CICS for MVS/ESA

System Definition Guide

Version 4 Release 1

SC33-1164-03

CICS for MVS/ESA

System Definition Guide

Version 4 Release 1

SC33-1164-03

Note!

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

Second edition (April 1997)

This edition applies to Version 4 Release 1 of the IBM licensed program Customer Information Control System/Enterprise Systems Architecture (CICS/ESA), program number 5655-018, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Consult the latest edition of the applicable IBM system bibliography for current information on this product.

This is the second edition of the System Definition Guide for CICS/ESA 4.1. It is based on the first edition, SC33-1164-00, which is now obsolete. Changes from the first edition are marked by the ‘+’ sign to the left of the changes. The vertical lines in the left-hand margins indicate changes made between the CICS/ESA 3.3 edition and the CICS/ESA 4.1 first edition.

The CICS/ESA 3.3 edition remains applicable and current for users of CICS/ESA 3.3.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

At the back of this publication is a page entitled “Sending your comments to IBM”. If you want to make comments, but the methods described are not available to you, please address them to:

IBM United Kingdom Laboratories Limited, Information Development,
Mail Point 095, Hursley Park, Winchester, Hampshire, England, SO21 2JN.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Notices

The following paragraph does not apply in any 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 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.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. 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 of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact Laboratory Counsel, MP151, IBM United Kingdom Laboratories, 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.

IBM may have patents or pending patent applications covering subject matter 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 the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, U.S.A..

Trademarks and service marks

The following terms, used in this publication, are trademarks or service marks of IBM Corporation in the United States or other countries:

ACF/VTAM
AD/Cycle
BookManager
CICS
CICS/ESA
CICS/MVS
CICS OS/2
CICS/VSE
COBOL/370
DATABASE 2
DB2
DFSMS
Enterprise Systems Architecture/370
ESA/390
Hiperbatch
IBM
IBMLink
IMS/ESA
Language Environment
MVS/DFP
MVS/ESA
MVS/SP
MVS/XA
NetView
Processor Resource/Systems Manager
PR/SM
RACF
SAA
Systems Application Architecture
VSE/ESA
VTAM

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 MVS/Enterprise System Architecture (MVS/ESA) environment.

If you need to know where programming interface information is described, or about the definitions of the different types of information in the CICS library, you should read the *CICS Family: Library Guide*.

Who this book is for

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/ESA operating system, and either have previous experience of CICS, or at least be familiar with the concepts and terminology from reading either the *CICS Family: General Information* or the *CICS/ESA 3.3 Facilities and Planning Guide*. 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. For example, see 1 2 if your task is defining CICS data sets.

Notes on terminology

"CICS" is used throughout this book to mean CICS/ESA, although the formal abbreviation "CICS/ESA" is retained when quoting the titles of other CICS publications.

| "CICS/ESA Version 3" is used throughout this book to mean CICS/ESA 3.1.1 or
| later.

"RACF" is used throughout this book to mean the 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. In countries where the dollar is not the national currency, the local currency symbol should be used.

"MVS" is used throughout this book to mean the MVS/ESA operating system.

Determining if a publication is current

IBM regularly updates its publications with new and changed information. When first published, both hardcopy and BookManager softcopy versions of a publication are in step, but subsequent updates will probably be available in softcopy before they are available in hardcopy.

For CICS books, these softcopy updates appear regularly on the *Transaction Processing and Data Collection Kit* CD-ROM, SK2T-0730-xx. Each reissue of the collection kit is indicated by an updated order number suffix (the -xx part). For example, collection kit SK2T-0730-06 is more up-to-date than SK2T-0730-05. The collection kit is also clearly dated on the cover.

Here's how to determine if you are looking at the most current copy of a publication:

- A publication with a higher suffix number is more recent than one with a lower suffix number. For example, the publication with order number SC33-0667-02 is more recent than the publication with order number SC33-0667-01. (Note that suffix numbers are updated as a product moves from release to release, as well as for hardcopy updates within a given release.)
- When the softcopy version of a publication is updated for a new collection kit the order number it shares with the hardcopy version does not change. Also, the date in the edition notice remains that of the original publication. To compare softcopy with hardcopy, and softcopy with softcopy (on two editions of the collection kit, for example), check the last two characters of the publication's filename. The higher the number, the more recent the publication. For example, DFHPF104 is more recent than DFHPF103. Next to the publication titles in the CD-ROM booklet and the readme files, asterisks indicate publications that are new or changed.
- Updates to the softcopy are clearly marked by revision codes (usually a "#" character) to the left of the changes.

Book structure

1 1. Installing resource definitions ... pages 1—93
Describes how to install the various resources needed to run a CICS region, such as control tables and user application programs.

1 2. Defining data sets ... pages 95—208
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.

1 3. System initialization ... pages 209—366
Describes the system initialization parameters that you can code to initialize a CICS region tailored for your installation.

Bibliography

CICS/ESA 4.1 library

Evaluation and planning		
Release Guide	GC33-1161	April 1997
Migration Guide	GC33-1162	April 1997
General		
CICS Family: Library Guide	GC33-1226	April 1995
Master Index	SC33-1187	October 1994
User's Handbook	SX33-1188	April 1997
Glossary (softcopy only)	GC33-1189	n/a
Administration		
Installation Guide	GC33-1163	April 1997
System Definition Guide	SC33-1164	April 1997
Customization Guide	SC33-1165	April 1997
Resource Definition Guide	SC33-1166	April 1997
Operations and Utilities Guide	SC33-1167	April 1997
CICS-Supplied Transactions	SC33-1168	April 1997
Programming		
Application Programming Guide	SC33-1169	October 1994
Application Programming Reference	SC33-1170	April 1997
System Programming Reference	SC33-1171	April 1997
Sample Applications Guide	SC33-1173	October 1994
Distributed Transaction Programming Guide	SC33-1174	October 1994
Front End Programming Interface User's Guide	SC33-1175	October 1994
Diagnosis		
Problem Determination Guide	SC33-1176	October 1994
Messages and Codes	GC33-1177	April 1997
Diagnosis Handbook	LX33-6093	October 1994
Diagnosis Reference	LY33-6082	April 1997
Data Areas	LY33-6083	April 1997
Supplementary Data Areas	LY33-6081	October 1994
Closely-Connected Program Interface	LY33-6084	November 1996
Communication		
Intercommunication Guide	SC33-1181	April 1997
Server Support for CICS Clients	SC33-1591	February 1996
CICS Family: Inter-product Communication	SC33-0824	October 1996
CICS Family: Communicating from CICS on System/390	SC33-1697	October 1996
Special topics		
Recovery and Restart Guide	SC33-1182	October 1994
Performance Guide	SC33-1183	October 1994
CICS-IMS Database Control Guide	SC33-1184	October 1994
CICS-RACF Security Guide	SC33-1185	October 1994
Shared Data Tables Guide	SC33-1186	October 1994
External CICS Interface	SC33-1390	April 1997
CICS ONC RPC Feature for MVS/ESA Guide	SC33-1119	February 1996
CICS Web Interface Guide	SC33-1892	November 1996

The book that you are reading was republished in hardcopy format in April 1997 to incorporate updated information previously available only in softcopy. The right-hand column in the above table indicates the latest hardcopy editions of the CICS/ESA books available in April 1997. A book with a date earlier than April 1997 remains the current edition for CICS/ESA 4.1. Note that it is possible that other books in the library will be updated after April 1997.

When a new order is placed for the CICS/ESA 4.1 product, the books shipped with that order will be the latest hardcopy editions.

The style of IBM covers changes periodically. Books in this library have more than one style of cover.

For information about the softcopy books, see “Determining if a publication is current” on page x. The softcopy books are regularly updated to include the latest information.

Other CICS books

- *CICS Application Migration Aid Guide*, SC33-0768
- *CICS Application Programming Primer (VS COBOL II)*, SC33-0674
- *CICS/ESA Facilities and Planning Guide* for CICS/ESA Version 3 Release 3, SC33-0654
- *CICS/ESA XRF Guide* for CICS/ESA Version 3 Release 3, SC33-0661
- *CICS Family: API Structure*, SC33-1007
- *CICS Family: General Information*, GC33-0155
- *IBM CICS Transaction Affinities Utility MVS/ESA*, SC33-1159

CICS Clients

- *CICS Clients: Administration*, SC33-1436
- *CICS Family: Client/Server Programming*, SC33-1435

Books from related libraries

Systems Network Architecture (SNA)

Systems Network Architecture - Function Description of Logical Unit Types, GC20-1868
Systems Network Architecture - Types of Logical Unit to Logical Unit Sessions, GC20-1869.

Systems Application Architecture (SAA)

IBM System Application Architecture Common Programming Interface Reference Manual, SC09-1308.

Advanced communications function for VTAM (ACF/VTAM)

Network Program Products General Information, GC30-3350
Advanced Communications Function for VTAM Installation and Resource Definition, SC23-0111
Advanced Communications Function for VTAM Customization, SC23-0112
Advanced Communications Function for VTAM Operation, SC23-0113
Advanced Communications Function for VTAM Messages and Codes, SC23-0114
Advanced Communications Function for VTAM Diagnosis Guide, SC23-0116
Advanced Communications Function for VTAM Diagnosis Reference, LY30-5582

Advanced Communications Function for VTAM Data Areas, LY30-5584
Advanced Communications Function for VTAM Programming, SC23-0115
Advanced Communications Function for VTAM Reference Summary, SC23-0135.

Telecommunications Access Method (TCAM)

TCAM Concepts and Applications, GC30-2049
TCAM System Programmer's Guide, TCAM Level 10, GC30-2051
TCAM Application Programmer's Guide, TCAM Level 10, GC30-3036
TCAM Macro Reference Guide, TCAM Level 10, GC30-2052.

Virtual Storage Access Method (VSAM)

MVS/ESA Integrated Catalog Administration: Access Method Services Reference, GC26-4135
MVS/ESA Catalog Administration Guide, GC26-4138
MVS/ESA Data Administration Guide, GC26-4140
MVS/ESA VSAM Administration Guide, GC26-4151
MVS/ESA VSAM Administration: Macro Instruction Reference, GC26-4152
MVS/ESA Catalog User's Guide, GC26-4041.

DATABASE 2 (DB2)

Application Programming and SQL Guide, SC26-4377
Administration Guide, SC26-4374
Command and Utility Reference, SC26-4378.

Programming language support

VS COBOL II Application Programming Guide, SC26-4045
VS COBOL II Installation and Customization manual, SC26-4048
OS PL/I Version 2 Installation and Customization under MVS Guide, SC26-4311
IBM C/370 Installation Guide, GC09-1331
IBM C/370 Programming Guide, SC09-1384.
Language Environment/370 Programming Guide, SC26-4818
Language Environment/370 Planning for Installation and Customization Guide, SC26-4817.

Information Management System (IMS)

Title	IMS		
	VS 2.2	ESA 3.1	ESA 4.1
Installation Guide	SC26-4172	SC26-4276	SC26-4276
Installation Listings	SC26-4215	----	----
System Definition Reference	SC26-4216	SC26-4278	SC26-4278
Utilities Reference	SC26-4173	SC26-4284	----
Utilities Reference: Database	----	----	SC26-4627
Utilities Reference: Data Communication	----	----	SC26-4628
Utilities Reference: Systems	----	----	SC26-4629

IBM CICS VSAM Recovery MVS/ESA

IBM CICS VSAM Recovery MVS/ESA General Information, GH19-6707
IBM CICS VSAM Recovery MVS/ESA User's Guide and Reference, SH19-6970
IBM CICS VSAM Recovery MVS/ESA Implementation Guide, SH19-6971
IBM CICS VSAM Recovery MVS/ESA Messages and Problem Determination, SH19-4006.

Summary of changes

+ Changes for the second edition of the CICS/ESA 4.1 book

+ This book is the second edition of the *System Definition Guide* for CICS/ESA 4.1. + Changes that were made for the first edition are still indicated by vertical bars to + the left of the changes. Changes made for this second edition are indicated by the + '+' symbol to the left of the changes. Users of the first edition can therefore see + what has changed since that first edition was published. Softcopy versions of this + book use both these revision indicators and use the '#' symbol to show further + changes since this second hardcopy edition of the book was published.

+ Significant changes for this second edition include:

+ - The following new system initialization parameters:
  + – CDSASZE
  + – DAE
  + – FSSTAFF
  + – PSTYPE
  + – RDSASZE
  + – SDSASZE
  + – SYDUMAX
  + – TRDUMAX
  + – TRTRANSZ
  + – TRTRANTY
  + – UDSASZE
  + – VTPREFIX
+ - Support for application programs written in C++.
+ - New translator options OOCOBOL, COBOL3 and LINKAGE/NOLINKAGE.
+ - New global user exit XDUREQC.

| Changes to the first edition of the CICS/ESA 4.1 book

| The main changes made to this book for CICS/ESA 4.1 include:

| - The number of DSAs has increased from five to eight. The new DSAs in this | release are:
  | – The shared dynamic storage area (SDSA)
  | – The extended shared dynamic storage area (ESDSA)
  | – The read-only dynamic storage area (RDSA).

  | You do not need to specify and tune the sizes of the DSAs and their storage | cushions; CICS dynamically manages the individual DA and cushion sizes | automatically, to optimize storage usage. Also, you no longer need to restart | CICS to tune these areas enabling CICS regions to operate continuously for | longer than in previous releases.

PN70228

| The following change was made by APAR PN70228.

+ Two new system initialization parameters, DSALIM and EDSALIM, set the
+ overall dynamic storage limits both above and below the 16MB boundary.
+ Within these overall storage limits, CICS can control the individual DSA sizes
| and their associated storage cushions.

| • The removal of the following system initialization parameters:

| – CDSASZE
| – UDSASZE
| – ECDSASZE
| – EUDSASZE
| – ERDSASZE
| – CSCS
| – USCS
| – ECSCS
| – EUSCS
| – ERSCS.

| • The DB2 resource control table suffix is now specified on the DSN2STRT
| option of the INITPARM system initialization parameter, and any DSNCRCTx
| parameter on the PARM statement of the CICS startup job is ignored.

| • The removal of support for OS/VS COBOL programs.

| • The removal of support for IMS/VS 2.2.

| • The removal of support for CICS program control tables (PCTs) and program
| properties tables (PPTs). You cannot migrate PCTs and PPTs at
| CICS/ESA 4.1; you must do so at your earlier release of CICS.

| • The removal of support for CICS signon tables (SNTs). You should define
| operator definitions in the RACF database.

| • The CSECT for the VS COBOL II interface module, DFHECI, has reverted to
| DFHECI (from DFHELII).

| • The addition of new system initialization parameters, and the alteration and
| removal of some existing system initialization parameters, as listed in the
| *CICS/ESA Migration Guide*.

| • Clarification of the resource definition part of CICS system definition, in
| Chapter 1, “Resource definition—an introduction” on page 3, and as described
| in the *CICS/ESA Resource Definition Guide*.

Changes to the CICS/ESA 3.3 edition of this book.

The main changes made to this book for CICS/ESA 3.3 include:

- The removal of the DSASZE, EDSASZE, ESCS, and SCS system initialization parameters. They are replaced by new parameters to control the storage protection facilities available in CICS/ESA 3.3. The new system initialization parameters are:
  - CDSASZE
  - UDSASZE
  - ECDSASZE
  - EUDSASZE
  - ERDSASZE
  - CSCS
  - USCS

  - ECSCS
  - EUSCS
  - ERSCS.

- The addition of the FEPI system initialization parameter and the transient data queues, CSZL and CSZX, in support of the new Front End Programming Interface (FEPI) feature
- Chapter 4, “Installing application programs” on page 25 is substantially reworked to give you more detailed information about installing application programs in an easy-to-follow format
- The incorporation of information about installing user-replaceable programs that was in chapter 1.5 of the 3.2.1 edition of this book into the *CICS/ESA Customization Guide*.

Changes to the CICS/ESA 3.2.1 edition of this book

The CICS/ESA 3.2.1 edition of the System Definition Guide is based on the 3.1.1 edition.

The main changes made to this book for CICS/ESA 3.2.1 include:

- Incorporation of the security information that was in chapter 1.8 of the 3.1.1 edition of this book into the *CICS/ESA CICS-RACF Security Guide*
- Removal of CICS internal security, and transfer of its function to the Resource Access Control Facility (RACF)
- Freeing of more virtual storage by moving CICS tables and BMS map sets above the 16MB[1] line
- Removal of macro-level programming support, and change of user-replaceable programs to command-level
- Removal of support for user-replaceable security programs
- Addition of Advanced-Program-to-Program (APPC) session security
- Provision of the DISCARD command for CICS RDO resources
- Additions to transient data destinations used by CICS
- Support for CICS operations from MVS extended consoles (time sharing option (TSO) and automated processes, such as NetView)
+ - Support for IMS/ESA 4.1 with local DL/I
- Support for backup and restoration of VSAM files open for update, known as “backup while open” (BWO)
- Support for Common Programming Interface for Communications (CPI Communications)
- Support for Systems Application Architecture (SAA) resource recovery
- Support for Language Environment/370
- Support for extended recovery across MVS images in a sysplex using MVS/ESA SP 4.1 or later and PR/SM

[1] MB equals 1 048 576 bytes.

- Changes and additions to the CICS system initialization parameters
- Provision of jobs to create the CICS data sets (needed for the CICS-supplied installation verification procedures (IVPs) and useful as a basis for your own data sets)
- Provision of a new CICS startup procedure, DFHSTART, that can be used to start the CICS pregenerated system, CICS MRO regions, or as a basis for your own startup procedures
- Changes to all the CICS library names known to SMP/E (see Table 1) to conform to the MVS product packaging rules, including changes to supplied JCL procedures containing references to the libraries.

Table 1. CICS/ESA 3.2.1 library name changes

Old library name	**New distribution library**	**New target library**
CEELIB	-	SDFHC370
COBLIB	-	SDFHCOB
DCEELIB	ADFHC370	-
DCOBLIB	ADFHCOB	-
DLOADLIB	ADFHMOD	-
DPL1LIB	ADFHPL1	-
JCLLIB	ADFHINST	SDFHINST
LOADLIB	-	SDFHLOAD
LOADLIB1	-	SDFHAUTH
MACLIB	ADFHMAC	SDFHMAC
PL1LIB	-	SDFHPL1
PROCLIB	ADFHPROC	SDFHPROC
SAMPLIB	ADFHSAMP	SDFHSAMP
SOURCE	ADFHSRC	SDFHSRC
-	ADFHAPD1	SDFHAPD1
-	ADFHAPD2	SDFHAPD2
-	ADFHMSGS	SDFHMSGS

Under the new installation process for CICS/ESA 3.3, most CICS libraries now have both an SMP/E distribution library (ADFHxxxx) and an SMP/E target library (SDFHxxxx).

Part 1. Installing resource definitions

After you have installed CICS, as described in the *CICS/ESA Installation Guide*, you need to install all the resources that your CICS regions need to run user transactions, and define those resources to CICS. This section of the book describes how to install resources and resource definitions in a CICS region.

It consists of the following chapters:

Chapter 1. Resource definition—an introduction

Before you can use CICS, you must supply it with information about the resources it should use, and how it should use them. Some examples of resources are:

- Terminals
- Files
- Programs
- Journals
- Transactions
- Connections
- Databases

Your CICS system has to know which resources to use, what their properties are, and how they are to interact with each other.

You supply this information to CICS by using one or more of the four methods of **resource definition**:

- **Resource definition online (RDO)**: This method uses the CICS-supplied online transactions CEDA, CEDB, and CEDC. Definitions are stored on the CICS system definition file (CSD), and are installed into an active CICS from the CSD.
- **DFHCSDUP offline utility**: This method also stores definitions in the CSD. DFHCSDUP allows you to make changes to definitions in the CSD by means of a batch job submitted offline.
- **Automatic installation (autoinstall)**: Autoinstall minimizes the need for a large number of definitions, by dynamically creating new definitions based on a "model" definition provided by you.
- **Macro definition**: You can use assembler macro source to define resources. Definitions are stored in assembled tables in a program library, from where they are installed during CICS initialization.

Depending on the resources you want to define, you can use one or more of these methods.

Table 2 on page 4 shows you the methods you can use for each resource.

Table 3 on page 5 suggests some of the things you should consider when deciding which definition method to use when there is a choice.

Table 2. Resources and how you can define them

Resource	RDO	DFHCSDUP	Autoinstall	Macro
Connections	Yes	Yes	Yes	No
Databases	No	No	No	Yes
Files (BDAM)	No	No	No	Yes
Files (VSAM)	Yes	Yes	No	No
Journals	No	No	No	Yes
Local shared resource (LSR) pools	Yes	Yes	No	No
Mapsets	Yes	Yes	Yes	No
Partitionsets	Yes	Yes	Yes	No
Partners	Yes	Yes	No	No
Profiles	Yes	Yes	No	No
Programs	Yes	Yes	Yes	No
Queues (destinations)	No	No	No	Yes
Recoverable service elements	No	No	No	Yes
Sessions	Yes	Yes	No.	No
Shared data tables	Yes	Yes	No	No
Temporary storage	No	No	No	Yes
Terminals (non-VTAM)	No	No	No	Yes
Terminals (VTAM)	Yes	Yes	Yes	No
Transactions	Yes	Yes	No	No
Transaction classes	Yes	Yes	No	No
Typeterms	Yes	Yes	No	No

Table 3. Methods of resource definition

Method	Description	Advantages	Disadvantages
Resource definition online (RDO)	This method uses the CEDA transaction, which allows you to define, alter, and install resources in a running CICS system.	• RDO is used while CICS is running, so allows fast access to resource definitions.	• Because CEDA operates on an active CICS system, care should be taken if it is used in a production system, and you should use some form of auditing as a control mechanism.
DFHCSDUP offline utility	DFHCSDUP is an offline utility which allows you to define, list, and modify resources by means of a batch job. DFHCSDUP can be invoked as a batch program or from a user-written program running either in batch mode or under TSO - using the second method, you can specify up to five user exit routines within DFHCSDUP.	• You can modify or define a large number of resources in one job. • You can run DFHCSDUP against a CSD as long as it is not in use.	• You cannot install resources into an active CICS system. • If you are sharing a CSD between regions, it must not be in use by another region while you are running DFHCSDUP.
Automatic installation (autoinstall)	This applies to VTAM terminals, LU62 sessions, programs, mapsets, and partitionsets. You set up "model" definitions using either RDO or DFHCSDUP, then CICS can create and install new definitions for these resources dynamically, based on the models.	• If you have large numbers of resources, a lot of time could be taken up defining them, and if they are not all subsequently used, storage is also wasted for their definitions. Using autoinstall reduces this wasted time and storage.	• You must spend some time initially setting up autoinstall in order to benefit from it.
Macro tables	Using this method, you code and assemble macroinstructions to define resources in the form of tables.	• Where possible, you should use the other methods.	• You can change the definitions contained in the tables while CICS is running, but you must stop and restart CICS if you want it to use the changed tables. • You must do time-consuming assemblies to generate macro tables.

For information about CICS resource definition, see the *CICS/ESA Resource Definition Guide*. For information about the DFHCSDUP utility, see the *CICS/ESA Operations and Utilities Guide*.

Resource definitions in the CSD are stored in **groups**. You specify the resource definitions needed by a particular run of CICS by a **list** of groups to be installed during CICS initialization, using the GRPLIST=listname system initialization parameter. You can also use the CEDA INSTALL command to install a resource definition or group of definitions defined in the CSD[2] dynamically on a running CICS region.

[2] The CSD is independent of the running CICS region, because when you install the definitions in the CICS region, CICS copies the information and keeps it in its own storage. Because CICS does this, you can modify the CSD without interfering with the running CICS region. You can also change the definitions in the running CICS region by reinstalling them, or add more definitions by installing new ones.

You should limit read/write access to resource definitions in the CSD to a small number of people. To do this, you can:

- Protect groups of resources by using the CEDA command LOCK
- Protect the list of resource groups specified in the system initialization parameter GRPLIST by using the CEDA command LOCK
- Use the CEDB transaction to create resource definitions, but not to INSTALL them
- Use the CEDC transaction for read-only access to resource definitions.

CICS control tables contain resource definition records for resources that cannot be defined in the CSD. The tables and their resource definitions are created by using the CICS table assembly macro instructions. You must use macro instructions to define non-VTAM networks and terminals, non-VSAM files, databases, journals, queues, and resources for monitoring and system recovery. For more information about defining resources in CICS control tables, see Chapter 2, “Defining resources in CICS control tables” on page 7.

Using the CSD and control tables together

In two cases, you can mix resources defined in the CSD with resources defined in control tables. These are:

1. On a cold start, you can mix file control resources defined in the CSD with those that were defined using DFHFCT macros. BDAM file definitions are loaded from the DFHFCT load module first, then the definitions for other types of files are loaded from the RDO groups specified in the GRPLIST system initialization parameter. Any definitions in the FCT other than for BSAM files (to allow migration of the FCT to the CSD) are ignored. Once CICS is running, you can use CEDA commands to add more file resource definitions.
2. You can also mix terminal resource definitions for non-VTAM[3] terminals defined in a TCT with resource definitions for VTAM terminals defined using RDO.

   However, avoid duplicate terminal IDs, because a TCT entry using the same terminal ID (TERMIDNT in the TCT) as a VTAM terminal in the CSD (TERMINAL name in the CSD), prevents CICS installing the VTAM definition.

[3] Non-VTAM terminals. TCT entries can be for TCAM DCB terminals, BSAM sequential devices, logical device codes (LDCs), and remote BTAM terminals required for ISC/MRO purposes.

Chapter 2. Defining resources in CICS control tables

This chapter describes what you should do to define resource definitions in CICS control tables. The tables and their resource definitions are created by using macros. You must use macros to define: non-VTAM networks and terminals, non-VSAM files, databases, journals, queues, and resources for monitoring and system recovery. For details about defining resource definitions in CICS control tables, and about migrating your tables to the CSD, see the *CICS/ESA Resource Definition Guide*.

For each of the CICS tables (listed on page 8) complete the following steps:

1. Code the resource definitions you require.
2. Assemble and link-edit these definitions, using the CICS-supplied procedure DFHAUPLE, to create a load module in the required CICS load library. The load library is either CICS410.SDFHLOAD or CICS410.SDFHAUTH, which you must specify by the NAME parameter of the DFHAUPLE procedure. The CICS-supplied macros used to create the CICS tables determine whether tables are loaded above the 16MB line. All tables, other than the JCT and TCT, are loaded above the 16MB line.
3. Name the suffix of the load module by a system initialization parameter. For most of the CICS tables, if you do not require the table you can code *tablename=NO*. The exceptions to this rule are as follows:
   - The CLT. Specifying CLT=NO causes CICS to try and load DFHCLTNO. The CLT is only used in the alternate CICS, when you are running CICS with XRF, and is always required in that case.
   - The SIT. Specifying SIT=NO causes CICS to try and load DFHSITNO. The SIT is always needed, and you can specify the suffix by coding the SIT system initialization parameter.
   - The TCT. Specifying TCT=NO causes CICS to load a dummy TCT named DFHTCTDY, as explained on page 320.
   - The TLT. Terminal list tables are specified by program entries in the CSD, and do not have a system initialization parameter.
   - The MCT. Specifying MCT=NO causes the CICS monitoring domain to build 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 (or classes) are active.
4. If you are running CICS with XRF, the active and the alternate CICS regions share the same versions of tables. However, to provide protection against DASD failure, you might want to run your active and alternate CICS regions from separate sets of load libraries–in which case, you should make the separate copies ***after*** generating your control tables.

Table 4 lists all the CICS tables that can be assembled, link-edited, and installed in your CICS libraries for use in your CICS system.

Table 4. CICS tables that you can assemble, link-edit, and install in CICS libraries

Table	Module Name	Abbreviation	Required load library
Command list table	DFHCLTxx	CLT	SDFHAUTH
Data conversion table	DFHCNV	CNV	SDFHLOAD
Destination control table	DFHDCTxx	DCT	SDFHLOAD
DL/I data management block	DFHDMBxx	DMB	SDFHLOAD
File control table	DFHFCTxx	FCT	SDFHLOAD
Journal control table	DFHJCTxx	JCT	SDFHLOAD
Monitor control table	DFHMCTxx	MCT	SDFHLOAD
Program list table	DFHPLTxx	PLT	SDFHLOAD
DL/I program specification block	DFHPSBxx	PSB	SDFHLOAD
Recoverable service element table	DFHRSTxx	RST	SDFHAUTH
System initialization table	DFHSITxx	SIT	SDFHAUTH
System recovery table	DFHSRTxx	SRT	SDFHLOAD
Terminal control table	DFHTCTxx	TCT	SDFHLOAD
Terminal list table	DFHTLTxx	TLT	SDFHLOAD
Temporary storage table	DFHTSTxx	TST	SDFHLOAD
Transaction list table	DFHXLTxx	XLT	SDFHLOAD

You can generate several versions of each CICS control table by specifying SUFFIX=xx in the macro that generates the table. This suffix is then appended to the default 6-character name of the load module.

To get you started, CICS provides the sample tables listed in Table 5 in the CICS410.SDFHSAMP library:

Table 5. Sample CICS system tables in the CICS410.SDFHSAMP library

Table	Suffix	Notes
Command list table (CLT)	1$	XRF regions only
Destination control table (DCT)	2$	
DL/I data management block (DMB)	2$	For DL/I IVP, DFHIVPDL
Journal control table (JCT)	2$	
Monitor control table (MCT)	A$	For a CICS AOR
Monitor control table (MCT)	F$	For a CICS FOR
Monitor control table (MCT)	T$	For a CICS TOR
Monitor control table (MCT)	2$	
DL/I program specification block (PSB)	1$	For DL/I IVP, DFHIVPDL
System initialization table (SIT)	$$	Default system initialization parameters
System initialization table (SIT)	6$	
System recovery table (SRT)	1$	
Terminal control table (TCT)	5$	Non-VTAM terminals only

Unless you have TCAM terminals or are using sequential devices, you do not need a TCT and should specify TCT=NO. (For information about the effect of TCT=NO, see page 320.) You define VTAM terminals in the CSD only, either explicitly or by means of autoinstall model definitions, but for non-VTAM terminals you must use

DFHTCT macros. For a summary of how you can define files and terminals to CICS, depending on the access methods you are using for these resources, see Table 2 on page 4.

Although you can modify and reassemble the tables while CICS is running, you must shut down and restart CICS to make the new tables available to CICS. (The command list table (CLT) is an exception in that a new table can be brought into use without shutting down either the active CICS region or the alternate CICS region.)

Migration

If you are using RACF 1.9, define operator characteristics in the RACF database. If you are using an SNT, containing operator characteristics, with an earlier release of CICS, you should transfer these characteristics into the CICS segment of the RACF 1.9 database. For migration considerations that apply specifically to security, see the *CICS/ESA CICS-RACF Security Guide*.

| Local BTAM terminals are not supported in CICS/ESA 4.1 regions. However, BTAM terminals in CICS regions running at an earlier release of CICS are supported by transaction routing from those CICS regions to the CICS/ESA 4.1 region.

| The PCTs, PPTs, SNTs, VTAM TCTs, and local BTAM TCTs are not supported in
| CICS/ESA 4.1. You can use other control tables that you used with earlier releases. But, first check that the source statements are still valid (some parameters may have changed, or been added or deleted) then reassemble the tables against the CICS/ESA 4.1 libraries.

If you used PCTs, PPTs, or TCTs containing VTAM terminal definitions with an
| earlier release of CICS, these ***must*** be migrated to the CSD. You cannot migrate
| PCTs and PPTs at CICS/ESA 4.1; you must do so at your earlier release of CICS.
| However, CICS/ESA 4.1 does provide macros (for migration purposes) for you to
| reassemble your TCTs against before migrating them to the CSD.

| 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 (to allow migration of the FCT
| to the CSD) are ignored. ***Before reassembling an FCT, remove all non-BDAM
| entries, because non-BDAM entries are ignored and cause an MNOTE during
| assembly***. For the latest SIT definitions, see Chapter 21, “CICS system initialization parameters” on page 211.

| Defining control tables to CICS

You can assemble and link-edit more than one version of a table, and (except for
the CNV) and use a suffix to distinguish them. To specify which version you want
| CICS to use, you code a system initialization parameter of the form *tablename=xx*.
| (For example, TCT=5$.) However, you can code the SIT=xx parameter only as a
| startup override; that is, not in the DFHSIT table. For details of all the CICS

| system initialization parameters, see Chapter 21, “CICS system initialization
| parameters” on page 211.

| Other tables that have special requirements are program list tables (PLTs), terminal
list tables (TLTs), and transaction list tables (XLTs). For each TLT, autoinstall for
programs must be active or you must specify a program resource definition in the
CSD, using the table name (including the suffix) as the program name. PLTs or
XLTs are autoinstalled if there is no program resource definition in the CSD. For
| example, to generate a TLT with a suffix of AA (DFHTLTAA), the CEDA command
| would be as follows:

```
| CEDA DEFINE PROGRAM(DFHTLTAA) GROUP(grpname) LANGUAGE(ASSEMBLER)
```

See 267 for information about single and two-character suffixes on MCTs.

For information about program and terminal list tables, see the *CICS/ESA Resource Definition Guide*.

The DFHCNV conversion table is required when communicating with CICS on a
non-System 390 platform. For information about the Data Conversion Process, see
the *CICS Family: Communicating from CICS on System/390*.

The command list table (CLT) is used only by the alternate CICS in a CICS system running with XRF=YES, and differs in many other respects from the other CICS tables. For more guidance information, including information on resource definition specific to the CICS extended recovery facility, see the *CICS/ESA 3.3 XRF Guide*.

Assembling and link-editing control tables

To assemble and link-edit your tables, write a job that invokes the CICS-supplied sample procedure DFHAUPLE. The DFHAUPLE procedure needs the following libraries to be online:

Table 6. Library requirements for the DFHAUPLE procedure

Library name	Tables required for
SYS1.MACLIB	All
CICS410.SDFHMAC	All
CICS410.SDFHLOAD	All except the CLT, RST, and SIT
CICS410.SDFHAUTH	CLT, RST, and SIT
SMP/E global zone	All
SMP/E target zone	All

When you have assembled and link-edited your tables, define them to CICS by system initialization parameters.

The DFHAUPLE procedure

The DFHAUPLE procedure, shown in Figure 1 on page 12, is tailored to your CICS environment and stored in the CICS410.XDFHINST library when you run the DFHISTAR job. (For information about DFHISTAR, see the *CICS/ESA Installation Guide*.) It consists of the following steps:

1. ASSEM: This step puts your table definition macros into a temporary partitioned data set (PDS) member that is used as input to the ASM and SMP steps. The BLDMBR step subsequently adds further members to this temporary PDS.
2. ASM: In this assembly step, SYSPUNCH output is directed to a temporary sequential data set. This output consists of IEBUPDTE control statements, linkage editor control statements, SMP control statements, and the object deck.
3. BLDMBR: In this step, the IEBUPDTE utility adds further members to the temporary PDS created in the ASSEM step. These members contain linkage editor control statements and SMP control statements, and the object deck from the assembly step.
4. LNKEDT: The link-edit step uses the contents of the PDS member LNKCTL as control statements. The object code produced in step 2 is included from the temporary PDS. The output is placed in the load library specified by the NAME parameter on the procedure. You must specify NAME=SDFHAUTH for the CLT, RST, and SIT, and NAME=SDFHLOAD for all the others.
5. ZNAME: This step creates a temporary data set, which is passed to the SMP/E JCLIN job step, containing a SET BDY command defining a target zone name. This tells SMP/E which target zone to update.
6. SMP: The SMP step uses the temporary PDS members MACROS, SMPCNTL, SMPJCL1, SMPJCL2, LNKCTL, SMPEOF, and the object deck to update the control data set (CDS).
7. DELTEMP: This final step deletes the temporary partitioned data set, &&TEMPPDS.

   This step must run successfully if you want SMP to reassemble CICS tables automatically when applying later maintenance.

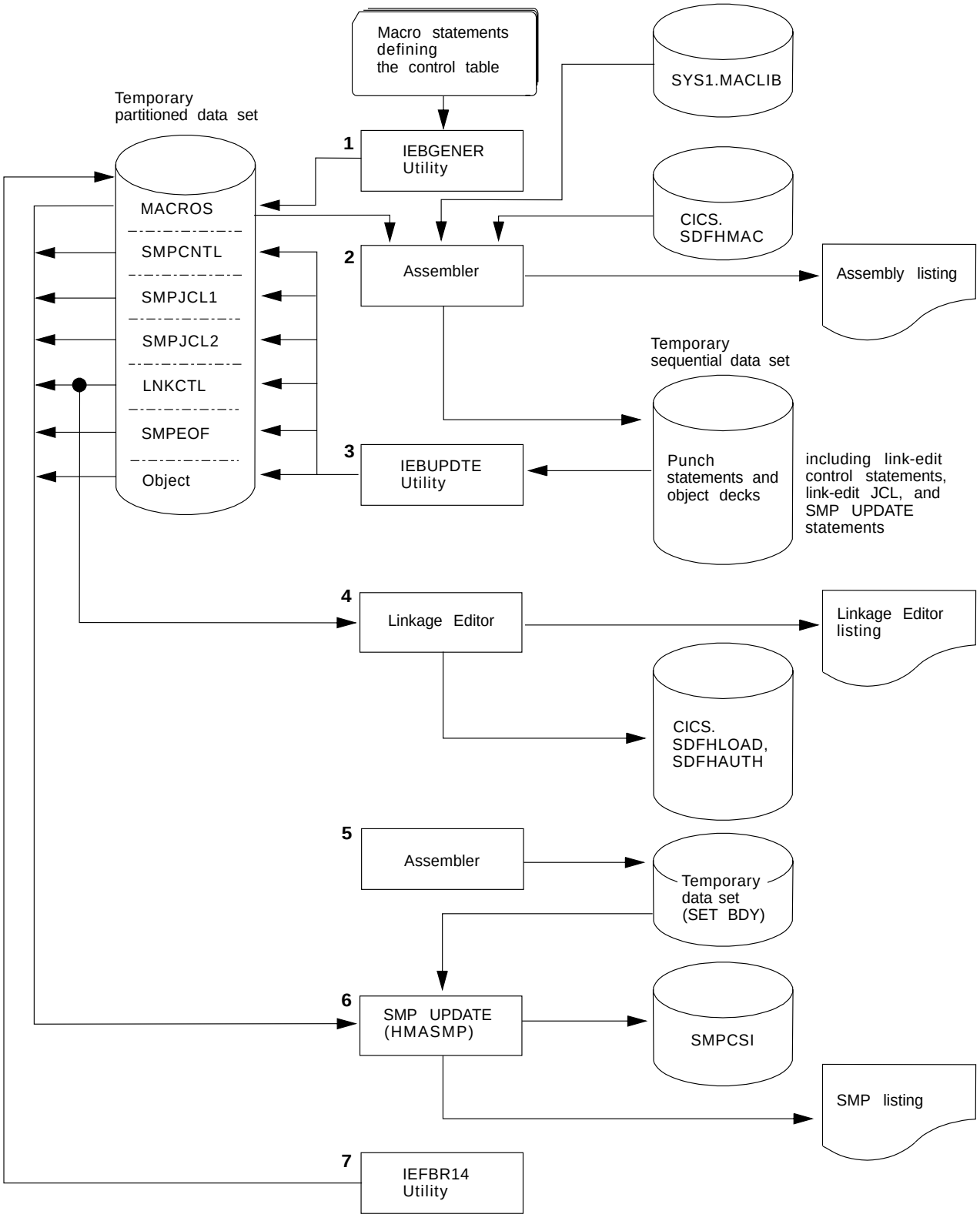


Figure 1. Assembling and link-editing the control tables using the DFHAUPLE procedure

Sample job stream for non-XRF tables

You can use the following job stream to assemble and link-edit all of the CICS control tables except for the CLT and the RST:

```
//jobname      JOB     (accounting information),
//             CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//ASMTAB       EXEC    PROC=DFHAUPLE[,NAME={SDFHLOAD|SDFHAUTH}]
//*
//ASSEM.SYSUT1 DD   *
         .
      Control table macro statements
         .
/*
```

Figure 2. Job stream for non-XRF tables

Note: Specify NAME=SDFHAUTH on this job for the system initialization table only; link-edit all other tables (except the CLT and RST) into SDFHLOAD.

Sample job stream for XRF-related tables

If you use CICS with XRF, there are two other tables that are assembled and link-edited using the DFHAUPLE procedure. These are:

CLT The command list table. This gives a list of MVS system commands and messages to the operator that CICS issues if a takeover occurs.

RST The recoverable service table. 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.

Both of these tables must be link-edited, with the reentrant attribute, into an APF authorized library. You specify that the table is reentrant on the RENTATT parameter of the DFHAUPLE procedure. A sample job to call the DFHAUPLE procedure for the CLT and RST is as follows:

```
//jobname      JOB     (accounting information),
//             CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//             EXEC    PROC=DFHAUPLE,NAME=SDFHAUTH,RENTATT=RENT
//*
//ASSEM.SYSUT1 DD   *
         .
      Control table macro statements
         .
/*
```

Figure 3. Job stream for XRF-related tables

Chapter 3. Installing map sets and partition sets

This chapter describes how to assemble and link-edit map sets and partition sets for use with the basic mapping (BMS) facility of CICS, assuming that they are defined by CICS-supplied macros.

CICS also supports the definition of BMS map sets and partition sets interactively by using licensed programs such as the IBM Screen Definition Facility II (SDF II), program number 5665-366.

For information about writing programs to use BMS services, see the *CICS/ESA Application Programming Guide*.

CICS loads BMS map sets and partition sets above the 16MB line if you specify the residency mode for the map set or partition set as RMODE(ANY) in the link-edit step. If you are using either map sets or partition sets from before CICS/ESA 4.1, you can load them above the 16MB line by link-editing them again with RMODE(ANY). For examples of link-edit steps specifying RMODE(ANY), see the sample job streams in this chapter.

CICS provides the DFHASMVS procedure (installed in the CICS410.SDFHPROC library) that you can use to assemble and link-edit map sets.

Note: The DFHASMVS procedure refers to the MVS library SYS1.MODGEN. If you have not yet restructured MVS/ESA (moving members from SYS1.AMODGEN to SYS1.MODGEN), change the SYS1.MODGEN reference to SYS1.AMODGEN in the DFHASMVS procedure, until you have restructured MVS/ESA. When you have restructured MVS/ESA, you must return the SYS1.AMODGEN reference to SYS1.MODGEN.

Installing map sets

This section first describes the types of map sets, how you define them, and how CICS recognizes them. This is followed by a description of how to prepare physical map sets and symbolic description map sets separately. Finally, there is a description of how to prepare both physical and symbolic description map sets in one job. In these descriptions, it is assumed that the SYSPARM parameter is used to distinguish the two types of map sets.

Types of map sets

To install a map set, you must actually prepare two types of map sets:

- A **physical** map set, used by BMS to translate data from the standard device-independent form used by application programs to the device-dependent form required by terminals.
- A **symbolic description** map set, used in the application program to define the standard device-independent form of the user data. This is a DSECT in assembler language, a data definition in COBOL, a BASED or AUTOMATIC structure in PL/I, and a “struct” in C/370.

Physical map sets must be cataloged in the CICS load library. Symbolic description map sets can be cataloged in a user copy library, or inserted directly into the application program itself.

The map set definition macros are assembled twice; once to produce the physical map set used by BMS in its formatting activities, and once to produce the symbolic description map set that is copied into the application program.

Defining the type of map set you require

The two types of map set can be distinguished by either:

- The TYPE operand of the DFHMSD macro
- Use of the SYSPARM operand on the EXEC statement of the job used to assemble the map set.

If you use the SYSPARM operand for this purpose, the TYPE operand of the DFHMSD macro is ignored. Using SYSPARM allows both the physical map set and the symbolic description map set to be generated from the same unchanged set of BMS map set definition macros.

Map sets can be assembled as either **unaligned** or **aligned** (an aligned map is one in which the length field is aligned on a halfword boundary). Use unaligned maps except in cases where an application package needs to use aligned maps.

The SYSPARM value alone determines whether the map set is aligned or unaligned, and is specified on the EXEC PROC=DFHASMVS statement. The TYPE operand of the DFHMSD macro can only define whether a physical or symbolic description map set is required. The SYSPARM operand can also be used to specify whether a physical map set or a symbolic description map set (DSECT) is to be assembled, in which case it overrides the TYPE operand. If neither operand is specified, an unaligned DSECT is generated.

For the possible combinations of operands to generate the various types of map set, see Table 7.

Table 7. SYSPARM and DFHMSD operand combinations for map assembly

Type of map set	SYSPARM operand of EXEC DFHASMVS statement	TYPE operand of DFHMSD macro
Aligned symbolic description map set (DSECT)	A A ADSECT	Not specified DSECT Any (takes SYSPARM)
Aligned physical map set	A AMAP	MAP Any (takes SYSPARM)
Unaligned symbolic description map set (DSECT)	Not specified Not specified DSECT	Not specified DSECT Any (takes SYSPARM)
Unaligned physical map set	Not specified MAP	MAP Any (takes SYSPARM)

How CICS recognizes map set type

For releases of CICS earlier than 1.6.0, BMS is generated by the DFHSG PROGRAM=BMS macro, and assumes that all map sets are aligned, or that all map sets are unaligned. Thus aligned and unaligned map sets cannot be mixed.

