging profiles data set creating 125
 - defining
 - as remote files 129
 - as VSAM non-RLS files 128
 - as VSAM RLS files 127
- DFHDPFMX
 - debugging profiles data set creating 125
- DFHDYP, dynamic transaction routing program
 - coding the DTRPGM system initialization parameter 174, 175
- DFHGCD, global catalog data set 66
- DFHISTAR job 7
- DFHJUP, CICS journal utility program 39
- DFHLCD, local catalog data set 72
- DFHLRQ, BTS data set 331
- DFHMQCOD 203
- DFHNCMN, named counter server region program 403
- DFHNCO macro 399
- DFHNCOPT, named counter options table 399
- DFHRPL, module load library 328
- DFHRSTAT 386
- DFHSIT keywords and operands 137
 - undefined keywords error message 300
- DFHSM (data facility hierarchical storage manager)
 - backup while open 11
- DFHSTART, sample startup procedure 321
- DFHTCTDY, dummy TCT 302
- DFHXQMN system initialization parameter
 - for pool list structure creation 350
- DFHXQMN, TS server program 350
- DFHXRCTL, XRF control data set 89
- DFLTUSER, system initialization parameter 171, 281
- DIP (data interchange program) 171
- DIP, system initialization parameter 171
 - discovering 317
- DISMACP, system initialization parameter 171
- DL/I
 - PDIR, system initialization parameter 212
 - specifying security checking of PSB entries 262, 290
- DOCCODEPAGE, system initialization parameter 172
- domains
 - kernel 72
 - parameters 72
- DSA (dynamic storage areas)
 - CDSA 337
 - CICS-key storage 334
 - cushions 338
 - CWAKEY, storage key for the CWA 335
 - ECDSA 250, 337
 - ERDSA 334, 337
- DSA (dynamic storage areas) *(continued)*
 - ESDSA 337
 - EUDSA 250, 338
 - key-0 storage 334
 - RDSA 334, 337
 - RENTPGM, storage for read-only DSAs 325
 - RENTPGM, system initialization parameter 221
 - SDSA 337
 - SOS (short-on-storage) 338
 - STGPROT, system initialization parameter 238
 - storage protection facilities 334
 - TCTUAKKEY, storage key for terminal control user areas 335
 - UDSA 337
- DSALIM, system initialization parameter 172
- DSHIPIDL, system initialization parameter 173
- DSHIPINT, system initialization parameter 173
- DSNTIA1 328
- DSNTIAC 328
- DSNTIAR 328
- DSRTPGM, system initialization parameter 174
- DTIMOUT (deadlock timeout interval) 339
- DTRPGM, system initialization parameter 174
- DTRTRAN, system initialization parameter 175
- dump analysis and elimination
 - system initialization parameter 169
- dump data sets 81, 175, 176
 - dump table facility 81
 - job control statements to allocate 84
 - space calculations 81
- dump utility program, 84
- DUMP, system initialization parameter 175
- DUMPDS, system initialization parameter 175
- dumps
 - controlling with dump table 81
 - effect of START= parameter 314
- DUMPSW, system initialization parameter 176
- DURETRY, system initialization parameter 176
- dynamic allocation
 - ADYN, dynamic allocation transaction 102
 - DFH99, sample DYNALLOC utility program 102
- dynamic allocation of the CSD 62
- dynamic transaction routing program, DFHDYP
 - coding the DTRPGM system initialization parameter 174, 175

E

ECDSA (extended CICS-key DSA) 337
ECDSASZE, system initialization parameter 177, 268
EDSALIM, system initialization parameter 177
EJBROLEPRFX, system initialization parameter 178
emergency restart
 resource backout 245
 START system initialization parameter 234
ENCRYPTION, system initialization parameter 179
EODI, system initialization parameter 180
ERDSA (extended read-only DSA) 334, 337
ERDSASZE, system initialization parameter 180, 268
ESDSA (shared DSA) 337
ESDSASZE, system initialization parameter 181, 269
ESMEXITS, system initialization parameter 181, 266
EUDSA (extended user DSA) 338
EUDSASZE, system initialization parameter 181, 269
exception class monitoring 202
exit interval, region 190
exits
 FE, global trap exit 250
 IEFUSI, exit routine 333
 XDUCLSE, dump global user exit 83, 85
 XDUOUT, dump global user exit 83, 86
 XDUREQ, dump global user exit 83, 86
 XDUREQC, dump global user exit 83, 86
external security interface 227, 283
extrapartition transient data 21
 CSSL, and other destinations used by CICS 329
extrapartition transient data queues 329

F

facilities
 auxiliary trace autoswitch facility 154
 generalized trace facility (GTF) 8
 storage protection facilities 334
 temporary storage 15
FCQRONLY, system initialization parameter 182
FCT (file control table)
 specifying the FCT suffix 182
FCT, system initialization parameter 182
FE global trap exit 250
FEPI (front end programming interface)
 FEPI, system initialization parameter 182
field name start character 183
field separator characters 183

file control table (FCT)
 specifying the FCT suffix 182
FILSTAT operand 105
FLDSEP, system initialization parameter 183
FLDSTRT, system initialization parameter 183
FORCEQR, system initialization parameter 184
frequency, activity keypoint 151
front end programming interface (FEPI)
 FEPI, system initialization parameter 182
FSSTAFF, system initialization parameter 184
FULL option of system initialization parameter BMS 154