For CICS 1.6.0 and later releases, the physical map set indicates whether it was assembled for aligned or unaligned maps. This information is tested at execution time, and the appropriate map alignment used. Thus aligned and unaligned map sets can be mixed.

The alignment information is missing from physical map sets assembled before CICS 1.6.0. Use the system initialization parameter BMS to indicate whether map sets assembled before CICS 1.6.0 are aligned or unaligned. For information about the BMS system initialization options, see page 233.

Using extended data stream terminals

Applications and maps designed for the 3270 Information Display System run unchanged on devices supporting extensions to the 3270 data stream such as color, extended highlighting, programmed symbols, and validation. To use fixed extended attributes such as color, you only need to reassemble the physical map set. If dynamic attribute modification by the application program is needed, you must reassemble both the physical and symbolic description map sets, and you must reassemble or recompile the application program.

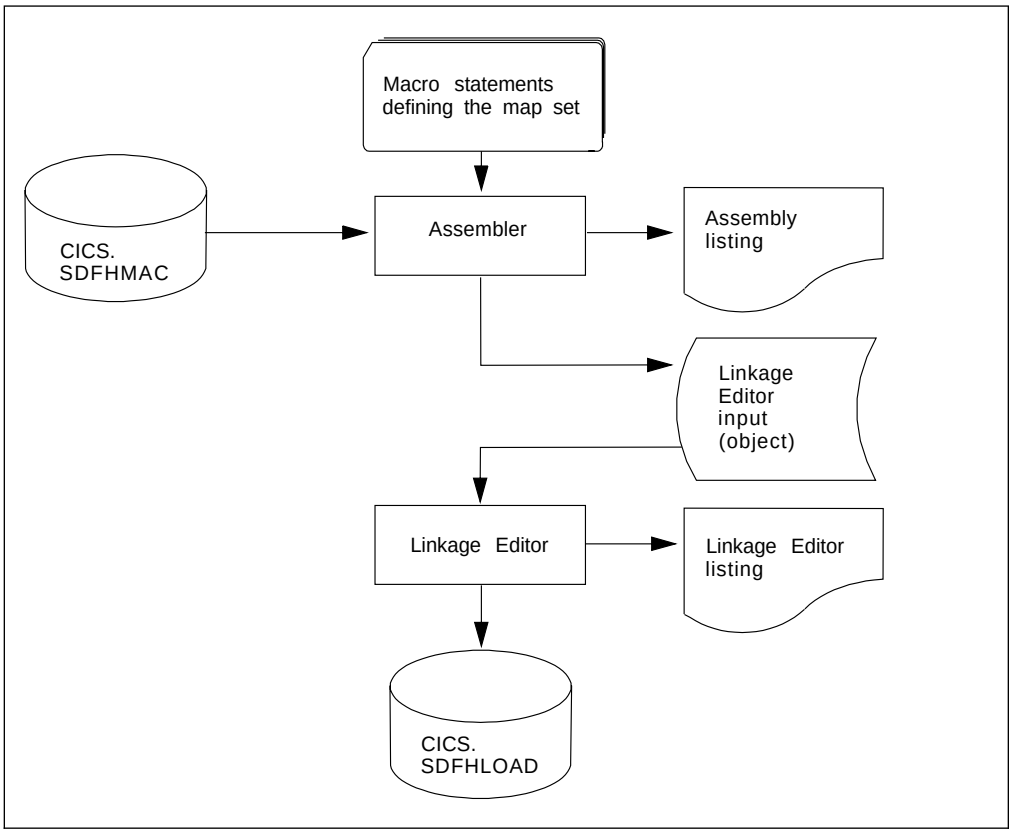


Figure 4. Installing physical map sets

Installing physical map sets

Figure 4 on page 17 shows the procedure for installing physical map sets.

Figure 5 gives an example job stream for the assembly and link-editing of physical map sets.

```
//PREP     JOB  'accounting information',CLASS=A,MSGLEVEL=1
//STEP1    EXEC PROC=DFHASMVS,PARM.ASSEM='SYSPARM(MAP)'          1
//SYSPUNCH DD   DSN=&&TEMP,DCB=(RECFM=FB,BLKSIZE=2960),
//         SPACE=(2960,(10,10)),UNIT=SYSDA,DISP=(NEW,PASS)
//SYSIN    DD   *
  ⋮
      Macro statements defining the map set
  ⋮
/*
//STEP2    EXEC PROC=DFHLNKVS,PARM='LIST,LET,XREF'               2
//SYSLIN   DD   DSN=&&TEMP,DISP=(OLD,DELETE)
//         DD   *
           MODE RMODE(ANY|24)                                    3
           NAME mapsetname(R)                                    4
/*
//
```

Figure 5. Assembling and link-editing a physical map set

Notes:

1 For halfword-aligned length fields, specify the option SYSPARM(AMAP) instead of SYSPARM(MAP).

2 Physical map sets are loaded into CICS-key storage, unless they are link-edited with the RMODE(ANY) and RENT options. If they are link-edited with these options, they are loaded into key-0 protected storage, provided that RENTPGM=PROTECT is specified on the RENTPGM initialization parameter.

However, it is recommended that map sets should not be link-edited with the RENT or the REFR options because, in some cases, CICS modifies the map set.

For more information about the storage protection facilities available in CICS/ESA 4.1, see “Storage protection” on page 358.

3 The MODE statement specifies whether the map set is to be loaded above (RMODE(ANY)) or below (RMODE(24)) the 16MB line. RMODE(ANY) indicates that CICS can load the map set anywhere in virtual storage, but tries to load it above the 16MB line, if possible.

4 Use the NAME statement to specify the name of the physical map set that BMS loads into storage. If the map set is device-dependent, derive the map set name by appending the device suffix to the original 1- to 7-character map set name used in the application program. The suffixes to be appended for the various terminals supported by CICS BMS depend on the parameter specified in the TERM or SUFFIX operand of the DFHMSD macros used to define the map set. For programming information giving a complete list of map set suffixes, see the *CICS/ESA Application Programming Reference* manual.

To use a physical map set, you must define and install a resource definition for it. You can do this either by using the program autoinstall function or by using the CEDA DEFINE MAPSET and INSTALL commands, as described in “Defining programs, map sets, and partition sets to CICS” on page 65.

Installing symbolic description map sets

Symbolic description map sets enable the application programmer to make symbolic references to fields in the physical map set. Figure 6 shows the preparation of symbolic description map sets for BMS.

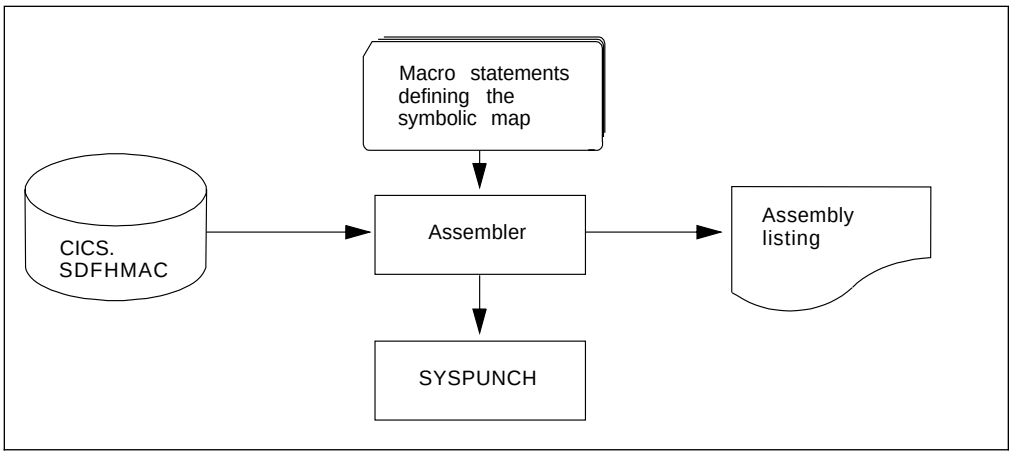


Figure 6. Installing symbolic description map sets using the DFHASMVS procedure

To use a symbolic description map set in a program, you must assemble the source statements for the map set and obtain a punched copy of the storage definition through SYSPUNCH. The first time this is done, you can direct the SYSPUNCH output to SYSOUT=A to get a listing of the symbolic description map set. If many map sets are to be used at your installation, or there are multiple users of common map sets, establish a private user copy library for each language that you use.

When a symbolic description is prepared under the same name for more than one programming language, a separate copy of the symbolic description map set must be placed in each user copy library. You must ensure that the user copy libraries are correctly concatenated with SYSLIB.

You need only one symbolic description map set corresponding to all the different suffixed versions of the physical map set. For example, to run the same application on terminals with different screen sizes, you would:

1. Define two map sets each with the same fields, but positioned to suit the screen sizes. Each map set has the same name but a different suffix, which would match the suffix specified for the terminal.
2. Assemble and link-edit the different physical map sets separately, but create only one symbolic description map set, because the symbolic description map set would be the same for all physical map sets.

You can use the sample job stream in Figure 7 to obtain a listing of a symbolic description map set. It applies to all the programming languages supported by CICS.

```
//DSECT    JOB  'accounting information',CLASS=A,MSGLEVEL=1
//ASM      EXEC PROC=DFHASMVS,PARM.ASSEM='SYSPARM(DSECT)'
//SYSPUNCH DD   SYSOUT=A
//SYSIN    DD   *
  ⋮
      Macro statements defining the map set
  ⋮
/*
//
```

Figure 7. Listing of a symbolic description map set

If you want to assemble symbolic description map sets in which length fields are halfword-aligned, change the EXEC statement of the sample job in Figure 7 to the following:

```
//ASSEM  EXEC PROC=DFHASMVS,PARM.ASSEM='SYSPARM(ADSECT)'
```

To obtain a punched copy of a symbolic description map set, code the //SYSPUNCH statement in the above example to direct output to the punch data stream. For example:

```
//SYSPUNCH DD   SYSOUT=B
```

To store a symbolic description map set in a private copy library, use job control statements similar to the following:

```
//SYSPUNCH DD   DSN=USER.MAPLIB.ASM(map set name),DISP=OLD
//SYSPUNCH DD   DSN=USER.MAPLIB.COB(map set name),DISP=OLD
//SYSPUNCH DD   DSN=USER.MAPLIB.PLI(map set name),DISP=OLD
```

Installing physical and symbolic description maps together

Figure 8 on page 21 shows the DFHMAPS procedure for installing physical and symbolic description maps together. The DFHMAPS procedure consists of the following four steps, shown in Figure 8:

1. The BMS macros that you coded for the map set are added to a temporary sequential data set.
2. The macros are assembled to create the physical map set. The MAP option is coded in the SYSPARM global variable in the EXEC statement (PARM='SYSPARM(MAP)').
3. The physical map set is link-edited to the CICS load library.
4. Finally, the macros are assembled again, this time to produce the symbolic description map set. In this step, DSECT is coded in the SYSPARM global variable in the EXEC statement (PARM='SYSPARM(DSECT)'). Output is directed to the destination specified in the //SYSPUNCH DD statement. In the DFHMAPS procedure, that destination is the CICS410.SDFHMAC library.

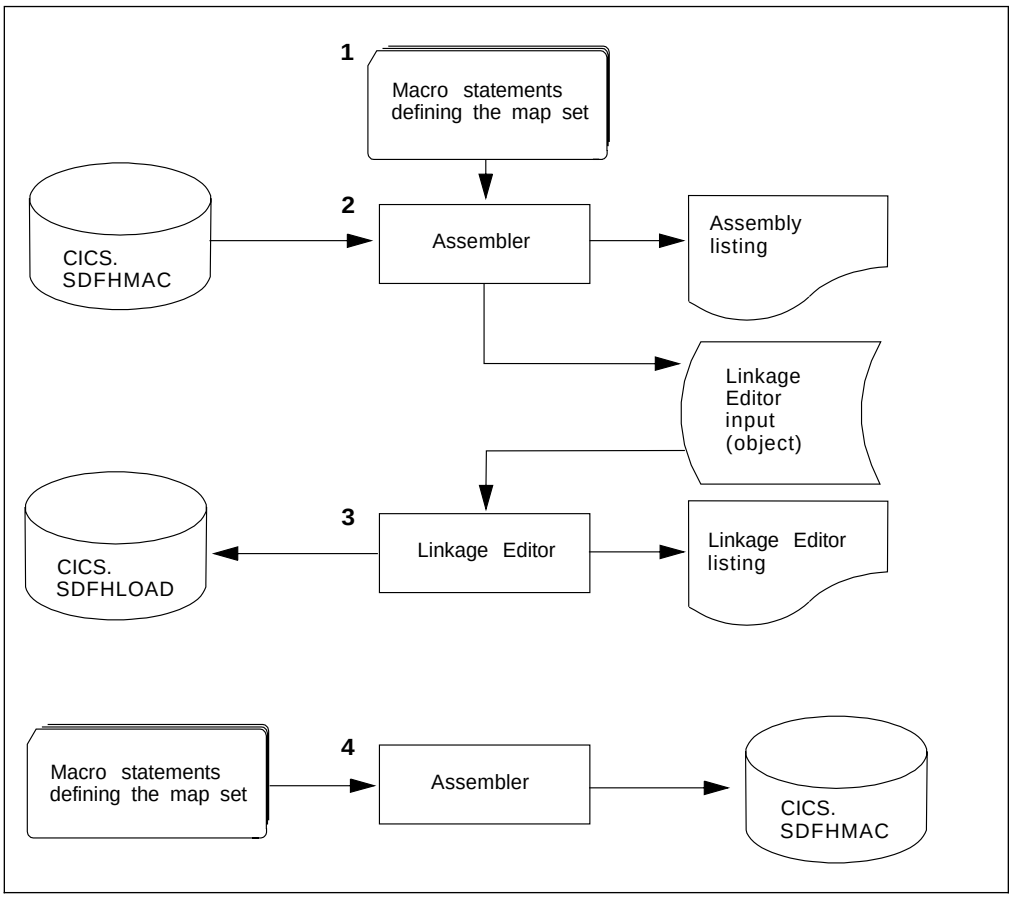


Figure 8. Installing a physical map set and a symbolic description map set together

Job stream to install both physical and symbolic description maps

The load module from the assembly of the physical map set and the source statements for the symbolic description map set can be produced in the same job by using the sample job stream in Figure 9.

```
//PREPARE  JOB  'accounting information',CLASS=A,MSGLEVEL=1
//ASSEM    EXEC PROC=DFHMAPS,MAPNAME=mapsetname,RMODE=ANY|24   (see note)
//SYSUT1   DD   *
  ⋮
      Macro statements defining the map set
  ⋮
/*
//
```

Note: The RMODE statement specifies whether the map set is to be loaded above (RMODE=ANY) or below (RMODE=24) the 16MB line.
RMODE=ANY indicates that CICS can load the map set anywhere in virtual storage, but tries to load it above the 16MB line, if possible.

Figure 9. Installing physical and symbolic description maps together

The DFHMAPS procedure produces map sets that are not halfword-aligned. If you want the length fields in input maps to be halfword-aligned, you have to code A=A on the EXEC statement. In the sample job in Figure 9 on page 21, change the EXEC statement to:

```
//ASSEM  EXEC  PROC=DFHMAPS,MAPNAME=mapsetname,A=A
```

This change results in the SYSPARM operands in the assembly steps being altered to SYSPARM(AMAP) and SYSPARM(ADSECT) respectively.

The DFHMAPS procedure directs the symbolic description map set output (SYSPUNCH) to the CICS410.SDFHMAC library. Override this by specifying DSCTLIB=name on the EXEC statement, where “name” is the name of the chosen user copy library.

Installing partition sets

Partition sets are installed in the same way as physical map sets (as illustrated in Figure 4 on page 17). There is no concept of a symbolic description partition set.

The job stream in Figure 10 is an example of the assembly and link-edit of partition sets.

```
//PREP     JOB  'accounting information',CLASS=A,MSGLEVEL=1
//STEP1    EXEC PROC=DFHASMVS
//SYSPUNCH DD   DSN=&&TEMP,DCB=(RECFM=FB,BLKSIZE=2960),
//         SPACE=(2960,(10,10)),UNIT=SYSDA,DISP=(NEW,PASS)
//SYSIN    DD   *
  ⋮
      Macro statements defining the partition set
  ⋮
/*
//STEP2    EXEC PROC=DFHLNKVS,PARM='LIST,LET,XREF'                 1
//SYSLIN   DD   DSN=&&TEMP,DISP=(OLD,DELETE)
//         DD   *
           MODE RMODE(ANY|24)                                      2
           NAME partitionsetname(R)                                3
/*
//
```

Figure 10. Assembling and link-editing a partition set

Notes:

1 A partition set is loaded into CICS-key storage, unless it is link-edited with the RMODE(ANY) and RENT options. If it is link-edited with these options, it is loaded into key-0 protected storage, provided that RENTPGM=PROTECT is specified on the RENTPGM initialization parameter.

For more information about the storage protection facilities available in CICS/ESA 4.1, see “Storage protection” on page 358.

2 The MODE statement specifies whether the partition set is to be loaded above (RMODE(ANY)) or below (RMODE(24)) the 16MB line. RMODE(ANY) indicates that CICS can load the partition set anywhere in virtual storage, but tries to load it above the 16MB line, if possible.

3 Use the NAME statement to specify the name of the partition set which BMS loads into storage. If the partition set is device-dependent, derive the partition set name by appending the device suffix to the original 1- to 7-character partition set name used in the application program. The suffixes that BMS appends for the various terminals depend on the parameter specified in the SUFFIX operand of the DFHPSD macro that defined the partition set.

For programming information giving a complete list of partition-set suffixes, see the *CICS/ESA Application Programming Reference* manual.

To use a partition set, you must define and install a resource definition for it. You can do this either by using the program autoinstall function or by using the CEDA DEFINE PARTITIONSET and INSTALL commands, as described in the *CICS/ESA Resource Definition Guide*.

Chapter 4. Installing application programs

This chapter describes what you have to do to install an application program to run under CICS. It provides the following information:

- "CICS-supplied facilities for installing programs" on page 26 describes the CICS-supplied procedures, translators, and interface modules that you can use to install application programs.
- "Adding CICS support for programming languages" on page 28 describes what you must do to add support for the programming languages that can be used with CICS. You should normally complete the actions in this section before installing your application programs.
- "Preparing to install application programs" on page 36 describes points that you should consider when preparing to install application programs and outlines the steps that you must complete to install an application program to run under CICS.
- "Using the CICS-supplied procedures to install application programs" on page 45 describes how to use the CICS-supplied procedures to install application programs.
- "Using your own job streams" on page 60 describes important points that you should consider if you intend using your own JCL to install application programs.

In this chapter, **application program** generally means any user program that uses the CICS command-level application programming interface (API). Such programs can also use:

- SQL statements
- DLI requests
- Common programming interface (CPI) statements
- SAA Resource Recovery statements
- CICS/ESA front end programming interface (FEPI) commands
- External CICS interface commands.

For information about writing CICS application programs, see the *CICS/ESA Application Programming Guide*.

Note: If you are developing application programs to use the CICS dynamic transaction routing facility, you are recommended to use the IBM CICS Transaction Affinities Utility MVS/ESA to detect whether the programs are likely to cause inter-transaction affinity. The IBM CICS Transaction Affinities Utility MVS/ESA is provided as a separate program product that you can run on CICS/ESA 4.1 and on earlier releases of CICS, and is described in the *IBM CICS Transaction Affinities Utility MVS/ESA User's Guide*, SC33-1159.

CICS-supplied facilities for installing programs

This section describes the CICS-supplied procedures, translators, and interface modules that you can use to install application programs.

The CICS-supplied procedures

CICS supplies job control statements (JCL) for the translate, compile, and link-edit steps, in separate cataloged procedures for each programming language supported. These procedures, installed in the CICS410.SDFHPROC library, should have been copied into a procedure library following installation of CICS. There is a list and brief description of these CICS-supplied procedures in the *Program Directory*.

Each procedure has a name of the form DFHwxTyL, where the variables w, x, and y depend on the type of program (Language Environment, batch, or online) and the programming language. Using the preceding naming convention, the procedure names are given in Table 8.

Table 8. Procedures for installing application programs

Language	*Not* Language Environment			Language Environment		
	Batch	**Online**	**EXCI**	**Batch**	**Online**	**EXCI**
Assembler	DFHEBTAL	DFHEITAL	DFHEXTAL	-	-	-
C	-	DFHEITDL	DFHEXTDL	-	DFHYITDL	DFHYXTDL
C++	-	-	-	-	DFHYITEL	DFHYXTEL
COBOL	DFHEBTVL	DFHEITVL	DFHEXTVL	DFHYBTVL	DFHYITVL	DFHYXTVL
PL/I	DFHEBTPL	DFHEITPL	DFHEXTPL	DFHYBTPL	DFHYITPL	DFHYXTPL

The CICS-supplied translators

The following CICS-supplied translators are installed in the CICS410.SDFHLOAD library:

Assembler	DFHEAP1$
C	DFHEDP1$
COBOL	DFHECP1$
PL/I	DFHEPP1$

For a description of the translation process, and information about the translator options that you can specify, see the *CICS/ESA Application Programming Guide*.

Dynamic invocation of the translator

You can invoke the command-level language translator dynamically from a batch Assembler-language program using an ATTACH, CALL, LINK, or XCTL macro; or from a C, PL/I, or COBOL program using CALL. If you use ATTACH, LINK, or XCTL, use the appropriate translator load module, DFHExP1$, where x=A for Assembler language, x=C for COBOL, x=D for C, or x=P for PL/I.

If you use CALL, specify PREPROC as the entry point name to invoke the translator.

In all cases, pass the following address parameters to the translator:

- The address of the translator option list
- The address of a list of DD names used by the translator (this is optional).

These addresses must be in adjacent fullwords, aligned on a fullword boundary. Register 1 must point to the first address in the list, and the high-order bit of the last address must be set to one, to indicate the end of the list. This is the case whether one or two addresses are passed.

Translator option list: The translator option list must begin on a halfword boundary. The first two bytes contain a binary count of the number of bytes in the list (excluding the count field). The remainder of the list can contain any of the translator option keywords, separated by commas, blanks, or both.

Data definition (DD name) list: The DD name list must begin on a halfword boundary. The first two bytes contain a binary count of the number of bytes in the list (excluding the count field). Each entry in the list must occupy an 8-byte field. The sequence of entries is:

Entry	Standard DD name	Entry	Standard DD name	Entry	Standard DD name
1	not applicable	3	not applicable	5	SYSIN
2	not applicable	4	not applicable	6	SYSPRINT
				7	SYSPUNCH

If you omit an applicable entry, the translator uses the standard DD name. If you use a DD name less than 8 bytes long, fill the field with blanks on the right. You can omit an entry by placing X'FF' in the first byte. You can omit entries at the end of the list entirely.

The CICS-supplied interface modules

Each of your application programs to run under CICS must contain one or more interface modules (also known as **stubs**) to use the following CICS facilities:

- The EXEC interface
- The CPI Communications facility
- The SAA Resource Recovery facility.

The interface modules and their use is described in the following sections.

The EXEC interface modules

Each of your CICS application programs must contain an interface to CICS. This takes the form of an EXEC interface module, used by the CICS high-level programming interface. The module, installed in the CICS410.SDFHLOAD library, must be link-edited with your application program to provide communication between your code and the EXEC interface program, DFHEIP.

The following interface modules are required for the assembler, C, COBOL, and PL/I programming languages:

Language Environment-conforming languages require the DFHELII interface module; otherwise:

Table 9. Non Language Environment-conforming language and Interface module name

Language	Interface module name
Assembler	DFHEAI and DFHEA10
C	DFHELII
COBOL	DFHECI
PL/I	DFHPL1OI supplied by PL/I (and DFHEPI, which is part of the PL/I DFHPL1OI module)

The CPI Communications interface module

Each of your CICS application programs that uses the Common Programming Interface for Communications (CPI Communications) must contain an interface to CPI Communications. This takes the form of an interface module, used by the CICS high-level programming interface, common to all the programming languages. The module, DFHCPLC, installed in the CICS410.SDFHLOAD library, must be link-edited with each application program that uses CPI Communications.

The SAA Resource Recovery interface module

Each of your CICS application programs that uses SAA Resource Recovery must contain an interface to SAA Resource Recovery. This takes the form of an interface module, used by the CICS high-level programming interface, common to all the programming languages. The module, DFHCPLRR, installed in the CICS410.SDFHLOAD library, must be link-edited with each application program that uses the SAA Resource Recovery facility.

Adding CICS support for programming languages

This section describes what you must do to add support for the programming languages that can be used with CICS. You should normally complete appropriate actions described in the following sections before installing your application programs.

To write a new CICS application program, you can use Assembler language, C, COBOL, or PL/I, and you request CICS services through the command-level application programming interface (API).

CICS provides the support needed for application programs written in Assembler language.

If you intend to run OS/VS COBOL programs in a CICS environment which has VS COBOL II or Language Environment support installed, see “OS/VS COBOL and storage protection” on page 34.

PN47159

The following change was made by APAR PN47159.

+ **Note:** If you run OS/VS COBOL applications with storage protection active and
+ do not use VS COBOL II, you must include the OS/VS COBOL run-time library in
+ the DFHRPL concatenation.

Before you can use application programs written in C, you must install the C library and compiler, and generate CICS support for C. (See “Installing CICS support for C/370 and IBM C/C++ for MVS/ESA” on page 34.)

Before you can use application programs written in VS COBOL II, you must install CICS support for VS COBOL II. (See “Installing CICS support for VS COBOL II, COBOL/370 and IBM COBOL for MVS and VM” on page 32.)

| Before you can use application programs written in PL/I, you must install resource
| definitions needed for run-time support of PL/I. (See “Installing CICS run-time support for PL/I and IBM PL/I for MVS and VM.” on page 35.) Also, if you want your PL/I application programs to run with the PL/I shared library facility, you must generate the PL/I shared library modules. (See “Generating PL/I shared library support for CICS” on page 36.)

Before you can run application programs compiled by a Language Environment-conforming compiler under CICS, you must have installed Language Environment support. (See “Language Environment support.”)

Language Environment support

This section describes CICS support for Language Environment and what you must do to install that support.

Language Environment support is a run-time library that establishes a common execution environment for a number of SAA languages. To run programs compiled by a Language Environment-conforming compiler under CICS you must have installed Language Environment support.

The CICS-Language Environment interface is enabled automatically if CICS can:

+ 1. Load the Language Environment interface modules, CEECCICS and
+ CEECTCB, from STEPLIB
2. Successfully call the CEECCICS module to initialize the interface.

Language Environment initialization takes place during CICS initialization. The CEECCICS module is loaded, followed by a partition initialization call to it, before the start of second phase PLT processing. If Language Environment cannot successfully complete the initialization of all languages supported by CICS, or can only initialize some of them, it issues messages to inform the user. If Language Environment initialization fails completely, it may be because the CEECCICS module could not be loaded, or something went wrong during the loading of a particular language routine.

Installing CICS support for Language Environment

For Language Environment support to be installed correctly by CICS, you must:

- Specify enough storage for the ERDSA to run CICS and Language Environment together. They need a minimum of 3500KB. To this minimum, add an amount of storage sufficient for your own requirements.
- Ensure the CICS-Language Environment interface module, CEECCICS, and the Language Environment module, CEECTCB, are installed in one of the APF-authorized libraries defined in the STEPLIB concatenation in the CICS startup job. For example, include the Language Environment SCEERUN library in the STEPLIB concatenation, first making sure it is APF-authorized.
- Add the program resource definitions for the Language Environment language interface modules to the CICS CSD. These are supplied as DEFINE statements in the CEECCSD member of the SCEESAMP library. After you have defined the program resource definitions, add the resource group to a CICS startup group list named in the GRPLIST system initialization parameter.

  Alternatively, CICS can create and install the resource definitions dynamically using program autoinstall. For more information about installing program resource definitions, see "Defining programs, map sets, and partition sets to CICS" on page 65. For information, see the *IBM Language Environment for MVS & VM Installation and Customization on MVS* manual, SC26-4817.
- Define the Language Environment transient data destinations, CEEMSG, CEEOUT, CESE, and CESO. The CICS-supplied destination control table, DFHDCT2$, contains sample entries for CEEMSG, CEEOUT, CESE, and CESO.

  For information about the attributes needed for Language Environment transient data destinations, see the *IBM Language Environment for MVS & VM Programming Guide*, SC26-4818.
- Replace any VS COBOL II or C/370 library entries in the CICS STEPLIB and DFHRPL concatenations with entries for the Language Environment libraries as follows:
  - Include the SCEERUN library, which contains CEECCICS and CEECTCB, in the STEPLIB.
  - Include the SCEERUN and SCEECICS libraries in DFHRPL.

Language Environment support for COBOL

To run COBOL programs compiled by a Language Environment conforming compiler under CICS, you must have installed support for Language Environment.

Note: It is no longer necessary to specify the language.

CICS can create and install program resource definitions automatically or you can create them specifically in the CSD, and install them by using the GRPLIST system initialization parameter or CEDA INSTALL command. For more information about installing program resource definitions, see "Defining programs, map sets, and partition sets to CICS" on page 65.

You can run existing VS COBOL II programs (not compiled by a Language Environment-conforming compiler) as in previous releases of CICS.

If the CICS-Language Environment interface is not enabled for COBOL, the CICS-VS COBOL II interface is enabled automatically if CICS can:

1. OS load the VS COBOL II interface module, IGZECIC, from STEPLIB
2. Successfully call the IGZECIC module to initialize the VS COBOL II interface.

For information about Language Environment support for programming languages, see the *Program Directory for IBM Language Environment for MVS and VM*.

Language Environment support for C and C++.

| To run C programs compiled by a Language Environment-conforming compiler under CICS, you must have:

| • Installed support for Language Environment
| • Installed resource definitions for the programs with the LANGUAGE attribute as either of the following:
  – LANGUAGE(C)
  – LANGUAGE(LE370).

+ CICS/ESA 4.1 supports application programs written in C++ that:

+ • Are compiled using the IBM C++ for MVS/ESA Version 3 compiler (5655-121)
+ • Execute with the run-time libraries provided by the Language Environment for MVS & VM (MVS feature) (5688-198)

| Code LANGUAGE(LE370) if:

| • The program exploits Language Environment multi-language support.
| • You intend to use Language Environment support at run time. If you don't intend to use Language Environment support at run time, code LANGUAGE(C).
+ • The program is a C++ program.
# • The program has been compiled by the C/370 compiler
# • Has been compiled by the IBM C/C++ for MVS/ESA compiler.

| CICS can create and install program resource definitions automatically or you can create them specifically in the CSD, and install them by using the GRPLIST system initialization parameter or CEDA INSTALL command. For more information about installing program resource definitions, see "Defining programs, map sets, and partition sets to CICS" on page 65.

You can run existing C programs that have not been compiled with the Language-Environment conforming compiler as in previous releases of CICS.

For information about Language Environment support for programming languages, see the *Program Directory for IBM Language Environment for MVS and VM*.

PN86930

The following change was made by APAR PN86930.

+ If Version 3 Release 2 of the C/C++ compiler is used to compile a C++ program,
+ the CXX parameter must be specified when options are passed to the compiler,

+ otherwise the C compiler will be invoked. Don't specify CXX if a C program is to be
+ compiled. Refer to the *IBM C/C++ for MVS/ESA Compiler and Run-Time Migration*
+ *Guide Version 3 Release 2*, SC33-2002, for further information.

Language Environment support for PL/I

To run PL/I programs compiled by a Language Environment-conforming compiler under CICS, you must have:

+ • Installed support for Language Environment
+ • Link-edited the programs with REPLACE PLISTART unless the fix of APAR PN43983 (PTF UN51883) is applied, or Language Environment version 1.3 is used.
+ • Installed resource definitions for the programs with the LANGUAGE attribute as either of the following:
+ – LANGUAGE(PLI)
+ – LANGUAGE(LE370).

Code LANGUAGE(LE370) if:

- The program exploits Language Environment multi-language support.
- You intend to use Language Environment support at run time. If you don't intend to use Language Environment support at run time, code LANGUAGE(PLI).

• The program has been compiled by the IBM SAA AD/Cycle PL/I compiler

• Has been compiled by the IBM PL/I for MVS & VM compiler.

CICS can create and install program resource definitions automatically or you can create them specifically in the CSD, and install them by using the GRPLIST system initialization parameter or CEDA INSTALL command. For more information about installing program resource definitions, see "Defining programs, map sets, and partition sets to CICS" on page 65.

You can run existing PL/I programs that have not been compiled with the Language Environment-conforming compiler as in previous releases of CICS.

For information about Language Environment support for programming languages, see the *Program Directory for IBM Language Environment for MVS and VM*.

+ Installing CICS support for VS COBOL II, COBOL/370 and IBM COBOL + for MVS and VM

+ Language Environment is a prerequisite for IBM COBOL for MVS and VM and
+ COBOL/370. Language Environment incorporates the run-time libraries required for
+ IBM COBOL for MVS and VM and COBOL/370.

+ For information see the *IBM Language Environment for MVS & VM Installation and*
+ *Customization on MVS* manual, SC26-4817.

+ If you are not using Language Environment with your CICS system and you intend
+ to use application programs written in VS COBOL II, you must have installed CICS
| support for VS COBOL II, as follows:

1. Place the following four VS COBOL II library routines in an APF-authorized library in the STEPLIB concatenation of your CICS startup job (for example, in the CICS410.SDFHAUTH library), or in an APF-authorized library in the MVS LNKLSTnn concatenation:
   - IGZ9CIC (and its alias IGZECIC)
   - IGZ9WTO (and its alias IGZEWTO)
   - IGZ9OPD (and its alias IGZEOPD)
   - IGZCMTxx, where xx represents the first two letters of the language specified on the MVS LANGUAGE option for your site (for example, IGZCMTEN).

   After you have installed VS COBOL II, the IGZ9CIC, IGZ9WTO, and IGZ9OPD routines are located in the SYS1.COB2CICS library and the IGZCMTxx routine is located in the SYS1.COB2LIB library.

   Notes:

   a. Alternatively, the SYS1.COB2CICS library can be APF-authorized and placed before SYS1.COB2LIB library in the STEPLIB or JOBLIB.

   b. To use VS COBOL II, you must have the CICS-VS COBOL II interface module, IGZECIC, in an APF-authorized library in the CICS STEPLIB concatenation. Do not put it in the LPA, because CICS does not search the LPA for this module.

2. Include the libraries containing the VS COBOL II library routines in the DFHRPL concatenation of your CICS startup job. VS COBOL II requires two packages of subroutines, known as COBPACKs. These subroutines are in two categories: (1) general and (2) environment-specific, containing system-specific logic. The COBPACKs you need are:

   IGZCPCC This module contains the CICS environment-specific modules, and is supplied in the SYS1.COB2CICS library.

   IGZCPAC This module contains the general VS COBOL II subroutines, and is supplied in the SYS1.COB2LIB library.

   Ensure that the SYS1.COB2CICS library is in front of the SYS1.COB2LIB library in the DFHRPL concatenation.

3. Create and install resource definitions for the COBPACKs as programs, with the LANGUAGE(ASSEMBLER) attribute. CICS can create and install program resource definitions automatically or you can create them specifically in the CSD, and install them by using the GRPLIST system initialization parameter or CEDA INSTALL command. For more information about installing program resource definitions, see “Defining programs, map sets, and partition sets to CICS” on page 65.

   The IGZECIC module does ***not*** have to be defined to CICS as a program resource in the CSD. For any other VS COBOL II library routine modules not included in COBPACKs, you should create and install program resource definitions (either automatically or specifically).

If you choose to define the VS COBOL II COBPACKs automatically, remember to specify ACTIVE on the PGAIPGM system initialization parameter (the default is INACTIVE). The PGAIPGM system initialization parameter is described on page 276.

| If you choose to define the VS COBOL II COBPACKs specifically, you can use the
| following commands:

```
| DEFINE PROGRAM(IGZCPCC) GROUP(cob2grp) LANGUAGE(ASSEMBLER) CEDF(NO)
| DEFINE PROGRAM(IGZCPAC) GROUP(cob2grp) LANGUAGE(ASSEMBLER) CEDF(NO)
| ADD GROUP(cob2grp) LIST(listname)
```

| For information about installing VS COBOL II support for CICS and about running
| VS COBOL II applications with CICS, see the *VS COBOL II Installation and
| Customization* manual.

+ OS/VS COBOL and storage protection

+ If you intend to run OS/VS COBOL programs in a CICS environment which has
+ storage protection active, and has VS COBOL II or LE support installed, then all
+ re-entrant OS/VS COBOL compatibility modules (that is, those whose names begin
+ with ILB) must be included in an authorized library. This is so that they are loaded
+ into Key 0 storage to be accessible to programs running in Key 9.

+ Installing CICS support for C/370 and IBM C/C++ for MVS/ESA

+ Language Environment is a prerequisite for IBM C/C++ for MVS/ESA. Language
+ Environment incorporates the run-time libraries required for IBM C/C++ for
+ MVS/ESA.

+ For information see the *IBM Language Environment for MVS & VM Installation and
+ Customization on MVS* manual, SC26-4817.

+ If you are not using Language Environment with your CICS system and you intend
+ to install any C/370 programs, you must have:

1. Installed the C/370 library and compiler (by using SMP/E or in accordance with information given in the *IBM C/370 Installation Guide*). This creates a data set called EDC.V1R2M0.SEDCLINK, which contains the two load modules for CICS-C/370 support, EDCCICS and EDCXV. For guidance information about installing C/370, see the *IBM C/370 Installation Guide*.
2. Generated CICS support for C/370. This process, which defines the C/370 support modules and libraries to CICS, is described in "Generating CICS support for C/370."

For information about C/370, see the *IBM System Application Architecture Common Programming Interface Reference Manual* and the *IBM C/370 Programming Guide*.

Generating CICS support for C/370

To generate CICS support for C/370:

1. Copy the CICS-C/370 interface module, EDCCICS, from EDC.V1R2M0.SEDCLINK to one of the authorized libraries defined in the STEPLIB DD statement in the CICS startup job stream.

2. Create and install a resource definition for the EDCXV module with the LANGUAGE(ASSEMBLER) attribute.

   CICS can create and install program resource definitions automatically or you can create them specifically in the CSD, and install them by using the GRPLIST system initialization parameter or CEDA INSTALL command. For more information about installing program resource definitions, see "Defining programs, map sets, and partition sets to CICS" on page 65.

   If you choose to define the EDCXV module specifically, you could use the following commands to define the module and add the group in which it is defined to the list, INSTLIST, to be installed at CICS initialization:

   ```
   DEFINE PROGRAM(EDCXV) GROUP(C370) LANGUAGE(ASSEMBLER)
   ADD GROUP(C370) LIST(INSTLIST)
   ```

3. Define the C/370 run-time library, EDC.V1R2M0.SEDCLINK, in the DFHRPL statement in the CICS startup job stream.

4. You can create and install resource definitions for the C/370-provided locales[4] in the same way as for EDCXV. These locales are EDC$GERM, EDC$USA, EDC$FRAN, EDC$ITAL, and EDC$SPAI. These are all load modules in the EDC.V1R2M0.SEDCLINK data set.

Installing CICS run-time support for PL/I and IBM PL/I for MVS and VM.

Language Environment is a prerequisite for IBM PL/I for MVS and VM. Language Environment incorporates the run-time libraries required for IBM PL/I for MVS and VM.

For information see the *IBM Language Environment for MVS & VM Installation and Customization on MVS* manual, SC26-4817.

CICS provides run-time support for PL/I version 2.3. Programs compiled against earlier releases of PL/I run with the PL/I version 2.3 run-time support.

For CICS support for PL/I, you must create and install some resource definitions, either automatically or specifically. For information about the definitions required, see the *OS PL/I Version 2 Installation and Customization under MVS Guide*.

CICS can create and install program resource definitions automatically or you can create them specifically in the CSD, and install them by using the GRPLIST system initialization parameter or CEDA INSTALL command. For more information about installing program resource definitions, see "Defining programs, map sets, and partition sets to CICS" on page 65.

The IBM-supplied group of CSD definitions, DFHPLI, is not supplied in CICS/ESA 4.1 and is not added to a compatibility group. For an explanation about compatibility groups in general, see "Sharing the CSD between CICS/ESA 4.1 and earlier releases" and "CICS supplied compatibility groups" on page 149. (The *CICS/ESA Resource Definition Guide* lists the contents of each compatibility group.)

[4] Locale is the term defined by the American National Standard for Information Systems (ANSI) to denote a C/370 programming language environment for a given national language. The C/370-supplied locales provide a C/370 programming language environment for German (EDC$GERM), American English (EDC$USA), French (EDC$FRAN), Italian (EDC$ITAL) and Spanish (EDC$SPAI).

Generating PL/I shared library support for CICS

If you want your PL/I application programs to run with the PL/I shared library facility, ensure that you generate the PL/I shared library modules. To do this run the stage 1 and stage 2 generation jobs described in the *OS PL/I Version 2 Installation and Customization under MVS Guide*. The stage 1 job assembles the PL/I macro, PLRSHR, supplied in the SYS1.SHRMAC library. The assembly of the PLRSHR macro generates the stage 2 jobs to assemble and link-edit the following modules:

- PLISHRE
- IBMBPSLA
- IBMBPSMA
- IBMBPSRA
- IBMTPSLA
- IBMTPSRA.

When you have completed the stage 2 jobs, which link-edit the PL/I shared library modules into the SYS1.PLIBASE library (or another suitable library), ensure that modules IBMBPSLA and IBMBPSMA are also installed in one of the libraries in the CICS DFHRPL library concatenation (for example, the CICS410.SDFHLOAD
| library) or in the LPA.

When you start up CICS, it attempts to load the modules IBMBPSLA and IBMBPSMA into the CICS nucleus. If this load fails (for example, because the modules are not found), PL/I shared library support is not available.

You must also run the job shown in Figure 11, to link-edit module PLISHRE into a CICS DFHRPL library, for example CICS410.SDFHLOAD.

```
//PLISHRE  JOB  'accounting information',CLASS=A,MSGCLASS=A
//LNKEDIT  EXEC DFHLNKVS,NAME=SDFHLOAD,INDEX=CICS410,
//              INDEX2=CICS410
//SYSPUNCH DD DUMMY
//SYSLIB   DD DSN=SYS1.PLIBASE,DISP=SHR
#           DD DSN=SYS1.SIBMBASE,DISP=SHR
//SYSLIN   DD *
           REPLACE IBMBPIR1
           INCLUDE SYSLIB(PLISHRE)
           NAME    PLISHRE(R)
/*
//
```

Figure 11. Job to link-edit PLISHRE

Preparing to install application programs

This section describes the following points that you should consider about application programs to be installed and the libraries that they are to be installed into:

- Using BMS map sets in programs
- MVS residence and addressing mode
- Program eligibility for the MVS link pack area (LPA)

- Installing programs in load library secondary extents.

This section also outlines the steps that you must complete to install application programs to run under CICS. (See "Overview of installing application programs" on page 44.)

Using BMS map sets in application programs

This section describes what you must do to use BMS map sets in application programs.

Before you install an application program to run under CICS, you must:

- Create any BMS map sets used by the program, as described in Chapter 3, "Installing map sets and partition sets" on page 15.
- Include the physical map sets (used by BMS in its formatting activities) in a library that is in the DFHRPL concatenation.
- Either include the symbolic map sets (copied into the application programs) in a user copy library or insert them directly into the application program source.

The DFHMAPS procedure writes the symbolic map set output to the library specified on the DSCTLIB parameter, which defaults to the CICS410.SDFHMAC library. If you want to include symbolic map sets in a user copy library:

- Specify the library name by the *DSCTLIB=name* operand on the EXEC statement for the DFHMAPS procedure used to install physical and symbolic map sets together.
- Include a DD statement for the user copy library in the SYSLIB concatenation of the job stream used to assemble and compile the application program.

  If you choose to let the DFHMAPS procedure write the symbolic map sets to the CICS410.SDFHMAC library (the default), include a DD statement for the CICS410.SDFHMAC library in the SYSLIB concatenation of the job stream used to compile the application program. This is not necessary for the DFHEITAL or DFHEBTAL procedures used to assemble Assembler-language programs, because these jobs already include a DD statement for the CICS410.SDFHMAC library in the SYSLIB concatenation.
- For PL/I, specify a library that has a block size of 400 bytes. This is necessary to overcome the blocksize restriction on the PL/I compiler.

For more information about installing map sets, see Chapter 3, "Installing map sets and partition sets" on page 15. For information about writing programs to use BMS services, see the *CICS/ESA Application Programming Guide*.

MVS residence and addressing modes

This section describes the effect of the MVS residence and addressing modes on application programs, how you can change the modes, and how you can make application programs permanently resident. An application written to run on MVS/370 can run on an MVS/ESA system, if it is link-edited with the AMODE(24) and RMODE(24) options.