G

GCD (global catalog data set)
 buffer space 70
 description 65
 job control statement for CICS execution 66
 job control statements to define and initialize 65
 space calculations 67
generalized trace facility (GTF) 8
generic applid for CICS XRF regions 152
global catalog
 AMP parameter for specification 329
 for resource definitions 308
 use in restart 309
global catalog data set (GCD)
 buffer space 70
 description 65
 job control statement for CICS execution 66
 job control statements to define and initialize 65
 space calculations 67
global trap exit, FE 250
GMTEXT, system initialization parameter 185
GMTRAN, system initialization parameter 186
GNTRAN, system initialization parameter 187
good morning message 185
good morning transaction 186
GOOD MORNING transaction 223
group list, RDO 188
GRPLIST, system initialization parameter 188
GTF (generalized trace facility) 8
GTFTR, system initialization parameter 190

H

high-performance option (HPO) 190, 266
HPO (high-performance option) 190, 266

HPO, system initialization parameter 190, 266

I

I/O error handling 92
ICP (interval control program) 190
ICP, system initialization parameter 190
ICV, system initialization parameter 190
ICVR, system initialization parameter 191
ICVTSD, system initialization parameter 192
IDCAMS, AMS utility program 95
identity class monitoring 202
IEFUSI, exit routine 333
IEV017 error message 300
IIOPLISTENER, system initialization parameter 192
INFOCENTER, system initialization parameter 192
INITIAL
 system initialization parameter START 235
initial start
 START system initialization parameter 234
INITPARM, system initialization parameter 193
internal trace, main storage 193
interregion communication (IRC) 193
intersystem communication (ISC) 193
interval control program (ICP) 190
intervals, activity keypoint 151
intrapartition transient data 21, 329
intrapartition transient data queues
 defining the intrapartition data set 21
INTTR, system initialization parameter 193
IRC (interregion communication) 193
IRCSTRT, system initialization parameter 193
ISC (intersystem communication) 193
ISC, system initialization parameter 193

J

Japanese language feature
 installing definitions in the CSD 62
JCL (job control language)
 CICS startup 321
 as a batch job 321
 as a started task 341
JES delay interval for XRF 194
JESDI, system initialization parameter 194
job control language (JCL)
 for CICS as a batch job 321
 for CICS as a started task 341
job stream
 CICS startup 320
jobs
 DFHISTAR 7
journaling
 BWO 11

journaling (*continued*)
 specifying security checking for
 journal entries 260, 288
 XJCT, system initialization
 parameter 260, 288
 JOURNALMODEL definitions 33
 JVMCCSIZE, system initialization
 parameter 194
 JVMCCSTART, system initialization
 parameter 194
 JVMLEVEL0TRACE 143
 JVMLEVEL0TRACE, system initialization
 parameter 195, 269
 JVMLEVEL1TRACE 143
 JVMLEVEL1TRACE, system initialization
 parameter 196, 270
 JVMLEVEL2TRACE 143
 JVMLEVEL2TRACE, system initialization
 parameter 196, 270
 JVMPROFILEDIR, system initialization
 parameter 196
 JVMUSERTRACE 143
 JVMUSERTRACE, system initialization
 parameter 196, 270

K

key-0 storage 334
 keypoint frequency 151
 KEYRING, system initialization
 parameter 197
 keys for page-retrieval 229

L

Language Environment runtime library,
 SCEERUN 322
 Language Environment runtime library,
 SCEERUN2 322
 LCD (local catalog data set)
 description 71
 job control statement for CICS
 execution 72
 job control statements to define and
 initialize 72
 use in restart 309
 LGDFINT system initialization
 parameter 197
 LGNMSG, system initialization
 parameter 197
 libraries
 SCEERUN, Language Environment
 runtime library 322
 SCEERUN2, Language Environment
 runtime library 322
 list structure
 storage considerations 403
 list structure, defining 386, 402
 LLACOPY macro 198
 LLACOPY, system initialization
 parameter 198
 local catalog data set (LCD)
 description 71
 job control statement for CICS
 execution 72

local catalog data set (LCD) (*continued*)
 job control statements to define and
 initialize 72
 use in restart 309
 LOCALCCSID, system initialization
 parameter 198
 locating modules in the relocatable
 program library 198
 log defer time interval 197
 log manager 27
 log streams
 mapping system log and journal
 names to 34
 LOGA transient data destination 329
 logging
 defining CICS journals for general
 logs
 forward recovery logs 31
 user journals 32
 defining CICS logs 27
 defining CICS system logs 27
 defining dummy logs 28
 JOURNALMODEL definitions 33
 log autoinstall 33
 logon data, VTAM 197
 LPA (link pack area)
 LPA system initialization
 parameter 199
 PRVMOD system initialization
 parameter 217, 271
 LPA, system initialization parameter 199