A command-level program written in Assembler language, C, VS COBOL II or later, or PL/I Version 1 Release 5.1 or later, can reside above the 16MB line, and address areas above that line. The program can contain both CICS and EXEC DLI

commands. However, if you are running with the CICS local DL/I interface, and a command-level program contains CALL DLI statements, the ***program*** can reside above the 16MB line, but the call ***parameter list*** and the call ***parameters*** must reside below the line. If you are using remote DL/I or DBCTL, and a command-level program contains CALL DLI statements, the program, the call parameter list, and the call parameters can reside above the 16MB line.

For information about considerations and restrictions that apply to such programs, see the *CICS/ESA Application Programming Guide* manual.

Establishing a program's addressing mode

Every program that executes in MVS/ESA is assigned two attributes: an addressing mode (AMODE), and a residency mode (RMODE). AMODE specifies the addressing mode in which your program is designed to receive control. Generally, your program is designed to execute in that mode, although you can switch modes in the program, and have different AMODE attributes for different entry points within a load module. The RMODE attribute indicates where in virtual storage your program can reside. Valid AMODE and RMODE specifications are:

AMODE=24	Specifies 24-bit addressing mode
AMODE=31	Specifies 31-bit addressing mode
AMODE=ANY	Specifies either 24- or 31-bit addressing mode
RMODE=24	Indicates that the module must reside in virtual storage below 16MB. You can specify RMODE=24 for 31-bit programs that have 24-bit dependencies
RMODE=ANY	Indicates that the module can reside anywhere in virtual storage.

Note: C language programs must be link-edited with AMODE=31.

If you do not specify any AMODE or RMODE attributes for your program, MVS assigns the system defaults AMODE=24 and RMODE=24. To override these defaults, you can specify AMODE and RMODE in one or more of the following places. Assignments in this list overwrite assignments later in the list.

1. On the linkage editor MODE control statement:

   ```
   MODE  AMODE(31),RMODE(ANY)
   ```

2. Either of the following:

   a. In the PARM string on the EXEC statement of the linkage editor job step:

      ```
      //LKED    EXEC  PGM=IEWL,PARM='AMODE=31,RMODE=ANY,..'
      ```

   b. On the LINK TSO command, which causes processing equivalent to that of the EXEC statement in the linkage editor step.

3. On AMODE or RMODE statements within the source code of an assembler program. (You can also set these modes in VS COBOL II by means of the compiler options; for guidance information about VS COBOL II compiler options, see the *VS COBOL II Application Programming Guide*.)

4. The link-edit modules DFHECI and DFHEPI assign AMODE=31 and RMODE=ANY to COBOL and PL/I programs.

For information about these modes and the rules that govern their use, see the *MVS/ESA Conversion Notebook Volume 1 for SP Version 3*.

CICS address space considerations

Table 10 gives the valid combinations of the AMODE and RMODE attributes and their effects:

Table 10. Valid AMODE and RMODE specifications and their effects

AMODE	RMODE	Residence	Addressing
24	24	Below 16MB line	24-bit mode
31	24	Below 16MB line	31-bit mode
ANY	24	Below 16MB line	31-bit mode
31	ANY	Above 16MB line	31-bit mode

The following example shows linkage editor control statements for a program coded to 31-bit standards:

```
//LKED.SYSIN DD *
   MODE AMODE(31),RMODE(ANY)
   NAME    anyname(R)    ("anyname" is your load module name)
/*
//
```

Making programs permanently resident

If you define a program in the CSD with the resident attribute, RESIDENT(YES), it is loaded on first reference. This applies to programs link-edited with either RMODE(ANY) or RMODE(24). However, be aware that the storage compression algorithm that CICS uses does not remove resident programs.

If there is not enough storage for a task to load a program, the task is suspended until enough storage becomes available. If any of the DSAs get close to being short on storage, CICS frees the storage occupied by programs that are not in use. (For more information about the dynamic storage areas in CICS/ESA 4.1, see “Storage protection” on page 358.)

Instead of making RMODE(24) programs resident, you can make them non-resident and use the library lookaside (LLA) function. The space occupied by such a program is freed when its usage count reaches zero, making more virtual storage available. LLA keeps its library directory in storage and stages (places) copies of LLA-managed library modules in a data space managed by the virtual lookaside facility (VLF). CICS locates a program module from LLA’s directory in storage, rather than searching program directories on DASD. When CICS requests a staged module, LLA gets it from storage without any I/O.

Preparing applications to run in the link pack area

Programs written in Assembler language, C, VS COBOL II or later, or PL/I Release 5.1, can reside in the link pack area (LPA). To do so, they must be read-only and have been link-edited with the RENT and REFR options. Other requirements are as follows:

Assembler
Use the RENT assembler option.

C Use the RENT compiler option.

COBOL
Do not overwrite WORKING STORAGE. (The CICS translator generates a CBL statement with the required compiler options, RENT and RES, unless you specify the translator option NOCBLCARD.)

PL/I Version 1 Release 5.1 or later
Do not overwrite STATIC storage. (The CICS translator inserts the required REENTRANT option into the PROCEDURE statement.)

If you want CICS to use modules that you have written to these standards, and installed in the LPA, you must specify USELPACOPY(YES) on the program resource definitions in the CSD.

For information about installing CICS modules in the LPA see the *CICS/ESA Installation Guide*.

Preparing application programs to run in the RDSAs

Programs that are eligible to reside above the 16MB boundary, and are read-only, can reside in the CICS extended read-only DSA (ERDSA). Therefore, to be eligible for the ERDSA, programs must be:

- Properly written to read-only standards
- Written to 31-bit addressing standards
- Link-edited with the RENT attribute and the RMODE(ANY) residency attribute.

Note: OS/VS COBOL or pre-CICS/VS 1.6 (DFHE type program stub) programs are not re-entrant and therefore cannot be loaded into read-only storage.

If the default RENTPGM=PROTECT is specified as a system initialization parameter, these programs should be link-edited with the NORENT option and not RENT or REFR.

If RENTPGM=NOPROTECT is specified as a system initialization parameter, they may be link-edited using the RENT option. However, in this case they will be loaded into the CDSA or ECDSA.

Programs that are ***not*** eligible to reside above the 16MB boundary, and are read-only, can reside in the CICS read-only DSA (RDSA) below the 16MB boundary. Therefore, to be eligible for the RDSA, programs must be:

- Properly written to read-only standards
- Link-edited with the RENT attribute.

Note: When you are running CICS with RENTPGM=PROTECT specified as a system initialization parameter, the RDSAs are allocated from key-0 read-only storage.

Programs link-edited with RENT and RMODE(ANY) are automatically loaded by CICS into the ERDSA.

ERDSA requirements for the specific languages are described in the following sections.

Assembler

For Assembler programs that you want CICS to install in the ERDSA, you should assemble and link-edit them with the following options:

1. The RENT assembler option
2. The linkage-editor RENT attribute
3. The RMODE(ANY) residency mode.

Note: If you specify these options, you must ensure that the program is truly read-only (that is, does not modify itself in any way—for example, by writing to static storage), otherwise storage exceptions will occur. The program must also be written to 31-bit addressing standards. See the *CICS/ESA Problem Determination Guide* for some possible causes of storage protection exceptions in programs resident in the ERDSA.

The CICS-supplied procedure, DFHEITAL, has a LNKPARM parameter that specifies the XREF and LIST options only. To link-edit an ERDSA-eligible program, override LNKPARM from the calling job, specifying the RENT and RMODE=ANY options in addition to any others you require. For example:

```
//ASMPROG  JOB 1,user_name,MSGCLASS=A,CLASS=A,NOTIFY=userid
//EITAL    EXEC DFHEITAL,
           .
           (other parameters as necessary)
           .
//         LNKPARM='LIST,XREF,RMODE=ANY,RENT'
```

Apar PQ20857

Documentation for Apar PQ20857 added 23/04/99

Note: The CICS EXEC interface module for assembler programs (DFHEAI) specifies AMODE(ANY) and RMODE(ANY). However, because the assembler defaults your application to AMODE(24) and RMODE(24), the resulting load module will also be AMODE(24) and RMODE(24).

If you want your application to be link-edited as AMODE(31) and RMODE(ANY), you are recommended to use appropriate statements in your assembler program. For example

```
MYPROG    CSECT
MYPROG    AMODE 31
MYPROG    RMODE ANY
```

There are two ways of setting AMODE and RMODE.

You can set the required AMODE and RMODE specification using linkage editor (or binder) control information on the JCL PARM keyword. For example;

```
//EITAL  EXEC DFHEITAL,
         LNKPARM='LIST,XREF,RENT,AMODE(31),RMODE(ANY)'
```

Alternatively, you can use the MODE control statement in the SYSIN dataset in the linkage editor or binder step in your JCL.

When using the binder, you may see unexpected warning messages about conflicting AMODE and RMODE specifications.

C

For C programs that you want CICS to load into the ERDSA, you should assemble and link-edit them with the following options:

1. The RENT compiler option.
2. The RMODE(ANY) residency mode.

The CICS-supplied procedures, DFHEITDL and DFHYITDL, have a LNKPARM parameter that specifies a number of link-edit options. To link-edit an ERDSA-eligible program, override this parameter from the calling job, and add RENT to the other options you require. You do not need to add the RMODE=ANY option, because the CICS EXEC interface module for C (DFHELII) is link-edited with AMODE(31) and RMODE(ANY). Therefore, your program is link-edited as AMODE(31) and RMODE(ANY) automatically when you include the CICS EXEC interface stub, see Table 9 on page 28.

The following sample job statements show the LNKPARM parameter with the RENT option added:

```
//C370PROG   JOB 1,user_name,MSGCLASS=A,CLASS=A,NOTIFY=userid
//EITDL    EXEC DFHEITDL,
           .
           (other parameters as necessary)
           .
//         LNKPARM='LIST,MAP,LET,XREF,RENT'
```

COBOL

CICS VS COBOL II and later programs are automatically eligible for the ERDSA, because:

- If you use the translator option, CBLCARD (the default), the required compiler option, RENT, is included automatically on the CBL statement generated by the CICS translator. If you use the translator option, NOCBLCARD, you must specify the RENT option by using the VS COBOL II macro IGYCOPT.
- The VS COBOL II compiler automatically generates code that conforms to read-only and 31-bit addressing standards.
- The CICS EXEC interface module for COBOL (DFHECI) is link-edited with AMODE(31) and RMODE(ANY). Therefore, your program is link-edited as AMODE(31) and RMODE(ANY) automatically when you include the CICS EXEC interface stub, see Table 9 on page 28.

However, you must specify the reentrant attribute to the linkage-editor. The CICS-supplied procedure, DFHEITVL, has a LNKPARM parameter that specifies a number of link-edit options. To link-edit an ERDSA-eligible program, override this parameter from the calling job, and add RENT to any other options you require. For example:

```
//COB2PROG JOB 1,user_name,MSGCLASS=A,CLASS=A,NOTIFY=userid
//EITVL    EXEC DFHEITVL,
           .
           (other parameters as necessary)
           .
//         LNKPARM='LIST,XREF,RENT'
```

PL/I

CICS PL/I programs are generally eligible for the ERDSA, provided they do not modify static storage. The following requirements are enforced, either by CICS or PL/I:

- The required REENTRANT option is included automatically, by the CICS translator, on the PL/I PROCEDURE statement.
- The PL/I compiler automatically generates code that conforms to 31-bit addressing standards.
+ - The CICS EXEC interface module for PL/I (DFHEPI, which is part of the PL/I
+ DFHPL1OI module) is link-edited with AMODE(31) and RMODE(ANY). Therefore, your program is link-edited as AMODE(31) and RMODE(ANY)
# automatically when you include the CICS EXEC interface stub, see Table 9 on page 28.

However, you must specify the reentrant attribute to the linkage-editor. The CICS-supplied procedure, DFHEITPL, has a LNKPARM parameter that specifies a number of link-edit options. To link-edit an ERDSA-eligible program, override this parameter from the calling job, and add RENT to any other options you require. For example:

```
//PLIPROG  JOB 1,user_name,MSGCLASS=A,CLASS=A,NOTIFY=userid
//EITPL    EXEC DFHEITPL,
           .
           (other parameters as necessary)
           .
//         LNKPARM='LIST,XREF,RENT'
```

Note: You must not specify the RENT attribute on the link-edit step unless you have ensured the program is truly read-only (and does not, for example, write to static storage), otherwise storage exceptions will occur. See the *CICS/ESA Problem Determination Guide* for some possible causes of storage protection exceptions in programs resident in the ERDSA.

Installing programs in load library secondary extents

CICS supports load library secondary extents that are created while CICS is executing. If you define libraries in the DFHRPL concatenation with primary and secondary extents, and secondary extents are added while CICS is running, as a result of link-editing into the DFHRPL library, the CICS loader detects the occurrence, closes, and then reopens the library. This means that you can introduce new versions using the CEMT NEWCOPY command, even if the new copy of the program has caused a new library extent.

However, this can increase the search time when loading modules from the secondary extents. You should avoid using secondary extents if possible.

Overview of installing application programs

This section outlines the steps that you must complete to install application programs to run under CICS. For detailed information about using the CICS-supplied procedures to install application programs, see “Using the CICS-supplied procedures to install application programs” on page 45. If you want to use your own JCL to install application programs, see “Using your own job streams” on page 60.

1. Translate the program source code, turning CICS commands into code understood by the compiler.

   Notes:

   a. For a program that does not use CICS commands and is only invoked by a running transaction (and never directly by CICS task initiation), no translator step is needed.

   b. CICS command-level programs that access DL/I services through either the DL/I CALL ***or*** EXEC DLI interfaces must also be translated. Applications that access DATABASE 2 (DB2) services using the EXEC SQL interface need an additional precompilation step. For information about this step, see the *IBM DATABASE 2 Application Programming and SQL Guide*.

2. Compile or assemble the translator output to produce object code.
3. Link-edit the object module to produce a load module, which you store in an application load library that is concatenated to the DFHRPL DD statement of the CICS startup job stream. Additional INCLUDE statements are required for applications that access DB2 services using the EXEC SQL interface. For information about these extra statements, see the *IBM DATABASE 2 Application Programming and SQL Guide*.
4. Create resource definition entries, in the CSD, for any transactions that invokes the program.
5. Do one of the following:
   - If you are using program autoinstall, ensure that the autoinstall user-replaceable module can correctly install a resource definition for the program.
   - If you are not using program autoinstall, create a resource definition entry in the CSD for the program.

If you have macro-level programs from an earlier release of CICS, recode them as command-level programs. Furthermore, references to the CSA or to the TCA are not allowed. You can specify YES for the system initialization parameter DISMACP to cause CICS to disable any transaction whose program invokes a CICS macro or references the CSA or the TCA.

CICS provides a utility program, DFHMSCAN, to identify the macro-level programs used by your CICS applications. For information about using the DFHMSCAN utility to identify macro-level programs, see the *CICS/ESA Operations and Utilities Guide*.

Using the CICS-supplied procedures to install application programs

The following sections describe the use of CICS-supplied procedures to install application programs to run under CICS.

Common considerations when installing application programs

This section outlines points to consider when installing application programs as described in the following sections.

- All the sample job streams in the following sections apply if your program source statements are imbedded in the text. If your program source is a member of a partitioned data set, replace the //TRN.SYSIN statement by a statement in one of the following forms:

  //TRN.SYSIN DD DSN=partition.dataset.name(programname),DISP=SHR
  //SYSIN DD DSN=partition.dataset.name(programname),DISP=SHR

- If you want your application program to use CPI Communications or SAA Resource Recovery you must make the appropriate interface modules available to your program. For information about the CPI Communications interface module and the SAA Resource Recovery interface module, see page 28.

- If you want your application program to reside in the MVS link pack area (LPA), you must specify appropriate options when installing your program. Options appropriate to each language are given for the sample job streams in the following sections. For information on preparing programs to run in the link pack area (LPA), see page 39.

  For information on preparing programs to run in the read-only DSAs, see page 40.

- If you want your application program to use BMS maps, you must first prepare the map sets. For more information, see “Using BMS map sets in application programs” on page 37.

Using the CICS-supplied interface modules

The CICS-supplied procedures to install your online application programs in a CICS library specify the CICS library member that contains the INCLUDE statement for the appropriate language EXEC interface module. These library members are named DFHEILIx, where x is A for assembler, C for COBOL, D for C, or P for PL/I.

If your application program is to use CPI Communications or the SAA Resource Recovery facility, you should do either of the following:

- Add appropriate INCLUDE statements to the LKED.SYSIN override in the job used to call the CICS-supplied procedure to install your application program. Add the following INCLUDE statements:
  - INCLUDE SYSLIB(DFHCPLC) if your program uses CPI Communications
  - INCLUDE SYSLIB(DFHCPLRR) if your program uses SAA Resource Recovery
- Rely on the linkage editor automatic library-call mechanism to include the necessary modules.

For example, the DFHEITVL procedure uses the following statements:

```
//COPYLINK EXEC PGM=IEBGENER,COND=(7,LT,COB)
//SYSUT1   DD DSN=&INDEX..&LIB(&STUB),DISP=SHR
//SYSUT2   DD DSN=&&COPYLINK,DISP=(NEW,PASS),
//            DCB=(LRECL=80,BLKSIZE=400,RECFM=FB),
//            UNIT=&WORK,SPACE=(400,(20,20))
//SYSPRINT DD SYSOUT=&OUTC
//SYSIN    DD DUMMY
  ⋮
//SYSLIN   DD DSN=&&COPYLINK,DISP=(OLD,DELETE)
//         DD DSN=&&LOADSET,DISP=(OLD,DELETE)
//         DD DDNAME=SYSIN
```

In this VS COBOL II example, the symbolic parameters STUB and LIB default to DFHEILIC and SDFHCOB. The DFHEILIC member contains the statement INCLUDE SYSLIB(DFHECI).

When using the CICS-supplied procedures to install OS/VS COBOL or VS COBOL II application programs, you may need to change the EXEC interface module being used by the procedure so that it matches the module specified on any linkedit ORDER statement relating to the EXEC interface module. The linkedit procedures can be used as supplied if you have not specified an ORDER statement for the module, or if you have specified ORDER DFHECI. If you have specified ORDER DFHELII, you should do one of the following:

- Modify the procedure you are using (DFHEITCL or DFHEITVL) so the symbolic parameters STUB and LIB default to DFHEILID and SDFHC370, instead of DFHEILIC and SDFHCOB.
- Add the symbolic parameter overrides STUB=DFHEILID and LIB=SDFHC370 to the EXEC statement in the JCL you are using to call the linkedit procedure.

The following examples illustrate the use of the symbolic parameters STUB and LIB as overrides in combination with the linkedit ORDER statement. The procedure called in the examples is DFHEITVL, but they are equally applicable to the OS/VS COBOL linkedit procedure DFHEITCL.

Example 1:

- No linkedit ORDER statement specified.
- No symbolic parameter overrides required.
- The EXEC interface module used defaults to DFHECI.

```
//jobname    JOB  accounting info,name,MSGLEVEL=1
//           EXEC PROC=DFHEITVL
//TRN.SYSIN    DD      *
 CBL   XOPTS(Translator options . . .)
             COBOL source statements
/*
//LKED.SYSIN DD      *
               NAME    anyname(R)
/*
//
```

Example 2:

- Linkedit ORDER statement specified for DFHECI.
- No symbolic parameter overrides required.

+ • The EXEC interface module used defaults to DFHECI.

```
+ //jobname    JOB  accounting info,name,MSGLEVEL=1
+ //           EXEC PROC=DFHEITVL
+ //TRN.SYSIN    DD      *
+  CBL   XOPTS(Translator options . . .)
+              COBOL source statements
+ /*
+ //LKED.SYSIN DD      *
+               ORDER DFHECI
+               NAME    anyname(R)
+ /*
+ //
```

+ **Example 3:**

+ • Linkedit ORDER statement specified for DFHELII.
+ • Overrides required for symbolic parameters STUB and LIB.
+ • The EXEC interface module used is DFHELII.

```
+ //jobname    JOB  accounting info,name,MSGLEVEL=1
+ //           EXEC PROC=DFHEITVL,STUB=DFHEILID,LIB=SDFHC370
+ //TRN.SYSIN    DD      *
+  CBL   XOPTS(Translator options . . .)
+              COBOL source statements
+ /*
+ //LKED.SYSIN DD      *
+               ORDER DFHELII
+               NAME    anyname(R)
+ /*
+ //
```

+ **Note:** If you are using a Lanuage Environment-conforming compiler, you should
+ ensure your linkedit procedure uses the EXEC interface module DFHELII.

If you want your COBOL program to use CPI Communications or SAA Resource Recovery, you can use either of the following LKED.SYSIN overrides in the sample job control statements given in Figure 15 on page 52:

- Add INCLUDE statements for the modules DFHCPLC (CPI Communications interface), and DFHCPLRR (SAA Resource Recovery interface).

  ```
  //LKED.SYSIN DD *
               INCLUDE SYSLIB(DFHCPLC)
               INCLUDE SYSLIB(DFHCPLRR)
               NAME anyname(R)
  /*
  ```

- Rely on the linkage editor automatic library-call mechanism to include the necessary modules.

  ```
  //LKED.SYSIN DD *
               NAME anyname(R)
  /*
  ```

In the DFHEITPL procedure for installing PL/I application programs, the SYS1.PL1BASE library is included in the SYSLIB concatenation. This is needed because the INCLUDE statement in the DFHEILIP module specifies INCLUDE SYSLIB(DFHPL1OI), where DFHPL1OI is the name of the PL/I interface module supplied by PL/I. For more information about the DFHPL1OI module, see the notes to Figure 19 on page 59.

In the DFHEILIP module, after the INCLUDE statement, there is the REPLACE PLISTART linkage editor command. This command causes the CSECT PLISTART, which is inserted by the compiler, to be removed because equivalent function is in the stub DFHPL1OI. The REPLACE PLISTART command is needed for programs to run under Language Environment/370 Version 1 Release 2, but is optional for other PL/I programs that run under CICS.

For more information about linkage editor requirements, see “Using your own job streams” on page 60.

Installing Assembler language application programs

You can use the DFHEITAL procedure (see Figure 12) to translate, assemble, and link-edit (to the CICS410.SDFHLOAD library) application programs written in Assembler language.

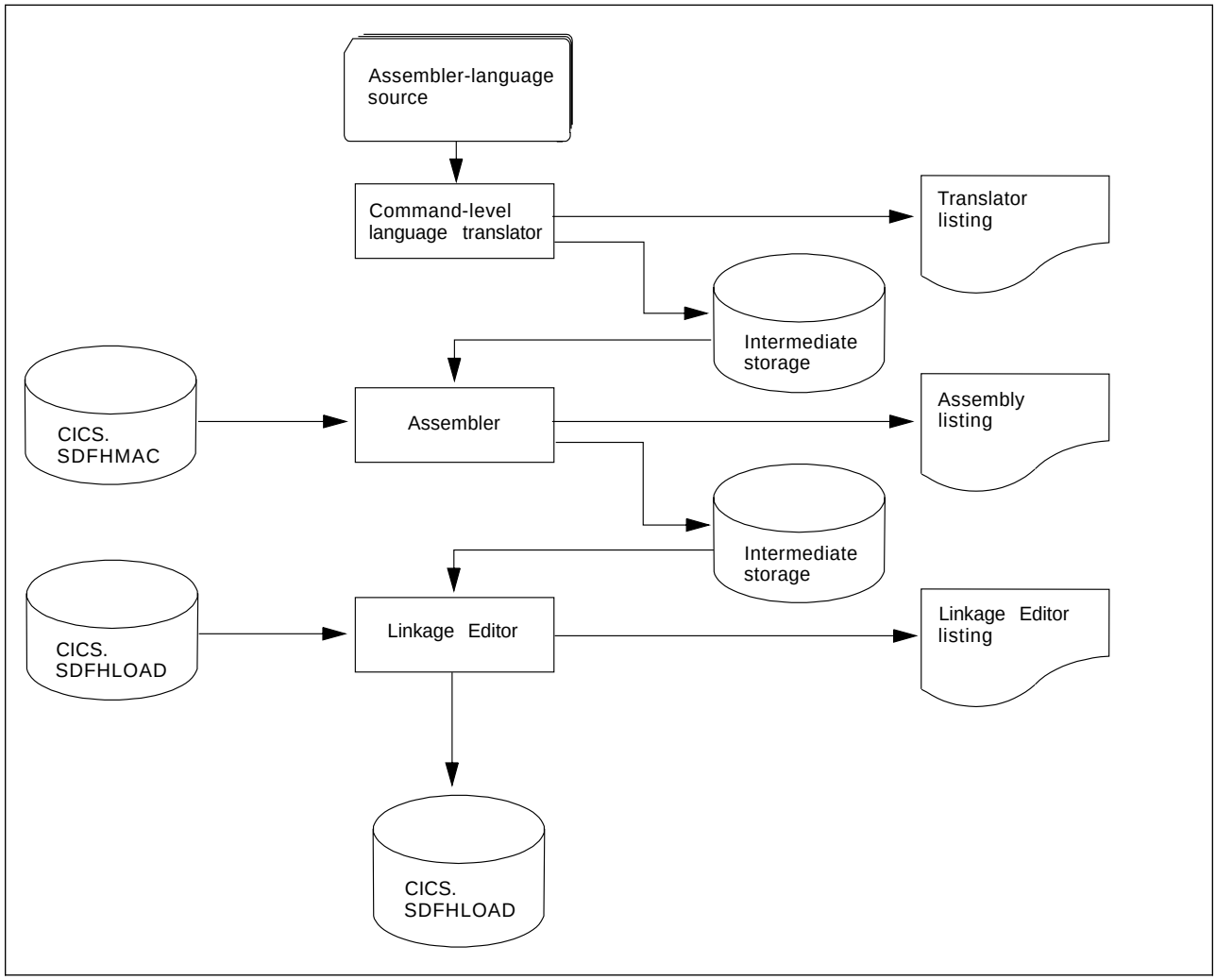


Figure 12. Installing Assembler language programs using the DFHEITAL procedure

Sample JCL to install Assembler-language application programs

You can use the sample job control statements shown in Figure 13 on page 50 to process application programs written in Assembler language. In the procedure name, “wx” depends on whether your programs are to run under Language Environment, use batch EXEC DLI calls, or otherwise. For the names of the CICS-supplied procedures, see Table 8 on page 26.

```
//jobname      JOB     accounting info,name,MSGLEVEL=1
//             EXEC    PROC=DFHwxTAL                          1
//TRN.SYSIN    DD      *                                      2
*ASM      XOPTS(translator options . . .)                     3
               .
               Assembler-language source statements
               .
/*
//LKED.SYSIN   DD      *
               NAME    anyname(R)
/*
//
```

where `anyname` is your load module name

Figure 13. Sample job control statements to call the DFHwxTAL procedures

Notes:

1 If you are installing a program into either of the read-only DSAs, see “Preparing application programs to run in the RDSAs” on page 40 for more details.

If you are installing a program that is to be used from the LPA, add:

- RENT to the PARM options in the EXEC statement for the ASM step of the DFHEITAL procedure
- RENT and REFR options to the LNKPARM parameter on the call to the DFHEITAL procedure.

(See “Preparing applications to run in the link pack area” on page 39.)

3 For information about the translator options you can include on the XOPTS statement, see the *CICS/ESA Application Programming Guide*.

Installing COBOL application programs

Figure 14 illustrates the flow of control in the cataloged procedures for COBOL and PL/I programs.

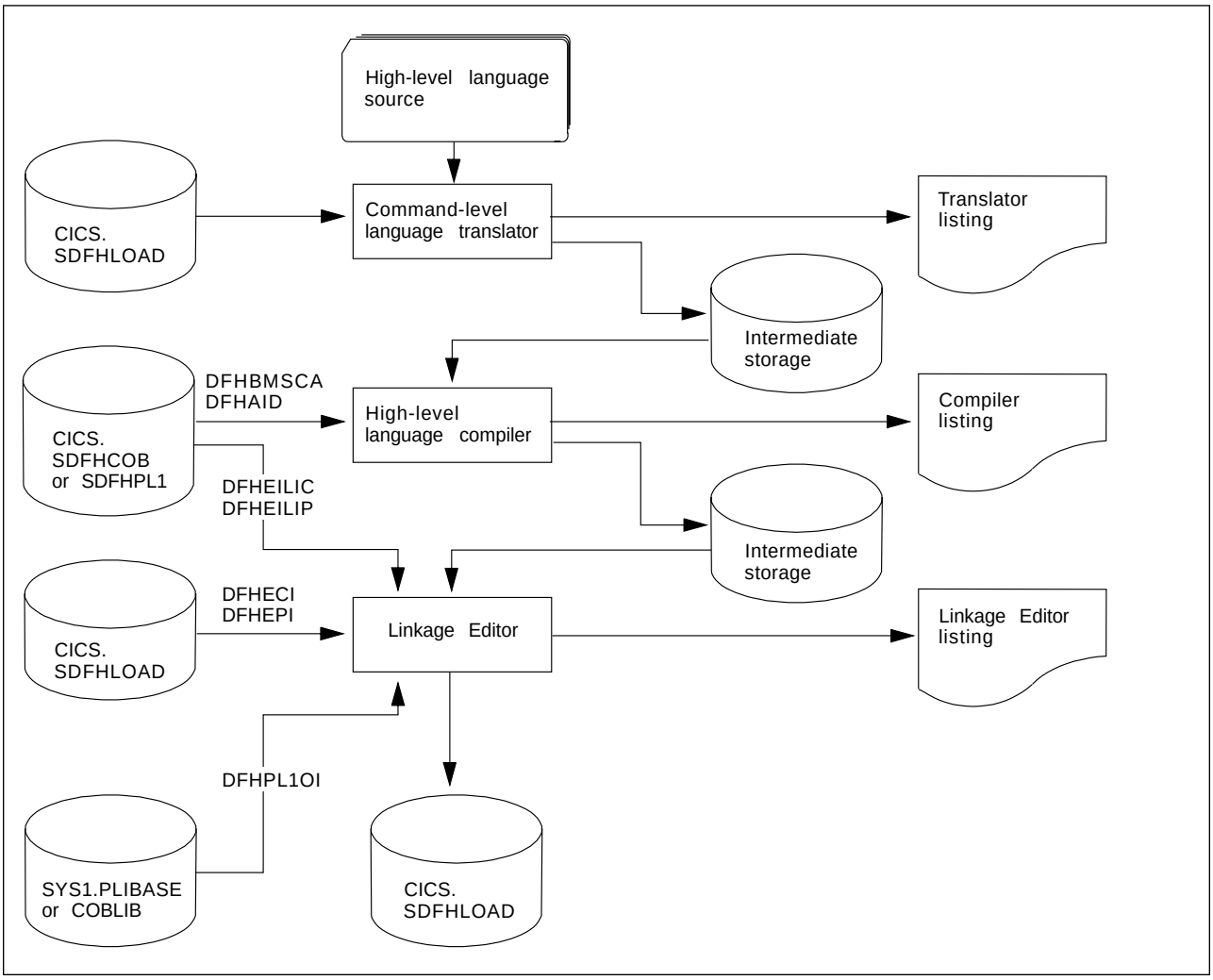


Figure 14. Installing COBOL and PL/I programs

Sample JCL to install COBOL application programs

You can use the job control statements shown in Figure 15 to process COBOL application programs. In the procedure name, "wx" depends on whether your programs are to run under Language Environment, use batch EXEC DLI calls, or otherwise. For the names of the CICS-supplied procedures, see Table 8 on page 26.

For information about adding CICS support for VS COBOL II, see "Installing CICS support for VS COBOL II, COBOL/370 and IBM COBOL for MVS and VM" on page 32.

```
//jobname    JOB  accounting info,name,MSGLEVEL=1
//           EXEC PROC=DFHwxTVL                                    1
//TRN.SYSIN  DD      *                                            2
 CBL   XOPTS(Translator options . . .)                            3
              .
             COBOL source statements
              .
/*
//LKED.SYSIN DD      *                                            4
               NAME    anyname(R)
/*
//
```

where `anyname` is your load module name

Figure 15. Sample job control statements to call the DFHwxTVL procedures

Notes for installing COBOL programs

1 Translator options:

The following translator options should be specified in accordance with the version of the COBOL compiler invoked in the compile step.

OOCOBOL for the IBM COBOL for MVS and VM release 2 compiler. This option is only needed if the object-oriented syntax (such as class-id and method-id), which is new in this compiler, is used in the application program. The OOCOBOL option implies the COBOL3, ANSI85 and COBOL2 translator options.

COBOL3 for the SAA AD/Cycle COBOL/370 compiler. COBOL3 implies the ANSI85 and COBOL2 translator options.

ANSI85 for the VS COBOL II compiler. This option specifies that the translator is to translate VS COBOL II programs that implement the ANSI85 standards. ANSI85 implies the COBOL2 option.

COBOL2 for the VS COBOL II compiler.

Compiler options:

To compile a VS COBOL II application program, you need the compiler options: RENT, RES, NODYNAM, and LIB. If you use the translator option, CBLCARD (the default), the CICS translator automatically generates a CBL statement containing these options. You can prevent the generation of a CBL or PROCESS card by

specifying the translator option NOCBLCARD and the compiler option ALOWCBL=NO.

The PARM statement of the COB step in DFHwxTVL specifies values for the compiler options. For example,

```
//COB     EXEC PGM=IGYCRCTL,REGION=&REG,
//        PARM='NODYNAM,LIB,OBJECT,RENT,RES,APOST,MAP,XREF'
```

It does not specify values for the SIZE and BUF options. The defaults are SIZE=MAX, and BUF=4K. SIZE defines the amount of virtual storage available to the compiler, and BUF defines the amount of dynamic storage to be allocated to each compiler buffer work file. You can change these options with a PARM.COB parameter in the EXEC statement that invokes the procedure. For example:

```
EXEC  PROC=DFHwxTVL,PARM.COB='SIZE=512K,BUF=16K,.,.,.'
```

Ensure that the APOST|QUOTE option in effect for the COBOL compiler matches that for the translator.

There is no BATCH compiler option for VS COBOL II.

If your application program uses the CALL DL/I interface, you must specify the DATA(24) compiler option. For information about VS COBOL II compiler options, and about requirements for programs using the CALL DL/I interface, see the *VS COBOL II Application Programming Guide*.

You can change compiler options by using any of the following methods:

- The parameter override in the PARM statement on the COB step of the DFHwxTVL procedure
- A CBL statement at the start of the source statements in the job stream used to call the DFHwxTVL procedure.

  Note: If you use a CBL statement, you need a parameter override of BATCH on the EXEC PROC=DFHwxTVL statement.

A parameter override cancels ***all*** the options specified in the procedure JCL. Therefore, if you use a parameter override, ensure that all the options you want are in the override or in a CBL statement.

- The VS COBOL II macro, IGYCOPT.

  This is needed if you do not use a CBL statement; that is, have specified the translator option NOCBLCARD and the compiler option ALOWCBL=NO.

For background information about the translator option CBLCARD|NOCBLCARD, see the *CICS/ESA Application Programming Guide*. If you choose to use the NOCBLCARD option, you must also specify the VS COBOL II compiler option ALOWCBL=NO to prevent an error message of IGYOS4006-E being issued. The ALOWCBL=NO option can be overridden at compile time by the JCL PARM option or a TSO command. For more information about the ALOWCBL PARM option, see the *Application Programming Guide for MVS and CMS*. For more information about the ALOWCBL compiler option, see the *VS COBOL II Installation and Customization for MVS* manual.

2 If you have no input for the translator, you can specify `DD DUMMY` instead of `DD *`. However, if you specify `DD DUMMY`, you must also code a suitable DCB

operand. (The translator does not supply all the data control block information for the SYSIN data set.)

3 Any translator options you specify should include the type of COBOL translator option, COBOL2.

For information about the translator options you can include on the XOPTS statement, see the *CICS/ESA Application Programming Guide*.

4 You can ignore weak external references unresolved by the linkage editor.

The link-edit job step requires access to the libraries containing the environment-specific modules for CICS, the general VS COBOL II library subroutines, and the Language Environment link-edit modules, as appropriate. The required libraries are included in the SYSLIB concatenation of procedures for VS COBOL II. Override or change the names of these libraries if the modules and library subroutines are installed in libraries with different names.

If you are installing a program into either of the read-only DSAs, see "Preparing application programs to run in the RDSAs" on page 40 for more details.

If you are installing a program that is to be used from the LPA, add the RENT and REFR options to the LNKPARM parameter on the call to the DFHEITVL procedure. (See "Preparing applications to run in the link pack area" on page 39.)

Apar PQ04566

Documentation for Apar PQ04566 added 05/12/97

Sample job stream to install OS/VS COBOL application programs

You can use the job control statements shown in Figure 16 to process OS/VS COBOL application programs. In the procedure name: x is "B" for batch or BMP EXEC DLI programs, and "I" for all other programs.

```
//jobname    JOB  accounting info,name,MSGLEVEL=1
//           EXEC PROC=DFHExTCL                          1
//TRN.SYSIN    DD      *                                 2
 CBL   XOPTS(Translator options . . .)                   3
              .
             OS/VS COBOL source statements
              .
/*
//LKED.SYSIN DD      *                                   4
               NAME    anyname(R)
/*
//
```

where `anyname` is your load module name

Figure 16. Sample job control statements to call the DFHExTCL procedures

Notes for installing an OS/VS COBOL program:

1 **Compiler options:**

Three OS/VS COBOL compiler options are required when compiling CICS application programs: NODYNAM, NOTRUNC, and LIB. These options are provided by default in the PARM statement in the DFHExTCL procedures.

Two further compiler options are also provided in the PARM statement in the DFHExTCL procedure. These are SIZE=256K and BUF=16K; you may need to modify these values to suit your installation. The value of the BUF operand depends on the block sizes of the data sets used by the OS/VS COBOL compiler. You can modify these PARM statement with a PARM.COB parameter in the EXEC statement that invokes the procedure. For example:

```
EXEC  PROC=DFHExTCL,PARM.COB='SIZE=512K,BUF=32K,.,.,.'
```

Ensure that the value of the LANGLVL option in effect for the OS/VS COBOL compiler matches that for the translator. APOST is a default option for the translator, and the procedure specifies APOST as a compiler option.

CICS supports the RES option with OS/VS COBOL. RES can save virtual storage, thus indirectly improving the working set and potentially reducing paging.

However, you cannot use the RES compiler attribute with the CALL statement in an EXECKEY(USER) program. OS/VS COBOL applications compiled with the RES compiler attribute will cause a protection exception (an ASRA transaction abend) if they execute a COBOL CALL statement (not an EXEC CICS LINK) to get to a subroutine linked-edited with the application while running with EXECKEY(USER) in a system with storage protection active. For more information, see the *CICS/ESA Application Programming Guide*.

You can change compiler options by using the parameter override on the COB step of the DFHExTCL procedure, or by a CBL statement at the start of the source statements. If you use a CBL statement, you need a parameter override of BATCH on the EXEC PROC=DFHExTCL statement. A parameter override cancels ***all*** the options specified in the procedure JCL. Therefore, if you use a parameter override, ensure that all the options you want are in the override or in a CBL statement.

2 If you use `DD DUMMY` instead of `DD *`, you must also code a suitable DCB operand. (The translator does not supply all the data control block information for the SYSIN data set.)

3 For guidance information about the translator options you can include on the XOPTS statement, see the *CICS/ESA Application Programming Guide*.

4 **Linkage editor considerations:** You can ignore weak external references unresolved by the linkage editor.

If you are installing a program into the extended read-only DSA (ERDSA), see “Preparing application programs to run in the RDSAs” on page 40 for further guidance.

If you are installing a program that is to be used from the LPA, add the RENT and REFR options to the LNKPARM parameter on the call to the DFHEITCL procedure. (See “Preparing applications to run in the link pack area” on page 39.)

Installing C application programs

Figure 17 on page 57 illustrates the flow of control in the cataloged procedures for C command-level programs.

Before you can install any C programs, you must have installed the C library and compiler and generated CICS support for C. (See "Installing CICS support for C/370 and IBM C/C++ for MVS/ESA" on page 34.)

Sample JCL to install C application programs

You can use the job control statements shown in Figure 18 to process C application programs. In the procedure name, `w` and `x` depend on whether the program to be installed is C, is to run under Language Environment, or is to use the external CICS interface. For the names of the CICS-supplied procedures, see Table 8 on page 26.

```
//jobname    JOB  accounting info,name,MSGLEVEL=1
//           EXEC PROC=DFHwxTDL                                    1
//TRN.SYSIN  DD      *                                            2
#pragma XOPTS(Translator options . . .)                           3
             .
             C source statements
              .
/*
//LKED.SYSIN DD      *                                            4
               NAME    anyname(R)
/*
//
```

where `anyname` is your load module name

Figure 18. Sample JCL to call the DFHwxTDL procedures

Notes for installing a C program:

1 Compiler options:

You can code compiler options by using the parameter override (PARM.C) in the EXEC statement that invokes the procedure, or by a C statement at the start of the source statements. If you use a C statement, you need a parameter override of BATCH on the EXEC PROC=DFHEITyL statement.

You can compile your C/370 applications under Version 1 Release 2 of the C/370 compiler and run them under Version 1 Release 2 of the C/370 library.

2 If you have no input for the translator, you can specify `DD DUMMY` instead of `DD *`. However, if you specify `DD DUMMY`, you must also code a suitable DCB operand. (The translator does not supply all the data control block information for the SYSIN data set.)

3 Translator options: For information about the translator options you can include on the XOPTS statement, see the *CICS/ESA Application Programming Guide*.

4 If you are installing a program into either of the read-only DSAs, see "Preparing application programs to run in the RDSAs" on page 40 for more details.

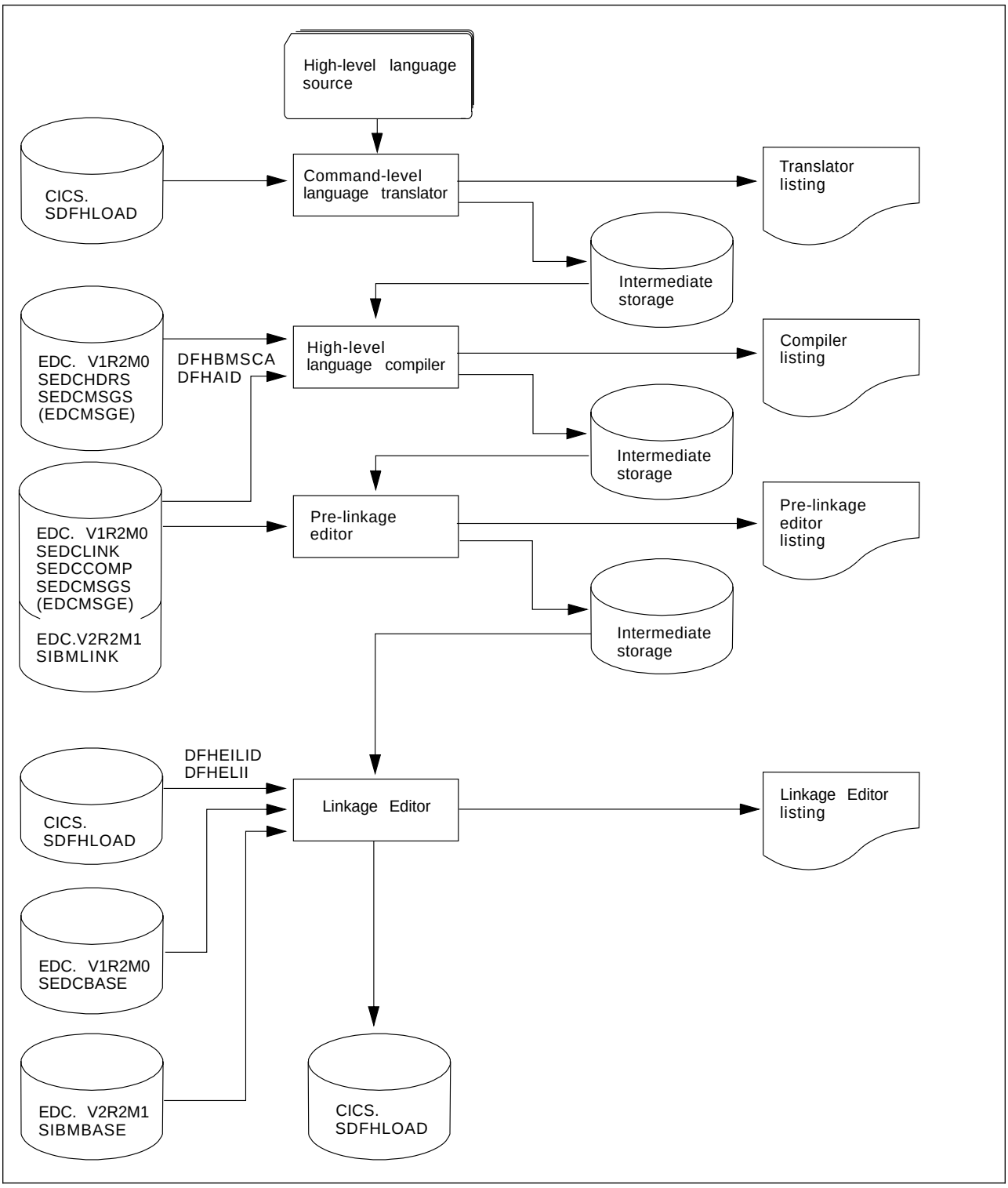


Figure 17. Installing C programs using the DFHxyTzL procedure

If you are installing a program that is to be used from the LPA, add the RENT and REFR options to the LNKPARM parameter on the call to the DFHEITyL procedure. (See “Preparing applications to run in the link pack area” on page 39.)