M

macros
 DEVTYPE (MVS macro) 83
 MVS SDUMP 8
 MAXJVMTCBS, system initialization
 parameter 134, 199
 MAXOPENTCBS, system initialization
 parameter 134, 199
 MAXSOCKETS, system initialization
 parameter 200
 MAXSSLTCBS, system initialization
 parameter 200
 MAXXPTCBS, system initialization
 parameter 134, 200
 MCT (monitoring control table) 200
 MCT, system initialization
 parameter 200
 message case 205
 message level 206
 message, good morning 185
 messages
 crucial and non-crucial messages 91
 messages data set
 job control statements to define and
 load 117
 messages data set for CMAC facility 117
 MINIMUM option of system initialization
 parameter BMS 154
 MN, system initialization parameter 201
 MNCONV, system initialization
 parameter 202
 MNEXC, system initialization
 parameter 202

MNFREQ, system initialization
 parameter 202
 MNIDN, system initialization
 parameter 202
 MNPER, system initialization
 parameter 203
 MNRES, system initialization
 parameter 203
 MNSYNC, system initialization
 parameter 203
 MNTIME, system initialization
 parameter 203
 module load library concatenation,
 DFHRPL 328
 monitoring 201, 314
 exception class 202
 identity class monitoring 202
 performance class 203
 transaction resource monitoring 203
 monitoring control table (MCT) 200
 MQCONN, system initialization
 parameter 203
 MRO (multiregion operation)
 batching 204
 batching requests 204
 extend lifetime of long-running
 mirror 205
 long-running mirror 205
 MROBTCH, system initialization
 parameter 204
 MROFSE, system initialization
 parameter 205
 MROLRM, system initialization
 parameter 205
 MSGCASE, system initialization
 parameter 205
 MSGLVL, system initialization
 parameter 206
 multiregion operation (MRO)
 batching 204
 batching requests 204
 extend lifetime of long-running
 mirror 205
 long-running mirror 205
 MVS SDUMP macro 82
 MVS START command, to start
 CICS 341
 MXT, system initialization
 parameter 206

N

named counters
 automatic restart manager
 parameters 406
 controlling server regions 408
 debug trace parameters 407
 defining an options table 399
 defining and starting server 403
 deleting or emptying pools 411
 dumping pool list structures 414
 list structure parameters 407
 list structure, defining 402
 options table
 defining 399
 making available to CICS 402
 parameters 400

- named counters (*continued*)
 - parameters, server 404
 - server overview 397
 - server parameters 404
 - SET command 409
 - starting a server 397
 - unloading and reloading pools 412
 - warning parameters 408
 - XES 409
- NATLANG, system initialization
 - parameter 207
- NCPLDFT, system initialization
 - parameter 208
- NEWSIT, system initialization
 - parameter 209, 270
 - effect on warm start 310
- NODDS option of system initialization
 - parameter BMS 155
- NONRLSRECOV, system initialization
 - parameter 210

O

- open destination request limit 211
- open transaction environment 134
- open transaction environment (OTE)
 - TCBs 134
- operator communication for initialization
 - parameters 308
- OPERTIM, system initialization
 - parameter 210
- OPNDLIM, system initialization
 - parameter 211
- OPNDST write-to-operator timeout
 - limit 211
- OTE TCBs 134
- overriding system initialization
 - parameters
 - from the console 307
 - from the SYSIN data set 306
- overview of coupling facility data table
 - server 363
- overview of temporary storage data
 - sharing server 349

P

- PA keys for page-retrieval 229
- PA keys for screen copying 215
- page-chaining command character
 - string 213
- page-copying command character
 - string 213
- page-purging command character
 - string 213
- page-retrieval command character
 - string 213
- page-retrieval keys 229
- PARM startup parameter
 - system initialization parameters 324
- PARMERR, system initialization
 - parameter 211
- PARMSRCE parameter 317
- PARMTYPE parameter 317
- PDI, system initialization parameter 211

- PDIR, system initialization
 - parameter 212
- performance class monitoring 203
- persistent sessions 217, 218
- persistent verification delay 219
- PF keys for page-retrieval 229
- PGCHAIN, system initialization
 - parameter 213
- PGCOPY, system initialization
 - parameter 213
- PGPURGE, system initialization
 - parameter 213
- PGRET, system initialization
 - parameter 213
- PLT (program list table)
 - system initialization programs 214
 - system termination programs 215
- PLTIPI, system initialization
 - parameter 214
- PLTIPIUSR, system initialization
 - parameter 214, 282
- PLTSD, system initialization
 - parameter 215
- pools
 - TCBs 136
- PRGDLAY, system initialization
 - parameter 215
- primary delay interval for XRF 211
- PRINT, system initialization
 - parameter 215
- program list table (PLT) 320
 - system initialization programs 214
 - system termination programs 215
- program specification block (PSB)
 - PDIR, system initialization
 - parameter 212
 - specifying security checking of PSB
 - entries 262, 290
 - PRTYAGE, system initialization
 - parameter 217
 - PRVMOD, system initialization
 - parameter 217, 271
- PSB (program specification block)
 - PDIR, system initialization
 - parameter 212
 - specifying security checking of PSB
 - entries 262, 290
 - PSBCHK, system initialization
 - parameter 217, 282
 - PSDINT, system initialization
 - parameter 217
 - PSTYPE, system initialization
 - parameter 218
- purge delay time interval, BMS 215
- PVDELAY, system initialization
 - parameter 219