C language programs must be link-edited with AMODE=31, so the DFHwxTDL procedures specify AMODE(31) by default.

Installing PL/I application programs

Figure 14 on page 51 illustrates the flow of control in the cataloged procedures for PL/I programs.

+ PL/I-CICS application programs must be link-edited in a different way from non-CICS PL/I applications. This is because the normal entry point, control section PLISTART, is not required on CICS systems. Instead, the module DFHPL1OI is provided by PL/I, which acts as the entry point to the program and must be link-edited with the application program. The CICS loader requires that DFHPL1OI be positioned at the head of the load module.

+ For example, assuming that the PL/I application is made up of members PLIAPP and EXTSUB, which reside in the library defined by ddname L1, and that the SYSLIB data set contains DFHPL1OI, then the following linkage editor statements should be used:

```
INCLUDE  SYSLIB (DFHPL1OI)
      Ensure that DFHPL1OI is at the head of the load module.
REPLACE  PLISTART
      Delete unwanted CSECTs from following INCLUDE.
INCLUDE  L1 (PLIAPP)
      Application procedure (1).
REPLACE  PLISTART
      Delete unwanted CSECTs from following INCLUDE.
INCLUDE  L1 (EXTSUB)
      Application procedure (2).
INCLUDE  DFHSHRE (PLISHRE)
      Optional.  Use if shared library is to be used.
NAME     APROG (R)
      Optional.  Defines name of load module.
```

+ For more information about preparing PL/I programs, see the *OS PL/I Version 2 Programming Guide*, SC26-4307-02.

Sample JCL to install PL/I application programs

You can use the job control statements shown in Figure 19 on page 59 to process | PL/I application programs. In the procedure name, “wx” depends on whether your | programs are to run under Language Environment, use batch EXEC DLI calls, or | otherwise. For the names of the CICS-supplied procedures, see Table 8 on | page 26.

```
//jobname    JOB  accounting info,name,MSGLEVEL=1
//           EXEC PROC=DFHwxTPL                                    1
//TRN.SYSIN  DD      *                                            2
*PROCESS    XOPTS(translator options...)PL/I compiler options...; 3
              .
             PL/I source statements                                4
               .
/*
//LKED.SYSIN DD      *                                            5
               NAME    anyname(R)
/*
//
```

where `anyname` is your load module name

Figure 19. Sample job control statements to call the DFHwxTPL procedures

Notes for installing a PL/I program:

These notes apply to the DFHEITPL procedure only.

1 The PL/I compiler step (EXEC PGM= statement) specifies IEL0AA as the name of the PL/I compiler. If you are using IBM PL/I for MVS & VM, change the name to IEL1AA.

In the DFHEITPL procedure, the link-edit step includes module DFHPL1OI. This module is generated during the installation of PL/I and is normally placed either in the CICS410.SDFHLOAD library, the SYS1.PLIBASE library, or a user library. Find out which library contains this module, and either copy the module to the CICS410.SDFHLOAD library or concatenate the appropriate library in LKED.SYSLIB.

If you include the PL/I REPORT and COUNT execution time options, output goes to the CPLI transient data destination. (See the *OS PL/I Optimizing Compiler: Programmer's Guide*.) There is an example of this transient data queue coded in the sample destination control table in the CICS410.SDFHSAMP library.

2 If you have no input for the translator, you can specify `DD DUMMY` instead of `DD *`. However, if you specify `DD DUMMY`, you must also code a suitable DCB operand. (The translator does not supply all the data control block information for the SYSIN data set.)

3 Translator and compiler options:

For information about the translator options you can include on the XOPTS statement, see the *CICS/ESA Application Programming Guide*.

Ignore the message from the PL/I compiler: "`IEL0548I  PARAMETER TO MAIN PROCEDURE NOT VARYING CHARACTER STRING`".

Warning messages may appear from the PL/I compiler stating that arguments and parameters do not match for calls to procedure DFHxxxx. These messages indicate that arguments specified in operands to CICS commands may not have the correct data type. Carefully check all fields mentioned in these messages, especially **receiver** fields.

4 If you include the CALL PLIDUMP statement in an application program, output goes to the CPLD transient data destination. (See the *OS PL/I Optimizing Compiler: Programmer's Guide*.) An example of the CPLD transient data queue is coded in the sample destination control table in the CICS410.SDFHSAMP library.

5 Linkage editor considerations:

You can ignore weak external references unresolved by the linkage editor.

If you are installing a program into either of the read-only DSAs, see "Preparing application programs to run in the RDSAs" on page 40 for more details.

If you are installing a program that is to be used from the LPA, add the RENT and REFR options to the LNKPARM parameter on the call to the DFHEITPL procedure. (See "Preparing applications to run in the link pack area" on page 39.)

If you want to use the PL/I shared library facility, you must generate the module PLISHRE (see "Generating PL/I shared library support for CICS" on page 36) before you compile and link-edit your application programs. When you have re-link-edited the PLISHRE module into the CICS410.SDFHLOAD library, put the INCLUDE SYSLIB(PLISHRE) control statement immediately after the INCLUDE SYSLIB(DFHPL1OI) statement in the DFHEILIP member in the CICS410.SDFHPL1 library. You must also code a LKED.SYSLIB DD statement to concatenate the library that contains the PLISHRE module in front of the SYS1.PLIBASE library. For example:

```
//LKED.SYSLIB   DD DSN=CICS410.SDFHLOAD,DISP=SHR
//              DD DSN=SYS1.PLIBASE,DISP=SHR
```

Using your own job streams

If you want to write your own JCL to translate, assemble (or compile), and link-edit your application programs, you can use the supplied cataloged procedures as a model. They are installed in the CICS410.SDFHPROC library.

The rest of this section summarizes the important points about the translator and each of the main categories of program. For simplicity, the following discussion states that you load programs into CICS410.SDFHLOAD or IMS.PGMLIB. In fact, you can use any libraries, but only when they are either included in the DFHRPL library concatenation in the CICS job stream, or included in the STEPLIB library concatenation in the batch job stream (for a stand-alone IMS batch program or a CICS shared database program).

IMS library names

The IMS libraries referred to in the job streams are identified by IMS.libnam (for example IMS.PGMLIB). If you use your own naming convention for IMS libraries, please rename the IMS libraries accordingly.

Translator requirements

The CICS translator requires a minimum of 256KB of virtual storage. You may need to use the translator options CICS and DLI. (See the following sections.) For VS COBOL II programs, use the COBOL2 translator option.

If you have no input for the translator, you can specify `DD DUMMY` instead of `DD *`. However, if you specify `DD DUMMY`, you must also code a suitable DCB operand. (The translator does not supply all the data control block information for the SYSIN data set.)

Online programs that use EXEC CICS or EXEC DLI commands

1. Always use the translator option CICS. If the program issues EXEC DLI commands, use the translator option DLI.
2. The linkage editor input (defined by the SYSLIN DD statement) must include the correct interface module ***before*** the object deck. Therefore, place an INCLUDE statement for the interface module before the object deck. Also put ORDER statements before the INCLUDE statements, and an ENTRY statement after all the INCLUDE statements.

The interface modules are:

DFHEAI	Assembler
DFHELII	C
DFHECI	COBOL
DFHPL1OI	PL/I
DFHELII	Language Environment-conforming languages

Note: Assembler and PL/I programs require additional modules, DFHEAI0 and DFHEPI respectively. In each case, inclusion of the interface module named above generates a reference to the additional module, and it is automatically included.

In the CICS-supplied procedures, the input to the linkage editor step (defined by the SYSLIN DD statement) concatenates a library member with the object deck. This member contains an INCLUDE statement for the required interface module. For example, the DFHEITVL procedure concatenates the library member DFHEILIC, which contains the following INCLUDE statement:

```
INCLUDE SYSLIB(DFHECI)
```

3. Place the load module output from the linkage editor (defined by the SYSLMOD DD statement) in CICS410.SDFHLOAD, or a user-defined application program library.

Figure 20 shows sample JCL and an inline procedure, based on the CICS-supplied procedure DFHEITVL, that can be used to install VS COBOL II application programs. The procedure does not include the COPYLINK step and concatenation of the library member DFHEILIC that contains the INCLUDE statement for the required interface module (as included in the DFHEITVL procedure). Instead, the JCL provides the following INCLUDE statement:

```
INCLUDE SYSLIB(DFHECI)
```

If this statement was not provided, the linkage editor would return an error message for unresolved external references, and the program output would be marked as not executable.

```
 //COB2APPL JOB accounting info,name,MSGCLASS=A,MSGLEVEL=(1,1),
//         CLASS=A
//*
//*********************************************************************
//* THIS JOB CAN BE USED TO INSTALL A VS COBOL II PROGRAM.
//*
//* THIS JOB ILLUSTRATES THE USE OF:
//*
//* 1) AN INLINE PROCEDURE BASED ON THE CICS=SUPPLIED PROCEDURE, DFHEITVL
//*
//* 2) INCLUDE AND ORDER STATEMENTS FOR INTERFACE MODULE DFHECI
//*
//*********************************************************************
//*
```

Figure 20 (Part 1 of 3). Sample user-defined JCL to install a VS COBOL II program

```
//MYEITVL PROC SUFFIX=1$,
// INDEX='CICS410',
// INDEX2='CICS410',
//       OUTC=A,
//       REG=5M,
//       LNKPARM='XREF,RENT,REFR',
//       WORK=SYSDA
//*
//*      THIS PROCEDURE CONTAINS 3 STEPS
//*      1.   EXEC THE COBOL TRANSLATOR (USING SUFFIX 1$)
//*      2.   EXEC THE VS COBOL II COMPILER
//*      3.   LINKEDIT THE OUTPUT TO SDFHLOAD LIBRARY
//*
//TRN    EXEC PGM=DFHECP&SUFFIX,
//       PARM='COBOL2',
//       REGION=&REG
//STEPLIB DD DSN=&INDEX..SDFHLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=&OUTC
//SYSPUNCH DD DSN=&&SYSCIN,
//       DISP=(,PASS),UNIT=&WORK,
//       DCB=BLKSIZE=400,
//       SPACE=(400,(400,100))
//COB    EXEC PGM=IGYCRCTL,REGION=&REG,
//       PARM='NODYNAM,LIB,OBJECT,RENT,RES,APOST,MAP,XREF'
//STEPLIB DD DSN=PP.COB2.COB2COMP,DISP=SHR
//SYSLIB DD DSN=&INDEX..SDFHCOB,DISP=SHR
//       DD DSN=&INDEX..SDFHSAMP,DISP=SHR
//*      DD DSN=&INDEX2..SAMPLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&OUTC
//SYSIN    DD DSN=&&SYSCIN,DISP=(OLD,DELETE)
//SYSLIN DD DSN=&&LOADSET,DISP=(MOD,PASS),
//       UNIT=&WORK,SPACE=(80,(250,100))
//SYSUT1 DD UNIT=&WORK,SPACE=(460,(350,100))
//SYSUT2 DD UNIT=&WORK,SPACE=(460,(350,100))
//SYSUT3 DD UNIT=&WORK,SPACE=(460,(350,100))
//SYSUT4 DD UNIT=&WORK,SPACE=(460,(350,100))
//SYSUT5 DD UNIT=&WORK,SPACE=(460,(350,100))
//SYSUT6 DD UNIT=&WORK,SPACE=(460,(350,100))
//SYSUT7 DD UNIT=&WORK,SPACE=(460,(350,100))
//*
//LKED   EXEC PGM=IEWL,REGION=&REG,
//       PARM='&LNKPARM',
//       COND=(5,LT,COB)
//SYSLIB DD DSN=&INDEX..SDFHLOAD,DISP=SHR
//       DD DSN=SYS2.COB2.COB2CICS,DISP=SHR
//       DD DSN=SYS2.COB2.COB2LIB,DISP=SHR
//* SYSLMOD DD DSN=&INDEX2..LOADLIB,DISP=SHR
//SYSLMOD DD DSN=&INDEX2..SDFHLOAD,DISP=SHR
//SYSUT1  DD UNIT=&WORK,DCB=BLKSIZE=1024,
//       SPACE=(1024,(200,20))
//SYSPRINT DD SYSOUT=&OUTC
//SYSLIN DD DSN=&&LOADSET,DISP=(OLD,DELETE)
//       DD DDNAME=SYSIN
// PEND
//*
```

Figure 20 (Part 2 of 3). Sample user-defined JCL to install a VS COBOL II program

```
//COB2TEST  EXEC MYEITVL,
//             INDEX='CICS410', QUALIFIER FOR CICS/ESA 4.1 LIBRARIES
//             INDEX2='user.qualif',    QUALIFIER FOR USER LIBRARIES
//             OUTC='*',
//             REG='2048K',          SYSOUT CLASS
//             WORK='SYSDA',      UNIT FOR TEMPORARY DATA SETS
//             LNKPARM='XREF,LIST,RENT'
//TRN.SYSIN DD *
        .
        VS COBOL II source statements
        .
/*
//LKED.SYSIN DD *
  INCLUDE SYSLIB(DFHECI)
  NAME name(R)
/*
//
```

Figure 20 (Part 3 of 3). Sample user-defined JCL to install a VS COBOL II program

Online programs that use the CALL DLI interface

1. Specify the translator option CICS, but not the translator option DLI.

   Note: For a program that does not use CICS commands and is only invoked by a running transaction (and never directly by CICS task initiation), no translator step is needed.

2. The interface module, DFHDLIAI, is automatically included by the linkage editor. If you use an INCLUDE statement in the linkage editor input, place it ***after*** the object deck.

3. Include the module, DLIUIB.

4. Place the load module output from the linkage editor (defined by the SYSLMOD DD statement) in CICS410.SDFHLOAD, or a user-defined application program library.

Batch, BMP, or shared database programs that use EXEC DLI commands

1. The translator option DLI is required. Do not specify the translator option CICS.

2. The INCLUDE statement for the interface module must ***follow*** the object deck in the input to the linkage editor (defined by the SYSLIN DD statement). The interface module, DFSLI000, which resides on IMS.RESLIB, is the same for all programming languages supported by IMS. If you include CICS410.SDFHLOAD in the input to the linkage editor (defined by the SYSLIB DD statement), concatenate it ***after*** IMS.RESLIB.

3. Place the load module output from the linkage editor (defined by the SYSLMOD DD statement) in IMS.PGMLIB, or a library concatenated in the STEPLIB DD statement of the batch job stream.

Batch, BMP, or shared database programs that use DL/I CALL commands

If you want to prepare assembler, COBOL, or PL/I programs that use the DL/I CALL interface, do not use any of the CICS-supplied procedures. Programs that contain CALL ASMTDLI, CALL CBLTDLI, or CALL PLITDLI should be assembled or compiled, and link-edited, as IMS applications, and are not subject to any CICS requirements. See the relevant IMS manual for information about how to prepare application programs that use the DL/I CALL interface.

Defining programs, map sets, and partition sets to CICS

To be able to use a program that you have installed in one of the load libraries specified in your CICS startup JCL, the program, and any map sets and partition sets that it uses, must be defined to CICS. To do this, CICS uses the resource definitions MAPSET (for map sets), PARTITIONSET (for partition sets), and PROGRAM (for programs). You can create and install such resource definitions in any of the following ways:

- CICS can dynamically create, install, and catalog a definition for the program, map set, or partition set when it is first loaded, by using the autoinstall for programs function.
- You can create a specific resource definition for the program, map set, or partition set and install that resource definition in your CICS region.

  You can install resource definitions in either of the following ways:

  – At CICS initialization, by including the resource definition group in the group list specified on the GRPLIST system initialization parameter

  – While CICS is running, by the CEDA INSTALL command.

For information about defining programs to CICS, see the *CICS/ESA Resource Definition Guide*.

Chapter 5. Defining DL/I support

This chapter describes what you need to do to enable your CICS region to work with local and remote DL/I, and with CICS shared database support. For information about adding system and resource definitions for use with DBCTL, see the *CICS/ESA CICS-IMS Database Control Guide*.

CICS can provide DL/I database support by using the IBM product Information Management System/Enterprise Systems Architecture (IMS/ESA) Database Manager Version 3 (5665-408) Release 1 or later.

Note: IMS/ESA 4.1 is the last IMS release to support CICS local DL/I.

You can use DL/I support with CICS through:

- Database control (DBCTL)
- CICS remote DL/I support, also known as **function shipping**
- CICS local DL/I support
- CICS shared database support (for batch applications).

IMS library names

The IMS libraries referred to in the job streams are identified by IMS.libnam (for example IMS.PGMLIB). If you use your own naming convention for IMS libraries, please rename the IMS libraries accordingly.

Note: Not all releases of IMS can be used with the new storage protection facilities available in CICS/ESA 4.1. Storage protection facilities are available for CICS ***local*** DL/I, but there are restrictions on the use of storage protection when running DBCTL. Table 11 summarizes the IMS releases that can be used with storage protection.

Table 11. Summary of IMS releases that can be used with storage protection

Release of IMS	CICS with local DL/I	CICS with DBCTL
IMS/ESA 3.1	Storage protection	Storage protection not available
IMS/ESA 4.1	Storage protection	Storage protection
Later IMS/ESA releases	n/a	Storage protection

For more information about storage protection, see “Storage protection” on page 358.

PDIRs and DDIRs

A directory of program specification blocks (PDIR) is a list of program specification blocks (PSBs) that define, for DL/I, the use of databases by application programs. A directory of data management blocks (DDIR) is a list of data management blocks (DMBs) that define, for DL/I, the physical characteristics of these databases.

Your CICS region needs one of the following:

- A PDIR and a DDIR, to access a database that it owns (local DL/I support)
- A PDIR only, to access a database owned by a remote CICS region (remote DL/I support).

Your CICS region does not need a PDIR and DDIR to access a DL/I database owned by DBCTL. For information about accessing DL/I databases owned by DBCTL, see the *CICS/ESA CICS-IMS Database Control Guide*.

Adding remote DL/I support

Remote DL/I support is included in CICS/ESA 4.1, and works with IMS 3.1 and 4.1 (or later). Usually, you use remote DL/I support, with either MRO or ISC connections, to access databases owned by another CICS region. You can also use CICS remote DL/I support to access, via another CICS region connected to DBCTL, databases owned by DBCTL. CICS regions accessing databases owned by DBCTL (that is, connected to DBCTL) must be running on the same MVS image as the DBCTL system. A simple overview is given in Figure 21.

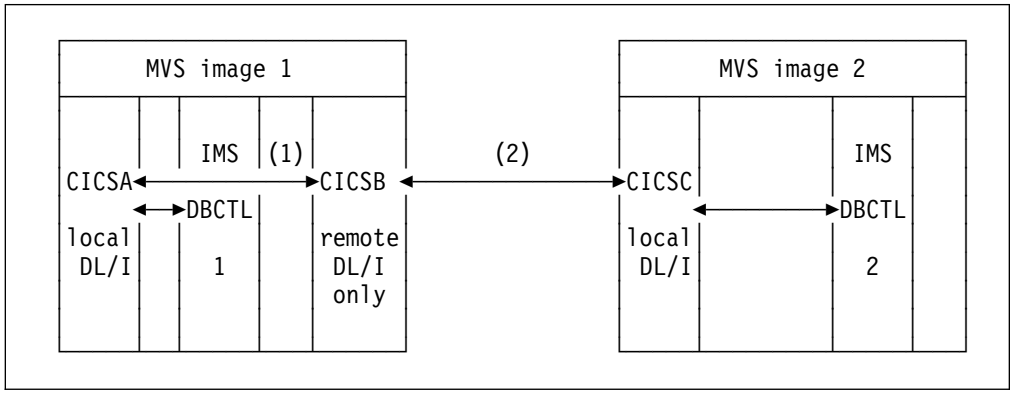


Figure 21. Using CICS remote DL/I support to access DBCTL databases

Notes:

1. CICSB uses remote DL/I to access, via CICSA, databases owned by DBCTL 1 in MVS image 1. This is only needed if CICSB is not connected to DBCTL 1.
2. CICSB uses remote DL/I to access, via CICSC, databases owned by DBCTL 2 in MVS image 2.
3. CICSA (connected to DBCTL 1) is in the same MVS image as DBCTL 1. CICSC (connected to DBCTL 2) is in the same MVS image as DBCTL 2.

For information about accessing DL/I databases owned by DBCTL, see the *CICS/ESA CICS-IMS Database Control Guide*.

To add support in CICS for remote database access, you must:

1. Code, assemble, and link-edit a program specification blocks directory (PDIR).
2. Code the CICS system initialization parameters for remote DL/I support.

Defining a PSB directory

You must code entries in a program specification block directory (PDIR), to indicate the identity of the remote CICS region, or regions, to which you want CICS to function ship DL/I requests. You do this by coding the SYSIDNT parameter in DFHDLPSB TYPE=ENTRY macros, which you assemble and link-edit to create a PDIR. You must also code the MXSSASZ parameter. You can, optionally, code the RMTNAME parameter to define the name by which the PSB is known in the remote CICS region. For information about creating PDIRs, see the *CICS/ESA Resource Definition Guide*.

Coding CICS system initialization parameters for remote DL/I support

The following is a summary of the DL/I parameters that you can, or must, code as CICS system initialization parameters:

```
DBP=1$                                  (MANDATORY FOR REMOTE DL/I)
DLI=REMOTE                              (MANDATORY FOR REMOTE DL/I)
PDIR={YES|xx}       SUFFIX OF PSB DIRECTORY
PSBCHK={NO|YES}     SECURITY CHECK OF REMOTE TERMINAL INITIATING A TRANSACTION
XPSB={YES|name|NO} PSB ENTRIES TO BE CHECKED BY RACF
```

For details of these (and other) system initialization parameters, see Chapter 21, “CICS system initialization parameters” on page 211.

Adding CICS local DL/I support

Before reading this section, read the chapter in the *CICS/ESA Installation Guide* that describes generating CICS local DL/I support.

When defining resources to CICS for CICS local DL/I, you must complete the steps listed below, described in the following sections:

1. Code, assemble, and link-edit program specification block directories (PDIRs), and data management block directories (DDIRs).
2. Code DL/I-related system initialization parameters.
3. Decide whether you need to use the DL/I global user exits, XDLIPRE and XDLIPOST. For background information about these exits, see the *CICS/ESA CICS-IMS Database Control Guide*.
4. Decide whether you require IMS data-sharing support, and code the necessary resource definitions.
5. Define and initialize a CICS system log for CICS, and give the emergency journal data set the DD name DFHJ01X. (See Chapter 11, “Defining data sets for journaling and archiving” on page 123.)
6. Decide whether you also want to run a CICS shared database region, in which case you must code the necessary resource definitions in the CSD. (See “CICS shared database support” on page 73.)
7. Decide whether you also want to access databases owned by DBCTL. For information about accessing databases owned by DBCTL, see the *CICS/ESA CICS-IMS Database Control Guide*.

The definition of DD statements for DL/I databases, and the use of IMS dynamic allocation and deallocation as an alternative to DD statements, is discussed in Chapter 19, “DD statements for DL/I data sets” on page 201.

Defining PSB and DMB directories

For CICS to access the IMS DL/I facility you must generate a PDIR and a DDIR. For information about how to code these directories, using DFHDLPSB macros for the PDIR, and DFHDLDBD macros for the DDIR, see the *CICS/ESA Resource Definition Guide*.

If you are running CICS with XRF, note that the PDIRs and DDIRs are loaded by the alternate CICS during its initialization, not at takeover time. This means that if you regenerate the directories between initialization and takeover, it is the versions loaded at initialization time that are used.

Assembling and link-editing PDIRs and DDIRs

Use a copy of the CICS-supplied procedure, DFHAUPLE, to assemble each directory.

Notes:

1. The CICS macros invoke corresponding IMS macros during the assembly of the directories, which means that the IMS.GENLIB libraries must be included in the SYSLIB concatenation. Therefore, before using the procedure, enable the // DD DSN=IMS.... statements, as directed in the DFHAUPLE procedure. You must also modify the DSN parameters if your OPTIONS, GENLIB, GENLIBA, and GENLIBB data sets have a prefix other than IMS.
2. PDIRs and DDIRs must be link-edited as AMODE(24) and RMODE(24).
3. When running with DBCTL, you must define the PDIR and DDIR to DBCTL by using IMS-supplied macros, during the sysgen of the IMS/DM system.

Coding CICS system initialization parameters for local DL/I support

The following list is a summary of the CICS system initialization parameters that are available for use with CICS local DL/I support:

Parameter	Meaning
DDIR	Suffix of the DMB directory
DLDBRC	Database recovery control
DLI|DL1	Type of CICS DL/I support
DLIOLIM	I/O error limit
DLIRLM	Name of the IMS resource lock manager
DLLPA	Use DL/I modules from the LPA
DLMON	Activate the DL/I DB monitor
DLTHRED	Number of threads through the CICS local DL/I interface
DLXCPVR	Page-fixing of ISAM or OSAM buffers
DMBPL	DMB pool size
ENQPL	ENQ control block space
PDIR	Suffix of the PSB directory
PISCHD	Program isolation scheduling
PSBCHK	DL/I security checking of remote terminal
PSBPL	PSB pool size
XPSB	PSB entries are to be checked by RACF.

If the CICS local DL/I interface has been generated to support IMS/ESA 4.1 and you are not using the IMS resource lock manager (IRLM), the system initialization parameter PISCHD is overridden, and program isolation scheduling is forced. It is therefore unnecessary to specify the PISCHD option if you use IMS/ESA 4.1. If you use local DL/I and change from intent scheduling to program isolation scheduling, you must specify the size of the ENQPOOL that you need, by using the system initialization parameter ENQPL.

For **dynamic transaction backout** purposes, you must define the correct version of DFHDBP:

DBP=2$|xx|1$ If you code DLI=YES, also code DBP=2$ to include the DL/I version of the dynamic backout program for CICS local DL/I support. Alternatively, if you have generated a user-defined DFHDBP with local DL/I support, specify DBP=xx where xx is the suffix of the DFHDBP module. If you code DL/I=REMOTE or NO, you can code DBP=1$. (See "Using a suffix to select the dynamic backout program" on page 319.)

For **shared database support**, code the following additional parameters:

IRCSTRT=YES To start interregion communication during CICS initialization. (Alternatively, you can issue the CEMT SET IRC OPEN command from the master terminal.)

ISC=YES To include the intercommunication group of programs.

For details of these (and all other) CICS system initialization parameters, see Chapter 21, "CICS system initialization parameters" on page 211.

IMS data sharing

If IMS data-sharing support is required, code the following CICS system initialization parameter:

```
DLDBRC=YES
```

Coding DLDBRC=YES generates support for sharing at the **database** level.

If you code DLDBRC=YES in a CICS region that operates in a non-data-sharing environment, you can use DBRC for database recovery, but only when you also code the RECOVCTL parameter in the DBRC INIT.RECON command. This command initializes the RECON data sets. For guidance information about DBRC commands, see the *IMS Utilities Reference* manual.

For information about system logging with DBRC, see "IMS database recovery control (DBRC) feature" on page 72.

If **block level** data sharing is required, code the following CICS system initialization parameters:

```
DLDBRC=YES
DLIRLM=YES|irlmname
```

If DLIRLM=YES|irlmname is coded with DLDBRC=NO (that is, not an IMS data sharing environment), the IRLM is used to handle program isolation scheduling. DLIRLM=YES|irlmname forces PISCHD=YES.

You do not need to code the DLIRLM system initialization parameter if you are using data sharing at the database level.

IMS database recovery control (DBRC) feature

DBRC is a prerequisite if you want to operate CICS as a subsystem in an IMS data-sharing environment, but even if you are not using IMS data-sharing, CICS can optionally use IMS database recovery control (DBRC) to recover your local IMS databases. However, if you decide to use DBRC, CICS must keep its system log on disk or standard-labeled tape.

If the CICS system log entry in the JCT specifies the PAUSE option on the JOUROPT parameter (for a journal defined on disk), CICS warns the operator to copy each disk log data set to a standard-labeled tape before it is overwritten. However, when CICS receives a reply to the outstanding WTOR message, that the data set has been copied, CICS cannot check that the journal is actually archived. One way to ensure that journal data sets are not overwritten is to specify the automatic journal archiving option (JOUROPT=AUTOARCH). In the event of an automatic archive job failing, CICS does not reuse the data set until the archive job has been rerun successfully and the control data set, DFHJACD, updated.

CICS regions using DBRC and a disk system log have a special log-copying utility, DFSUARC0. Use the IMS utility DFSUARC0 for the system log copy function, whether you are manually controlling the archiving or using the automatic archiving facility. However, if your CICS region abnormally terminates while writing to the disk log, you cannot use the DFSUARC0 utility until the log data set in use at the time of failure is closed correctly.

How you ensure that the CICS system log is closed correctly following an abnormal termination depends on whether you are running CICS with XRF.

- If you are using XRF, and an alternate CICS region performs a takeover you must wait until the emergency restart is completed. Even if you are not using XRF, you can perform an emergency restart with START=AUTO. After an emergency restart, the log data set in use at the time of the failure is closed correctly, and you can then archive it.
- If you are not using XRF, you can restart CICS with the system initialization parameter START=LOGTERM, instead of a full emergency restart. (See page 294.) After a start with START=LOGTERM, the log data set in use at the time of the failure is closed correctly, and CICS terminates. You can then archive the system log by running the DFSUARC0 utility.

Tapes produced by the DFSUARC0 utility can be used as input to the IMS recovery utilities, but not to a CICS emergency restart.

Note: CICS creates the log (and other journal data sets) with undefined (U) record format, but lays out the records in a form compatible with variable blocked (VB) format. The DFSUARC0 utility requires VB format, so you must include the DCB subparameter RECFM=VB in your JCL. This requirement is met in the skeletal JCL supplied by DBRC.

The DFSUARC0 utility may also require a valid LRECL parameter to be present, so include the DCB subparameter for LRECL, equal to the maximum blocksize minus 4. The maximum blocksize is defined by the BUFSIZE parameter in the JCT.

CICS resource definitions needed to support data sharing

Ensure that the group list that you use to initialize CICS includes a copy of the CICS-supplied group, DFHDLI. The DFHDLI group is created by DFHCSDUP when you initialize the CSD, and is included in the IBM-defined group list, DFHLIST.

Defining and initializing a CICS system log

Define a system log for CICS use with local DL/I support, to ensure data integrity. The CICS system log can be defined on disk or tape. For information about defining a system log, see Chapter 11, “Defining data sets for journaling and archiving” on page 123.

CICS shared database support

CICS shared database support needs CICS local DL/I support, which you must generate as described in the *CICS/ESA Installation Guide*; but you do not need to generate any CICS shared database modules. These are supplied pregenerated in CICS410.SDFHAUTH.

If you want to use a DL/I shared database, you must define the link between the CICS region and the batch region that shares the DL/I database with CICS. Use the following CEDA commands to create the resource definitions needed in the CSD:

```
DEFINE CONNECTION(@BCH) ACCESSMETHOD(IRC|XM)
DEFINE SESSIONS(@BCH@B) CONNECTION(@BCH)
             PROTOCOL(LU61) RECEIVECOUNT(n) RECEIVEPFX(@B)
```

If you are migrating a TCT from an earlier release of CICS to your CSD, the following entry in the TCT produces the CONNECTION-SESSIONS pair of definitions identical to those described above in the CEDA DEFINE examples:

```
DFHTCT TYPE=IRCBCH,SYSIDNT=@BCH,ACCMETH=IRC|(IRC,XM),SESNUMB=n
```

Chapter 6. Defining DB2 support

This chapter outlines what you must do to define CICS support for IBM DATABASE 2 (DB2), which uses the CICS-DB2 attachment facility supplied with CICS/ESA 4.1. You may already have completed the procedure described in this chapter if you installed CICS support for DB2 as part of the CICS installation process described in the *CICS/ESA System Definition Guide*.

Figure 22 illustrates the CICS–DB2 connection.

The CICS/ESA 4.1 CICS-DB2 attachment facility

You can use the CICS/ESA 4.1 CICS-DB2 attachment facility with CICS/ESA 4.1 regions ***only***. If you wish to use CICS regions from before CICS/ESA 4.1 with DB2, those CICS regions must continue to use the CICS-DB2 attachment facility provided before CICS/ESA 4.1. (An attachment facility that is compatible with earlier releases of CICS is included on the DB2 product tape.)

The CICS/ESA 4.1 CICS-DB2 attachment facility enables application programs running under CICS/ESA 4.1 to process commands from DB2 Release 2.3 or later.

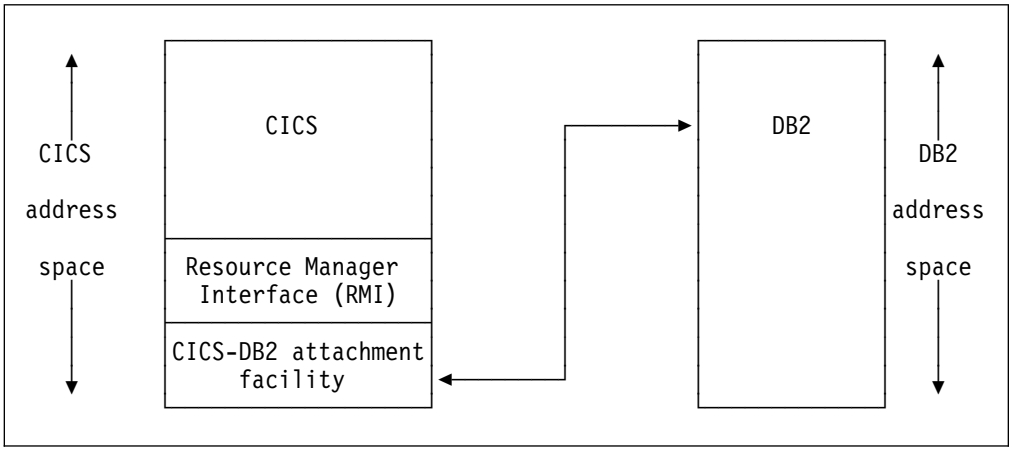


Figure 22. Overview of the CICS–DB2 connection

To define CICS-DB2 support, complete the following steps:

1. Install and test DB2, as described in the *IBM DATABASE 2 Installation Guide*.
2. Assemble and link-edit the resource control table (RCT), DSN2CT, for the CICS-DB2 attachment facility.

   Notes:

   a. The resource control table, DSN2CTxx, is for use with CICS/ESA 4.1 ***only***. If you have CICS regions from before CICS/ESA 4.1 attached to DB2, those CICS regions should continue to use the RCT, DSNCRCTx.

   b. The macro used to assemble the RCT is called DSNCRCT (as for RCTs from before CICS/ESA 4.1). This enables you to use the JCL and data that you used for your RCTs from before CICS/ESA 4.1, and only change

ASM.SYSLIB to specify the CICS410.SDFHMAC library, to generate the DSN2CT module for the new CICS-DB2 attachment facility.

You can use the DFHAUPLE procedure to assemble and link edit your resource control table. The following is an example of JCL that you can use to assemble your resource control table:

```
//RCT    EXEC DFHAUPLE,SMPPGM=IEFBR14
// PARM.LNKEDT='AMODE=24,RMODE=24,LIST,XREF,LET,NCAL'
//ASSEM.SYSUT1 DD DSN=DSNxxx.SDSNSAMP(DSN8FRCT),DISP=SHR
//LNKEDT.SYSLMOD DD DSN=DSNxxx.SDSNLOAD,DISP=SHR
```

This job does not need to reference the DB2 macro library (DSNxxx.SDSNMACS). However, if the DB2 macro library has been added to the SYSLIB statement of the ASM step in the DFHAUPLE procedure, it must be concatenated **after** the CICS macro library. The LNKEDT.SYSLMOD library can be any library that is concatenated in the STEPLIB DD statement of your CICS initialization JCL.

Note: The resource control table must be linked RMODE(24).

3. Modify your CICS startup JCL as follows:
   - Concatenate the following libraries on the STEPLIB DD statement:
     - CICS410.SDFHAUTH, for access to some of the CICS-DB2 attachment facility modules
     - DSNxxx.SDSNLOAD (after the CICS libraries)
     - The library containing the DSN2CTxx module.
   - There should be no DB2 libraries in the DFHRPL DD statement. If DB2 libraries are required in the DFHRPL concatenation by an application, or other product, they should be placed after the CICS libraries.
4. Remove programs DSNCCOM0, DSNCCOM1, and DSNCCOM2 from your PLTs, if those programs are specified there.
5. (Optional) Enable CICS to access appropriate CICS-DB2 attachment facility modules during the PLT initialization stage of CICS startup.

   To enable CICS to connect to DB2 during the PLT initialization stage of CICS startup, you must specify the program, DSN2COM0 in your startup PLT. SQL programs can run in PLT initialization after DSN2COM0 has run.
6. (Optional) Enable CICS to access appropriate CICS-DB2 attachment facility modules during the PLT termination stage of CICS shutdown.

   To enable CICS to disconnect from DB2 during the PLT termination stage of CICS shutdown, you must specify the DSN2COM2 program in your shutdown PLT.
7. (Optional) Specify the suffix of the RCT and DB2 subsystem ID to be used.

   You can specify the suffix of the RCT to be used during the PLT initialization stage of CICS startup, by using the DSN2STRT option on the INITPARM system initialization parameter, as follows:

   ```
   INITPARM=(DSN2STRT='xx,yyyy')
   ```

   where xx is the 2-character RCT suffix and yyyy is the 4-character DB2 subsystem ID.

If you specify a DB2 subsystem ID in INITPARM, it is used during PLT processing at startup, or by a DSNC command that omits to specify a resource control table suffix and DB2 subsystem ID. If you do not specify the DSN2STRT option, the default RCT, DSN2CT00, is used.

Table 12 gives examples of how the values specified determine which DB2 suffix and subsystem ID are used.

Table 12. Selection of DB2 suffix and subsystem ID

INITPARM parameter	DSNC STRT command	DB2 suffix	DB2 subsystem ID
INITPARM=AA,BBBB	None	AA	BBBB
INITPARM=AA,BBBB	DSNC STRT CC	CC	BBBB
INITPARM=AA,BBBB	DSNC STRT ,DDDD	AA	DDDD
INITPARM=AA,BBBB	DSNC STRT EE,FFFF	EE	FFFF
None specified	None specified	00 (default suffix)	Taken from resource control table

8. Code the MVS REGION parameter on the JOB statement, and the CICS system initialization parameters, to provide resources for threads. For more information about storage requirements for a CICS region, see “Storage requirements for a CICS region” on page 356. Allow sufficient LSQA storage outside the dynamic storage areas to support DB2 and user requirements. For example, allow a minimum number of bytes for each thread.
9. Define the CICS-DB2 attachment facility programs and transactions to CICS.

   CICS/ESA 4.1 supplies a resource group, DFHDB2, for all the programs and transactions of the CICS/ESA 4.1 CICS-DB2 attachment facility. When you initialize the CICS system definition data set (CSD), this resource group is installed in the CSD and added to the default startup group list, DFHLIST.

   To ensure that the resource definitions required for DB2 support are available when you initialize your CICS system, ensure that the DFHDB2 and DFHRMI resource groups[5] are added to the group list that you use to start your CICS/ESA 4.1 regions.
10. Remove from your CICS/ESA 4.1 group list any DB2 definitions for earlier releases. Check your CICS/ESA 4.1 group lists for the following resource definitions, and remove any you find from the group list:
    - DB2 program defintions:

DSNCCOM0	DSNCUEXT	DSNCSTRT	DSNCEDON
DSNCCOM1	DSNCEXT1	DSNCSTOP	DSNCMSG0
DSNCCOM2	DSNCEXT2		

    - DB2 transaction definitions:

[5] The DFHRMI resource group defines CICS task-related user exit modules and the DFHDB2 resource group defines modules and transactions which are specifically designed to attach CICS to DB2.

DSNC
Any transaction that executes program DSNCCOM0, DSNCCOM1, or DSNCCOM2.
Note: The program names for the new attachment facility have been renamed from DSNCxxxx to DSN2xxxx.

Note: If you share your CSD between CICS/ESA 4.1 and earlier releases of CICS that use DB2, you must use separate group lists for the different CICS releases. That is, the CICS/ESA 4.1 DB2 definitions must be in a different group list to the DB2 definitions for earlier releases. For more information about sharing the CSD between CICS releases, see "Sharing the CSD between CICS/ESA 4.1 and earlier releases" on page 148.

To be able to use CICS applications to access DB2, you must design, code, prepare, and test them for DB2. For information, see the *CICS/ESA Application Programming Guide*.

Chapter 7. Defining terminal resources

This chapter describes how to define to CICS the terminals (and logical units) that it is to use, and how it is to use them. You define terminals to CICS in one of two ways, depending on the type of terminal access method you are using:

1. ACF/VTAM terminals are defined in the CSD either explicitly, or by using model terminal definitions if you are using the CICS automatic installation facility (**autoinstall**). Using autoinstall, you leave it to CICS to install the terminal resource definition dynamically at logon time. CICS obtains the information needed to create a terminal control table terminal entry (TCTTE) from the TERMINAL and TYPETERM definitions recorded in the CSD. For guidance information about this process, see the *CICS/ESA Resource Definition Guide*.

   You can add VTAM definitions to the CSD offline using the DEFINE command of the CICS utility program, DFHCSDUP, or online using the CEDA DEFINE command. If you want the terminal definitions installed during CICS initialization, you must add the names of the groups containing the definitions to a group list used during a cold start. Otherwise you can install a group of definitions using the CEDA INSTALL GROUP(groupname) command online. For details of the GRPLIST system initialization parameter, see Chapter 21, “CICS system initialization parameters” on page 211.

   Each terminal must also be defined to ACF/VTAM in a VTAM definition statement.

2. Non-VTAM terminals are defined in a terminal control table (TCT) using DFHTCT macros.

   During CICS initialization, CICS loads the TCT specified by the TCT system initialization parameter, and those terminals defined in the TCT are installed as CICS resources. You must also make these terminals known to the operating system, and include a DD statement in the CICS startup job stream for each terminal.

If you are running CICS with XRF, see “XRF considerations” on page 93.

VTAM terminals

If your CICS system is to communicate with terminals or other systems using VTAM services, you must:

1. Define CICS to ACF/VTAM with an APPL statement in SYS1.VTAMLST. For more information about defining an APPL statement for CICS, see the *CICS/ESA Installation Guide*.
2. Define to VTAM the terminal resources that CICS is to use. For more information about defining terminal resources to VTAM, see “Defining CICS terminal resources to VTAM” on page 80.
3. Define to CICS the terminal resources that it is to use. For more information about defining terminal resources to CICS, see “Defining terminal resources to CICS” on page 80.

Defining CICS terminal resources to VTAM

Each terminal, or each logical unit (LU) in the case of SNA terminals, that CICS is to use must be defined to VTAM. The terminals can be defined as local or remote.

Local VTAM terminals
can be SNA terminals connected to a channel-attached cluster controller, or they can be non-SNA 3270 terminals connected via a local control unit.

Remote VTAM terminals
are attached to an SNA cluster controller, which is connected via an SDLC line with a channel-attached communications controller. The communications controller may also be loaded with code to enable remote terminals to be connected to it by a binary synchronous (BSC) line.

You define terminals, controllers, and lines in VTAM tables[6] as nodes in the network. Each terminal, or each logical unit (LU) in the case of SNA terminals, must be defined in the VTAM tables with a VTAM node name that is unique throughout the VTAM domain.

If you are using VTAM 3.3 or later, you can define the AUTINSTMODEL name, printer, and alternate printer to VTAM by using VTAM MDLTAB and ASLTAB macros. These definitions are passed to CICS to select autoinstall models and printers.

For information about defining resources to VTAM, see the *ACF/VTAM Installation and Resource Definition* manual.

Defining terminal resources to CICS

A given VTAM terminal (or logical unit) may be defined explicitly in the CICS system definition file (CSD), in which case it has a TERMINAL name, and a NETNAME (which is the same as the VTAM node name). Terminals defined in this way have terminal control table entries (TCTTEs) installed at CICS startup.

If a terminal does not have an explicit definition in the CSD, CICS can create and install a definition dynamically for the terminal when it logs on, using the CICS autoinstall facility. CICS can autoinstall terminals by reference to TYPETERM and model TERMINAL definitions created with the AUTINSTMODEL and AUTINSTNAME attributes. For information about TYPETERM and TERMINAL definitions, see the *CICS/ESA Resource Definition Guide*.

If you use autoinstall, you must ensure that the CICS resource definitions correctly match the VTAM resource definitions. For programming information about VTAM logmode definitions and their matching CICS autoinstall model definitions, see the *CICS/ESA Customization Guide*.

If you specify the system initialization parameter TCTUALOC=ANY, CICS stores the terminal control table user area (TCTUA) for VTAM terminals above the 16MB line if possible. (See page 302 for more information about the TCTUALOC parameter.)

---

6 VTAM has tables describing the network of terminals with which it communicates. VTAM uses these tables to manage the flow of data between CICS and the terminals.

Defining the terminal shutdown time limit

You can specify a time limit within which all VTAM terminals used by CICS must shut down, when CICS is shutting down. (This is to prevent a hung terminal stopping CICS shutting down.) You specify this time limit on the TCSWAIT system initialization parameter. You can also specify actions that CICS is to take, if the time limit is exceeded. You specify the actions on the TCSACTN system initialization parameter. More information about choosing appropriate values for TCSWAIT and TCSACTN is given in the following sections.

Choosing an appropriate value for TCSWAIT

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
```

Note: If you *do not* want a time limit (that is, you assume that all terminals never hang), specify the TCSWAIT=NO system initialization parameter.

Specifying that CICS is only to report hung terminals

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.

Specifying that CICS is to force close all hung terminals

To attempt to force-close all hung terminals specify the TCSWAIT=mm (with an appropriate time interval) and TCSACTN=UNBIND system initialization parameters.

Specifying that CICS is to force close some hung terminals

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.

Apar PQ15611

Documentation for Apar PQ15611 added 09/07/98

To attempt to force-close the CICS VTAM ACB if there are any hung terminals, specify the system initialization parameters TCSWAIT=***mm*** (with an appropriate time interval) and TCSACTN=FORCE.

Limitations of the terminal shutdown time limit facility

The following limitations apply to the terminal shutdown time limit:

- The terminal control shutdown time limit facility is only for VTAM terminals and VTAM intersystem connections.
- For all CICS-supported VTAM terminals, including LU Type 6.2 single-session APPC terminals (but excluding LU Type 6.1 connections and LU Type 6.2 parallel connections), the following facilities are provided:

| – The TCSWAIT-controlled shutdown timing mechanism.
| – The TCSACTN- and DFHZNEP-controlled, optional, force close mechanism.
| – The following messages:

```
DFHZC2350 Threshold exceeded. Sessions still active: ...
DFHZC2351 Terminal still active. Reason: ...
```

| • For all VTAM intersystem connections, including both LU Type 6.1 connections and LU Type 6.2 parallel connection (but not LU Type 6.2 single-session APPC terminals), the following facilities are provided:
| – The TCSWAIT-controlled shutdown timing mechanism.
| – The message: `DFHZC2352 Connection still active`
| • The force-close action on a hung terminal (no quiesce protocol, issue VTAM CLSDST, send UNBIND to the terminal) only ATTEMPTS to shutdown the terminal; there is no guarantee that all terminals will shutdown in all circumstances.

Apar PQ15611

Documentation for Apar PQ15611 added 09/07/98

To guarantee that all VTAM terminals will shutdown, and the VTAM ACB will be
closed, you must specify TCSACTN=FORCE.

TCAM terminals

CICS supports the DCB interface of ACF/TCAM (also known as the GET/PUT interface) in an SNA or non-SNA environment. This section describes the DFHTCT macros you must code to define the CICS terminals connected to this interface.

With the CICS support of the TCAM DCB interface, each TCAM communication line has associated with it two “sequential” queues: the input process queue and the output process queue. CICS routes messages for terminals connected using the TCAM DCB interface to the queue named in the DEST option of the SEND and CONVERSE commands.

CICS assumes that there is a user-written message control program (MCP) that processes messages on the TCAM queues. The TCAM MCP is responsible for polling and addressing terminals, translating code, and line control. Therefore, some DFHTCT operands that are associated with such activities are irrelevant in the TCAM DCB interface environment.

For programming information about the CICS/TCAM interface, see the *CICS/ESA Customization Guide*.

You code one DFHTCT TYPE=SDSCI macro for each input queue, and one for each output queue. The macros generate DCBs, corresponding to TPROCESS blocks. CICS treats a queue like a communication line. Each queue is described by a DFHTCT TYPE=LINE macro; this generates one TCT line entry (TCTLE) for each queue.

Note: TCAM DCBs are stored below the 16MB line.

Each TCAM terminal needs a DFHTCT TYPE=TERMINAL macro; this generates one TCT terminal entry (TCTTE) for each terminal. To avoid duplicating the TCTTEs for both the input queue and the output queue, describe all the terminals immediately after the DFHTCT TYPE=LINE macro for the output queue. Although attached to the output TCTLE, these TCTTEs are used for both input and output processing. You must also generate one dummy TCTTE for the input TCTLE; this need have only a TRMIDNT operand giving a dummy terminal identification and a LASTTRM operand.

Each input record from TCAM must contain the source terminal identification. CICS uses this identification as a search argument to find the corresponding TCTTE (by comparing against the NETNAME value for each TCTTE).

Note: The usual way to ensure that the input records contain the source terminal identification is to specify OPTCD=W in the DFHTCT TYPE=SDSCI macro. If you omit this specification, you are responsible for ensuring that the record contains a suitable source terminal identification.

By using the POOL feature (POOL=YES on the DFHTCT TYPE=LINE macro), you can establish a pool of common TCTTEs on the output TCTLE that do not contain terminal identifiers. As required, terminal identifiers are assigned to the TCTTEs or removed from association with the TCTTEs. For programming information about the TCTTEs, see the *CICS/ESA Customization Guide*.

Sequential (BSAM) devices

You can use a pair of input and output sequential data sets to simulate a terminal to CICS. For example, you can do this to test an application program before the intended terminal becomes available. You must code the following DFHTCT TYPE= macros:

```
DFHTCT TYPE=INITIAL,
       ACCMETH=(NONVTAM)    defining the access
                            method
```

(Define the following macro instructions contiguously.)

```
DFHTCT TYPE=SDSCI,
       DSCNAME=isadscn,     defining the input
       DDNAME=indd, ...      data set
DFHTCT TYPE=SDSCI,
       DSCNAME=osadscn,     defining the output
       DDNAME=outdd, ...     data set
DFHTCT TYPE=LINE,
       ISADSCN=isadscn,
       OSADSCN=osadscn, ...
DFHTCT TYPE=TERMINAL,
       TRMIDNT=name, ...
```

The two data sets defined by the DFHTCT TYPE=SDSCI macros simulate a CICS terminal known by the name specified in the TRMIDNT operand of the DFHTCT TYPE=TERMINAL macro. The DSCNAMEs of the input and output data sets must be specified in the ISADSCN and OSADSCN operands of the DFHTCT TYPE=LINE macro respectively.

You must code a DD statement for each sequential data set defined by an SDSCI macro. The DD name on the DD statement must be the same as the name coded on the DDNAME parameter (or, by default, on the DSCNAME parameter) of the SDSCI macro. For example, you could use the following DD statements for sequential input and output:

```
//CARDIN  DD  *,DCB=BLKSIZE=80
        .
    Statements containing valid transactions
        .
/*
//PRINTER DD  SYSOUT=A,DCB=BLKSIZE=132
```

This example of an I/O combination simulates a terminal to a CICS application program. There is an example of the SDSCI statements supporting this CARDIN/PRINTER combination in the copybook DFH$TCTS, which is defined in the sample TCT, DFHTCT5$. DFH$TCTS is supplied in CICS410.SDFHSAMP. Input to the application program is submitted through the input stream (CARDIN), and output to the terminal is sent to the output stream (PRINTER). If the BLKSIZE parameter is defined in the TCT for the data set, you can omit it from the JCL. However, if it is not defined in the TCT, the block size defaults to 0, and if you also omit it from the DD statement for the data set, you get message IEC141I 013-34. There are other examples of DD statements for I/O sequential data sets in some of the CICS-supplied installation verification procedures. You can find them in CICS410.SDFHINST after installation.

You can also use two DASD data sets to simulate a terminal. You must code a DD statement for each data set defined by an SDSCI macro, and the DD name on the DD statement must be the name coded on the DDNAME (or DSCNAME) parameter of the SDSCI macro. For example, you might code:

```
//DISKIN1  DD DSN=SIMULATD.TERMINAL.IN,
//         UNIT=3380,DISP=OLD,VOL=SER=volid
//DISKOT1  DD DSN=SIMULATD.TERMINAL.OUT,
//         UNIT=3380,DISP=OLD,VOL=SER=volid
```

Input from this simulated terminal is read from the DISKIN1 data set. Output to the terminal is written to the DISKOT1 data set.

Each statement in the input file (from CARDIN or DISKIN1 in the examples used above), must end with a character representing X'E0'. The standard EBCDIC symbol for this end-of-data hexadecimal value is a backslash (\) character, and this is the character defined to CICS in the pregenerated system. You can redefine this for your installation on the EODI system initialization parameter; see Chapter 21, “CICS system initialization parameters” on page 211 for details.

Using a sequential device with START requests

You can use a sequential device as the terminal specified on an EXEC CICS START REQUEST. This could be useful in situations where you need to associate the started task with a terminal, but where the termid specified does not need to represent a real terminal. For this purpose, define the sequential device as shown in Figure 23 on page 85, and add the required DD statement to the CICS startup JCL.

```
*
         DFHTCT TYPE=INITIAL,                                          X
               SUFFIX=xx,                                              X
               ACCMETH=(VTAM,NONVTAM)
*
         DFHTCT TYPE=SDSCI,                                            X
               DEVICE=1403,                                            X
               DSCNAME=PRNT001
*
         DFHTCT TYPE=LINE,                                             X
               ACCMETH=BSAM,                                           X
               INAREAL=80,                                             X
               TRMTYPE=CRLP,                                           X
               OSADSCN=PRNT001                                         X
               DFHTCT TYPE=TERMINAL,                                   X
               TRMIDNT=P001,                                           X
               ERRATT=NO,                                              X
               LPLEN=80,                                               X
               PGESIZE=(24,80),                                        X
               TRMSTAT=RECEIVE
*
         DFHTCT TYPE=FINAL
```

Figure 23. Example of TCT definitions needed to use BSAM device for START commands

Include the following DD statement in your CICS startup JCL to support the sequential device defined in Figure 23.

```
//PRNT001  DD  DUMMY
```

Terminating sequential input

End-of-file does not terminate sequential input. You should use CESF GOODNIGHT as the last transaction, to close the device and terminate reading from the device. Otherwise, CICS invokes the terminal error program (DFHTEP), and issues the messages in Table 13 at end-of-file on the sequential device:

Table 13. Warning messages if a sequential terminal is not closed

Message	***Destination***
DFHTC2507 *date time applid* Input event rejected return code *zz* {on line w/term|at term}*termid* {, trans}*tranid*{, rel line=} *rr,time*	CSMT
DFHTC2500 *date time applid* {Line|CU|Terminal} out of service {Term|W/Term} *termid*	CSMT

Use of CESF GOODNIGHT puts the sequential device into RECEIVE status and terminates reading from the device. However, if you close an input device in this way, the receive-only status is recorded in the warm keypoint at CICS shutdown. This means that the terminal is still in RECEIVE status in a subsequent warm start, and CICS does not then read the input file.

You can also use CESF LOGOFF to close the device and terminate reading from the device, but CICS still invokes DFHTEP to issue the messages in Table 13 at end-of-file. However, the device is left in TTI status, and is available for use when restarting CICS in a warm start.

If you want CICS to read from a sequential input data set, either during or following a warm start, you can choose one of the following methods:

- Close the input with CESF LOGOFF, and ignore the resultant messages. This leaves the terminal in TTI state, and CICS reads input automatically in the next startup.
- Do not close the input, and ignore the resultant messages. This leaves the terminal in TRANSCEIVE state, and CICS reads input automatically in the next startup.
- Close the input with CESF GOODNIGHT, and change the status from RECEIVE to TRANSCEIVE ***after*** CICS initialization by using the CEMT master terminal transaction (input through the console or a master terminal):

  ```
  CEMT SET TERMINAL(termid) TTI
  ```

- Code a user program, to be invoked from the program list table (PLT), to issue the appropriate EXEC CICS INQUIRE and EXEC CICS SET commands for each sequential device that is required to process input. For example, use the following statement to establish the state of a sequential terminal:

  ```
  EXEC CICS INQUIRE TERMINAL(termid) SERVSTATUS(cvda) TTISTATUS(cvda)
  ```

  For each terminal where SERVSTATUS returns DFHVALUE(INSERVICE) and TTISTATUS returns DFHVALUE(NOTTI), set the terminal to TRANSCEIVE with the following statement:

  ```
  EXEC CICS SET TERMINAL(termid) TTI
  ```

  For programming information about the use of EXEC CICS INQUIRE and EXEC CICS SET commands, see the *CICS/ESA System Programming Reference* manual. For programming information about writing post initialization-phase programs, see the *CICS/ESA Customization Guide*.

If you use BSAM devices for testing purposes, the final transaction to close down CICS could be CEMT PERFORM SHUT.

Console devices

You can operate CICS from a **console device**[7]

You can use a terminal as both a system console and a CICS terminal. To enable this, you must define the terminal as a console in the CSD. (You cannot define consoles in the TCT.)

If you are running CICS under MVS/ESA SP 4.1 or later, suitably authorized TSO users can enter MODIFY commands from terminals connected to TSO. To enable this, you must define the TSO user as a console device in the CSD.

You can use each console device for normal operating system functions and to invoke CICS transactions. In particular, you can use the console device for CICS master terminal functions to control CICS terminals or to control several CICS

[7] If you are running CICS under a release of MVS ***before*** MVS/ESA SP 4.1, a console device is a locally-attached system console. If you are running CICS under MVS/ESA SP 4.1 or later, a console device can be a locally-attached system console, a TSO user defined as a console, or an automated process such as Netview.

regions in conjunction with multiregion operation. Consequently, you can be a master terminal operator for several CICS regions.

You can also use console devices to communicate with alternate CICS regions if you are using XRF. Such communication is limited to the CICS-supplied transaction, CEBT.

For further guidance information about operating CICS from a console device, see the *CICS/ESA Operations and Utilities Guide*.

Defining console devices to CICS

You can define console devices to CICS by using either the DEFINE TERMINAL command of the DFHCSDUP utility, or the CEDA DEFINE TERMINAL command using RDO. The definition of consoles is summarized in Table 14.

Table 14. Summary of console definition

MVS/ESA version	CONSNAME(tso_userid)	CONSNAME(name)	CONSID(number)
V5	√	√	for migration
V4	√	√	√
V3			√

Note: Under MVS/ESA SP V5, you only need to define the console names (TSO userids) to CICS; you do not need to define the console names in a CONSOLnn member of the MVS SYS1.PARMLIB library.

For information about defining MVS consoles to CICS, see “Defining MVS consoles to CICS” on page 88. For information about defining TSO users to CICS, see “Defining TSO users as console devices” on page 88.

For information about defining console devices to MVS, see the *MVS/ESA Initialization and Tuning Reference*.

Having defined the console devices in the CSD, you must ensure that their resource definitions are installed in the running CICS region. You can install the definitions in one of two ways, as follows:

1. Include the group list that contains the resource definitions on the GRPLIST system initialization parameter in the CICS startup job.
2. During CICS execution, install the console device group by using the RDO command CEDA INSTALL GROUP(*groupname*), where *groupname* is the name of the resource group containing the console device definitions.

DFHLIST, the CICS-defined group list created when you initialize the CSD with the DFHCSDUP INITIALIZE command, does not include any resource definitions for console devices. However, there is a group called DFH$CNSL that contains definitions for three consoles. This group is intended for use with the installation verification procedures and the CICS-supplied sample programs. You can add this to your own group list, and alter the definitions to define your own console devices. Alternatively, if you decide to create new terminal definitions for your console devices, you can specify the CICS-supplied TYPETERM definition, DFHCONS, on the TYPETERM(*name*) parameter. This TYPETERM definition for console devices is generated in the group DFHTYPE when you initialize the CSD. For information about terminal definitions, see the *CICS/ESA Resource Definition Guide*.

Defining TSO users as console devices

If you are running CICS under MVS/ESA SP 4.1 or later, you can define TSO users as console devices.

To define a TSO user as a console device, you must use the CONSNAME(name) operand of the DEFINE TERMINAL command. You should define consoles to CICS with preset security by using the USERID operand, so that the CESN transaction does not have to be used by the TSO user. Otherwise, the TSO user's CICS signon password is displayed when entered for the CESN transaction. For an example of the DEFINE command required to define a TSO user, see Figure 24. For further information about defining consoles (or terminals) with preset security, see the *CICS/ESA CICS-RACF Security Guide*.

Defining MVS consoles to CICS

To use an MVS console as a CICS master terminal, you must define it to CICS by a terminal definition entry in the CSD. Each console you define is identified on the terminal definition by either CONSOLE(number) or CONSNAME(name).

For an example of the DEFINE command required to define a console, see Figure 24.

Although we recommend that you specify console names for all consoles used by CICS, you can use CONSOLE(number) instead of CONSNAME(name) when running under MVS/ESA SP 4.1 or later. However, when several MVS images are united to form a sysplex, the assignment of console identification numbers depends on the order in which the MVS images are IPLed. The identification numbers are determined from the sequence in which they are encountered in the several CONSOLnn members for the MVS images. Therefore, when you use MVS/ESA SP 4.1 or later in a sysplex, you should identify console devices attached to the sysplex by CONSNAME instead of CONSOLE.

```
//DEFTERM  JOB (accounting information),MSGCLASS=A,
//         MSGLEVEL=(1,1),CLASS=A,NOTIFY=userid
//CONSDEF  EXEC PGM=DFHCSDUP
//STEPLIB  DD DSN=CICS410.SDFHLOAD,DISP=SHR
//DFHCSD   DD DSN=CICS410.DFHCSD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
*
```

Figure 24 (Part 1 of 2). Defining consoles and a TSO user in the CSD using DFHCSDUP

```
*  Define a console for CICS under MVS before MVS/ESA SP 4.1
DEFINE TERMINAL(trmidnt)   GROUP(grpname)     TYPETERM(DFHCONS)
       CONSOLE(number)  DESCRIPTION(MVS CONSOLE number)
*  Define a console for CICS under MVS/ESA SP 4.1 or later
DEFINE TERMINAL(trmidnt)   GROUP(grpname)     TYPETERM(DFHCONS)
       CONSNAME(consname)  DESCRIPTION(MVS CONSOLE consname)
*  Define a TSO user as a console device for CICS under MVS/ESA SP 4.1
                                                      or later
DEFINE TERMINAL(trmidnt)   GROUP(grpname)     TYPETERM(DFHCONS)
       CONSNAME(tsouser)   DESCRIPTION(TSO USER tsouser)
       USERID(tsouser)
*
APPEND LIST(DFHLIST) TO(yourlist)
*
ADD GROUP(grpname) LIST(yourlist)
*
LIST   LIST(yourlist)  OBJECTS
/*
//
```

Figure 24 (Part 2 of 2). Defining consoles and a TSO user in the CSD using DFHCSDUP

Note: You must substitute your own values for the operands that are shown in lowercase in the DEFTERM job shown in Figure 24 on page 88.

GROUP(grpname)
A unique name for the group to which the console resource definition is to belong.

TERMINAL(trmidnt)
A unique 4-character terminal identifier as the name by which CICS is to know the console.

PN79353

The following change was made by APAR PN79353.

+ **CONSOLE(number)**
+ This specifies a number that is used to generate a specific terminal entry for
+ each console that can accept CICS commands.

+ If you are running CICS under MVS/ESA SP V4.1 or above, you can specify
+ numbers in the range 0 through 250, excluding 128. Numbers in the range 100
+ through 250 are migration ids only. Automated processes, such as Netview,
+ can use the migration ids to communicate with CICS rather than using a named
+ console. The migration ids are not defined in the SYS1.PARMLIB member, but
+ are dynamically allocated and re-allocated during MVS execution. Therefore,
+ you cannot rely on a particular migration id being allocated to a particular
+ automated process. You are recommended to specify console names for all
+ consoles used by CICS.

+ If you are running CICS under MVS/ESA SP V4.1 or earlier, you can only
+ specify numbers in the range 0 through 99. The numbers must match the
+ identification numbers assigned to the consoles according to their sequence in
+ the MVS SYS1.PARMLIB member, CONSOLnn.

Apar PQ15423

Documentation for Apar PQ15423 added 23/06/98

If you are running CICS under MVS/ESA SP V4.1 or above, you can use CONSNAME(INTERNAL) or CONSNAME(INSTREAM), depending on the CICS service level and the release of MVX being used, instead of 0 for this function.

CONSNAME(consname)

If CICS is running under MVS/ESA SP 4.1 or later, this specifies an 8-character console name, which is used to generate a specific terminal entry for each console that can accept CICS commands. This uniquely identifies the console device within a sysplex, regardless of which MVS image it is connected to. The names correspond to the names defined for the consoles in the MVS SYS1.PARMLIB member, CONSOLnn, or are activated by TSO users or automated processes dynamically.

To define a system console, code the console name, up to eight characters.

To define a TSO user's default console device, code the TSO userid. This uniquely identifies the TSO user as a console device within a sysplex, regardless of which MVS image the TSO session is connected to. However, a TSO user can use any other eight character name by specifying NAME(newname) on the CONSOLE command with the ACTIVATE keyword.

Apar PQ09813

Documentation for Apar PQ09813 added 07/07/98

Note: If the name DFHCONxx (where ***xx*** is any valid character) is specified, the terminal is used by CICS as a ***pool*** console for use when processing a modify command from a console with a name that is not defined to CICS.

USERID(tsouser)

The CICS preset security userid to be used for all security at this console device. This userid is permanently signed on when the console device is installed. That is, the userid cannot be signed off, and the security for the console is checked only when it is installed. You should ensure that only authorized users can install terminals with preset security.

Apar PQ09813

Documentation for Apar PQ09813 added 07/07/98

Note: Preset Security, if defined for pooled consoles (those with a Consname of DFHCONxx) allows each pooled console to be assigned only once as each is then permanently signed-on. This means that, once used, each pool entry is not returned to the pool but is permanently assigned to the console name that first used it. Preset security, if used for pooled consoles, should be defined with the same level of security for all consoles in the pool, as any pooled console may be selected for use when an unknown console requires a transaction to be run.

For further information, see the *CICS/ESA CICS-RACF Security Guide*.

TO(yourlist) and LIST(yourlist)
A unique name for *yourlist*. This new list of the groups of resource definitions to be installed at CICS startup must include all the CICS-supplied resources as well as your own.

If you have defined a console device in your CSD as one of the following:

- CONSOLE(00) for CICS running under MVS before MVS/ESA SP 4.1.

Apar PQ15423

Documentation for Apar PQ15423 added 23/06/98

- CONSNAME(INTERNAL) or CONSNAME(INSTREAM), depending on the CICS service level and the release of MVS being used, for CICS running under MVS/ESA SP 4.1 or later.

you can use it to issue commands using MVS job control language. It is also used by authorized programs that use the MGCR macro to issue MVS commands.

VTAM persistent sessions considerations

Persistent session support improves the availability of CICS. It benefits from VTAM 3.4.1 persistent LU–LU session improvements to provide restart-in-place of a failed CICS without rebinding.

CICS support of persistent sessions includes the support of all LU–LU sessions except LU0 pipeline and LU6.1 sessions. CICS determines for how long the sessions should be retained from the PSDINT system initialization parameter. This is a user-defined time interval. If a failed CICS is restarted within this time, it can use the retained sessions immediately—there is no need for network flows to rebind them.

You can change the interval using the CEMT SET VTAM command, or the EXEC CICS SET VTAM command, but the changed interval is not stored in the CICS global catalog, and therefore is not restored in an emergency restart.

If CICS is terminated through CEMT PERFORM SHUTDOWN IMMEDIATE, or if CICS fails, its sessions are placed in “recovery pending” state.

During emergency restart, CICS restores those sessions pending recovery from the CICS global catalog and the CICS system log to an “in session” state. This happens when CICS opens its VTAM ACB.

Subsequent processing is LU dependent: cleanup and recovery for non-LU6 persistent sessions are similar to that for non-LU6 backup sessions under XRF. Cleanup and recovery for LU6.2 persistent sessions maintain the bound session when possible but there are cases where it is necessary to unbind and rebind the sessions, for example, where CICS fails during a session resynchronization.

The end user of a terminal sees different symptoms of a CICS failure following a restart, depending on whether VTAM persistent sessions, or XRF, are in use:

- If CICS is running without VTAM persistent sessions or XRF, and fails, the user sees the VTAM logon panel followed by the “good morning” message (if AUTOCONNECT(YES) is specified for the TYPETERM resource definition).

- If CICS does have persistent session support and fails, the user perception is that CICS is "hanging": the screen on display at the time of the failure remains until persistent session recovery is complete. After a successful CICS emergency restart, the recovery options defined for the terminals or sessions take effect. The recovery options are specified on the RECOVOPTION parameter of the TYPETERM resource definition. If you specify SYSDEFAULT as the value for RECOVOPTION, the terminal user can clear the screen and continue to enter CICS transids. If you specify MESSAGE as the RECOVNOTIFY attribute of the TYPETERM resource definition, the user is notified of the successful recovery.

Unbinding sessions

Sessions held by VTAM in a recovery pending state are not always reestablished by CICS. CICS (or VTAM) unbinds recovery pending sessions in the following situations:

- If CICS does not restart within the specified persistent session delay interval
- If you perform a COLD start after a CICS failure
- If CICS restarts with XRF=YES (when the failed CICS was running with XRF=NO)
- If CICS cannot find a terminal control table terminal entry (TCTTE) for a session (for example, because the terminal was autoinstalled with AIRDELAY=0 specified)
- If a terminal or session is defined with the recovery option (RECOVOPT) set to UNCONDREL or NONE
- A connection is defined with the persistent session recovery option (PSRECOVERY) set to NONE.

In all these situations, the sessions are unbound, and the result is as if CICS has restarted following a failure without VTAM persistent session support.

There are some other situations where APPC sessions are unbound. For example, if a bind was in progress at the time of the failure, sessions are unbound.

Sessions not retained

There are some circumstances in which VTAM does not retain LU–LU sessions:

- VTAM does not retain sessions after a VTAM, MVS, or processor (CPC) failure.
- VTAM does not retain CICS sessions if you close VTAM with any of the following CICS commands:
  - SET VTAM FORCECLOSE
  - SET VTAM IMMCLOSE
  - SET VTAM CLOSED
- VTAM does not retain CICS sessions if you close the CICS node with the VTAM command VARY NET INACT ID=applid
- VTAM does not retain CICS sessions if you perform a normal CICS shutdown (with a PERFORM SHUTDOWN command).

Without persistent session support, all sessions existing on a CICS system are lost when that CICS system fails. In any subsequent restart of CICS, the rebinding of

sessions that existed before the failure depends on the terminal's AUTOCONNECT option. If AUTOCONNECT is specified for a terminal, the user of that terminal waits until the GMTRAN transaction has run before being able to continue working. If AUTOCONNECT is not specified for a terminal, the user of that terminal has no way of knowing (unless told by support staff) when CICS is operational again unless the user tries to log on. In either case, users are disconnected from CICS and need to reestablish a session, or sessions, to regain their working environment.

For CICS persistent session support, you need the VTAM persistent LU–LU session enhancements in VTAM 3.4.1 or later. CICS/ESA 4.1 functions with releases of VTAM earlier than 3.4.1, but in the earlier releases sessions are not retained in a bound state in the event of a CICS failure.

XRF considerations

If you intend operating CICS with the extended recovery facility (XRF), there are some more things you must consider when setting up your terminal network. For example, in an XRF environment, SNA VTAM terminals can be XRF-capable. This means that, if you have specified appropriate options in the TYPETERM definitions, XRF backup sessions can be established in parallel with the active sessions. (For guidance about defining extended recovery attributes for terminals, see the *CICS/ESA Resource Definition Guide*.)

The ability of a terminal to receive this XRF support is not determined by CICS, but by the terminal connection to CICS through ACF/NCP and ACF/VTAM. CICS gives each terminal the best support possible, based on the parameters passed to it from VTAM when the terminal logs on to CICS.

There are extra terminal definition keywords, that enable you to control the manner in which the XRF-capable terminals are supported by CICS.

Non-XRF-capable terminals must have their sessions reestablished to the new active CICS region after a takeover. How you do this depends on the network and the XRF configuration.

For further information about XRF-capable terminals, and non-XRF-capable terminals, see the *CICS/ESA 3.3 XRF Guide*.

Part 2. Defining data sets

Having defined your resources and installed your application programs (see Part 1, “Installing resource definitions” on page 1), you must now set up data sets and data definition statements. This part of the book consists of one chapter for each of the CICS facilities. Each chapter describes the facility, its function and usage, and the data sets needed to implement it on a running CICS region. If the data sets need preformatting, jobs that you can use for this purpose are shown.

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

- Chapter 8, “Preparing to set up CICS data sets” on page 97
- Chapter 9, “Defining the temporary storage data set” on page 111
- Chapter 10, “Defining transient data destination data sets” on page 115
- Chapter 11, “Defining data sets for journaling and archiving” on page 123
- Chapter 12, “Defining the CICS system definition data set” on page 139
- Chapter 13, “Defining and initializing the restart data set” on page 159
- Chapter 14, “Defining and using catalog data sets” on page 161
- Chapter 15, “Defining and using auxiliary trace data sets” on page 171
- Chapter 16, “Defining dump data sets” on page 175
- Chapter 17, “Defining the CICS availability manager data sets” on page 181
- Chapter 18, “Defining user files” on page 189
- Chapter 19, “DD statements for DL/I data sets” on page 201
- Chapter 20, “Defining the CMAC messages data set” on page 207.

Chapter 8. Preparing to set up CICS data sets

This chapter shows you how to define the data sets you need to run CICS. Some of these data sets are mandatory, whilst others are needed only if you are using the corresponding facilities. You may also need to provide data set definitions for user files, DL/I databases, and terminals other than VTAM terminals.

Space calculations are given so that you can calculate the space to allocate to the data sets, and the data definition statements to define them to the running CICS region.

CICS utility programs provided for postprocessing of the data sets are described in the *CICS/ESA Operations and Utilities Guide*.

Overview of setting up CICS data sets

Before you start setting up your CICS data sets, review the CICS programs you need, and their data set requirements. You then have to:

- Set up and catalog the data sets and libraries that are used by the CICS programs during execution.
- If necessary, initialize or preformat the data sets for use during CICS execution.
- RACF-protect the data sets to suit your security requirements.
- Include in the CICS startup job stream DD statements for the required data sets, but note that DD statements are ***not*** needed for the following:
  - User files for which you are using CICS dynamic allocation facilities
  - DL/I databases you are accessing through CICS local DL/I support if you are using IMS dynamic allocation facilities
  - DL/I databases you are accessing through CICS remote DL/I support or DBCTL.

  For more information about user file and DL/I file definitions, see 189 and 201.

Table 15 on page 98 summarizes the CICS data sets and their characteristics.

Data set naming conventions

There are no restrictions on the data set names you choose for CICS data sets, other than MVS constraints. In the examples in this book, CICS410 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.

You are recommended to use the *CTGI* naming convention, as described in the *MVS Sysplex Application Migration* manual, GC28-1211.[8] For example, if CICSHTH1 is the APPLID, the data set name for the CSD would be:

```
DFHCSD   DD  DSN=CICS410.CICSHTH1.DFHCSD,DISP=SHR
```

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 "Data set considerations when running CICS with XRF" on page 104.

Table 15 (Page 1 of 2). Summary of CICS data sets

Data set	DDNAME used by CICS	Block or control interval size (bytes)	Record format	Data set organization	Other comments
AUXILIARY TRACE (See page 171)	DFHAUXT DFHBUXT	4096	F	Sequential	3 See page 103 for information about GTF.
CAVM CONTROL (See page 181)	DFHXRCTL	4096 minimum	1	VSAM ESDS	Required if running CICS with XRF.
CAVM MESSAGE (See page 181)	DFHXRMSG	4096 minimum	1	VSAM ESDS	Required if running CICS with XRF.
CATALOGS (See page 161)	DFHGCD DFHLCD	8192 & 2048	VB	VSAM KSDS	Both data sets must be initialized before use (2).
CSD (See page 139)	DFHCSD	8192	VB	VSAM KSDS	---
DUMP (See page 175)	DFHDMPA DFHDMPB	32760 (tape) or 1 track (DASD)	V	Sequential	For CICS transaction dumps only; see page 103 for information about system dumps.

[8] The *CTGI* naming convention is a recommended example of a naming convention that you can use for CICS 4-character names, and 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

Where names are allowed to be up to eight characters long, as for CICS APPLIDs, the general recommendation is that the letters CICS are used for the first four characters, particularly for production regions.

Table 15 (Page 2 of 2). Summary of CICS data sets

Data set	DDNAME used by CICS	Block or control interval size (bytes)	Record format	Data set organization	Other comments
JOURNAL (See page 123)	DFHJnnx where: nn=id of journal x=A, B, or X	Defined in JCT as BUFSIZE= parameter (4)	U	Sequential	Defined by CICS as U, but record structure same as VB. May be processed as VB, if RECFM=VB is defined in JCL (5). These data sets must be formatted before they are used for the first time.
JOURNAL ARCHIVE CONTROL (See page 123)	DFHJACD	512	1	VSAM RRDS	Data set holds 199 records.
JOURNAL ARCHIVE PDS (See page 123)	DFHJPDS	---	FB	Sequential PO	Members of this PDS are named in the ARCHJCL parameter of the JCT entries.
JOURNAL ARCHIVE JCL OUTPUT (See page 123)	DFHJOUT	80	F	Sequential PS	---
MESSAGES (See page 207)	DFHCMACD	---	V	VSAM KSDS	Can be created and loaded by the DFHCMACI job.
RESTART (See page 159)	DFHRSD	2048 (data: maximum 2000, average 400)	V	VSAM KSDS	Must be initialized before use with a dummy record.
TEMPORARY STORAGE (See page 111)	DFHTEMP	See page 112	1	VSAM ESDS	---
TRANSIENT DATA EXTRA-PARTITION (See page 115)	From DSCNAME operand of DFHDCT TYPE=EXTRA macro	From BLKSIZE operand of DFHDCT TYPE=SDSCI macro	From RECFORM operand of DFHDCT TYPE=SDSCI macro	Sequential	The TYPE=SDSCI macro referred to has the same DSCNAME parameter as the TYPE=EXTRA macro.
TRANSIENT DATA INTRA-PARTITION (See page 115)	DFHINTRA	See page 117.	1	VSAM ESDS	---

Notes:

1 These data sets use control interval (CI) processing and therefore the record format is not relevant.

2 DFHGCD is the CICS global catalog data set, and in an XRF environment it is passively shared between the active and the alternate CICS regions. DFHLCD is the CICS local catalog data set, and this is a unique data set; each CICS region must have its own local catalog. See "Data set considerations when running CICS with XRF" on page 104 for an explanation of actively and passively shared data sets in an XRF environment.

3 The CICS utility program, DFHTU410, prints and formats auxiliary trace data. For information about this CICS utility program, see the *CICS/ESA Operations and Utilities Guide*.

4 The maximum block size on journal tapes is 32 760 bytes, and the maximum block size of 32 760 bytes for disk journal data sets is defined in its DCB in the journal formatting program, DFHJCJFP. You cannot override this block size by specifying a DCB parameter in the DD statement when you are formatting the journal data sets. You limit the actual size of the journal blocks by coding a BUFSIZE operand in the JCT entry for the journal. However, the actual size of the blocks written to journals by CICS can vary widely, up to the limit set by the BUFSIZE operand.

5 The IMS utilities require VB format, so, if you are using DL/I, include the DCB subparameter RECFM=VB on the DD statement for the system log and any other journal data sets that are processed by an IMS utility.

Multiple extents and multiple 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 might have to be much larger than your average needs to cater for exceptional cases, 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 are saving space until it becomes necessary to use it. As each extra extent becomes full, VSAM creates another. VSAM continues to create extra extents when 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.

Multiple extents over multiple volumes should be used 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, you should consider the following possible performance impacts:

- Using a large number of buffers means that for non-recoverable queues all processing can be performed without going to VSAM. This improves CICS performance. However, at shutdown all the buffers have to be flushed sequentially which can take a long time.
- If you specify a large number of buffers, and the number of queues is small, CICS takes longer to search down the chain of buffers for a particular queue.
- You can still specify only up to 255 VSAM strings. This means that there is no change on CICS waiting for VSAM strings.

CICS-supplied jobs to create CICS data sets

CICS supplies the following jobs that you can use to create the CICS data sets.

Job	Function
DFHCOMDS	Deletes and re-creates data sets common to all CICS regions. (See Table 16 on page 102.)
DFHDEFDS	Deletes and re-creates copies of data sets used only by one CICS region. (See Table 17 on page 102.) You run a separate copy of this job to create the data sets for each CICS region.
DFHALTDS	Deletes and re-creates the XRF data sets used by an alternate CICS region. (See Table 18 on page 102.) You run a separate copy of this job to create the XRF data sets for each alternate CICS region.
DFHCMACI	Deletes and re-creates the CICS messages data set, DFHCMACD, and loads it with the data from the CICS-supplied file, DFHCMACD, in the CICS410.SDFHMSGS target library.

When you ran the DFHISTAR job as part of the CICS installation or post-installation tasks, these jobs were tailored to your environment and stored in the library that

you specified on the LIB parameter of the DFHISTAR job (by default, CICS410.XDFHINST). If you have not yet run DFHISTAR, you should do so before running any of the CICS post-installation jobs.

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.

For information about these jobs and about generating new versions of them, see the *CICS/ESA Installation Guide*.

Table 16. CICS data sets created by the DFHCOMDS job

DFHCSD	CICS region definition data set
DFHJPDS	JCL data set containing DFH$ARCH
SYSIN	SYSIN data set

Table 17. CICS data sets created by the DFHDEFDS job

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
DFHGCD	CICS global catalog
DFHLCD	CICS local catalog
DFHINTRA	intrapartition transient data set
DFHJACD	automatic journal archive control data set
DFHJ01A	journal (A) data set
DFHJ01B	journal (B) data set
DFHJ01X	emergency journal data set
DFHRSD	restart data set
DFHTEMP	temporary storage data set
DFHXRCTL	XRF control data set
DFHXRMSG	XRF message data set
FILEA	sample program file

Table 18. CICS data sets created by the DFHALTDS job

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
DFHLCD	CICS local catalog

MVS system data sets used by CICS

Besides its own system data sets, summarized in Table 15 on page 98, CICS also uses some MVS data sets. These are listed in Table 19:

Table 19. MVS data sets used by CICS

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

Recalculate the size of these system data sets, taking into account the increased volumes of data that CICS generates. For example, for the SYS1.DUMPnn data sets you need at least 25 cylinders of a 3380 device, or the equivalent. For guidance information about calculating the size of SDUMP data sets, see the *MVS/ESA Initialization and Tuning Guide* manual.

The SDUMP data sets can become full with unwanted SDUMPs that precede ASRA, ASRB, and ASRD abends (after message DFHAP0001). To prevent this, suppress such SDUMPs as described on page 176.

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. Similarly, you must consider the amount of CICS monitoring data that is being written when CICS monitoring classes are active.

CICS can write records to SMF of up to 32756 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 16384 bytes (16KB) or 8192 bytes (8KB). 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.

If you are running CICS with GTF trace on, make allowance for CICS trace entries in the GTF data sets.

For background information about SMF, and about other SMF data set considerations, see the *MVS/ESA System Management Facilities (SMF) Guide*.

For programming information about CICS monitoring records and their sizes, see the *CICS/ESA Customization Guide*. For programming information about CICS statistics records and their sizes, see the *CICS/ESA Performance Guide*. For background information about GTF, see the *MVS/ESA Service Aids* manual.

Data set considerations when running CICS with XRF

There are some factors that you must consider regarding both the CICS system data sets and the user application data sets when you are running CICS with XRF. These considerations are about the type of data set sharing that takes place between the active and the alternate CICS regions, and about data set allocation and disposition.

Even if you intend to run CICS with XRF=NO to begin with, you are advised to think about data set dispositions with XRF from the start.

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 may be executing in different MVS images.

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

- For a given **file name**, do the active and alternate CICS regions:
  - Refer to separate data sets?
  - Refer to the same data set?
- For a given data set, is it required by the alternate CICS region:
  - Before takeover occurs?
  - After takeover occurs?
- For a given data set, is it allocated:
  - At job step initiation?
  - Dynamically?
- What facilities of MVS global resource serialization (GRS) or JES3 are being used? See the item about integrity of data on shared DASD on page 106.

The allocation of data sets, and how you specify the DISP parameter, are important factors when running CICS with XRF. The point at which data sets are allocated, and whether they are shared between active and alternate CICS regions must be considered. A shared data set, in XRF terms, means one that is required by both the active and alternate CICS regions, though not necessarily concurrently. (The DD statements refer to the same data set.) In an XRF environment, CICS classifies data sets as follows:

- Actively shared
- Passively shared
- 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 the:

- 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

These 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 the:

- CICS system definition data set (DFHCSD)
- Global catalog data set (DFHGCD)
- Restart data set (DFHRSD)
- CICS system log and other journal data sets (DFHJnnx)
- Automatic journal archiving data sets (DFHJACD and DFHJPDS)
- 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.

Data set allocation

If you define data sets to CICS and MVS using DD statements, they are allocated at job step initiation. This means that the value coded for the DISP parameter is critical for all shared data sets. However, if you are using dynamic allocation, the alternate CICS region does not allocate any data sets dynamically before takeover occurs.

DISP=SHR
allows data sets to be allocated concurrently by the active and alternate CICS regions. Unfortunately, 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, then consider using dynamic allocation with DISP=OLD.

Alternatively, this exposure can be reduced by using RACF protection, which can be applied to both user and system data sets.

DISP=NEW|OLD|MOD
requests exclusive use of a data set, so it follows that it may not be possible to start the alternate CICS region before the active CICS region terminates.

Integrity of data on shared DASD

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, even when DISP=OLD is specified. To prevent concurrent use, you can use either global resource serialization (GRS) or JES3, to provide global data set enqueuing in a multi-MVS environment. However, 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" on page 146.

Backup while open (BWO) of VSAM files

CICS/ESA 4.1, with MVS/DFP 3.2, DFHSM 2.5, and DFDSS 2.5, provides the support for some types of CICS VSAM data sets to be backed up by DFHSM and DFDSS while CICS is currently updating these data sets. This process is referred to as **backup while open (BWO)**. 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 via DFHSM and DFDSS and brought to a point of consistency by applying the forward recovery logs via a forward recovery utility such as CICSVR MVS/ESA.

BWO is available only for user files accessed by CICS File Control and for the CICS system definition (CSD) file.

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, and KSDS data sets are supported. MVS/DFP 3.2, DFHSM 2.5, and DFDSS 2.5 software products must also be installed.

CICS defines a data set as eligible for BWO when the FCT 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. BACKUPTYPE=STATIC, the default, defines a file as not eligible for BWO.

If DFHSM and DFDSS are to backup a data set while it is open, and the FCT specifies BACKUPTYPE=STATIC, all CICS files currently open for update against that data set must be closed before the backup can start.

The first time a file is opened against a VSAM base cluster data set since a CICS cold start, CICS checks the value specified for BACKUPTYPE. If this is DYNAMIC, CICS issues a call to MVS/DFP 3.2 callable services to update the ICF catalog to indicate that the base cluster data set is eligible for BWO while it is under the control of CICS.

Any subsequent file opened against the same cluster must have the same BACKUPTYPE attribute as that of the first file. If a mismatch is found, the file open fails.

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 starts, except for backout failures.) When CICS is terminated by a controlled normal shutdown, all CICS files are closed.

When the last file open for update (and defined as eligible for BWO) is closed against a base cluster data set, the MVS/DFP 3.2 callable services update the ICF catalog to indicate that this data set is no longer eligible for BWO. This prevents BWO during the batch window between CICS sessions.

Note: During the batch window between CICS sessions it is possible to update CICS user data sets by batch jobs (although, to maintain data integrity, this should only be done after a controlled normal shutdown, and ***never*** after an uncontrolled or immediate shutdown).

Any BWO backup made during a batch window after an uncontrolled or immediate shutdown should be discarded if batch updates are made. This is because those updates are not logged to CICS forward recovery logs and therefore the BWO backup could not be forward recovered to a point of consistency.

For a normal CICS shutdown, CICS also needs a **quiesced data set**[9] backup to be made after the batch updates and before the data set is made available to a subsequent CICS session so that CICS forward recovery can start from a consistent point.

Before DFHSM and DFDSS take a backup of any kind of a VSAM sphere, a call is made to examine the state of the ICF catalog to check if BWO is required. If so, the backup is made without attempting to obtain exclusive control and serialize updates to this data set.

When a backup copy of a data set is restored via DFHSM and DFDSS, and the backup was of a BWO type, the ICF catalog is updated to indicate that the data set needs to be forward recovered before it can be used. CICS checks this at data set open time and fails an FCT open if the catalog indicates that the data set is back-level.

[9] **Quiesced data set**: A data set against which all update activity has been quiesced so that DFHSM and DFDSS can have exclusive control while a backup is made.

The systems administrator must put appropriate procedures into place for BWO and for forward recovery, but these new procedures should be simpler than those currently in use. 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 may have been damaged while allocated to CICS. This operation may require backout of partially committed units of work, by CICS emergency restart.

The systems administrator must decide which VSAM user data sets are eligible for BWO, subject to the restrictions detailed in "Restrictions on BWO" applicable to heavily-updated KSDS data sets.

Effect of disabling activity keypointing

If activity keypointing is disabled in your CICS region (by specifying the system initialization parameter AKPFREQ=0), this has a serious effect on BWO support, because 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. For a heavily-updated data set that has been open for update for several days or weeks there could be a lot of forward recovery needed. For information about TURs and recovery points, see the *CICS/ESA 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.

Storage management facilities

The following storage management facilities are needed to use BWO:

- 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
- 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/ESA Storage Management Library: Storage Management Subsystem Migration and Planning Guide*, SC26-4659

  This describes migration to the Storage Management Subsystem and coexistence of SMS-managed storage with non-SMS-managed storage.

- *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 DFHSMS 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 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 introduces DFDSS and helps you evaluate its use.

Chapter 9. Defining the temporary storage data set

This chapter describes how to define 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. You save data in temporary storage queues that are identified by symbolic names. Temporary storage queues can be either in main storage or in VSAM-managed auxiliary storage.

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.

For background information about CICS temporary storage, see the *CICS/ESA Application Programming Guide*.

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.

Job control statements to define the temporary storage data set

To define a VSAM data set for auxiliary temporary storage, as a single extent data set on a single volume, you can use the sample job shown in Figure 25 on page 112.

Alternatively, you can run the CICS-supplied job DFHDEFDS (in CICS410.XDFHINST), to create the DFHTEMP data set as one of the data sets for a CICS region. For information about the DFHDEFDS job, see the *CICS/ESA Installation Guide*.

+ **Note:** 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.