R

- RACF (resource access control facility)
 - checking program entries with
 - RACF 262, 290
 - DFTUSER, system initialization
 - parameter 171, 281
 - establishing APPC sessions 227, 284
 - MRO bind-time security 227, 283
 - resource level checking 227, 284

- RACF (resource access control facility)
 - (*continued*)
 - SEC, system initialization
 - parameter 227, 283
 - SECPREFX, system initialization
 - parameter 228, 284
 - specifying a prefix to resource
 - name 228, 284
 - XAPPC, system initialization
 - parameter 256, 286
- RAMAX, system initialization
 - parameter 219
- RAPOOL, system initialization
 - parameter 220
- RBA (relative byte address) 23
- RDO (resource definition online)
 - CICS system definition data set
 - (CSD) 41
 - group list (GRPLIST) 188
- RDSA (read-only DSA) 334, 337
- RDSASZE, system initialization
 - parameter 221, 271
- read-only storage
 - system initialization parameter 221
- RECEIVE ANY (RA) maximum 219
- RECEIVE ANY (RA) pool size 220
- reconnection delay interval (XRF) 152
- reconnection transaction for XRF 223
- record-level sharing (RLS)
 - VSAM data sharing 96
- recovery for files 210
- region exit interval (ICV) 190
- REGION parameter for CICS
 - startup 333
- region status
 - starting a server 385
- region status server
 - list structure, defining 386
- relative byte address (RBA) 23
- RELOAD, named counter pool 412
- RENTPGM, storage for read-only
 - DSAs 325
- RENTPGM, system initialization
 - parameter 221
- request parameter list (RPL) 220
- resource access control facility (RACF)
 - checking program entries with
 - RACF 262, 290
 - DFTUSER, system initialization
 - parameter 171, 281
 - establishing APPC sessions 227, 284
 - MRO bind-time security 227, 283
 - resource level checking 227, 284
 - SEC, system initialization
 - parameter 227, 283
 - SECPREFX, system initialization
 - parameter 228, 284
 - specifying a prefix to resource
 - name 228, 284
 - XAPPC, system initialization
 - parameter 256, 286
- resource backout at emergency
 - restart 245
- resource definition online (RDO)
 - CICS system definition data set
 - (CSD) 41
 - command logs 59

resource definition online (RDO)
(continued)
 group list (GRPLIST) 188
 RESP, system initialization
 parameter 221
 RESSEC, system initialization
 parameter 221, 283
 RLS, record-level sharing
 VSAM data sharing 96
 RMTRAN, system initialization
 parameter 223
 RPL (request parameter list) 220
 RRMS, system initialization
 parameter 224
 RST, system initialization parameter 224
 RSTSIGNOFF, system initialization
 parameter 224
 RSTSIGNTIME, system initialization
 parameter 225
 RUWAPool, system initialization
 parameter 226

S

sample job streams
 CICS startup 320
 samples
 data for loading a BDAM data
 set 100
 DFH\$TDWT (transient data
 write-to-terminal program) 21
 DFHSTART, sample startup
 procedure 321
 JCL to create and load a BDAM data
 set 99
 sample job to define auxiliary data
 sets on disk 78
 screen copying 215
 SDSA (shared DSA) 337
 SDSASZE, system initialization
 parameter 226, 272
 SDTRAN, system initialization
 parameter 226
 SDUMP data sets 8
 SDUMP macro 82
 CICS retry interval 176
 DURETRY option 176
 SEC, system initialization
 parameter 227, 283
 SECPRFX, system initialization
 parameter 228, 284
 security
 for transactions 264, 292
 MRO bind-time security 227, 283
 of attached entries 264, 292
 of XRF data sets 88
 resource class names 257, 286
 SEC, system initialization
 parameter 227, 283
 SECPRFX, system initialization
 parameter 228, 284
 security checking
 for EXEC CICS system
 commands 257, 286
 for program entries 262, 290
 for temporary storage entries 264,
 292