```
//DEFTS    JOB accounting info,name
//AUXTEMP  EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
   DEFINE CLUSTER(NAME(CICS410.CNTL.CICSqualifier.DFHTEMP)-
          RECORDSIZE(4089,4089)            -                    1
          RECORDS(200)                     -
          NONINDEXED                       -
          CONTROLINTERVALSIZE(4096)        -
          SHAREOPTIONS(2 3)                -
          VOLUMES(volid))                  -
        DATA(NAME(CICS410.CNTL.CICSqualifier.DFHTEMP.DATA) -
          UNIQUE)
/*
```

Figure 25. Sample job defining an auxiliary temporary storage data set

Notes:

1 The RECORDSIZE value must be 7 bytes less than the CONTROLINTERVALSIZE. See “Space considerations” for information about calculating the control interval size.

Using multiple extents and multiple volumes

The job control statements in Figure 25 are for a single-extent data set defined on a single volume. That data set must be big enough to hold all your data. 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 “Multiple extents and multiple volumes” on page 100.

Space considerations

The amount of space allocated to temporary storage is expressed in two values that you must specify:

1. The control interval size
2. The number of control intervals in the 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
  - 20 bytes (for the temporary storage header)
  - 32 bytes if the storage is defined as recoverable and is requested by an EXEC CICS START TRANSID(name) FROM (data-area) command.

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 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 should 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 32KB.

  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 degrades performance.

Example: If you use BMS to write a 24 x 80 character screen to temporary storage, the data written occupies 1920 bytes. If you define the temporary storage queue as recoverable, you need a further 32 bytes, with another 20 bytes for the CICS temporary storage header, giving a total of 1972 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 CI size is larger than this, 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. You must consider space for dynamic transaction backout, especially the dynamic logging of DL/I database changes if you are running with CICS local DL/I support.

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 101. 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.

Job control statements for CICS execution

The DD name required by the temporary storage data set is DFHTEMP. For a CICS execution, you need a data definition statement for DFHTEMP in the startup job stream, such as:

```
//DFHTEMP  DD  DSN=CICS410.applid.DFHTEMP,DISP=SHR
```

XRF considerations

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.

Chapter 10. Defining transient data destination data sets

Data sets used for transient data destinations (queues) can be intrapartition or extrapartition. This chapter tells you how to define transient data destination data sets. The transient data intrapartition data set is a VSAM entry-sequenced data set (ESDS) used for queuing messages and data within the CICS region. Transient data extrapartition data sets are sequential files, normally on disk or tape; each queue can be used either to send data outside the CICS region or to receive data from outside the region. For background information about intrapartition and extrapartition transient data, see the *CICS/ESA Application Programming Guide*.

Messages or other data are addressed to a symbolic queue which you define in the destination control table (DCT) with the parameter TYPE=INTRA (for intrapartition queues) or TYPE=EXTRA (for extrapartition queues). The queues can be used as **indirect** destinations to route messages or data to other queues.

For information about coding destination control table entries, see the descriptions of the DFHDCT macro in the *CICS/ESA Resource Definition Guide*.

Queues used by CICS

System messages that CICS produces are commonly sent to transient data queues, either intrapartition or extrapartition. The following is a brief description of the queues used by CICS, supplied in the sample DCT (DFHDCT2$):

CADL CEDA VTAM resource log, indirect to CSSL.
CAIL Autoinstall terminal model resource log, indirect to CSSL.
CCPI CPI Communications message log, indirect to CSSL.
CCSE C/370 error data stream (stderr) log, indirect to CCSO.
CCSI C/370 input data stream (stdin) log. See note.
CCSO C/370 output data stream (stdout) log.
CDBC Database log, indirect to CSSL.
CDUL Dump message log, indirect to CSSL.
CESE Language Environment/370 runtime output.
CMIG Migration log to detect use of EXEC CICS ADDRESS CSA commands.
CPLD PL/I dumps, indirect to CPLI.
CPLI PL/I SYSPRINT output, direct to DD name PLIMSG.
CRDI RDO install log, indirect to CSSL.
CSCS Signon/sign-off security log, indirect to CSSL.
CSDL CEDA command log, indirect to CSSL.
CSFL File allocation message log, indirect to CSSL.
CSKL Transaction and profile resource log, indirect to CSSL.
CSML Signon/sign-off message log, indirect to CSSL.
CSMT Terminal error message and transaction abend message log, indirect to CSSL.
CSNE ZNAC-produced messages log, indirect to CSSL.
CSPL Program resource log, indirect to CSSL.
CSRL Partner resource log, indirect to CSSL.
CSSL Message log, direct to DD name MSGUSR (all the other general CICS queues are defined as indirect queues to CSSL).
CSTL Terminal I/O error log, indirect to CSSL.
CSZL The queue used for Front End Programming Interface (FEPI) messages. You do not have to define this queue if you do not have FEPI installed.

CSZX The queue used for Front End Programming Interface (FEPI) processing. You do not have to define this queue if you do not have FEPI installed.

Note: The queue name CCSI has been reserved for the C/370 input data stream (stdin), but any attempt to read from this stream causes EOF to be returned.

You should include in your CICS region all of the queues that CICS uses. 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. The sample destination control table contains definitions of all the queues that CICS uses, and you can use this as the basis of your own DCT. The sample table (DFHDCT2$) and the included copybooks are supplied in CICS410.SDFHSAMP. For information about the queues used by CICS, see the *CICS/ESA Resource Definition Guide*.

For information about the queues that CICS uses for RDO, see “Multiple extents and multiple volumes” on page 100.

For a way of printing these system messages on a local printer as they occur, see the transient data write-to-terminal sample program, DFH$TDWT. This sample program is supplied with the CICS pregenerated system in CICS410.SDFHLOAD, and the assembler source is in CICS410.SDFHSAMP. For programming information about DFH$TDWT, see the *CICS/ESA Customization Guide*.

Defining the intrapartition data set

You use job control statements to define the transient data intrapartition data set, which must be big enough to hold all the data for intrapartition queues. You can define the data set as any one of the following:

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

Figure 26 on page 117 shows job control statements to define a single extent data set on a single volume. Instead of defining one extent data set, which might have to be much larger than your average needs to cater for exceptional cases, you can define multiple extents and/or multiple volumes. For considerations about using multiple extents and/or multiple volumes, see “Using multiple extents and multiple volumes” on page 117.

```
//DEFDS     JOB  accounting info,name,MSGCLASS=A
//TDINTRA   EXEC PGM=IDCAMS
//SYSPRINT  DD   SYSOUT=A
//SYSIN     DD   *
       DEFINE CLUSTER -
                ( NAME(CICS410.applid.DFHINTRA)    -
                  RECORDSIZE(1529,1529)            -
                  RECORDS(100)                     -
                  NONINDEXED                       -
                  CONTROLINTERVALSIZE(1536)        -
                  VOL(volid))                      -
              DATA  -
                ( NAME(CICS410.applid.DATA.DFHINTRA))
/*
//
```

Figure 26. Sample job to define a transient data intrapartition data set

Job control statements to define the intrapartition data set

For an example of the JCL that you can use to define the transient data intrapartition data set, see Figure 26. If you allocate space in records, rather than tracks or cylinders, you need RECORDSIZE, and it should be 7 bytes less than the CONTROLINTERVALSIZE. (For full details, see the appropriate VSAM manuals.)

Alternatively, you can run the CICS-supplied job DFHDEFDS to create the DFHINTRA data set as one of the data sets for a CICS region. For information about the DFHDEFDS job, see the *CICS/ESA Installation Guide*.

+ **Note:** You must not define any extra associations for a transient data intrapartition
+ data set. (Do not, for example, define a PATH.) Doing so causes CICS startup to
+ fail.

Using multiple extents and multiple volumes

The job control statements in Figure 26 are for a single extent data set defined on a single volume. That data set must be big enough to hold all your data. 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 “Multiple extents and multiple volumes” on page 100.

Space considerations

Space is allocated to queues in units of a control interval. 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.

The CONTROLINTERVALSIZE parameter must be large enough to hold the longest record, plus the 32 bytes that CICS requires for its own purposes. The maximum control interval size is 32KB.

Size of the intrapartition data set

- Make the data set large enough to avoid a NOSPACE condition. If a queue has the reusable queue space option, a control interval allocated to the queue is released for reuse when all records stored in that control interval have been read by EXEC CICS READQ TD requests.

  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 GET requests. Space is also released by an EXEC CICS DELETEQ TD request, regardless of whether the queue is defined as reusable.

- The intrapartition data set should hold at least two control intervals.

Intrapartition data set restriction

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.

Number of VSAM buffers and strings

You can 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 101. You should specify a value to suit your CICS region.

Job control statements for CICS execution

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 a CICS execution, you need a data definition statement such as:

```
//DFHINTRA  DD  DSNAME=CICS410.applid.DFHINTRA,DISP={OLD|SHR}
```

XRF considerations

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, therefore specify DISP=SHR on the DD statement.

Defining extrapartition data sets

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

You should define transient data extrapartition data sets used as queues for CICS messages with a record length of 120 bytes and a record format of V or VB.

You need a DD statement for each entry representing an extrapartition data set in the destination control table (DCT). The DD name must match that specified in the DSCNAME operand of the DFHDCT TYPE=SDSCI macro. You can catalog these data sets in a VSAM user catalog.

You could use the DD statements shown in Figure 27 for the extrapartition data set entries in the sample DCT supplied in CICS410.SDFHLOAD. In these sample DD statements, the LOGUSR queue is defined as a sequential file on disk, and CICS410.LOGUSR is a new data set to be cataloged; the MSGUSR and PLIMSG queues are routed to SYSOUT.

```
//LOGUSR  DD   DSN=CICS410.applid.LOGUSR,DISP=(NEW,KEEP),
//         DCB=(DSORG=PS,RECFM=V,BLKSIZE=136),
//         VOL=SER=volid,UNIT=3380,SPACE=(CYL,2)
//MSGUSR  DD   SYSOUT=A
//PLIMSG  DD   SYSOUT=A
```

Figure 27. Sample JCL to define transient data extrapartition data sets

Note: Change the space allocation given in this sample job stream to suit your own installation's needs.

The DFHCXRF data set

Besides any extrapartition data sets that you might define using DFHDCT macros, there is a special extrapartition queue that CICS creates dynamically. This has the destination 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 as operands in the DCT. If you code DFHCXRF as a DSCNAME, or code CXRF as a DESTID, the DFHDCT macro generates an MNOTE 8 error message.

Although this 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.

DFHCXRF data set in active CICS regions

CICS uses the CXRF queue during:

- CICS startup, for any CICS components that need to write to transient data queues before transient data initialization has ended. Requests to write to any CICS transient data queue (DESTID=Cxxx) before the queue is ready are redirected to DFHCXRF. Requests from CICS components to write CICS transient data before even CXRF has been created fail with a QIDERR condition.

- CICS shutdown, after DFHSTP has flushed the intrapartition transient data buffers. Requests to any CICS intrapartition transient data queue after this point are redirected to DFHCXRF. This is because once the buffers have been flushed to DFHINTRA, the DCTEs for the intrapartition queues must not be modified before they are warm-keypointed by DFHWKP later in the shutdown process.'

If you want to take advantage of the special CXRF queue, you must include a DD statement for DFHCXRF. (For example, see Figure 28.) If you omit the DD statement, transient data write requests redirected to CXRF fail with a NOTOPEN condition.

DFHCXRF data set in 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.

DD statements for the DFHCXRF data set

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 Figure 28 in a startup job stream for an alternate CICS region.

```
//DFHCXRF  DD SYSOUT=*

or

//DFHCXRF  DD DSN=CICS410.applid.DFHCXRF,DISP=(NEW,KEEP),
//            DCB=(DSORG=PS,RECFM=V,BLKSIZE=136),
//            VOL=SER=volid,UNIT=3380,SPACE=(TRK,5)
```

Figure 28. 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. (See “The DFHCXRF data set” on page 119.)

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 11. Defining data sets for journaling and archiving

This chapter describes how to create and initialize the following data sets, which CICS uses for journaling in a running CICS region:

- System log data sets that CICS uses to record changes to recoverable resources
- User journal data sets for use by your own applications, and for auto-journaling file control, terminal control, and forward recovery logs in file control
- A journal archive data set, for use by the CICS automatic journal archive facility
- A partitioned data set, for the members containing the skeletal JCL used by the archive job submission program
- A sequential data set to receive the journal archive job that is output by the archive job submission program.

If you want CICS to write journal records, you must create and initialize the journal data sets you need. Each journal data set (or each journal tape) must also be defined with an operating system data definition statement in the CICS startup job stream. The DD statements defining data sets for CICS journals have the form DFHJnnA or DFHJnnB, where “nn ” is the journal identification. For all journals except the CICS system log, the journal identification in CICS journal DD statements corresponds to the JFILEID operand that you specify in the journal definition entries in the journal control table (JCT). For the system log you must specify JFILEID=SYSTEM in the JCT, and this corresponds to a journal identification of 01 in the DD statement. If you define the system log on disk, you can optionally include an emergency journal data set with the DD name DFHJ01X, but note that this data set is mandatory if you are using the CICS local DL/I interface. DFHJ01X does not need an explicit reference in the JCT; you simply define it in your CICS startup JCL with a DD statement. For information about how to define journals in the JCT, see the *CICS/ESA Resource Definition Guide*.

The CICS system log can be used to return recoverable resources to their committed state after an abnormal termination of CICS. For information about this use of the CICS system log, see the *CICS/ESA Recovery and Restart Guide*.

Defining and formatting CICS journals on disk

You define disk journals on sequential data sets that are reused for output when full. This is true whether a disk journal is defined in the JCT as one or two data sets (JTYPE=DISK1 or JTYPE=DISK2). You must also format the required disk data sets for journal output before CICS can use them for the first time. Once they have been formatted, data sets are reused for successive CICS executions. During initialization, CICS determines which data sets to open for journaling, and where to begin journaling within the data sets, according to:

- The options defined in the JOUROPT parameter in the JCT
- The type of start CICS is performing
- The status of the journal data sets, in the global catalog.

For information about where CICS begins recording on journal data sets, see the *CICS/ESA Recovery and Restart Guide*.

The CICS system log is sensitive to the physical characteristics of the DASD it resides on, and cannot be moved to another type of DASD. If you move your CICS data sets from one type of DASD to another (for example, from 3380 devices to 3390 devices), you must:

- Preformat new copies of your journal data sets (on the new DASD) by using DFHJCJFP. (See "Preformatting journal data sets.")
- Perform a cold start of CICS.

Preformatting journal data sets

You preformat each data set, including the emergency system log data set (DFHJ01X), by running the DFHJCJFP utility program. If you do not preformat a journal data set, CICS may not be able to open it. In the unlikely event that CICS does open an unformatted journal data set, wrong data may appear in the journal.

Do not reformat journal data sets used by CICS

Warning: Except for the CICS system log emergency data set, do not format journal data sets again after they have been used by CICS, unless you intend to use them for another purpose, or have a particularly good reason for needing to reformat them. Reformatting destroys the contents of a journal data set, and if you reformat the system log, CICS cannot use it in an emergency restart.

| However:

- if you use local DL/I and DBRC, but do ***not*** use CICS automatic journal archiving, then DFHJ01X ***must be reformatted*** after an emergency restart. For more information, see "Reformatting the DFHJ01X data set."
- You should format any journals that have been used in a previous run of CICS with a date/time stamp greater than the current date before they are used with the current (lower) date/time. This is to ensure that no residual log records from earlier runs of CICS are left with a higher date/time stamp, which may cause any subsequent emergency restart to fail.

Reformatting the DFHJ01X data set

If you are running CICS with local DL/I support, and using DBRC, but ***not*** using CICS automatic journal archiving, then always reformat the DFHJ01X data set after CICS has used it in an emergency restart. This ensures that the archive utility only copies records from one execution of CICS if you have to archive an emergency system log data set that was not closed correctly. However, if you are running CICS with automatic journal archiving, and you have to restart a failed emergency restart, CICS ensures that the emergency system log data set is closed correctly before submitting the archive job.

After a data set has been successfully formatted, a system console message shows the number of tracks formatted:

```
DFHJC4599   JOURNAL DATA SET INITIALIZED - nnnnn TRACKS AVAILABLE
```

Resetting the journal status

CICS maintains a record of the status of disk journal data sets in one of two ways. These are:

1. If you are using the CICS automatic journal archiving facility, the status (open, ready for use, or not ready for use) is maintained in the journal archive control data set (DFHJACD).
2. If you are not using automatic journal archiving, the status is maintained in the CICS global catalog.

If you want to change back from using automatic journal archiving to manual archiving, you must ensure that, when the change takes place, the appropriate journal data sets have already been archived using the automatic archiving JCL. This ensures that the status on the JACD data set is set to “READY FOR USE”, and that no unexpected automatic archive jobs will be submitted during the next CICS startup.

| If you are managing your own archiving of disk journals, and you specify JOUROPT=PAUSE, CICS issues message DFHJC4583 when a journal disk data set is full and ready for copying. If you fail to reply to DFHJC4583 messages, it is possible for both data sets of a journal to have a status “NOT READY” at CICS initialization. If this occurs, you can reset the journal status by specifying the system initialization parameter, JSTATUS=RESET. However, you must ensure that ***all*** journal data sets have been copied before you specify the JSTATUS parameter. Specifying RESET causes the status of all journal data sets to be set to “READY FOR USE”, and the journals repositioned according to the rules in the *CICS/ESA Recovery and Restart Guide*.

Job control statements for formatting journals on disk

You can run the CICS-supplied job DFHDEFDS to format the journal data sets as some of the data sets for a CICS region. For information about the DFHDEFDS job, see the *CICS/ESA Installation Guide*.

Alternatively, you can use the sample job in Figure 29 to format journal data sets on disk.

```
//DSKJRNLS JOB accounting info,name,MSGLEVEL=1
//STEPA    EXEC PGM=DFHJCJFP
//STEPLIB  DD  DSN=CICS410.SDFHLOAD,DISP=SHR
//JOURNAL  DD  DSN=CICS410.applid.DFHJ01A,VOL=SER=volid,       1a  2
//         SPACE=(CYL,10,,CONTIG),UNIT=3380,
//         DISP=(NEW,CATLG)
//*
```

Figure 29 (Part 1 of 2). Sample job for formatting journal data sets on disk

```
//STEPB    EXEC PGM=DFHJCJFP
//STEPLIB  DD  DSN=CICS410.SDFHLOAD,DISP=SHR
//JOURNAL  DD  DSN=CICS410.applid.DFHJ01B,VOL=SER=volid,       1a  2
//         SPACE=(CYL,10,,CONTIG),UNIT=3380,
//         DISP=(NEW,CATLG)
//*
//STEPC    EXEC PGM=DFHJCJFP
//STEPLIB  DD  DSN=CICS410.SDFHLOAD,DISP=SHR
//JOURNAL  DD  DSN=CICS410.applid.DFHJ01X,VOL=SER=volid,       1b  2
//         SPACE=(CYL,10,,CONTIG),UNIT=3380,
//         DISP=(NEW,CATLG)
//*
//STEPD    EXEC PGM=DFHJCJFP
//STEPLIB  DD  DSN=CICS410.SDFHLOAD,DISP=SHR
//JOURNAL  DD  DSN=CICS410.applid.DFHJ02A,VOL=SER=volid,       1c  2
//         SPACE=(CYL,10,,CONTIG),UNIT=3380,
//         DISP=(NEW,CATLG)
//
```

Figure 29 (Part 2 of 2). Sample job for formatting journal data sets on disk

Notes:

1 Assumptions:

a The system log is defined on 2 data sets (JTYPE=DISK2).

b The CICS local DL/I interface is installed, so DFHJ01X is required.

c A user journal is defined with journal identification 2 (JFILEID=2) and on a single data set (JTYPE=DISK1).

2 Coding the journal data definition statements:

- The DD name for the journal data definition statement must be JOURNAL.
- Change the space allocations to suit your installation's needs, but you must allocate at least 3 tracks.
- Do not remove the CONTIG attribute, because each data set space must be contiguous.
- The maximum block size for disk journal data sets is 32760 bytes, and is set in its DCB by the DFHJCJFP program. Therefore, do ***not*** code the DCB parameter on the DD statements to specify a blocksize for the journal data sets. (Specifying a DCB parameter can lead to unpredictable errors when the journal is used subsequently, such as in an emergency restart.) You limit the actual size of the journal blocks by coding a BUFSIZE operand in the JCT entry for the journal. However, the actual size of the blocks written to journals by CICS can vary widely, up to the limit set by the BUFSIZE operand. For information about specifying the BUFSIZE parameter, see the *CICS/ESA Resource Definition Guide*.

Space considerations

If you choose to have the system log on disk, you must allocate enough space for each data set. For a successful restart, the system log needs to be large enough to hold, at any one time, at least one complete keypoint. It must also hold all data logged from the start of the task (or logical unit of work) to the moment of failure, for each task that does logging.

Minimum space allocation

You must preallocate at least 3 tracks of disk space for each data set used for journaling, and all of the disk space within each data set must be contiguous. If you do not allocate at least three tracks, the formatting job fails with message DFHJC4598 (insufficient space).

In practice, allocate enough space in each data set to hold at least two to three times the amount of data logged during the processing of the longest task. Also specify the activity keypoint frequency (AKPFREQ) as a system initialization parameter so that CICS records at least two or three keypoints while writing to each system log data set. You may decide to allocate enough space on the combined log data sets to avoid having to switch more than once during a CICS run.

Variables that affect the system log's space requirements include:

- The loading on the system log. This, in turn, depends on parameters chosen during CICS table preparation, such as recoverable destination control table destinations and user journaling.
- The duration of the longest-running task.
- The frequency at which keypoints are written.
- The buffer size of the system log.
- The frequency of WAIT requests.
- The track capacity of the device on which the system log resides.

Because CICS uses the DFHJ01X data set only for output from transaction backout processing during an emergency restart, and only to hold information for inflight tasks, it can clearly be much smaller than the DFHJ01A and DFHJ01B data sets. However, if the DFHJ01X data set is too small, you see the following message:

```
DFHJC4582  SYSTEM LOG DATASET DFHJ01X IS FULL.  CICS ABNORMALLY TERMINATED
```

You could then allocate a larger data set and rerun emergency restart.

Job control statements for CICS execution

If you catalog and format the journal data sets on disk as shown in Figure 29 on page 125, the data definition statements for the CICS execution are as follows:

```
//DFHJ01A   DD  DSN=CICS410.applid.DFHJ01A,DISP=SHR
//DFHJ01B   DD  DSN=CICS410.applid.DFHJ01B,DISP=SHR
//DFHJ01X   DD  DSN=CICS410.applid.DFHJ01X,DISP=SHR
//DFHJ02A   DD  DSN=CICS410.applid.DFHJ02A,DISP=SHR
```

Defining and formatting CICS journals on tape

Tape journals generally consist of a series of physical reels (tape volumes) used in sequence. The volumes are mounted on either one or two tape drives, according to the JTYPE parameter in the JCT and the corresponding DD statements.

CICS supports the use of unlabeled tapes or standard-labeled tapes for journals.

For information about how to define CICS journals on tape, see the JCT chapter in the *CICS/ESA Resource Definition Guide*.

During emergency restart, the CICS recovery utility program, DFHRUP, must find the last record written. It needs to have this record to process the system log from the point where it was positioned when the system terminated abnormally. If new tapes are used in system logs and an abnormal termination causes no end-of-file mark to be written, difficulties occur in finding the last record written; for example, the tape may run off the end of the reel.

To avoid this, use the CICS-provided utility program, DFHFTAP, to preformat the tapes before they are used as CICS journals for the first time. Do not run the DFHFTAP utility program on tapes if they have been formatted before and contain journal records that you might need.

You can use the DFHFTAP utility program to format some tape volumes in sequence. These may be standard-labeled or unlabeled. You specify whether the tapes are labeled or not by the LABEL parameter on the DD statement for the tape in the DFHFTAP job.

If you are formatting standard-labeled tapes, the DFHFTAP utility program issues the message DFHJC6110D after formatting each tape volume asking you to supply the volume identifier of the next tape volume for formatting, or end the process.

If you ask for a further volume to be formatted, the operating system requests you to mount the identified volume. You can mount a volume that was previously labeled (by IEHINITT) with the required volume identifier. If you mount a labeled tape with a different volume identifier or an unlabeled tape, the operating system requests permission to overwrite the existing volume identifier with the required one or to create an appropriate new volume label. In this way, you can use the DFHFTAP utility program to initialize tape volumes with volume labels and to format them for CICS journal use.

Job control statements for formatting tapes for journal use

If you are using standard-labeled tapes, put the procedure shown in Figure 30 on page 129 in an executable procedure library, such as SYS1.PROCLIB, for use as a means of formatting some tapes.

```
//TAPJRNLS  PROC VOLID=,DSN='DUMMY.JRNL'
//STEPA     EXEC PGM=DFHFTAP
//STEPLIB   DD   DSN=CICS410.SDFHLOAD,DISP=SHR
//SYSPRINT  DD   SYSOUT=A
//DFHTAPE   DD   UNIT=(TAPE,,DEFER),LABEL=(,SL),DSN=&DSN,
//          DISP=(NEW,KEEP),VOL=SER=&VOLID
//          PEND
```

Figure 30. Example of a cataloged procedure for formatting tapes

You can invoke this procedure with the following EXEC statement:

```
//    EXEC  PROC=TAPJRNLS,VOLID=LOG001,DSN='CICS410.SYSTEM.LOG'
```

In the example shown in Figure 30, the procedure formats a tape volume intended for use as a CICS journal. The volume serial identifier refers to the first tape volume to be formatted. It is not necessary to code the data set name on the DSN parameter when formatting tapes for journals, but it is recommended that you do so. Code the same data set name for each volume being formatted for the same journal. You supply the volume identifiers of later volumes interactively.

The DD name for the tape data definition statement must be DFHTAPE.

If you use unlabeled tapes, the data definition statement for the tape is:

```
//DFHTAPE DD UNIT=(TAPE,,DEFER),LABEL=(,NL),DISP=(NEW,KEEP),
//           VOL=SER=LOG001
```

Job control statements for CICS execution

You must indicate the existence of tape labels by the LABEL=(,SL) operand on the DD statements defining the journal. Where the journal type is TAPE2, both DD statements must have the same data set name, so that only a single data set name is needed when the journal is read subsequently. You supply the volume identifiers to CICS by adding volumes to a series, and this volume series information is stored in the global catalog. During initialization, CICS obtains the volume identifier for opening a tape journal either from the cataloged series information, or from the DD statement for the DFHJ01A data set, as follows:

- When the SERIES startup parameter is ***not*** coded in the startup job stream:
  - If a volume series is established for the journal in the catalog, CICS obtains volume information from the series, for both warm and cold starts
  - If you are using a new JCT that includes a labeled-tape journal that does not have a cataloged series, CICS obtains the volume information from the DD statement for the DFHJ01A data set.
- When SERIES=PURGE ***is*** coded in the startup job stream:
  - CICS obtains the volume information from the DD statement for the DFHJ01A data set.

Retention periods are not used by CICS, but may be important for external control
purposes. Data definition statements for labeled journals have the format:

```
# //DFHJnny DD  DISP=OLD,DSN=dsname,
//            LABEL=(,SL[,RETPD=nn]),
//            UNIT=(TAPE,,DEFER),
//            VOL=SER=serial
```

For example, you could define a system log written to two drives and a user journal on a single drive with the following statements:

```
//DFHJ01A DD  UNIT=(TAPE,,DEFER),LABEL=(,SL,RETPD=20),
//            DISP=SHR,VOL=SER=jrnl1a,DSN=CICS410.SYSTEM.LOG
//DFHJ01B DD  UNIT=(TAPE,,DEFER),LABEL=(,SL,RETPD=20),
//            DISP=(SHR),VOL=SER=DUMMY,DSN=*.DFHJ01A
#//DFHJ02A DD  UNIT=(TAPE,,DEFER),LABEL=(,SL),
//            DISP=SHR,VOL=SER=jrnl2a,DSN=CICS410.AUDIT
```

CICS uses the volume identifier from the DD statement of the DFHJ01A data set to specify the first volume in a journal series during startup only when there is no volume series information available in the global catalog from previous runs. Typically this is when you are starting CICS with a newly initialized catalog, or when you specify the SERIES=PURGE system initialization parameter. The volume identifier is not used in any circumstances if the JCT entry for a journal specifies OPEN=DEFERRED. The volume parameter for the B data set is coded as VOL=SER=DUMMY because it is not used by CICS. You supply further volume identifiers in the series interactively, by EXEC CICS or CEMT commands. All volume series are managed by the CICS volume manager.

For a system log defined on two drives and using unlabeled tapes, the data definition statements might be:

```
//DFHJ01A DD  UNIT=(TAPE,,DEFER),LABEL=(,NL),DISP=SHR,
//            VOL=SER=volid
//DFHJ01B DD  UNIT=(TAPE,,DEFER),LABEL=(,NL),DISP=SHR,
//            VOL=SER=volid
```

Although the LABEL= parameter may specify an unlabeled tape, the volume (VOL=) parameter is also needed. If your journals are on unlabeled tapes, the operating system uses the volume identifiers in the DD statements to prompt the operator to mount the required tape volume. Unlabeled tapes are identified by the external paper label on the reel of tape, which is only significant to the operators, and cannot be checked by the system. Paper labels should conform to your installation's standard operating procedures, but should also contain a volume identifier that can be coded on the VOL=SER parameter in the JCL.

If you define the system log on tape (labeled or unlabeled), and CICS is performing an emergency restart, CICS prompts you to mount the log tape that was in use when CICS terminated abnormally. The message specifies that the tape should be mounted on the tape drive (address) referred to in the DFHJ01A DD statement.

Warning: If your log is on ***unlabeled*** tapes, CICS acts as if any tape mounted is the correct one. You must take care, therefore, when mounting unlabeled tapes.

Writing an EOF mark on a journal or system log tape

You can use the end-of-file utility program, DFHTEOF, offline to reposition and correctly close a journal tape after an uncontrolled shutdown.

CICS also uses the DFHTEOF utility program during a CICS restart. In an emergency restart, CICS checks the closed status of the system log recorded in the global catalog. If the global catalog shows that the system log was successfully closed following the uncontrolled shutdown, the DFHTEOF utility program does not need to search for the end of the log.

However, if the global catalog does not show that the log was closed, the DFHTEOF utility program writes an end-of-file (EOF) mark after the last valid record written before the failure. It does this by reading the tape forward from the load point and checking the label record in each journal block.

If the system log (or any other CICS-created tape journal) is to be used by DL/I or by a user-written program that does not include such label-checking logic, you must write an EOF mark after the last valid record. Do this even when a region ABEND caused the abnormal termination, in case the ABEND routine itself failed to close the tapes.

A sample job stream to run the DFHTEOF utility program is shown in Figure 31.

```
//TEOF      JOB  accounting info,name,MSGLEVEL=1
//JOBLIB    DD   CICS410.SDFHAUTH,DISP=SHR
//          EXEC PGM=DFHTEOF
//DFHTAPE   DD   UNIT=(180,,DEFER),LABEL=(,SL),DSN=AUDIT,
//          DISP=OLD,VOL=SER=AUD001
//
```

Figure 31. Sample job to run the tape-end-of-file utility, DFHTEOF

Ensure that the correct volume (that is, the last volume that was mounted for system log output during the previous execution) is mounted and positioned at its load point.

Execution of the DFHTEOF utility program at CICS restart uses the DFHJ01A DD statement (that is, the statement that exists for the system log).

Warning: Do not attempt to use the DFHTEOF utility program to write an EOF on an SMF format journal tape. The DFHTEOF utility program uses the time stamp in the label record to determine the position of the last record written, but SMF format tapes do not have the equivalent fields at the same location.

User journals using SMF

CICS user journals that are for output only (but not the system log) can optionally record data on SMF data sets using the SMF 110 type record format. This facility is an alternative to writing journal data to a CICS disk or tape journal. To do this, you must create the necessary JCT entries by specifying FORMAT=SMF and JTYPE=SMF on the DFHJCT TYPE=ENTRY macro.

For programming information about SMF format, see the *CICS/ESA Customization Guide*.

XRF considerations

When running CICS with XRF=YES, you must define the system log as a disk journal with two data sets; otherwise CICS abends.

The active and alternate CICS regions must refer to the same system log data sets (including the emergency journal data set, DFHJ01X, if it is being used), because the alternate uses information in the log to effect its emergency restart. While the alternate is in standby mode it does not require the system log, but opens the log during emergency restart following a takeover.

Your user journals may be on disk or tape. The user journals are opened in the usual way during the emergency restart that forms the takeover process.

Although your user journals can be on disk or tape, if your CICS region depends heavily on user journals, consider the following points:

Disk journals

The use of disk journals provides faster switching, but unless you are using automatic journal archiving (JOUROPT=AUTOARCH) it could be at the expense of operator control over archiving.

With disk journaling, you should archive each data set before reuse to preserve journal data that might be required for batch backout or forward recovery purposes. This archive must be done either by the operator, or automatically by the CICS automatic archiving program. (See “Journal archive data sets” on page 133.) To allow for a fast switch at takeover time, you are strongly recommended to define two data sets for each disk journal.

(If you define a single data set for use as a disk journal, the data set must always be archived when it becomes full, before the alternate can takeover and reuse it. The time taken to archive the data set creates an unacceptable delay during the takeover period.)

The data sets are allocated at job step initiation. The JCL defining the journal data sets must specify DISP=SHR. The need to archive journals means that this is the normal status.

Tape journals

Tape journals provide implicit archiving at the expense of slower switching. Although tape journaling covers the question of archiving, there is additional operator involvement and time required to make the tape journal available to the alternate. There is also the manual switching of tape units, and the movement of the journal tapes to consider. For these reasons tape journaling is probably unacceptable where you want a fast takeover.

Defining user journals on standard-labeled tape for XRF

The DD statements are defined differently for the active and alternate CICS regions of an XRF system.

DD statements for the active CICS region

You can define a user journal on standard-labeled tape (specified with JTYPE=TAPE2 in the JCT in this instance), with the following DD statements in the startup job stream for the active CICS region:

```
//DFHJ02A DD UNIT=(TAPE,,DEFER),LABEL=(,SL),
          DISP=SHR,VOL=SER=tape1a,DSN=AUDIT
#         //DFHJ02B DD UNIT=(TAPE,,DEFER),LABEL=(,SL),
          DISP=SHR,VOL=SER=tape1b,DSN=*.DFHJ02A
```

where:

tape1a is optionally the first volume serial of the journal.

tape1b is a dummy name.

yyddd is the expiry date expressed in year and number of days.

DD statements for the alternate CICS region

To avoid delaying the execution of other jobs and any risk of confusion, do not specify real volume serial identifiers in the DD statements for the alternate CICS. They are ***never*** used during initialization or takeover. CICS always uses the next in the series from the volume information in the global catalog during a takeover. Use dummy serial numbers that do not exist in your installation, making sure that you use a different dummy number for each different journal data set.

The following DD statements define a user journal for the alternate CICS:

```
#         //DFHJ02A DD UNIT=(TAPE,,DEFER),LABEL=(,SL),
          DISP=SHR,VOL=SER=dummy1
#         //DFHJ02A DD UNIT=(TAPE,,DEFER),LABEL=(,SL),
          DISP=SHR,VOL=SER=dummy2
```

where:

dummy1 is the dummy serial number for the first journal data set.

dummy2 is the dummy serial number (different from dummy1) for the second journal data set.

yyddd is the expiry date expressed in year and number of days.

Journal archive data sets

You can specify that you want CICS to archive a disk journal data set automatically when it is closed for output, but only when you have defined the journal in the JCT with two data sets (JTYPE=DISK2). The automatic journal archiving facility caters for all the data sets defined for a journal, including the emergency system log data set, DFHJ01X. You request automatic journal archiving using the DFHJCT TYPE=ENTRY macro, by specifying the AUTOARCH option on the JOUROPT parameter.

The automatic journal archiving facility does not normally require any action by an operator, except to mount tapes if the archive is to tape. The only time that an

operator needs to intervene is if an automatic journal archive has failed for some reason, and as a result a journal data set is not ready for use when CICS tries to open it for output. If this occurs, CICS prompts the operator with message DFHJC4542D or DFHJC4543D, which may be followed by DFHJC4544D, depending on the reply to the other message. For information about how to respond to these messages, see the *CICS/ESA Messages and Codes* manual.

If you specify JOUROPT=AUTOARCH, CICS requires the following data sets:

- A **journal archive control data set** (JACD), which must be defined as a VSAM relative-record data set (RRDS)
- A **journal partitioned data set** (JPDS), each member of which contains skeletal JCL for use by the automatic archive job submission program
- A **sequential data set** (JOUT), to receive the JCL output from the journal archive submission program, and usually specified in the DD statement as the JES internal reader.

Defining the journal archive control data set (JACD)

A journal archive control data set (JACD) is defined when you run the DFHDEFDS job as part of the CICS post-installation tasks. Alternatively, you can use the sample job in Figure 32 on page 135 to create a JACD. For information about creating CICS data sets, see the *CICS/ESA Installation Guide*.

CICS uses the JACD to store control information about the journal data sets. The JACD must have the following attributes:

- **Share option 4**. This allows both CICS and the batch archive job to update the data set, and also ensures that the data set buffers are refreshed for each access. This data set is read only infrequently, and refreshing the buffers each time is not a performance consideration, but it is important for ensuring data integrity.
- **Control interval (CI) size of 512 bytes**. This ensures that each JACD record is also a physical block.
- **Record size of 505 bytes**. This record size is the CI size minus 7 (the 7 bytes required for VSAM CI information), and ensures that the control interval contains only one JACD record.
- **Number of records as 199**. This allows one record for each of the possible disk journal data sets that can be defined to CICS, including the emergency system log data set, DFHJ01X.

You do not have to initialize this data set; CICS initializes it for you. For each JCT entry defined with JOUROPT=AUTOARCH, records are added or updated as follows:

- If there is already an entry for the journal data set on the JACD, and the journal data set name and the PDS member name of the archive JCL are the same as in the JCT, CICS does not change the record.
- If there is an entry on the JACD, but the journal data set name or the archive JCL PDS name has changed, CICS updates the record, and issues message DFHJC4539.
- If there is no entry on the JACD for the journal data set, CICS adds a new record.

Note: All added and updated records are given a status of READY.

CICS ***never*** deletes records from the JACD.

```
//DFHJACDC JOB accounting info,,
//   MSGCLASS=A,MSGLEVEL=(1,1),
//   CLASS=A,NOTIFY=userid
//DEFDS  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 /*                                                                 */
 /*     DEFINE THE JOURNAL ARCHIVE DATA SET                         */
 /*                                                                 */
        DEFINE CLUSTER                          -
             ( NAME(CICS410.applid.DFHJACD)     -
               NUMBERED                         -
               RECORDS( 199 )                   -
               RECORDSIZE( 505 505 )            -
               CONTROLINTERVALSIZE( 512 )       -
               SHAREOPTIONS( 4 )                -
               VOLUMES( CIC310 )  )             -
               DATA                             -
               ( NAME(CICS410.applid.DFHJACD.DATA) )
 /*                                                                 */
```

Figure 32. Sample job to define a journal archive data set

Block size warning when archiving

Do not try to change the block size of journal data sets when you copy them. CICS journal control writes a sequence number at the start of each block, and this is essential for the operation of the batch backout operation. If you reblock the journal data, the sequence number is not at the start of each block as before.

Defining a partitioned data set (JPDS) for the skeletal JCL members

The JPDS contains skeletal JCL for use by the journal archive submission program. The journal archive submission program, DFHJASP, uses the appropriate member for each journal, obtaining the member name from the ARCHJCL parameter in the JCT. The JPDS is a read-only data set to CICS, but it must be defined with DISP=SHR so that the data set can be updated from a batch or TSO environment during CICS execution.

You can use the sample skeletal JCL, supplied in member DFH$ARCH in CICS410.SDFHINST as a basis for writing your own archiving JCL.

You can use the sample job shown in Figure 33 to create a suitable JPDS data set and also copy into it the sample JCL. Alternatively, you can run the CICS-supplied job DFHCOMDS to create and load the DFHJPDS data set as one of the data sets common to all CICS regions. For information about the DFHCOMDS job, see the *CICS/ESA Installation Guide*.

Archiving system log data sets with CICS local DL/I

The sample skeletal JCL DFH$ARCH contains a dummy IEFBR14 job step for the journal data set copy step. If you are running CICS with local DL/I support, modify this to use DFSUARC0 to copy the system log data sets.