security *(continued)*
 security checking *(continued)*
 of DB2 resources 258
 of destination control entries 258,
 286
 of enterprise bean method
 calls 259, 287
 of EXEC-started transaction
 entries 261, 289
 of file control entries 259, 288
 of HFS files 260, 288
 of journal entries 260, 288
 of PSB entries 262, 290
 of resources 263, 291
 specifying a prefix to resource
 name 228, 284
 using RACF to establish APPC
 sessions 227, 284
 XAPPC, system initialization
 parameter 256, 286
 XCMD, system initialization
 parameter 257, 286
 XDB2, system initialization
 parameter 258
 XDCT, system initialization
 parameter 258, 286
 XEJB, system initialization
 parameter 259, 287
 XFCT, system initialization
 parameter 259, 288
 XHFS, system initialization
 parameter 260, 288
 XJCT, system initialization
 parameter 260, 288
 XPCT, system initialization
 parameter 261, 289
 XPPT, system initialization
 parameter 262, 290
 XPSB, system initialization
 parameter 262, 290
 XRES, system initialization
 parameter 263, 291
 XTRAN, system initialization
 parameter 264, 292
 XTST, system initialization
 parameter 264, 292
 sequence numbers
 See named counters 397
 sequential terminal devices
 logical close to quiesce 332
 SET command, named counters 409
 SETXCF, delete named counter pool 411
 sharing the CSD 47
 freeing internal locks 53
 protection by internal locks 49
 sharing between CICS regions 49
 single keystroke retrieval (SKR) 229
 SIT (system initialization table)
 DFHSIT keywords and operands 137
 DFHSIT TYPE=CSECT 137
 DFHSIT TYPE=DSECT 137
 supplying system initialization
 parameters to CICS 303
 SIT parameters
 retrieving information 317
 SIT, system initialization parameter 228,
 272

SKR (single keystroke retrieval) 229
 SKRxxxx, system initialization
 parameter 229
 SMS (storage management
 subsystem) 11, 12, 13
 SNSCOPE, system initialization
 parameter 229, 285
 SOS (short-on-storage) 338
 space calculations
 auxiliary trace data set 79
 CSD 42
 defining data sets 3
 disk space 42
 dump data sets 81
 global catalog 67
 XRF message data set 90
 SPCTR, system initialization
 parameter 230
 SPCTRxx, system initialization
 parameter 230, 272
 SPOOL, system initialization
 parameter 233
 SRBSVC, system initialization
 parameter 233
 SRT (system recovery table) 233, 234
 SRT, system initialization parameter 233
 SRVERCPsystem initialization
 parameter 234
 SSLDELAY, system initialization
 parameter 234
 SSLTCB, system initialization
 parameter 234
 STANDARD option of system
 initialization parameter BMS 154
 STANDBY start option 236
 standby start-up for XRF 236
 START command, MVS 341
 START, system initialization
 parameter 234
 (option,ALL) 236
 START=AUTO 309
 START=COLD 311
 START=INITIAL 311
 START=STANDBY 312
 started task, CICS as a 341
 STARTER, system initialization
 parameter 236, 266
 starting CICS regions 319
 as a started task 341
 MVS START command 341
 sample job stream 320
 specifying the type of startup 308
 START=AUTO 309
 START=COLD 311
 START=INITIAL 311
 START=STANDBY, for an XRF
 alternate CICS 312
 startup procedure, DFHSTART 321
 STATEOD, system initialization
 parameter 236
 STATINT, system initialization
 parameter 237
 statistics 314
 statistics sample program,
 DFH0STAT 340
 STATRCDD, system initialization
 parameter 237

STGPROT, system initialization
parameter 238

STGRVCVY, system initialization
parameter 238

STNTR, system initialization
parameter 239

STNTRxx, system initialization
parameter 239, 275

storage calculations 18

storage cushion size
short-on-storage warnings 340

storage management subsystem
(SMS) 11, 12, 13
backup while open facility 11
introduction 13
release information 12

storage protection for CICS regions 334

storage protection system initialization
parameter, STGPROT 238

storage requirements for CICS
regions 333

storage trace
auxiliary 153
main 193
trace option in transaction dump 251
trace table size in main storage 250
trace table size in transaction
dump 251

strings and buffers, VSAM 248, 251

SUBTSKS, system initialization
parameter 242

SUFFIX, system initialization
parameter 243, 266

supervisor call (SVC)
type 3, DFHCSVC 157
type 6, DFHHPSVC 190, 233, 266

surveillance signal for XRF 89, 148

SVC (supervisor call)
type 3, DFHCSVC 157
type 6, DFHHPSVC 190, 233, 266

SYDUMAX, system initialization
parameter 243

SYSIDNT, system initialization
parameter 243

SYSIN data stream 325

SYSARM resource table 317

system data sets 3

system dumps 82

system identifier, system initialization
parameter SYDUMAX 243

system identifier, system initialization
parameter SYSIDNT 243

system initialization
for an alternate CICS (XRF=YES)
START=STANDBY 312
how CICS determines the type of
startup 308
START=AUTO 309
START=COLD 311
START=INITIAL 311