```
//DFHJNLCR JOB accounting information,
//   MSGCLASS=A,MSGLEVEL=(1,1),
//   CLASS=A,NOTIFY=USERID
//*********************************************************************
//* This job CREATES and LOADS the JCL PDS data set with DFH$ARCH,
//* the sample automatic archiving skeletal JCL supplied in the
//* CICS410.SDFHINST library.
//*********************************************************************
//DEFPDS   EXEC PGM=IEBUPDTE,PARM=MOD,
//             COND=(8,LT,DELJNLS)
//SYSUT1   DD DSN=CICS410.SDFHINST,DISP=SHR
//SYSUT2   DD DSN=CICS410.applid.DFHJPDS,DISP=(NEW,CATLG),VOL=SER=volid,
//         DCB=(DSORG=PO,RECFM=FB,LRECL=80,BLKSIZE=400),
//         UNIT=SYSDA,SPACE=(TRK,(2,,1))
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
./     REPRO   NAME=DFH$ARCH
/*
//
```

Figure 33. Sample job to copy CICS-supplied skeletal JCL

You can see from the sample skeletal JCL in DFH$ARCH, that the JCL you write for use by the automatic journal archive submission program, DFHJASP, should be written using symbolic parameters (for example, those defined with the percent symbol, like %JJ in the sample). These parameters enable you to use a single skeletal JCL source file to archive all the journals defined to your CICS regions. DFHJASP resolves the symbolic parameters into the correct values for the journal it is archiving at the time it submits the job to DFHJOUT.

For information about the sample skeletal JCL, the symbolic parameters that are available to you, and the journal archive utility program, DFHJACDU, see the *CICS/ESA Operations and Utilities Guide*.

Job control statements for CICS execution

If you specify JOUROPT=AUTOARCH, you must include a DD statement for the following CICS data sets in the CICS startup job stream:

DFHJACD The journal archive control data set.

DFHJPDS The journal partitioned data set containing the archive JCL.

DFHJOUT The sequential data set for the JCL output by the journal archive submission program.

For example:

```
//DFHJACD   DD  DSN=CICS410.applid.DFHJACD,DISP=SHR
//DFHJPDS   DD  DSN=CICS410.applid.DFHJPDS,DISP=SHR
//DFHJOUT   DD  SYSOUT=(A,INTRDR)
```

Although DFHJACD may also be accessed from an archive batch job, you do not need to specify any AMP subparameters on the DD statement.

Note that you should increase the number of internal readers defined by your JES parameters to allow for the number of readers assigned by DFHJOUT DD statements in concurrent CICS jobs.

Journal utility programs (DFHJUP and DFHJACDU)

CICS provides two journal utility programs:

DFHJUP
The CICS journal utility. You can use the DFHJUP utility program to select, print, or copy data held on CICS journal data sets.

DFHJACDU
If you are using the CICS automatic archiving facility, the DFHJACDU utility program is used to check, and then update, the journal archive control data set (JACD).

For information about these utility programs, see the *CICS/ESA Operations and Utilities Guide*.

Chapter 12. Defining the CICS system definition data set

This chapter describes how to define and initialize a system definition data set (CSD) that CICS needs to store definitions of the resources that it uses.

This chapter also discusses some considerations regarding the use of the CEDA transaction, particularly when a CSD is being shared by more than one CICS region.

You may have met 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. For information about DFHIVPBT and DFHIVPOL, see the *CICS/ESA Installation Guide*. However, the CSD created by the IVPs is limited in size, and initialized with the CICS-supplied resource definitions only.

A CSD is mandatory for some resource definitions. If you are creating a CSD for the first time, go through the steps listed below under "Summary of steps to create a CSD": The remainder of this chapter 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/ESA Version 4. For information about upgrading your CSD, see the *CICS/ESA Operations and Utilities 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.

Summary of steps to create a CSD

If you do not already have a CSD in use at your installation:

1. Decide how much disk space you require.
2. Decide whether the CSD is to be eligible for backup-while-open (BWO[10]). If so, you must install the following software products:
   - MVS/DFP 3.2
   - DFHSM 2.5
   - DFDSS 2.5

   If the CSD data set is to be eligible for BWO, it must be defined on SMS-managed DASD and must have an ICF catalog. If the CSD is eligible for BWO you must code the CSDBKUP=DYNAMIC system initialization parameter.
3. Define and initialize the CSD.
4. Decide what CICS file processing attributes you want for your CSD, and code these in as system initialization parameters.
5. Decide what backup and recovery procedures you require for your CSD. If you decide to make the CSD a CICS recoverable resource, code the CSDRECOV operand accordingly. You must also specify support for dynamic transaction backout, by coding the DBP system initialization parameter.

[10] Eligibility for BWO means that DFHSM 2.5 and DFDSS 2.5 can back up the CSD while the data set is open for update.

6. Decide if you want to use command logs for RDO; see “RDO command logs” on page 154 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 including the necessary DD statement in the CICS startup job stream or by using dynamic allocation.

   When you have started CICS, test the RDO transactions CEDA, CEDB, and CEDC. For information about these transactions, see the *CICS/ESA Resource Definition Guide*.
8. Finally, if you want to restrict access to particular CICS-supplied transactions by applying security, create a new group list (to use instead of the IBM-defined list, DFHLIST). This list should contain the resource definitions for the transactions CEMT, CEST, CECI, CEDA, CEDB (and any others that you think need to be secure) with the transaction security values appropriate to your installation. For information about how to do this, see the *CICS/ESA Resource Definition Guide*.

When you are certain that RDO is installed correctly, you can then plan to migrate your CICS tables to the CSD. The PCT, PPT, and FCT (for all but BDAM files to be loaded on a CICS cold start) are obsolete in CICS/ESA Version 4, and these must be migrated to your CSD. You cannot migrate PCTs and PPTs at CICS/ESA 4.1; you must do so at your earlier release of CICS. (For more information, see Chapter 2, “Defining resources in CICS control tables” on page 7.)

You define file control resource definitions for the CSD by specifying CSDxxxxx system initialization parameters, which are described in Chapter 21, “CICS system initialization parameters” on page 211.

For information about migrating CICS control tables to RDO using the MIGRATE command, see the *CICS/ESA Operations and Utilities Guide*.

Calculating disk space

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

- Each resource definition (for example each program, transaction and terminal) needs one record; the sizes of these definition records are:

Resource	Definition record size
Program	144 bytes
Transaction	168 bytes
Profile	156 bytes
Partition set	146 bytes
Map set	146 bytes
Terminal	313 bytes
Typeterm	336 bytes
Connection	184 bytes
Session	260 bytes
File	264 bytes
LSR pool	182 bytes

- Each group requires two 122-byte records.
- Each group list requires two 122-byte records.
- Each group name within a list requires one 68-byte record.

In your calculation, allow for approximately 565 CICS-supplied resource definitions of various types, which are loaded into the CSD when you initialize the CSD with the utility program, DFHCSDUP. Finally, add a suitable contingency (approximately 25%), and use your calculated figure when you define the VSAM cluster for the CSD. (See the sample job in Figure 34.)

Defining and initializing the CICS system definition

Before you can use the CSD, you must define it as a VSAM KSDS data set, and initialize it using the DFHCSDUP utility program. (See Figure 34.)

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.

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

```
//DEFINIT  JOB  accounting information
//DEFCSD   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//AMSDUMP  DD   SYSOUT=A
//SYSIN    DD   *
   DEFINE CLUSTER -
            (NAME(CICS410.applid.DFHCSD) -
            VOLUMES(volid)               -
            KEYS(22 0)                   -                      1
            INDEXED                      -
            RECORDS(n1 n2)               -
            RECORDSIZE(120 500)          -                      2
            FREESPACE(10 10)             -
            SHAREOPTIONS(2))             -                      3
          DATA                           -
            (NAME(CICS410.applid.DATA.DFHCSD)   -
            CONTROLINTERVALSIZE(8192))          -
          INDEX                                 -
            (NAME(CICS410.applid.INDEX.DFHCSD))
/*
//INIT     EXEC PGM=DFHCSDUP,REGION=300K
//STEPLIB  DD DSN=CICS410.SDFHLOAD,DISP=SHR
//DFHCSD   DD DSN=CICS410.applid.DFHCSD,DISP=SHR                 4
//SYSPRINT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//SYSIN    DD *
           INITIALIZE
           LIST ALL OBJECTS
/*
//
```

Figure 34. Sample job to define and initialize the CSD

Notes:

1 The key length is 22 bytes, and the KEYS parameter must be coded as shown.

2 The average record size of 120 bytes is calculated 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 disk space" on page 140.

3 Code the SHAREOPTIONS parameter as shown.

4 The DDNAME for the CSD must be DFHCSD.

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 CEDA 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* manual.

File processing attributes for the CSD

File processing attributes for the CSD must be supplied as the following system initialization parameters:

CSDACC The type of access allowed
CSDBKUP Specify whether the CSD is eligible for BWO[11]
CSDBUFND The number of buffers for CSD data
CSDBUFNI The number of buffers for the CSD index
CSDDISP The disposition of the CSD data set
CSDDSN The JCL data set name (DSNAME) of the CSD
CSDFRLOG A forward recovery journal identifier
CSDJID An identifier for automatic journaling
CSDLSRNO A VSAM local shared resource pool
CSDRECOV Whether or not the CSD is recoverable
CSDSTRNO The number of strings for concurrent requests.

These parameters are described in greater detail in Chapter 21, "CICS system initialization parameters" on page 211.

[11] Eligibility for BWO means that DFHSM 2.5 and DFDSS 2.5 can back up the CSD while the data set is open for update.

Sharing and availability of the CSD

This section describes considerations that affect how you can implement the sharing of a CSD. 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.

To optimize the sharing of a CSD, you should observe the following considerations:

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 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.

For more information, see “Multiple users of the CSD within a CICS region” on page 144.

Shared user access from several CICS regions

- Several users in different CICS regions can access the CSD at the same time
- Only one CICS region should be given read/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.
- 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.
- 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:

  1. 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.
  2. Define the CSD as read-only to other CICS regions.
  3. For all regions other than the CSD-owning region:
     a. Redefine the CEDB transaction as a remote transaction (to be run in the CSD-owning region).
     b. 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.

- 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.
- 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 sample job stream on page 141.
- 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.

For more information about shared CSD access within one MVS image, see "Sharing a CSD by CICS regions within a single MVS image" on page 145. For more information about shared CSD access across several MVS images, see "Sharing a CSD in a multi-MVS environment" on page 146. For information about sharing the CSD between CICS/ESA 4.1 and earlier releases, see "Sharing the CSD between CICS/ESA 4.1 and earlier releases" on page 148.

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 149.

For information about the system initialization parameters for controlling access to the CSD, see "File processing attributes for the CSD" on page 142.

Multiple users of the CSD within a CICS region

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

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 sample job stream on page 141. 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 on page 143.

  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 CSDLSRNO=NONE, and the minimum values for CSDBUFNI and CSDBUFND, but at the expense of degraded performance.

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 150 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 system initialization parameters allow the following operations on the CSD:

CSDACC operand	*Operations*
READONLY	Read and browse.
READWRITE	Add, delete, update, read and browse.

Sharing a CSD in a multi-MVS environment

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 enqueuing (for example, with global resource serialization (GRS)).

These multi-MVS restrictions also apply to running the offline utility, DFHCSDUP. See Figure 35 on page 147 for examples of CSD usage in a multi-MVS environment.

Note: In Figure 35 on page 147, you can use a single CSD in place of the individual CSDs shown on MVS image CEC1, because they are all on the same MVS image. However, without GRS or some other form of global enqueuing, you cannot share a CSD that is shown on shared DASD.

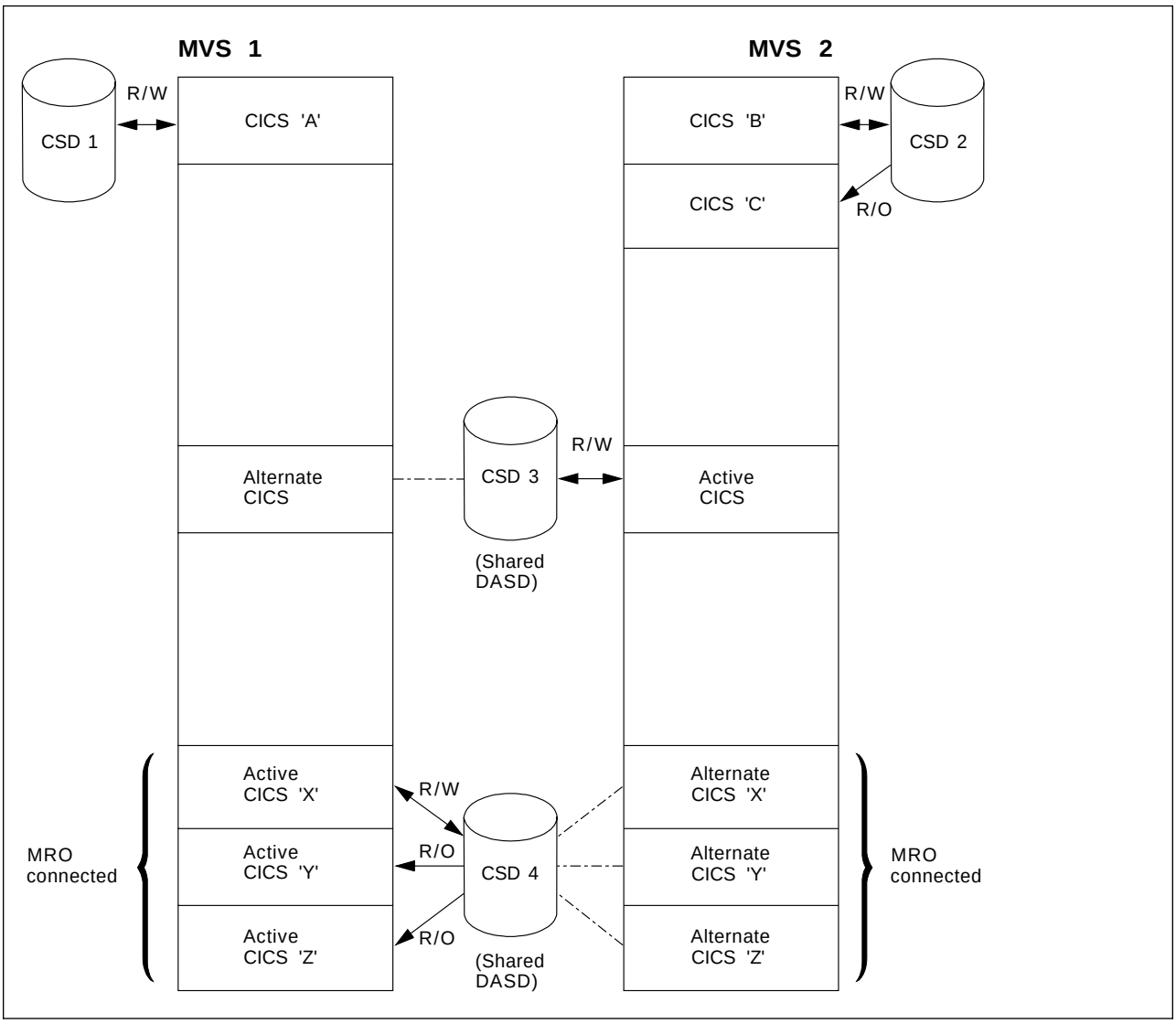


Figure 35. Some examples of CSD usage in a multi-MVS environment

Multiple users of one CSD across CICS or batch regions

The types of access needed in the four situations where the CSD may be used are shown in Table 20.

Table 20. CSD access

	Type of Activity	Access
1	CICS region performing initialization (cold 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
4	CICS regions performing emergency restart, and file backout is required	Read/write

Note the following limitations when the activities listed in Table 20 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 CEDx transactions are running in another CICS region that has the CSD defined for read-write access.

A CICS region starting with a cold start opens the CSD for read access only, 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.

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 CICS/ESA 4.1 and earlier releases

Resource attributes become obsolete when they have no relevance for a new release of CICS. In CICS/ESA 4.1, CICS continues to display these on CEDx panels, but they are displayed as protected fields, indicating that they are not supported by CICS/ESA 4.1. Using the ALTER command on definitions that specify obsolete attributes does not cause the loss of these attributes in CICS/ESA 4.1, so you can safely update resource definitions from a CICS/ESA 4.1 region. If you are sharing the CSD between a CICS/ESA 4.1 region and a

| CICS/ESA Version 3 or CICS/MVS Version 2 region, 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 can 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 higher level region, that is, CICS/ESA 4.1.

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

You can also use the CICS/ESA 4.1 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 on 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.

| If you share your CSD between CICS/ESA 4.1 and earlier releases of CICS that
| use DB2, you must use separate group lists for the different CICS releases. That
| is, the CICS/ESA 4.1 DB2 definitions must be in a different group list to the DB2
| definitions for earlier releases.

| For information about sharing the CSD between CICS releases, see the *CICS/ESA*
| *Migration Guide*.

CICS supplied compatibility groups

If you are sharing the CSD between CICS/ESA 4.1 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/ESA 4.1 level, some of the IBM groups referenced by your group list are deleted, and the contents transferred to one of the compatibility groups, DFHCOMP1, DFHCOMP2, or DFHCOMP3. 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/ESA 4.1, see the *CICS/ESA 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” on page 144, 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.”

Planning for backup and recovery

To guard against system failures that affect your CSD, take a backup copy of the CSD at regular intervals. Then, if the CSD is corrupted for any reason, you can restore it to its state at the last backup. To keep the backup of the CSD as up-to-date as possible, make an image copy of your CSD before each CEDA update activity.

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, a BWO of the CSD can be restored via DFHSM and DFDSS, then forward recovered to the point of corruption via a forward recovery utility.

If activity keypointing is disabled in your CICS region (by specifying the system initialization parameter AKPFREQ=0), this has a serious affect on BWO support. For more information about the effect of AKPFREQ=0 on BWO, see “Effect of disabling activity keypointing” on page 108.

You can use the system initialization parameter CSDBKUP=DYNAMIC|STATIC to define the CSD as eligible for BWO. Specify CSDBKUP=DYNAMIC for BWO support or STATIC (the default) for a “normal” quiesced backup. If you specify CSDBKUP=DYNAMIC, CSDRECOV=ALL is also required. For more information about BWO, see “Backup while open (BWO) of VSAM files” on page 106.

If you code CSDRECOV=ALL as a system initialization parameter, all changes (after images) made by CICS to the CSD are logged in the CICS journal defined by the CSDFRLOG system initialization parameter. If you code CSDRECOV=ALL, but omit CSDFRLOG, CICS uses the system log as the journal for CSD recovery. Using the latest backup copy and the after images from the relevant CICS journal, 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 the 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 “RDO command logs” on page 154 for more information.

The CSDBKUP, CSDRECOV, and CSDFRLOG system initialization parameters interact according to how they are specified. Table 21 and Table 22 on page 152 summarize their effects when the SIT is assembled and during CICS override processing, respectively.

Table 21. CSDBKUP and related parameters at SIT assembly time

CSDRECOV	CSDFRLOG	CSDBKUP [1]	Result
ALL	FRLOG from 01 through 99.	Either DYNAMIC or STATIC	OK
ALL	NO	Either DYNAMIC or STATIC	CSDFRLOG defaults to 01 — the system log.
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 requires CSDRECOV=ALL.

Note:

1 When CSDBKUP=DYNAMIC, the CSD is eligible for BWO.

Table 22. CSDBKUP and related system initialization parameters during CICS override processing			
CSDRECOV	**CSDFRLOG**	**CSDBKUP [1]**	**Result**
ALL	FRLOG from 01 through 99.	Either DYNAMIC or STATIC.	OK
ALL	NO	Either DYNAMIC or STATIC.	CSDFRLOG defaults to 01 — the system log.
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.
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.

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. This means that you must include support for dynamic transaction backout by coding the DBP system initialization parameter. To do this, code:

DBP=1$, if you have ***not*** installed CICS local DL/I support

DBP=2$, if you have installed CICS local DL/I support.

Other recovery considerations

When you are deciding what to code on the CSDRECOV parameter, consider the following factors:

- 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 and availability of the CSD" on page 143: For information about using the DFHCSDUP utility to access the CSD, see "Accessing the CSD by the offline utility program, DFHCSDUP" on page 154.

CEDA command syncpoint criteria

You can issue CEDA in two ways:

1. A single command entered on the command line
2. A series of single commands from within 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, 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.

RDO command logs

If you want to record RDO commands, specify the DCT entries for the CADL, CAIL, CRDI, CSDL, CSFL, CSKL, CSPL, and CSRL extrapartition transient data destinations that CICS can use as the logs. These 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 of 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.

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, code DCT entries shown in Figure 36 on page 155. (These definitions are included in the sample DCT copy member, DFH$DCTR.) If you like, you can direct these logs to any other transient data queue, or define them as extrapartition data sets.

```
CADL   DFHDCT TYPE=INDIRECT,     CEDA VTAM RESOURCE LOGGING          X
             DESTID=CADL,                                            X
             INDDEST=CSSL
*
CAIL   DFHDCT TYPE=INDIRECT,     AUTOINSTALL TERMINAL MODEL LOGGING
             DESTID=CAIL,                                            X
             INDDEST=CSSL
*
CRDI   DFHDCT TYPE=INDIRECT,      RDO INSTALL LOGGING                X
             DESTID=CRDI,                                            X
             INDDEST=CSSL
*
CSDL   DFHDCT TYPE=INDIRECT,     CEDA COMMAND LOGGING                X
             DESTID=CSDL,                                            X
             INDDEST=CSSL
*
CSFL   DFHDCT TYPE=INDIRECT,     FILE ALLOCATION MESSAGES            X
             DESTID=CSFL,                                            X
             INDDEST=CSSL
*
CSKL   DFHDCT TYPE=INDIRECT,     TRANSACTION+PROFILE RESOURCE LOGGING X
             DESTID=CSKL,                                            X
             INDDEST=CSSL
*
CSPL   DFHDCT TYPE=INDIRECT,     PROGRAM RESOURCE LOGGING            X
             DESTID=CSPL,                                            X
             INDDEST=CSSL
*
CSRL   DFHDCT TYPE=INDIRECT,     PARTNER RESOURCE LOGGING            X
             DESTID=CSRL,                                            X
             INDDEST=CSSL
```

Figure 36. DCT entries for RDO command logs sent to CSSL

You must specify the block size, record size, and record format for these destinations in the DFHDCT TYPE=SDSCI macro. For example:

```
MSGUSR   DFHDCT TYPE=SDSCI,       CICS MESSAGES AND RDO LOGS        X
               BLKSIZE=136,                                          X
               BUFNO=1,                                              X
               DSCNAME=MSGUSR,                                       X
               RECFORM=VARUNB,                                       X
               RECSIZE=132,                                          X
               TYPEFLE=OUTPUT
```

Figure 37. Example DFHDCT TYPE=SDSCI statement for the RDO command logs

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.

- You can include the following DD statement in your CICS startup job stream:

  ```
  //DFHCSD  DD  DSN=CICS410.applid.DFHCSD,DISP=SHR
  ```

  You usually need the CSD DD statement to include DISP=SHR. (See "Sharing and availability of the CSD" on page 143.)

  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***.

- You may prefer to take advantage of the CICS dynamic allocation of the CSD. If so, 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. To dynamically allocate the CSD, specify the data set name (DSNAME) and disposition (DISP) of the CSD, using one of the following methods:
  - The system initialization parameters CSDDSN and CSDDISP
  - The command CEMT SET FILE
  - The command EXEC CICS SET FILE.

  CICS then 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.

  For more information about OPEN processing, see Chapter 18, "Defining user files" on page 189. For information about the parameters that you can code for the CSD in the SIT, see Chapter 21, "CICS system initialization parameters" on page 211.

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 the *CICS/ESA Resource Definition Guide*.

Moving your CICS tables to the CSD

When you have created a CSD, and initialized it with CICS-supplied definitions, you are ready to move the contents of your CICS tables to the CSD data set. You do this by running the offline utility, DFHCSDUP, using the MIGRATE command to specify the table you want to migrate. For information about the DFHCSDUP utility program and the available commands, see the *CICS/ESA Operations and Utilities Guide*. For information about moving the contents of CICS tables to the CSD, see the *CICS/ESA Resource Definition Guide*.

Release compatibility of the CSD

If you are using a CSD with an earlier release of CICS, upgrade your CSD as part of the process of migration. For information about upgrading the CSD, and about the release compatibility of the CSD after upgrading, see the *CICS/ESA Migration Guide*.

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 and specifying:

```
UPGRADE USING(DFHRDJPN)
```

For information about the DFHCSDUP utility program and the available commands, see the *CICS/ESA Operations and Utilities Guide*.

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). 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 page 104.)

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 “File processing attributes for the CSD” on page 142.

For more information about using the CSD as a recoverable file, see “Planning for backup and recovery” on page 150.

Chapter 13. Defining and initializing the restart data set

This chapter describes the restart data set and shows you how to define and initialize it.

The restart data set is a VSAM key-sequenced data set (KSDS). It is used during emergency restarts only, to temporarily hold the backout information read from the CICS system log. However, the restart data set must be available whenever CICS starts up; that is, you should always include a data definition statement for the restart data set in your CICS startup job stream.

Job control statements to define and initialize the restart data set

Before its first use, you must define and initialize the restart data set as a VSAM key sequenced data set (KSDS). To do this, you can use the sample job in Figure 38 on page 160. Alternatively, you can run the CICS-supplied job DFHDEFDS to create the restart data set as one of the data sets for a CICS region. For more information about the DFHDEFDS job, see the *CICS/ESA Installation Guide*.

The name in the CLUSTER definition must be the same as the DSN parameter in the DD statement for the restart data set in the CICS startup job stream. In Figure 38 on page 160, job step INITRSD initializes the data set with one record containing a 22-byte key, using the DFHINST0 member installed in the CICS410.SDFHINST library and specified by the RSDREC DD statement. You can use any value for the key, but a low-value (X'00') key gives the best performance when the data set is being used for an emergency restart. If you decide to use a low-value key, create a suitable record in a partitioned data set for use as the input file in the AMS REPRO job step.

Job control statements for CICS execution

If you defined the restart data set using the sample job shown in Figure 38 on page 160, the data definition statement for the CICS execution is:

```
//DFHRSD  DD  DSN=CICS410.applid.DFHRSD,DISP=OLD
```

Note: The restart data set is a passively shared data set when running CICS with XRF, so specify DISP=SHR in the startup job streams for the active and alternate CICS regions.

```
//DEFVSAM  JOB accounting info,,CLASS=A
//DEFRSD   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
      DEFINE CLUSTER(NAME(CICS410.applid.DFHRSD) -
               KEYS(22 0)-
               INDEXED -
               RECORDSIZE(400 2000)-
               RECORDS(100 100)-
               CONTROLINTERVALSIZE(2048)-
               SHAREOPTIONS(2)-
               VOLUMES(volid)) -
             DATA(NAME(CICS410.applid.DFHRSD.DATA)-
               UNIQUE)-
            INDEX(NAME(CICS410.applid.DFHRSD.INDEX)-
               UNIQUE)
//*
//INITRSD   EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT  DD SYSOUT=*
//RSD       DD DSN=CICS410.applid.DFHRSD,DISP=SHR
//SYSIN     DD *
  REPRO INFILE(RSDREC) OUTFILE(RSD)
/*
//RSDREC    DD DSN=CICS410.SDFHINST(DFHINST0),DISP=SHR
//*
```

Note: The RSD step is used to initialize the restart data set with low-values, using the DFHINST0 member installed in the CICS410.SDFHINST library and specified by the RSDREC DD statement.

```
//SYS01    DD DSN=CICS410.INITRSD(LOVALUES),DISP=SHR
//SYSIN    DD   *
       REPRO             -
           INFILE(SYS01) -
           OUTFILE(SYS02)
/*
//
```

Figure 38. Sample job to define and initialize the restart data set

Chapter 14. Defining and using catalog data sets

This chapter describes how to define and use the CICS **global** catalog data set (GCD), and the CICS **local** catalog data set (LCD) which CICS needs to catalog CICS system information. For the rest of this chapter, 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.)

Warning: Defining and initializing the CICS catalogs

You must define and initialize new CICS catalogs for CICS/ESA 4.1.

If, for any reason, you need to reinitialize one of the catalogs while running CICS, reinitialize them both. If you reinitialize only one of the catalogs, CICS fails on any subsequent restart.

For more information about how CICS uses the catalogs for startup and restart, see "The role of the CICS catalogs" on page 329.

The global catalog

The global catalog is a VSAM key-sequenced data set (KSDS). In an XRF environment there is only one global catalog. It is shared passively between the active and the alternate CICS regions. The global catalog is used:

- During a controlled shutdown, it is used to record warm keypoint information for a subsequent warm start.
- During the running of CICS, it holds the resource definitions that are ***installed***, either during initialization when CICS installs the group list, or by means of the RDO CEDA INSTALL command. These definitions can be for:
  - Programs
  - Transactions
  - Transaction classes
  - Mapsets
  - Terminals, including any that are autoinstalled
  - Sessions and connections, for communication with other CICS regions
  - Files.
- To hold the volume identifiers of any CICS journals written on standard-labeled tapes.
- To hold DL/I status information.
- To hold disk journal status information.

For further guidance information about what is written to the global catalog, and about how CICS uses the global catalog for startup and restart, see "The role of the CICS catalogs" on page 329.

Job control statements to define and initialize the global catalog

Before its first use, you must define and initialize the CICS global catalog as a KSDS. You can use the sample job in Figure 39 to do this, or you can run the CICS-supplied job, DFHDEFDS, to define and initialize the global catalog data set as one of the data sets for a CICS region. For information about the DFHDEFDS job, see the *CICS/ESA Installation Guide*.

In Figure 39, the job step INITGCD initializes the data set with one record containing a 28-byte key. You can also use the job step to reinitialize the global catalog without redefining it. You can use any value for the key, but generally it is best to use a key comprising all low-values (X'00'). If you use a key of low-values, create a suitable record in a partitioned data set for use as the input file in the AMS REPRO job step.

```
//GLOCAT   JOB accounting info,,CLASS=A
//DEFGCD   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
      DEFINE CLUSTER -
               (NAME(CICS410.applid.DFHGCD) -                    1
               INDEXED                      -
               CYLINDERS(n1 n2))            -                    2
               FREESPACE(10 10)             -
               SHAREOPTIONS(2)              -
               REUSE                        -                    5
               VOLUMES(volid))              -
             DATA                           -                    3
               (NAME(CICS410.applid.DATA.DFHGCD.)  -
               CONTROLINTERVALSIZE(8192)    -                    4
               KEYS(28 0))                  -
             INDEX                          -
               (NAME(CICS410.applid.INDEX.DFHGCD) -
               IMBED                        -
               REPLICATE)
/*
//INITGCD  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYS02    DD DSN=CICS410.applid.DFHGCD,DISP=SHR
```

Note: If you want to load a dummy record with a key comprising all low-values, use a DD statement for SYS01 that defines a PDS which contains a suitable dummy record. In this example, the PDS member, LOVALUES, must contain only one record with the first 28 characters defined as low-values (X'00').

```
//SYS01    DD DSN=CICS410.INITGCD(LOVALUES),DISP=SHR
//SYSIN    DD   *
       REPRO -
              INFILE(SYS01) -
              OUTFILE(SYS02) REUSE                               5
/*
//
```

Figure 39. Sample job to define and initialize the global catalog

Notes:

1 The data set name in the CLUSTER definition must 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's needs, and substitute your values for n1 and n2.

Whichever IDCAMS parameter you use for the GCD space allocation (CYLINDERS, TRACKS, or RECORDS), make sure that you specify a secondary extent. CICS abends if your GCD fills and VSAM cannot create a secondary extent.

3 This job does not specify a RECORDSIZE value for the global catalog, which therefore defaults to using an average and maximum record size of 4089 bytes, RECORDSIZE(4089 4089). If your maximum record size is greater than 4089, you must add a RECORDSIZE parameter to the sample job to specify your own value.
For information about record sizes, see Table 23 on page 165

4 You can vary the CONTROLINTERVALSIZE from the values shown in the VSAM definition. However, although larger values reduce the number of control interval (CI) and control area (CA) splits, other factors increase CICS shutdown times, and slow down a cold start.

This job stream does not specify a BUFFERSPACE parameter, although you can code an explicit value if you want to define buffers of a specific size. BUFFERSPACE is the minimum bufferspace permitted; VSAM defaults to a bufferspace value equal to twice the CI size of the data component, plus the CI size of the index, which gives a default of 20480 bytes in the example job. A larger minimum buffer size (bufferspace) may improve cold start and warm restart times, and may significantly reduce CICS shutdown times.

Another way to define buffer space for the GCD is by means of the AMP parameter on the DD statement for the GCD in the CICS startup job stream, which you can use to override the default or defined value. (Note, however, that the BUFSP parameter defines the maximum bufferspace. If you define a BUFFERSPACE value on the AMP parameter that is smaller than the BUFFERSPACE value specified in the DEFINE statement, the BUFFERSPACE value takes precedence.

For performance reasons, CICS defines a STRNO (number of strings) value of 32. Based on the example job stream in Figure 39 on page 162, the absolute minimum value of BUFSP is calculated as follows:

```
BUFND = (STRNO + 1)
BUFNI = STRNO
BUFSP = 33 * 8192 (BUFND * CI size) + 32 * 1024  (BUFNI * CI size) = 303104 bytes
```

Note: This is the smallest figure that can be used for BUFSP.

The principal factors affecting CICS startup and shutdown times are the number of resources defined in CICS tables, and the number of resources defined in the group list for those definitions managed by RDO.

5 To enable the global catalog to be opened again and again as a reusable cluster, you must specify the REUSE option on the DEFINE CLUSTER command, and the REPRO command used to re-initialize the global catalog.

Reusing the global catalog to perform a cold start

If you need to clear the catalog to perform a completely cold start, you can avoid deleting and redefining the data set by specifying the REUSE option on the cluster definition. If you specify REUSE on the cluster definition, you can include in the CICS cold start-job stream a job step to re-initialize the global catalog, as shown in the INITGCD job step in Figure 39 on page 162. To re-initialize the global catalog without redefining it, you must specify the REUSE option on the DEFINE CLUSTER command, and on the REPRO command of the INITGCD job step, as shown.

Be aware, however, that a start initiated by START=COLD is not entirely without reference to the previous run of a CICS system using the same global catalog. For example, if you are running CICS with local DL/I support, the global catalog may contain extended error queue elements (EEQEs) relating to DL/I write failures. This information is normally preserved across a cold start, but if you re-initialize the global catalog it is lost. For more information about the use of the global catalog in a cold start of CICS, see “Classes of start and restart” on page 329.

Space calculations

Each global catalog keypoint record has a 28-byte key.

To estimate the amount of space needed in your global catalog to keypoint installed resource definitions, table entries, and control blocks, use the sizes specified in Table 23 on page 165.

Each entry is one VSAM record, and the records for each type of table have different keys.

There are also records for each of the standard-labeled tape journal series that are keypointed during warm shutdown, and the restart control record.

PQ00144

The following change was made by APAR PQ00144.

+ The space requirements for a VSAM KSDS such as DFHGCD can vary for different CICS cold starts. This can occur even if no changes have been made to the CICS definitions to be stored on the VSAM KSDS. This is because VSAM will utilize 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 will call 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, 'freespace' is honored only once a CI split has occurred.

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

Table 23. Sizes for entries in the global catalog

Installed definition, table entry, or control block	***Number of bytes per entry***
Installed PARTNER definitions	96 bytes
Installed program definitions	44 bytes
Installed TRANSACTION definitions (without TPNAME)	112 bytes
Installed TRANSACTION definitions (with TPNAME or XTPNAME)	176 bytes
Installed VSAM file (or data table) definition	172 bytes
Installed TRANCLASS definitions	8 bytes
BDAM file control table entry (FCT)	96 bytes
Data set names (JCL or dynamically allocated)	52 bytes
DL/I extended error queue elements (EEQEs)	132 bytes
Terminal control table entry (TCT)	
Destination control table entry (DCT)	96 bytes
Dump table entry	48 bytes
Interval control element (ICE)	68 bytes
Automatic initiator descriptor (AID)	68 bytes
Temporary storage destination record	18 bytes
Temporary storage destination auxiliary record	6 bytes
Installed TYPETERM definitions **1**	530 bytes
Installed model TERMINAL definitions **1**	530 bytes
Volume control information for labeled tapes	44 bytes
Unit of recovery descriptors	200 bytes
Deferred work element (DWE) **2**	80 bytes

Notes:

1 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 via a group list. For example, if you start up 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. Allow space in your calculations for all autoinstall resources installed in your CICS region.

2 The value given is for a DWE chained off a unit of recovery descriptor, an LU6.1 TCTTE, or an APPC TCTTE.

Job control statement for CICS execution

If you define the global catalog using the sample job in Figure 39 on page 162, the data definition statement for the CICS execution is:

```
//DFHGCD  DD  DSN=CICS410.applid.DFHGCD,DISP=OLD
```

This is a minimum specification for a global catalog for use by a single CICS region. Add the relevant AMP subparameters to help improve restart and shutdown time. The AMP parameter is described in the *MVS/ESA JCL Reference* manual, and an example is shown in the CICS startup job stream in Chapter 23, “CICS startup” on page 337.

If you are running CICS with XRF, the global catalog is passively shared by the active and alternate CICS regions, and you must specify DISP=SHR.

The local catalog

CICS/ESA Version 4 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. The kernel, the domain manager, and the other domains all 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 “Classes of start and restart” on page 329.

The local catalog is a VSAM key-sequenced data set (KSDS). It 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.

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.

Information written to the local catalog

Before the local catalog can be used to bring up a CICS region, it must be initialized with the following data:

- The 16 domain manager parameter records, each of which contains information relating to one of the CICS domains. These records are identified by their domain names, which are:

Domain name in catalog	Description
DFHAP	Application domain
DFHCC	CICS local catalog domain
DFHDD	Directory manager domain
DFHDE	Distributed computing environment domain
DFHDM	Domain manager domain
DFHDS	Dispatcher domain
DFHDU	Dump domain
DFHGC	CICS global catalog domain
DFHKE	Kernel domain
DFHLD	Loader domain
DFHLM	Lock manager domain
DFHME	Message domain
DFHMN	Monitoring domain
DFHPA	System initialization parameter domain
DFHPG	Program manager domain
DFHSM	Storage manager domain
DFHST	Statistics domain
DFHTI	Timer domain
DFHTR	Trace domain
DFHUS	User domain
DFHXM	Transaction manager domain
DFHXS	Security domain.

- Four loader domain parameter records, which contain information relating to:
  - DFHDMP, the CSD file manager
  - DFHEITSP, the RDO language definition table
  - DFHOCP, the dynamic open/close program
  - DFHPUP, the CSD parameter utility program.

To enable you to initialize the local catalog correctly, with all the records in the correct sequence, there is a CICS-supplied utility called DFHCCUTL that you run immediately after you have defined the VSAM data set.

In addition to the information written to the local catalog when you first initialize it, the loader domain writes a program definition record for each CICS nucleus module. The number of records varies depending on the level of function you have included in your CICS region (for example, CICS local DL/I support adds a few more records). Allow for at least 75 of these loader-domain records.

Some domains also 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:

- Storage manager domain
- Dispatcher domain
- Parameter manager domain
- Message domain
- Dump domain.

Apar PQ07674

Documentation for Apar PQ07674 added 26/08/98

You can add records to the local catalog to enable the CICS self-tuning mechanism for storage manager domain subpools. For details of how to do this using the CICS-supplied utility program, DFHSMUTL, see the *CICS/ESA Operations and Utilities Guide*.

Finally, when you define the VSAM cluster for the local catalog, specify a secondary extent value as a contingency allowance. See the sample job in Figure 40.

Job control statements to define and initialize the local catalog

Before its first use, you must define and initialize the CICS local catalog as a VSAM key sequenced data set (KSDS). To do this, you can use the sample job in Figure 40. Alternatively, you can run the CICS-supplied job DFHDEFDS to create a local catalog for an active CICS region or the CICS-supplied job DFHALTDS to create a local catalog for an alternate CICS region. For information about the jobs DFHDEFDS and DFHALTDS, see the *CICS/ESA Installation Guide*.

```
//LOCAT    JOB accounting info,,CLASS=A
//DEFLCD   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
//*
         DEFINE CLUSTER -
                (NAME( CICS410.applid.DFHLCD)  -                  1
                 INDEXED                       -
                 RECORDS( 200 10 )             -                  2
                 FREESPACE(10 10)              -
                 SHAREOPTIONS( 2 )             -
                 REUSE                         -
                 VOLUMES( volid ))             -
```

Figure 40 (Part 1 of 2). Sample job to define and initialize the local catalog

```
          DATA                                  -
               (NAME( CICS410.applid.DATA.DFHLCD )   -
                KEYS( 28 0 )                     -
                RECORDSIZE( 45 124 )             -                  3
                CONTROLINTERVALSIZE( 2048 ))  -
          INDEX (NAME( CICS410.applid.INDEX.DFHLCD ) -
                 IMBED                           -
                 REPLICATE )
/*
//*************************************************************
//INITLCD EXEC PGM=DFHCCUTL
//*
//*            INITIALIZE THE CICS LOCAL CATALOG
//*
//STEPLIB   DD DSN=CICS410.SDFHLOAD,DISP=SHR
//SYSPRINT  DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=*
//DFHLCD    DD DSN=CICS410.applid.DFHLCD,DISP=SHR
//*
//
```

Figure 40 (Part 2 of 2). Sample job to define and initialize the local catalog

Notes:

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

```
DEFINE CLUSTER -
       (NAME( CICS410.DBDCCIC1.DFHLCD)
  ⋮
DEFINE CLUSTER -
       (NAME( CICS410.DBDCCIC2.DFHLCD)
  ⋮
```

2 Space for about 200 records should be adequate for the local catalog, but also specify space for secondary extents as a contingency allowance.

3 The local catalog records are small by comparison with the global catalog. Use the record sizes shown, which, in conjunction with the number of records specified, ensure enough space for the data set.

Job control statement for CICS execution

If you define the local catalog using the sample job in Figure 40 on page 168, the data definition statement for the CICS execution is:

```
//DFHLCD  DD  DSN=CICS410.applid.DFHLCD,DISP=OLD
```

Chapter 15. Defining and using auxiliary trace data sets

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

There are several types of tracing available in CICS to help you with problem determination, and these are described in the *CICS/ESA Problem Determination Guide*. 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 *MVS/ESA Service Aids* manual. For information about using CICS tracing for problem determination, see the *CICS/ESA Problem Determination Guide*.

Defining auxiliary trace data sets

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 DFHDFHTU410 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 may 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. 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.
TRTABSZ	Defines the size of the CICS internal trace table.
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.
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 21, “CICS system initialization parameters” on page 211.

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 guidance information about the CICS control options available with CETR and CEMT, see the *CICS/ESA CICS-Supplied Transactions* manual.

Job control statements to allocate auxiliary trace data sets

If you are defining auxiliary trace data sets on disk, you can use the job shown in Figure 41 to allocate and catalog them before running CICS.

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 DFHDEFDS and DFHALTDS, see the *CICS/ESA Installation Guide*.

```
//DEFTRCDS JOB (accounting information),
//             MSGCLASS=A,MSGLEVEL=(1,1),
//             CLASS=A,NOTIFY=userid
//*******************************************************************
//*            Create auxiliary trace data sets
//*******************************************************************
//ALLOCDS  EXEC PGM=IEFBR14
//DFHAUXT  DD  DSN=CICS410.applid.DFHAUXT,UNIT=3380,VOL=SER=volid,
//         DISP=(NEW,CATLG),DCB=(BLKSIZE=4096,RECFM=F,LRECL=4096),  1
//              SPACE=(CYL,(5,1))                                   2
//DFHBUXT  DD  DSN=CICS410.applid.DFHBUXT,UNIT=3380,VOL=SER=volid,
//         DISP=(NEW,CATLG),DCB=(BLKSIZE=4096,RECFM=F,LRECL=4096),  1
//              SPACE=(CYL,(5,1))                                   2
//
```

Figure 41. Sample job to define auxiliary trace data sets on disk

Notes:

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.

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 as shown in Figure 41 on page 173, you can define them to CICS in the startup job stream with the following DD statements:

```
//DFHAUXT  DD  DSN=CICS410.applid.DFHAUXT,DCB=BUFNO=n,DISP=SHR
//DFHBUXT  DD  DSN=CICS410.applid.DFHBUXT,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 DFHDFHTU410 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:

```
//DFHAUXT  DD  DSN=CICS410.applid.DFHAUXT,UNIT=3400,VOL=SER=volid,
//         DISP=(NEW,KEEP),LABEL=(,NL)
//DFHBUXT  DD  DSN=CICS410.applid.DFHBUXT,UNIT=3400,VOL=SER=volid,
//         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.

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

Trace utility program (DFHTU410)

If you write trace entries to CICS auxiliary trace data sets you can use the trace utility program, DFHTU410, to extract all or selected trace entries, and format and print the data.

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 DFHTU410, see the *CICS/ESA Operations and Utilities Guide*.

Chapter 16. Defining dump data sets

This chapter describes how to define the following two types of dump data sets that CICS uses for recording dumps as a consequence of a failure detected during CICS execution, or upon explicit request:

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

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.

You can set the options you want in the dump table in two ways:

1. Using the CEMT master terminal command
2. Using the EXEC API commands.

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 with a CEMT or EXEC CICS command. For information about the dump table options you can set, see the *CICS/ESA Problem Determination Guide*.

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 message DFHAP0001 or DFHSR0001). To prevent this from happening, you can suppress all SDUMPs preceding ASRA, ASRB and ASRD abends, or you can suppress some of them. “Suppressing system dumps that precede ASRx abends” on page 176 tells you how to do this.

System dumps

CICS produces a system dump using the MVS SDUMP macro.

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 a SYS1.DUMP data set, which you can process using the interactive problem control system (IPCS). For information about the SDUMP macro, and the associated SYS1.DUMP data sets, see the *MVS/ESA Authorized Assembler Programming Reference* and *MVS/ESA Planning: Problem Determination and Recovery* manuals.

+ The SDUMP macros issued by CICS normally contain the QUIESCE=NO
+ parameter. They may not contain the parameter 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. For more information on
+ this command see the *MVS/ESA System Commands* manual.

You should use the ADD function when changing the SDUMP options using the
CHNGDUMP command to ensure that the areas selected by CICS to dump are
included in the SYS1.DUMP output. It replaces any options specified by CICS when issuing the SDUMP in many cases. This can result in partial dumps being taken to SYS1.DUMP. MVS always includes LSQA and TRT in the dump but may exclude the private area if you use the wrong options in the update by the CHNGDUMP command. You must thoroughly review your use of the CHNGDUMP command when setting up your CICS region. For information about the CHNGDUMP command and the effect that altering its options has on the dump output from CICS, see the *MVS/ESA System Commands* manual.

If you are running CICS with XRF, 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 ADI (alternate delay interval) for the alternate is set too low. However, you can prevent SDUMPS of other address spaces from causing unnecessary takeovers when the alternate CICS is running on a different MVS image by setting the QUIESCE=NO option for SDUMP, 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 dumpcode SR0001.

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

CICS uses dumpcode 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, dumpcode AP0001 is used instead.

Where storage protection is not active, SDUMPs can be suppressed by suppressing dumpcode AP0001. However, note that this suppresses dumps for errors in both application *and* CICS programs.

For more information about the storage protection facilities available in CICS/ESA 4.1, see "Storage protection" on page 358.

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/ESA Messages and Codes* manual and the *CICS/ESA Problem Determination Guide*.

Processing system dumps

You can process a system dump using IPCS, either online under TSO, or by submitting a batch job to print it. IPCS is described in the *MVS/ESA IPCS User's Guide*. For information about the IPCS VERBEXIT parameters that you use with the CICS IPCS dump exit, see the *CICS/ESA Operations and Utilities Guide*.

The CICS transaction dump data sets

CICS records transaction dumps on a sequential data set, or a pair of sequential data sets, on disk or tape. The data sets must be defined in the CICS run with the DD names DFHDMPA and DFHDMPB, but if you define a single data set only, its DD name must be DFHDMPA. In this chapter, a reference to "the CICS dump data set" means either DFHDMPA or DFHDMPB. Note that 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 following message is sent to the console:

```
DFHDU0306  applid Unable to open Transaction Dump Data set
           dataset-text-descr
```

With two data sets, you can print transaction dumps from one data set while CICS is running. To do this, first use 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. You can print the completed

data set with the DFHDU410 dump utility program. For information about the DFHDU410 dump utility program, see the *CICS/ESA Operations and Utilities Guide*.

In addition to switching dump data sets explicitly, the operator can use CEMT SET DUMP AUTO to cause automatic switching when the current data set becomes full. (Note that this 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.
```

where “x” and “y” can have the value A or B. If you specified DISP=SHR for the dump data set, you can print the completed data set with the DFHDU410 utility program and then reissue the command: CEMT SET DUMP AUTO. This again switches data sets automatically (once only) when the current data set is full.