system initialization parameter
EJBROLEPRFX 178
ENCRYPTION 179

system initialization parameters 317
ADI 148
AIBRIDGE 148
AICONS 148

system initialization parameters
(continued)
AIEXIT 149
AILDELAY 149
AIQMAX 150
AIRDELAY 150
AKPFREQ 151
APPLID 151
AUTCONN 152
AUTODST 153
AUTORESETTIME 153
AUXTR 153
AUXTRSW 154
BMS 154
BRMAXKEEPTIME 155
CDSASZE 156, 267
CHKSTRM 156, 267
CHKSTSK 156, 268
CICSSVC 157
CILOCK 157
CLINTCP 157
CLSDSTP 158
CLT 158
CMDPROT 158
CMDSEC 159, 278
CONFDATA 160, 279
CONFTXT 161, 281
CPSMCONN 161
CSDACC 163
CSDBKUP 163
CSDBUFND 164
CSDBUFNI 164
CSDDISP 164
CSDDSN 165
CSDFRLOG 165
CSDINTEG 165
CSDJID 166
CSDLSRNO 167
CSDRECOV 167
CSDSTRNO 168
CWAKEY 169
DAE 169
DATFORM 169
DB2CONN 170
DBCTLCON 170
DEBUGTOOL 170
DFLTUSER 171, 281
DIP 171
DISMACP 171
DOCCODEPAGE 172
DSALIM (DSA storage limit) 172
DSHIPIDL 173
DSHIPINT 173
DSRTPGM 174
DTRPGM 174
DTRTRAN 175
DUMP 175
DUMPPDS 175
DUMPSW 176
DURETRY 176
ECDSASZE 177, 268
EDSALIM (EDSA storage limit) 177
entering at the console 307
EODI 180
ERDSASZE 180, 268
ESDSASZE 181, 269
ESMEXITS 181, 266

system initialization parameters
(continued)
EUDSASZE 181, 269
FCQRONLY 182
FCT 182
FEPI 182
FLDSEP 183
FLDSTRT 183
FORCEQR 184
from operator's console 304, 308
from the SYSIN data set 325
FSSTAFF 184
GMTEXT 185
GMTRAN 186
GNTRAN 187
GRPLIST 188
GTFTR 190
how to specify 133
HPO 190, 266
ICP 190
ICV 190
ICVTSD 192
IIOPLISTENER 192
in the PARM parameter 304, 306
in the SYSIN data set 304, 306
INFOCENTER 192
INITPARM 193
INTTR 193
IRCSTRT 193
ISC 193
JESDI 194
JVMCCSIZE 194
JVMCCSTART 194
JVMLEVEL0TRACE 195, 269
JVMLEVEL1TRACE 196, 270
JVMLEVEL2TRACE 196, 270
JVMPROFILEDIR 196
JVMUSERTRACE 196, 270
KEYRING 197
LGDFINT 197
LGNMSG 197
LLACOPY 198
LOCALCCSID 198
LPA 199
MAXJVMTCBS 134, 199
MAXOPENTCBS 134, 199
MAXSOCKETS 200
MAXSSLTCBS 200
MAXXPTCBS 134, 200
MCT 200
MN 201
MNCONV 202
MNEXC 202
MNFREQ 202
MNIDN 202
MNPER 203
MNRES 203
MNSYNC 203
MNTIME 203
MQCONN 203
MROBTCH 204
MROFSE 205
MROLRM 205
MSGCASE 205
MSGVLV 206
NATLANG 207
NCPLDFT 208

system initialization parameters

(continued)

NEWSIT 209, 270
 NONRLSRECOV 210
 OPERTIM 210
 OPNDLIM 211
 PARMERR 211
 PDI 211
 PDIR 212
 PGAICTLG 212
 PGAEXIT 212
 PGAIPGM 212
 PGCHAIN 213
 PGCOPY 213
 PGPURGE 213
 PGRET 213
 PLTPI 214
 PLTPIUSR 214, 282
 PLTSD 215
 PRGDLAY 215
 PRINT 215
 PRTYAGE 217
 PRVMOD 217, 271
 PSBCHK 217, 282
 PSDINT 217
 PSTYPE 218
 PVDELAY 219
 RAMAX 219
 RAPOOL 220
 RDSASZE 221, 271
 RENTPGM 221
 RESP 221
 RESSEC 221, 283
 RMTRAN 223
 RRMS 224
 RST 224
 RSTSIGNOFF 224
 RSTSIGNTIME 225
 RUWAPOL 226
 SDSASZE 226, 272
 SDTRAN 226
 SEC 227, 283
 SECPRFX 228, 284
 SIT 228, 272
 SKRxxxx 229
 SNSCOPE 229, 285
 SPCTR 230
 SPCTRxx 230, 272
 specifying on the PARM
 statement 324
 SPOOL 233
 SRBSVC 233
 SRT 233
 SRVERCP 234
 SSLDELAY 234
 SSLTCB 234
 START 234
 STARTER 236, 266
 STATEOD 236
 STATINT 237
 STATRCD 237
 STGPROT 238
 STGRVCY 238
 STNTR 239
 STNTRxx 239, 275
 SUBTSKS 242
 SUFFIX 243, 266

system initialization parameters

(continued)

SYDUMAX 243
 SYSIDNT 243
 SYSTR 244
 TAKEOVR 244
 TBEXITS 245
 TCAM 245
 TCP 246
 TCPIP 246
 TCSACTN 246
 TCSWAIT 247
 TCT 247
 TCTUAKY 247
 TCTUALOC 248
 TD 248
 TDINTRA 249
 TDSUBTASK 249
 TRANISO (transaction isolation) 249
 TRAP 250
 TRTABSZ 250
 TRTRANSZ 251
 TRTRANTY 251
 TS 251
 TST 252
 UDSASZE 252, 278
 UOWNETQL 253
 USERTR 253
 USRDELAY 253
 USSHOME 254
 VTAM 255
 VTPREFIX 255
 WEBDELAY 256
 WRKAREA 256
 XAPPK 256, 286
 XCFGROUP 257
 XCMD 257, 286
 XDB2 258
 XDCT 258, 286
 XEJB 259, 287
 XFCT 259, 288
 XHFS 260, 288
 XJCT 260, 288
 XLT 261
 XPCT 261, 289
 XPPT 262, 290
 XPSB 262, 290
 XRES 263, 291
 XRF 264
 XRFSOFF 264
 XRFSTME 264
 XTRAN 264, 292
 XTST 264, 292
 XUSER 265, 293
 System initialization parameters
 ICVR 191
 MXT 206
 system initialization table (SIT)
 DFHSIT keywords and operands 137
 supplying system initialization
 parameters to CICS 303
 system management facilities
 for CICS statistics 8
 system recovery table (SRT) 233, 234
 system spooling interface 233
 system startup 319
 startup job stream 320

system task 341

System-managed duplexing 425
 System-managed rebuild 423
 SYSTR, system initialization
 parameter 244

T

takeover action for XRF 244
 TAKEOVR, system initialization
 parameter 244
 TBEXITS, system initialization
 parameter 245
 TCAM, system initialization
 parameter 245
 TCB pools 136
 TCP, system initialization parameter 246
 TCPIP, system initialization
 parameter 246
 TCSACTN, system initialization
 parameter 246
 TCSWAIT, system initialization
 parameter 247
 TCT (terminal control table) 247
 TCT, system initialization parameter 247
 TCTUAKY, storage key for terminal
 control user areas 335
 TCTUAKY, system initialization
 parameter 247
 TCTUALOC, system initialization
 parameter 248
 TD, system initialization parameter 248
 TDINTRA, system initialization
 parameter 249
 TDSUBTASK, system initialization
 parameter 249
 temporary storage
 starting up temporary storage
 servers 349
 TS data sharing server 349
 VSAM buffers and strings 251
 temporary storage data sharing
 defining TS pools for TS data
 sharing 17
 temporary storage server
 sample startup job 351
 temporary storage table (TST) 252
 specifying security checking of
 temporary storage entries 264, 292
 terminal control table (TCT) 247
 dummy control table,
 DFHTCTDY 302
 terminal control table user area storage
 key
 system initialization parameter 247
 terminal scan delay, ICVTSD 192
 time interval, region exit 190
 timeout limit, user ID 253
 trace
 , trace utility program 80
 auxiliary storage trace 153
 auxiliary trace autoswitch facility 154
 auxiliary trace data sets for XRF 80
 AUXTR, system initialization
 parameter 153
 AUXTRSW, system initialization
 parameter 154

- trace (*continued*)
 - CETR, trace control transaction 78
 - CICS standard tracing, setting levels of 239, 275
 - controlling trace with CEMT or CETR 78
 - defining auxiliary trace data sets 77
 - DFHAUXT auxiliary trace data set 77
 - DFHBUXT auxiliary trace data set 77
 - GTFTR, system initialization parameter 78, 190
 - INTTR, system initialization parameter 78, 193
 - job control statements to allocate auxiliary trace data sets 78
 - option in transaction dump 251
 - sample job to define auxiliary data sets on disk 78
 - SM component, warning when setting trace level 239, 275
 - space calculations for auxiliary trace data sets 79
 - SPCTR, system initialization parameter 230
 - SPCTRxx, system initialization parameter 230, 272
 - special tracing, setting levels of 230, 272
 - starting auxiliary trace 77
 - STNTR, system initialization parameter 239
 - STNTRxx, system initialization parameter 239, 275
 - SYSTR, system initialization parameter 244
 - table size in main storage 250
 - table size in transaction dump 251
 - TRTABSZ, system initialization parameter 250
 - TRTRANSZ, system initialization parameter 251
 - TRTRANTY, system initialization parameter 251
 - USERTR, system initialization parameter 78, 253
 - using auxiliary trace data sets 77
 - using DFHDEFDS to allocate auxiliary trace data sets 78
 - trace utility program, 80
 - trademarks 430
 - TRANISO, system initialization parameter 249
 - transaction isolation 336
 - transaction list table, XLT 261
 - transaction resource monitoring 203
 - transactions
 - ADYN, dynamic allocation transaction 102
 - CSFU, CICS file utility transaction 103
 - transient data (extrapartition) data sets 21
 - defining 24
 - transient data (intrapartition) data set
 - defining 21
 - failure to open 23
 - transient data (intrapartition) data set (*continued*)
 - multiple extents and volumes 23
 - other considerations 23
 - transient data queues 21
 - transient data write-to-terminal sample program (DFH\$TDWT) 21
 - TRAP, system initialization parameter 250
 - TRTABSZ, system initialization parameter 250
 - TRTRANSZ, system initialization parameter 251
 - TRTRANTY, system initialization parameter 251
 - TS, system initialization parameter 251
 - TST (temporary storage table) 252
 - specifying security checking of temporary storage entries 264, 292
 - TST, system initialization parameter 252
 - TYPE=CSECT, DFHSIT 137
 - TYPE=DSECT, DFHSIT 137
 - types of data tables 105
- U**
- UDSA (user DSA) 337
 - UDSASZE, system initialization parameter 252, 278
 - UNLOAD, named counter pool 412
 - UOWNETQL, system initialization parameter 253
 - user file definitions 93
 - user files
 - coupling facility data table server 363
 - user ID timeout limit 253
 - USERTR, system initialization parameter 253
 - USRDELAY, system initialization parameter 253
 - USSHOME, system initialization parameter 254
 - utility programs
 - , dump utility program 84
 - DFHCCUTL, local catalog initialization utility program 73
 - DFHJUP, CICS journal utility program 39
 - IDCAMS, AMS utility program 95
- V**
- VERBEXIT, IPCS parameter 82
 - virtual telecommunications access method (VTAM)
 - ACB at CICS startup 314
 - high performance option (HPO) 190, 266
 - logon data 197
 - system initialization parameter, VTAM 255
 - VBUILD TYPE=APPL statement 151
 - virtual terminals
 - VTPREFIX 255
 - VSAM buffers and strings 248, 251
 - VSAM data sets 94
 - bases and paths 94
 - loading empty VSAM data sets 95
 - opening and closing 102
 - VSAM intrapartition data set 21
 - VTAM, system initialization parameter 255
 - VTPREFIX, system initialization parameter 255
- W**
- warm start 234
 - WEBDELAY, system initialization parameter 256
 - welcome (good morning) message 185
 - write-to-operator timeout limit 210
 - WRKAREA, system initialization parameter 256
- X**
- XAPPC, system initialization parameter 256, 286
 - XCFGROUP, system initialization parameter 257
 - XCMD, system initialization parameter 257, 286
 - XDB2, system initialization parameter 258
 - XDCT, system initialization parameter 258, 286
 - XDUCLSE, dump global user exit 83, 85
 - XDUOUT, dump global user exit 83, 86
 - XDUREQ, dump global user exit 83, 86
 - XDUREQC, dump global user exit 83, 86
 - XEJB, system initialization parameter 259, 287
 - XES, named counter server response to 409
 - XFCT, system initialization parameter 259, 288
 - XHFS, system initialization parameter 260, 288
 - XJCT, system initialization parameter 260, 288
 - XLT, system initialization parameter 261
 - XLT, transaction list table 261
 - XPCT, system initialization parameter 261, 289
 - XPPT, system initialization parameter 262, 290
 - XPSB, system initialization parameter 262, 290
 - XRES, system initialization parameter 263, 291
 - XRF (extended recovery facility)
 - actively shared data sets 9
 - ADI (alternate) 148
 - AIRDELAY parameter (active and alternate CICS) 150
 - allocation and dispositions of data sets 8
 - alternate delay 148
 - alternate delay interval 148

XRF (extended recovery facility)
(continued)

- APPLID system initialization
 - parameter 152
- AUTCONN, system initialization
 - parameter 152
- auxiliary trace data sets 80
- CLT system initialization
 - parameter 158
- command list table (CLT) 158
- control data sets 89
- CSD requirements 62
- data set considerations when running
 - CICS with XRF 8
- data set status 8
- DD statements in CICS startup job
 (DFHXRCTL) 88
- DFHCXRF data set for the alternate
 CICS 25
- DISP option for data sets 10
- DISP=SHR 62
- DUMP system initialization
 - parameter 175
- generic and specific applids 152
- GOOD MORNING transaction 223
- integrity of data on shared DASD 10
- JES delay interval 194
- JESDI system initialization
 - parameter 194
- passively shared data sets 9
- PDI system initialization
 - parameter 211, 223
- primary delay interval (PDI) 211
- reconnection delay 152
- reconnection transaction 223
- space calculations for
 - DFHXRMSG 90
- START=STANDBY (alternate) 236
- surveillance signal 148
- TAKEOVR system initialization
 - parameter 244
- transient data extrapartition data
 set 25
- user file definitions 104
- VTAM ACB at startup 314
- XRF system initialization
 - parameter 264
- XRF, system initialization parameter 264
- XRFSOFF, system initialization
 - parameter 264
- XRFSTME, system initialization
 - parameter 264
- XTRAN, system initialization
 - parameter 264, 292
- XTST, system initialization
 - parameter 264, 292
- XUSER, system initialization
 - parameter 265, 293

Readers' Comments — We'd Like to Hear from You

CICS Transaction Server for z/OS
Version 4 Release 1
System Definition Guide

Publication No. SC34-6999-02

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.
- Send a fax to the following number: +44 1962 816151
- Send your comments via email to: idrctf@uk.ibm.com

If you would like a response from IBM, please fill in the following information:

Name

Address

Company or Organization

Phone No.

Email address



Fold and Tape

Please do not staple

Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM United Kingdom Limited
User Technologies Department (MP095)
Hursley Park
Winchester
Hampshire
United Kingdom
SO21 2JN

Fold and Tape

Please do not staple

Fold and Tape



SC34-6999-02