You can define the CICS dump data sets DFHDMPA and DFHDMPB as temporary data sets for each CICS run. More commonly, you allocate and catalog them in advance, reuse them repeatedly, and do not delete them when the CICS job has completed. You can then use the DFHDU410 utility program to print the dump output at any time during or after the CICS run. Note that it is not practical to try to use temporary data sets when running CICS with XRF.

You do not need DCB parameters for dump data sets (but see “Copying disk dump data sets to tape” on page 179 for an exception). 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.

PN89471

The following change was made by APAR PN89471.

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

1. XDUCLSE, 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/ESA Customization Guide*.

Selecting the transaction dump data set at startup

You can 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. For more information about the DUMPDS parameter, see Chapter 21, “CICS system initialization parameters” on page 211.

Job control statements to allocate dump data sets

You can run the CICS-supplied job DFHDEFDS to allocate and catalog the dump data sets for an active CICS region or the CICS-supplied job DFHALTDS to allocate and catalog them for an alternate CICS region. For information about the jobs DFHDEFDS and DFHALTDS, see the *CICS/ESA Installation Guide*.

Alternatively, you can use the sample data definition statements in Figure 42 to allocate and catalog dump data sets on disk.

```
//DFHDMPA  DD  DSN=CICS410.applid.DFHDMPA,DISP=(NEW,CATLG),
//             UNIT=3380,VOL=SER=volid,SPACE=(CYL,(5,1))
//DFHDMPB  DD  DSN=CICS410.applid.DFHDMPB,DISP=(NEW,CATLG),
//             UNIT=3380,VOL=SER=volid,SPACE=(CYL,(5,1))
```

Figure 42. Sample job control statements for defining disk dump data sets

Note: Change the space allocations in this sample job stream to suit your own installation’s needs.

If you are running CICS with XRF, you must allocate different data sets for the alternate.

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 DFHDU410 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.

Copying disk dump data sets to tape

If you intend copying dump data sets to tape or disk, you must specify DCB parameters on the DD statements when allocating and cataloging the dump data sets, as follows:

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

Otherwise, if you do not intend copying dump data sets to tape or disk, it is not necessary to include DCB parameters when defining dump data sets on disk, as illustrated in the sample job in Figure 42.

Space calculations

For the initial installation of CICS, a dump data set of between 5 and 10MB should be enough. When normal operation begins, you can adjust this to suit your own installation's requirements.

Job control statements for CICS execution

The following DD statements for inclusion in the CICS startup job stream assume that the transaction dump data sets have been cataloged previously:

```
//DFHDMPA  DD  DSN=CICS410.applid.DFHDMPA,DISP=SHR
//DFHDMPB  DD  DSN=CICS410.applid.DFHDMPB,DISP=SHR
```

DISP=SHR enables each data set, if held on disk, to be processed by the DFHDU410 offline utility after the switch to the other data set has taken place.

The following are examples of DD statements for transaction dump data sets on unlabeled tapes:

```
//DFHDMPA  DD  DSN=CICS410.applid.DFHDMPA,UNIT=3400,VOL=SER=volid1,
//         DISP=(NEW,KEEP),LABEL=(,NL)
//DFHDMPB  DD  DSN=CICS410.applid.DFHDMPB,UNIT=3400,VOL=SER=volid2,
//         DISP=(NEW,KEEP),LABEL=(,NL)
```

Chapter 17. Defining the CICS availability manager data sets

This chapter tells you how to define the CICS availability manager (CAVM) data sets. The CAVM is the 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. 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.

Both the active and alternate CICS regions must refer to the same pair of data sets. You define these data sets, but must not try to initialize them, and you are recommended to place the data sets on separate volumes. The first time they are used, CICS recognizes them as a new pair of data sets. If they 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.

You must define a separate pair of data sets for each generic APPLID in use. If a CICS complex consists of, for example, five regions, five pairs of data sets must be defined.

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.

Why have two data sets? Because of RESERVE commands issued by other MVS images in a multi-MVS environment, a shared DASD volume may 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 is used:

- To record the presence or absence, identities, and current states of the active and alternate CICS regions' jobs
- 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.

JCL to define the XRF control data set

You must define the XRF control data set, but not initialize it. You can use the JCL statements in Figure 43 to define the XRF control data set. Alternatively, you can run the CICS-supplied job DFHDEFDS to define the XRF control data set as one of the data sets for a CICS region. For information about the DFHDEFDS job, see the *CICS/ESA Installation Guide*.

```
//CICSCTL   JOB   'accounting info',name,MSGCLASS=A
//XRCTL     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
       DEFINE CLUSTER -
                (NAME(CICS410.applid.DFHXRCTL) -
                 RECORDSIZE(4089 4089)         -                  1
                 CONTROLINTERVALSIZE(4096)     -                  2
                 RECORDS(4)                    -
                 NIXD                          -
                 SHAREOPTIONS(3,3)             -                  3
                 VOLUMES(volid1))              -
              DATA                             -                  4
                (NAME(CICS410.applid.DATA.DFHXRCTL))
/*
//
```

Figure 43. Sample job to define the XRF control data set

Notes:

1 The RECORDSIZE must be at least 4089.

2 The control interval sizes of the XRF control data set and the XRF message data set must be equal, and at least 4096 bytes.

3 The SHAREOPTIONS must be specified as 3,3.

4 The data set must be VSAM-ESDS.

Serializing access to the XRF control data set: 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, it would be unwise to 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.

Space calculations

Only four control intervals are needed.

Job control statements for CICS execution

The DD name required for CICS execution is DFHXRCTL. The following is an example of the JCL statement required:

```
//DFHXRCTL  DD    DSN=CICS410.applid.DFHXRCTL,DISP=SHR
```

The XRF message data set

The XRF message data set is used:

- Mainly to pass messages about the current states of specific resources from the active to the alternate CICS region
- 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.

JCL to define the XRF message data set

Like the XRF control data set, the XRF message data set must be defined but not initialized. You can use the sample job in Figure 44 on page 184 to define the XRF message data set. Alternatively, you can run the CICS-supplied job DFHDEFDS to define the XRF message data set as one of the data sets for a CICS region. For information about the DFHDEFDS job, see the *CICS/ESA Installation Guide*.

If you the sample JCL in Figure 44 on page 184, read the accompanying notes; the other options shown are suggestions only.

You should define the XRF message data set on a volume that is ***not*** subject to RESERVE activity, and should 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.

The XRF message data set is reserved for a short time for formatting when CICS uses it for the first time.

```
//CICSMSG   JOB   'accounting info',name,MSGCLASS=A
//XRMSG     EXEC  PGM=IDCAMS
//DDNAME2   DD    DISP=OLD,UNIT=3380,VOL=SER=volid2
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
       DEFINE CLUSTER -
                (NAME(CICS410.applid.DFHXRMSG) -
                 RECORDSIZE(4089 4089)         -                  1
                 CONTROLINTERVALSIZE(4096)     -                  2
                 RECORDS(1500)                 -
                 NIXD                          -
                 SHAREOPTIONS(3,3)             -                  3
                 VOL(volid2)                   -
                 FILE(DDNAME2))                -
              DATA                             -                  4
                (NAME(CICS410.applid.DATA.DFHXRMSG))
/*
//
```

Figure 44. Sample job to define the XRF message data set

Notes:

1 The RECORDSIZE must be at least 4089.

2 The control interval sizes of the XRF message data set and the XRF control data set must be equal, and at least 4096 bytes.

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.

3 The SHAREOPTIONS must be specified as 3,3.

4 The data set must not be indexed.

Space calculations

It is difficult to give a simple answer to the question: “How big should my XRF message data set be?”. A simple answer is that the size required 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 may 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 24 lists the sizes of the various messages sent to the data set.

Table 24. Sizes of messages sent to the XRF message data set

Type of TCT entry	***Bytes per install***
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 25. Additional space requirements (VTAM terminals only)

Bytes per logon	***Bytes per logoff***	***Bytes per signon***	***Bytes per sign-off***
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.

Job control statement for CICS execution

The DDNAME required for CICS execution is DFHXRMSG. The following JCL statement can be used:

```
//DFHXRMSG  DD    DSN=CICS410.applid.DFHXRMSG,DISP=SHR
```

Security

To ensure that the integrity and security of your CICS regions and terminal network are not compromised, you must protect your XRF data sets using RACF. 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.

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. 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 18. Defining user files

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

CICS application programs process files, which, to CICS, are logical views of a physical 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. 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 sets. These data sets 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 data tables to improve the performance and function of CICS installations 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 virtual storage, above the 16MB line. Each data table is associated with a VSAM KSDS, known as its **source data set**. For further information about data tables, see “CICS data tables” on page 197.

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. 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 190.

When you create a data set, you may 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 DSNAME), 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 “Backup while open (BWO) of VSAM files” on page 106.

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 via 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 via 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

As suggested above, there are several ways you can load data into an empty VSAM data set. VSAM imposes specific restrictions during initial data set load processing, when the data-set is said to be in **load mode**. These restrictions apply from the time that a file referencing the empty data set is opened. They continue while records are being loaded, and finish when the file is closed.

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

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

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.

Note: 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.

PN65947

The following change was made by APAR PN65947.

+ 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 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 may 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.

| **Note:** You cannot specify SERVREQ=REUSE on a data set's resource definition,
| even if you define the data set with the AMS REUSE attribute.

Fixed length records in VSAM data sets

Note that 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.

BDAM data sets

CICS supports access to keyed and nonkeyed BDAM data sets. To construct and format such data sets, you use BDAM.

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. Figure 45 is an example of this.

```
//BDAM EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN    DD DUMMY
//SYSPRINT DD SYSOUT=A
//SYSUT1   DD DSN=CICS410bdam.user.file.init,DISP=SHR        1
//SYSUT2   DD DSN=CICS410bdam.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 45. Sample JCL to create and load a BDAM data set

Notes:

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/ESA 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 46, 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 DATE FOR RECORD 99
1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
```

Figure 46. 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 use a mixture of definitions in the FCT and CSD 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 BSAM files (to allow migration of the FCT to the CSD) are ignored.

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.

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=CICS410.vsam.user.file,DISP=OLD
//BDAMFILE  DD   DSN=CICS410.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.

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 DSNAME and DISP file resource definition parameters

You can define a data set to CICS by specifying the DSNAME and DISP operands 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 DSNAME and DISP, do ***not*** provide a DD statement for the data set in the startup job stream.

If you use DSNAME and DISP on the file resource 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.

Note: If you define a data set to CICS using the DSNAME and DISP parameters in the resource definition, and also provide a DD statement in the CICS startup job stream, the attributes in the DD statement override those in the CICS resource definition.

For information about using the DSNAME and DISP parameters, see the *CICS/ESA 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 the *CICS/ESA CICS-Supplied Transactions* manual.

Dynamic allocation in an application program

You can assign the data set name dynamically from an application program by using the command:

```
EXEC CICS SET FILE(filename) DSNAME(datasetname) SHARE|OLD
```

The data set is then dynamically allocated in the same way as if you used the CEMT master terminal command, but only if the file status is CLOSED, and also DISABLED or UNENABLED. The file must be closed when you issue a command to change file attributes.

For programming information about the EXEC CICS SET command, see the *CICS/ESA System Programming Reference* manual.

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 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. 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.

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.

Closing files normally: You can close a file explicitly with one of the following commands:

```
CEMT SET FILE(filename) CLOSED
EXEC CICS SET FILE(filename) CLOSED
```

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.

Closing files using the FORCE option: You can also close a file using one of the following commands:

```
CEMT SET FILE(filename) FORCECLOSE
EXEC CICS SET FILE(filename) CLOSED FORCE
```

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.

The FORCE option and data integrity

Warning: 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. To ensure that they do, you can 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.
- If the data sets are allocated at job step initiation, the JCL defining the data sets must specify DISP=SHR.

CICS data tables

A data table is defined by means of the CEDA DEFINE FILE command. When a table is opened, CICS builds it by extracting data from the table's corresponding source VSAM data set and loading it into virtual storage above the 16MB 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/ESA Resource Definition Guide*. For programming information about the file control commands of the application programming interface, see the *CICS/ESA Application Programming Reference* manual.

CICS/ESA supports two types of data tables:

- **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/ESA Customization Guide*.

CICS-maintained data tables

All modifications to entries in a CICS-maintained data table are automatically reflected in the source data set. Similarly, all changes to the source data set are reflected in the data table if the changes are the result of file API requests issued in the CICS region in which the table resides.

If the source data set is shared, updates by other regions are not reflected in the table. A warning message is issued if the cross-region share options for the source data set would allow updates by other regions.

If a table is defined as recoverable, the full range of API file control commands is supported with full CICS integrity.

Requests that cannot be satisfied by reference to the data table alone, result in calls to VSAM to access the source data set. These calls could be sequential, generic or update requests, or direct read requests for records that are not contained in the data table, but are likely to be in the data set. Such calls can result in physical I/Os if the data is not held within the VSAM buffers.

User-maintained data tables

After the loading of a user-maintained data table at OPEN time, subsequent data table modifications are not reflected by CICS in the source data set. This has to be performed by the user's application if it is required.

Only a subset of the API file control commands is available. For programming information about those commands, see the *CICS/ESA Application Programming Reference* manual.

The global user exit can both select and ***modify*** records when constructing the user-maintained data table.

If a record is not found in the data table after loading, it is assumed that it does not exist; NOTFND is reported to the application. No attempt is made to access the source data set.

All record access requests perform well, because they never result in VSAM calls except during loading. This gives the optimum performance for those applications that do not require CICS to maintain data across a CICS restart.

Only dynamic backout recovery is supported for user-maintained data tables. After a restart or XRF takeover, a data table is reloaded from its source data set, which the application may or may not have kept in step with changes to the data table.

Loading from sources other than a VSAM data set can be done by having an empty VSAM KSDS source data set for the user-maintained data table. Application programs can then write records to the data table, which can be from any CICS file. This includes IMS/ESA or DB2 files accessible from the CICS address space.

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 virtual storage above the 16MB line of the CICS address space.

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 195.

Loading data tables

A data table is built automatically when it is opened. The table is constructed using a hashing technique, to speed up subsequent reference by keys.

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, all storage that was used to hold the hash tables and data table entries 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 196.

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.

Chapter 19. DD statements for DL/I data sets

This chapter describes the DD statements you need for DL/I data sets in your CICS startup job when using local DL/I support only.

You do not need any DD statements for DL/I in a CICS system using remote DL/I support only. If you are accessing DL/I databases from CICS through IMS database control (DBCTL), see the *CICS/ESA CICS-IMS Database Control Guide*.

DD statements for CICS local DL/I

To access DL/I databases with CICS local DL/I support, the DD statements you need fall into two categories:

1. The DD statements you need for your own user-defined DL/I databases
2. The DD statements that are required by IMS.

DL/I database data sets

You can either use the IMS dynamic allocation and deallocation facilities, or provide DD statements, for your CICS local DL/I databases. These two methods of allocating your DL/I databases are discussed below, together with information regarding the use of the DISP parameter.

Dynamic allocation of database data sets

If you use IMS dynamic allocation for your DL/I data sets, and you are not using data sharing, specify DISP=OLD. However, if you are using IMS data sharing, then you must specify DISP=SHR. When you are using dynamic allocation instead of JCL, the DISP parameter is specified in the IMS DFSMDA TYPE=DATASET macros. This macro is described in the *IMS Utilities Reference* manual. Each DFSMDA entry generates a module which is given the DBD name.

These rules regarding the DISP parameter also apply if you are running CICS with XRF. If you are using dynamic allocation, and running CICS with XRF=YES, DISP=OLD can be specified, but only if you are not using data sharing. The alternate CICS region does not allocate the data set until a PSB that references the data set is first scheduled after a takeover. See Table 26 on page 202 for a summary of which DISP option to specify in the various circumstances.

For information about dynamic allocation and deallocation of DL/I data sets, see the *CICS/ESA Recovery and Restart Guide*.

JCL allocation of data sets

If you are not using dynamic allocation and deallocation, you need DD statements in the CICS startup job stream for each physical DL/I database data set that is to be accessed. The DD names are those specified in the IMS DATASET macro during DBD generation. For an example of a job stream that includes DL/I DD statements, see the *CICS/ESA Operations and Utilities Guide*.

If you are running CICS with DL/I and using data sharing, or if you are running CICS with XRF, you must specify DISP=SHR; otherwise specify DISP=OLD, as shown in Table 26 on page 202.

Table 26. Selection of DISP parameter for DL/I databases

Type of CICS-IMS operation	**Dynamic allocation**	**Allocation by JCL**
Data sharing (with or without XRF)	DISP=SHR	DISP=SHR
No data sharing, ***without*** XRF	DISP=OLD	DISP=OLD
No data sharing, ***with*** XRF	DISP=OLD	DISP=SHR

Database recovery control (DBRC) registration for databases

When you specify DISP=SHR for your databases, you lose the protection against simultaneous usage that you get with DISP=OLD. Therefore, if you are running with DISP=SHR specified for your database data sets, you are recommended to register your databases with DBRC (using INIT.RECON command with the SHARECTL command). Note that when you are running CICS with XRF, DBRC control is not required as a means of maintaining integrity between the active and alternate CICS regions, but to protect your databases against access by other MVS jobs.

However, be aware that DBRC does not give you protection against any CICS region that is not signed on to DBRC. A database that is being shared has a JCL disposition of SHR and is defined with a VSAM cross-region or cross-system SHAREOPTION of 3 (or 4), which permits full sharing. Therefore, the database can be freely accessed by any region that is not signed on to DBRC. Your operating procedures must ensure that, while a database is being shared, it is never ***updated*** by a region that is not signed on to DBRC. Note also that if such a region reads the database, the read operations lack integrity.

XRF considerations

In an XRF system, whether you choose to use JCL or dynamic allocation for your DL/I database data sets, the active and alternate CICS regions must refer to the same data sets. However, you can minimize the risk of inconsistencies in data set naming between the active and alternate CICS regions' jobs by using dynamic allocation. For background information about dynamic allocation, see the *IMS Utilities Reference* manual.

When you are using IMS dynamic allocation instead of JCL, the DISP parameter is specified in the IMS DFSMDA TYPE=DATASET macros. For more information, see "Dynamic allocation of database data sets" on page 201.

DD statements for CICS shared databases

If you are using the CICS shared database facility, and you are not using the dynamic allocation and deallocation facilities of IMS, then you must include DD statements in the CICS startup job stream for all DL/I databases that might be accessed from a batch region. These DD statements are required even if the databases are not accessed directly by applications in the CICS region. For

information about how to create an IMS batch region to access shared DL/I databases, see the *CICS/ESA Operations and Utilities Guide*.

DD statements required for IMS system data sets

If you are running CICS with DL/I, there are some DD statements that you must include in the CICS startup job stream for specific IMS system data sets (for example, IMS libraries). These are for:

- The DL/I libraries, RESLIB and ACBLIB
- The DL/I library that contains the generated DFHDLQ module
- CICS shared databases
- DBRC data sets
- The integrated resource lock manager (IRLM)
- DFSVSAMP options.

These DD statements are described in the following sections.

DL/I library definitions

The IMS libraries that you need to define in the CICS startup job stream are RESLIB and ACBLIB. The requirements are described below, together with some XRF considerations, where applicable.

The DD statements for these DL/I libraries are shown later under the relevant library heading.

The RESLIB library

The RESLIB library (IMS.RESLIB) must be defined twice. First, you concatenate the RESLIB library with CICS410.SDFHAUTH in the STEPLIB DD statements. The RESLIB library must be an APF-authorized library. Second, you need another DD statement for RESLIB, this time with a DD name of DFSRESLB. IMS loads its modules from the DFSRESLB library. The DD statements for RESLIB might be as follows:

```
//*        The authorized library DD statement
//STEPLIB  DD  DSN=CICS410.SDFHAUTH,DISP=SHR
//         DD  DSN=reslib,DISP=SHR
//*        DFSRESLIB, required by IMS
//DFSRESLB DD  DSN=reslib,DISP=SHR
```

When you are running CICS with XRF, the active and alternate CICS regions can refer either to the same RESLIB, or to different RESLIBs. The alternate CICS region needs this data set ***before*** takeover.

The data set is allocated at job step initiation. If you use the same RESLIB for an active and alternate CICS region, it must be specified with DISP=SHR in the DD statement.

Applying service to RESLIB: If you want to apply IMS maintenance to RESLIB, the CICS system using that RESLIB must be shut down. For this reason, you may prefer to run the alternate CICS region with a separate RESLIB. Then, depending on the nature of the maintenance to be applied, you can shut down the alternate CICS region and bring it up again after the fix has been applied to the RESLIB used by the alternate CICS region. However, be aware that even when you run XRF with separate libraries, some maintenance may require that both active and

alternate CICS regions are shut down. Read the PTF cover memos carefully to determine whether this is so.

The ACBLIB library

Include a DD statement for the ACBLIB library (IMS.ACBLIB), with a DD name of IMSACB. For example:

```
//IMSACB   DD  DSN=IMS.ACBLIB,DISP=SHR
```

ACBLIB contains the IMS application control blocks (ACBs), and should include ACBs for programs running in IMS batch regions (using the CICS shared database facility), and for online transactions.

When you are running CICS with XRF, both the active and alternate CICS regions must refer either to the same ACBLIB, or to ACBLIBs that contain the same ACBs referenced by the active and alternate CICS regions. The alternate CICS region does ***not*** require this data set before takeover.

The data set is allocated at job step initiation, and therefore it must be specified with DISP=SHR in the DD statement.

The APF-authorized library containing DFHDLQ

You must include a DD statement in your STEPLIB concatenation for the APF-authorized library into which you generated the CICS-DLI initialization module, DFHDLQ. For information about generating the DFHDLQ module, see the *CICS/ESA Installation Guide*.

CICS shared databases and the PGMLIB library

CICS does not need the library containing the IMS batch and shared-database application programs, such as the PGMLIB library (IMS.PGMLIB) or its equivalent. CICS application programs are either in CICS410.SDFHLOAD, or a private library in the DFHRPL library concatenation.

However, if you are running CICS with XRF on two MVS images, ensure that the library containing the IMS batch and shared-database application programs is also accessible by the second MVS image. Alternatively, you can have a copy of the library available on the second MVS image. This enables you to restart the batch region programs, after a takeover by the alternate CICS region on the second MVS image, if the takeover is caused by the failure of the MVS image on which the active CICS region is running.

MVS checkpoint data sets

If checkpoints are being taken by the CICS shared-database job in an XRF environment, the MVS checkpoint data sets should be accessible to the shared batch job after a takeover occurs. This applies when the alternate CICS region is executing in a second MVS image, which means the shared-database job also has to be restarted on the second MVS image.

The IMS data sets required for DBRC users

The DBRC data sets that you need to define if you are running CICS with IMS data sharing, or using DBRC for recovery control without data sharing, are as follows:

- The RECON data sets
- JCLPDS, the skeletal JCL data set
- JCLOUT, the GENJCL output data set.

The requirements for these are briefly described below, together with some XRF considerations, where applicable. For more detailed information about these data sets, refer to the appropriate IMS manual. IMS manuals are listed in “Information Management System (IMS)” on page xiii.

DBRC RECON

RECON1, RECON2 (and optionally, RECON3) are the main DBRC data sets, used to keep track of the subsystems participating in data sharing. They must be defined as VSAM key sequenced data sets, with a key length of 24 bytes. The VSAM share options parameter must be specified as SHAREOPTIONS(3 3), and you are recommended to specify a different size for each RECON data set. For an example of the DEFINE CLUSTER statements you need for RECON data sets, see the relevant sample DFSIVJnn job in the *IMS Installation Listings* manual.

The active and alternate CICS regions must refer to the same RECON data sets, but the alternate CICS regions does not require the data sets before takeover.

You are recommended to use IMS dynamic allocation for the RECON data sets, in which case they are allocated during IMS initialization. DISP=SHR must be specified if you are using dynamic allocation. For information about IMS dynamic allocation, see the description of the DFSMDA macro in the *IMS Utilities Reference* manual. You can also use JCL to define the RECON data sets, in which case they are allocated at job step initiation. They must be specified with DISP=SHR in the DD statements.

DBRC JCLPDS

The JCLPDS data set contains skeletal JCL for use by the GENJCL function of the CBRC transaction. The same data set may be used by batch DBRC jobs.

Active and alternate CICS regions can refer to either the same, or a different, JCLPDS data set, but the alternate CICS region does not open the data set before takeover.

If the active and alternate CICS regions use the same data set, note that it is allocated at job step initiation, and therefore it must be specified with DISP=SHR in the DD statement.

DBRC JCLOUT

The JCLOUT data set contains the output produced from the GENJCL function of the CBRC transaction (the JCL to run the requested GENJCL function).

The active and alternate CICS regions do not normally refer to the same JCLOUT data set (for example, if it is defined with the SYSOUT parameter on the DD statement). The alternate CICS region does not open the data set before takeover.

JCLOUT is normally defined with the SYSOUT parameter specified to direct the output to the JES internal reader for execution. In this case, the data set is unique. For example, you could code the DD statement for JCLOUT as follows:

```
//JCLOUT  DD  SYSOUT=(A,INTRDR)
```

However, if you define JCLOUT as a data set on disk (with the DSNAME parameter coded), then you must do one of the following:

- Define unique data sets for both the active and alternate CICS region
- Define a data set with DISP=SHR specified.

IMS resource lock manager (IRLM)

The only data set required for the IMS resource lock manager (IRLM) is the RESLIB library (IMS.RESLIB) from which the IRLM code is loaded. This data set is allocated at job step initiation and a JCL status of DISP=SHR should be specified.

The DFSVSAMP options

You define the DL/I buffer pools on the DFSVSAMP DD statement. For information about the DFSVAMP options, see the *IMS Installation Guide* (Version 1), or the *IMS System Definition Reference* manual. For example, a DFSVSAMP DD statement might be as follows:

```
//DFSVSAMP DD  *
2048,4
/*
```

If you require the IMS DL/I trace facilities, specify them in the DFSVSAMP DD statement. For example, if the LOCK=OUT facility is specified, CICS writes DL/I trace and lock trace records to the CICS system log.

Note: If you include the IMS DL/I trace facility in the DFSVSAMP DD statement with the LOCK=OUT option, you can get large volumes of DL/I data written to the CICS system log.

For information about other IMS DL/I trace facilities, refer to the *IMS System Definition Reference* manual.

Chapter 20. Defining the CMAC messages data set

This chapter describes the VSAM key-sequenced data set (KSDS) called DFHCMACD. DFHCMACD is used by the CMAC transaction to provide online descriptions of the CICS messages and codes.

You can create the DFHCMACD data set and load it with the CICS-supplied messages and codes data by running the DFHCMACI job. Some IBM-supplied service may include changes to CICS messages and codes, and associated changes to the DFHCMACD data set. You can apply such service changes to the DFHCMACD data set by running the DFHCMACU job.

For more information about the DFHCMACI and DFHCMACU jobs, see the *CICS/ESA Installation Guide*.

Notes:

1. The DFHCMACD data set is accessed by the file CMAC, 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 needed by the CICS messages facility are in the CSD group DFHCMAC. The CICS IVPs have a DD statement for the CMAC file, but for dynamic allocation you should copy the supplied resource definition for the CMAC file and add the DSNAME option.
2. 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 DFHCMAC 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.
3. You should specify the DFHCMAC group of resources for the CICS messages facility only in those CICS regions that need to use the facility; for example on some terminal-owning regions, but perhaps not on data-owning regions.

Job control statements to define and load the messages data set

Before its first use, the DFHCMACD data set should be defined and loaded as a VSAM key sequenced data set (KSDS). The sample job in Figure 47 on page 208 shows you how to do this.

Note: You can define and load the DFHCMACD data set by running the DFHCMACI job.

```
//CMACJOB  JOB  'accounting information',name,MSGCLASS=A
//CMACDEF  EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT DD SYSOUT=*
//AMSDUMP  DD SYSOUT=*
//SYSIN    DD *
  DELETE CICS410.DFHCMACD
  SET MAXCC=0
  DEFINE CLUSTER (                          -
                NAME( CICS410.DFHCMACD )      -
                CYL(2,1)                      -
                KEYS( 9 0 )                   -
                INDEXED                       -
                VOLUME ( cmacvol)             -
                RECORDSIZE( 8192 32666 )      -
                FREESPACE( 5 5 )              -
                SHAREOPTIONS( 2 )             -
                )                             -
  INDEX         (                             -
                NAME( CICS410.DFHCMACD.INDEX )    -
                )                             -
  DATA          (                             -
                NAME( CICS410.DFHCMACD.DATA )     -
                )
/*
//CMACLOAD EXEC PGM=IDCAMS,REGION=1M
//SYSPRINT DD SYSOUT=*
//AMSDUMP  DD SYSOUT=*
//SYS01    DD DSN=CICS410.SDFHMSGS(DFHCMACD),DISP=SHR
//DFHCMACD DD DSN=CICS410.DFHCMACD,DISP=SHR
//SYSIN    DD *
  REPRO INFILE (SYS01)                    -
        OUTFILE (DFHCMACD)
/*
//
```

where `cmacvol` is the volume on which the DFHCMACD data set is to be created.

Figure 47. Sample job to define and initialize the CMAC data set

Job control statements for CICS execution

If you defined the messages data set using the sample job shown in Figure 47, the data definition statement for the CICS execution is:

```
//DFHCMACD  DD DSN=CICS410.DFHCMACD,DISP=SHR
```

Part 3. CICS system initialization

This section of the book describes how to define CICS system initialization parameters, how they are processed by CICS, and a startup job stream you can use to start a CICS system.

- Chapter 21, “CICS system initialization parameters” on page 211 describes the CICS system initialization parameters.
- Chapter 22, “Processing system initialization parameters” on page 321 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 23, “CICS startup” on page 337 describes a sample startup job stream, and a sample procedure for use as a started task.

Chapter 21. CICS system initialization parameters

This chapter ...

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.

"Changes to system initialization parameters" lists those system initialization parameters that have changed, are no longer supported, or have been added in CICS/ESA 4.1. Descriptions of all the system initialization parameters are given, starting on page 228.

Changes to system initialization parameters

Table 27 lists those CICS system initialization parameters that have changed, been removed, or been added in CICS/ESA 4.1. Descriptions of the changed and new system initialization parameters are given after the table, starting on page 228.

Table 27 (Page 1 of 4). System initialization parameters added, changed, or removed in CICS/ESA 4.1

	DFHSIT	[TYPE={**CSECT**|DSECT}] ⋮ ~~[,AMXT={**MXT-number**|number}]~~ ⋮ [CMDSEC={**ASIS**|ALWAYS} ~~[,CDSASZE={**1M**|number}]~~ **PN70228** The following change was made by APAR PN70228. [,CDSASZE={**0K**|number}] ⋮ [,CMDPROT={**YES**|NO}] ⋮ ~~[,CMXT=([n1][,n2]...)]~~ ~~[,CMXTLIM=([n1][,n2]...,n10)]~~ ⋮ ~~[,CSCS={**64K**|number}]~~ ⋮ [,DAE={**NO**|YES}] [,DSALIM={**5M**|number}] [,DTRTRAN={**CRTX**|name|NO}] ⋮ **PN88030** ECDSASZE has been re-introduced by APAR PN88030. [,ECDSASZE={**0K**|number}]

Table 27 (Page 2 of 4). System initialization parameters added, changed, or removed in CICS/ESA 4.1

		~~[,ECSCS={**256K**|number}]~~ [,EDSALIM={**20M**|number}] ⋮ **PN88030** ERDSASZE has been re-introduced by APAR PN88030. [,ERDSASZE={**0K**|number}] ~~[,ERSCS={**256K**|number}]~~ **PN88030** ESDSASZE has been added by APAR PN88030. [,ESDSASZE={**0K**|number}] ⋮ **PN88030** EUDSASZE has been re-introduced by APAR PN88030. [,EUDSASZE={**0K**|number}] ~~[,EUSCS={**256K**|number}]~~ ⋮ [,GNTRAN={**NO**|transaction-id}] ⋮ [,GRNAME=name] [,GRPLIST={**DFHLIST**|name| (name[,name2][,name3][,name4])}] ⋮ [,ICVR={**5000**|number}] ⋮ ~~[,ISRDELAY={**30**|number}]~~ ⋮ ~~.[,MAXSMIR={**999**|number}]~~ ⋮ [,MNCONV={**NO**|YES}] [,MNFREQ={**0**|hhmmss}] [,MNSUBSYS={**null**|xxxx}] [,MNSYNC={**NO**|YES}] [,MNTIME={**GMT**|LOCAL}] ⋮ [,MXT={**5**|number}] ⋮ [,PGAICTLG={**MODIFY**|NONE|ALL}] [,PGAIEXIT={**DFHPGADX**|name}] [,PGAIPGM={**INACTIVE**|ACTIVE}] ⋮ [,PLTPISEC={**NONE**|CMDSEC|RESSEC|ALL}] [,PLTPIUSR=userid] ⋮

Table 27 (Page 3 of 4). System initialization parameters added, changed, or removed in CICS/ESA 4.1

[,PSDINT={**0**|hhmmss}]

PQ01573

The following parameter was added by APAR PQ01573.

[,PSTYPE={**SNPS**|MNPS}]
⋮

PN70228

The following change was made by APAR PN70228.

[,RDSASZE={**0K**|number}]
⋮
[,RESSEC={**ASIS**|ALWAYS}
⋮

PN70228

The following change was made by APAR PN70228.

[,SDSASZE={**0K**|number}]
⋮
[,SEC={**YES**|NO|~~MIGRATE~~}]
⋮
⋮
[,SNSCOPE={**NONE**|CICS|MVSIMAGE|SYSPLEX}]
⋮
[,SPCTRxx={(1[,2][,3])|ALL|OFF}]
[,STNTRxx={(1[,2][,3])|ALL|OFF}]
⋮
[,SYDUMAX={**999**|number}]
⋮
[,TD=({**3**|number1}[,{**3**|number2}])]
[,TRANISO={**NO**|YES}]
⋮
[,TRDUMAX={**999**|number}]
⋮

PN60636

New SIT parameters TRTRANSZ and TRTRANTY added by PN60636

[,TRTRANSZ={**40**|number}]
[,TRTRANTY={**TRAN**|ALL}]
⋮
[,TS=([COLD][,{0|**3**|value-1}][,{**3**|value-2}])]
⋮
~~[,UDSASZE={**4M**|number}]~~

PN70228

The following change was made by APAR PN70228.

Table 27 (Page 4 of 4). System initialization parameters added, changed, or removed in CICS/ESA 4.1

		[,UDSASZE={**0K**|number}] ⋮ ~~[,USCS={**64K**|number}]~~ ⋮ [,USRDELAY={**30**|number}] ⋮ [,VTPREFIX={||character}] ⋮ [,XUSER={**YES**|NO}] ⋮
	END	DFHSITBA

For details of the new system initialization parameters, see the descriptions starting on page 228.

Specifying system initialization parameters

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 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.

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 the user-specified version of the CICS modules (for example, DFHDBPxx) and tables (for example, DFHJCTxx).
3. Special information used to control the initialization process.

The syntax of the system initialization parameters that can be coded in the DFHSIT macro is listed in Table 28 on page 215. Except for those parameters marked "**SIT macro only**," all the system initialization parameters can be provided at run time, although there are restrictions in some cases. The restrictions are explained at the end of the description of the system initialization parameter to which they

apply. See the CHKSTRM and DLDBRC parameters on page 235 and on page 246 for examples of such restrictions.

There are some other CICS system initialization parameters (and options of the parameters in Table 28) that you cannot define in the DFHSIT macro. (See "Initialization parameters that cannot be coded in the DFHSIT macro" on page 225.) The parameters that you cannot define in the DFHSIT macro are shown in Figure 49 on page 226.

System initialization keywords grouped by function

For a list of all the system initialization keywords grouped by their functional area, see Appendix, "System initialization parameters grouped by functional area" on page 369. This ensures that you do not miss coding an important parameter relating to a particular CICS function. For details of how to code a parameter, you still have to refer to the parameter descriptions that are listed alphabetically in this chapter.

Migration considerations

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/ESA 4.1 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 "DFHSIT, the default system initialization table" on page 220.

The DFHSIT macro parameters

Table 28 (Page 1 of 5). The DFHSIT macro parameters

	DFHSIT	[TYPE={**CSECT**|DSECT}] [,ADI={**30**|number}] + [,AIEXIT={**DFHZATDX**|DFHZATDY|name}] [,AILDELAY={**0**|hhmmss}] [,AIQMAX={**100**|number}] [,AIRDELAY={**700**|hhmmss}] [,AKPFREQ={**200**|number}] [,APPLID=({**DBDCCICS**|name1}[,name2])] [,AUTCONN={**0**|hhmmss}] [,AUXTR={**OFF**|ON}] [,AUXTRSW={**NO**|ALL|NEXT}] [,BMS=({MINIMUM|STANDARD|**FULL**}[,COLD] [,{**UNALIGN**|ALIGN}][,{**DDS**|NODDS}])] # [,CDSASZE={**0K**|number}] [,CICSSVC={**216**|number}]

Table 28 (Page 2 of 5). The DFHSIT macro parameters

		[,CLSDSTP={**NOTIFY**|NONOTIFY}]
		[,CLT=xx]
		[,CMDPROT={**YES**|NO}]
		[,CMDSEC={**ASIS**|ALWAYS}
		[,CONFDATA={**SHOW**|HIDETC}]
		[,CONFTXT={**NO**|YES}]
		[,CSDACC={**READWRITE**|READONLY}]
		[,CSDBKUP={**STATIC**|DYNAMIC}]
		[,CSDBUFND=number]
		[,CSDBUFNI=number]
		[,CSDDISP={OLD|SHR}]
		[,CSDDSN={name}]
		[,CSDFRLOG=number]
		[,CSDJID={**NO**|number}]
		[,CSDLSRNO={**1**|number|NONE|NO}]
		[,CSDRECOV={**NONE**|ALL|BACKOUTONLY}]
		[,CSDSTRNO={**2**|number}]
		[,CWAKEY={**USER**|CICS}]
+		[,DAE={**NO**|YES}]
		[,DATFORM={**MMDDYY**|DDMMYY|YYMMDD}]
		[,DBP={1$\|2$|xx|YES}] (Must specify in the SIT macro)
		[,DBUFSZ={**500**|number}]
		[,DCT=({**YES**|xx|NO}[,COLD])]
		[,DDIR={**YES**|xx}]
		[,DFLTUSER={**CICSUSER**|userid}]
		[,DIP={**NO**|YES}]
		[,DISMACP={**YES**|NO}]
		[,DLDBRC={**NO**|YES}]
		[,{DLI|DL1}=({**NO**|YES|REMOTE}[,COLD])]
		[,DLIOLIM={**100**|number}]
		[,DLIRLM={YES|name|**NO**}]
		[,DLLPA={**NO**|YES}]
		[,DLMON={**NO**|YES}]
		[,DLTHRED={**1**|number}]
		[,DLXCPVR={**NO**|YES}]
		[,DMBPL={**4**|number}]
		[,DSALIM={**5M**|number}]
		[,DSHIPIDL={**020000**|hhmmss}]
		[,DSHIPINT={**120000**|hhmmss}]
		[,DTRPGM={**DFHDYP**|program-name}]
		[,DTRTRAN={**CRTX**|name|NO}]
		[,DUMP={**YES**|NO}]
		[,DUMPDS={**AUTO**|A|B}]
		[,DUMPSW={**NO**|NEXT}]
		[,DURETRY={**30**|number-of-seconds|0}]
#		[,ECDSASZE={**0K**|number}]
		[,EDSALIM={**20M**|number}]
		[,ENQPL={**2**|number}]
		[,EODI={**E0**|xx}]
#		[,ERDSASZE={**0K**|number}]
#		[,ESDSASZE={**0K**|number}]

Table 28 (Page 3 of 5). The DFHSIT macro parameters

		[,ESMEXITS={**NOINSTLN**|INSTLN}] ***(SIT macro only)***
#		[,EUDSASZE={**0K**|number}]
		[,FCT={**YES**|xx|NO}]
		[,FEPI={**NO**|YES}]
		[,FLDSEP={**'____'**|'xxxx'}]
		[,FLDSTRT={**'_'**|'x'}]
+		[,FSSTAFF={YES|**NO**}]
		[,GMTEXT={**'WELCOME TO CICS/ESA'**|'text'}]
		[,GMTRAN={**CSGM**|CESN|name}]
+		[,GNTRAN={**NO**|transaction-id}]
|		[,GRNAME=name]
|		[,GRPLIST={**DFHLIST**|name|
|		(name[,name2][,name3][,name4])}]
		[,GTFTR={**OFF**|ON}]
		[,HPO={**NO**|YES}] ***(SIT macro only)***
		[,ICP=COLD]
		[,ICV={**1000**|number}]
		[,ICVR={**5000**|number}]
		[,ICVTSD={**500**|number}]
		[,INITPARM=(pgmname_1='parmstring_1'
		[, ... ,pgmname_n='parmstring_n'])]
		[,INTTR={**ON**|OFF}]
		[,IRCSTRT={**NO**|YES}]
		[,ISC={**NO**|YES}]
		[,JCT={**YES**|xx|NO}]
		[,JESDI={**30**|number}]
		[,LGNMSG={**NO**|YES}]
|		[,LLACOPY={**YES**|NO|NEWCOPY}]
		[,LPA={**NO**|YES}]
		[,MCT={**NO**|YES|xx}]
		[,MN={**OFF**|ON}]
|		[,MNCONV={**NO**|YES}]
		[,MNEVE={**OFF**|ON}]
		[,MNEXC={**OFF**|ON}]
|		[,MNFREQ={**0**|hhmmss}]
		[,MNPER={**OFF**|ON}]
|		[,MNSUBSYS={**null**|xxxx}]
|		[,MNSYNC={**NO**|YES}]
|		[,MNTIME={**GMT**|LOCAL}]
		[,MROBTCH={**1**|number}]
#		[,MROFSE={**NO**|YES}]
		[,MROLRM={**NO**|YES}]
		[,MSGCASE={**MIXED**|UPPER}]
		[,MSGLVL={**1**|0}]
|		[,MXT={**5**|number}]
		[,NATLANG=(**E**,x,y,z,...)]
		[,OPERTIM={**120**|number}]
		[,OPNDLIM={**10**|number}]
		[,PARMERR={**INTERACT**|IGNORE|ABEND}]
		[,PDI={**30**|number}]
		[,PDIR={**YES**|xx}]

Table 28 (Page 4 of 5). The DFHSIT macro parameters

		[,PGAICTLG={**MODIFY**|NONE|ALL}]
		[,PGAIEXIT={**DFHPGADX**|name}]
		[,PGAIPGM={**INACTIVE**|ACTIVE}]
		[,PGCHAIN=character(s)]
		[,PGCOPY=character(s)]
		[,PGPURGE=character(s)]
		[,PGRET=character(s)]
		[,PISCHD={**YES**|NO}]
		[,PLTPI={**NO**|xx|YES}]
		[,PLTPISEC={**NONE**|CMDSEC|RESSEC|ALL}]
		[,PLTPIUSR=userid]
		[,PLTSD={**NO**|xx|YES}]
		[,PRGDLAY={**0**|hhmm}]
		[,PRINT={**NO**|YES|PA1|PA2|PA3}]
		[,PRTYAGE={**32768**|number}]
		[,PSBCHK={**NO**|YES}]
		[,PSBPL={**4**|number}]
		[,PSDINT={**0**|hhmmss}]
		[,PSTYPE={**SNPS**|MNPS}]
		[,PVDELAY={**30**|number}]
		[,RAMAX={**256**|number}]
		Apar PQ15635 Documentation for Apar PQ15635 added 09/10/98
		[,RAPOOL={**50**|value1|(value1,value2)},FORCE]
		[,RDSASZE={**0K**|number}]
		[,RENTPGM={**PROTECT**|NOPROTECT}]
		[,RESP={**FME**|RRN}]
		[,RESSEC={**ASIS**|ALWAYS}
		[,RMTRAN=({**CSGM**|name1} [,{**CSGM**|name2}])]
		[,RST={**NO**|xx|YES}]
		[,SDSASZE={**0K**|number}]
		[,SEC={**YES**|NO}]
		[,SECPRFX={**NO**|YES}]
		[,SKRxxxx='page-retrieval-command']
		[,SNSCOPE={**NONE**|CICS|MVSIMAGE|SYSPLEX}]
		[,SPCTR={(**1,2**|1[,2][,3])|ALL|OFF}]
		[SPCTRxx={(**1,2**|1[,2][,3])|ALL|OFF}]
		[,SPOOL={**NO**|YES}]
		[,SRBSVC={**215**|number}]
		[,SRT={**YES**|xx|NO}]
		[,START={**AUTO**|COLD|STANDBY}]
		[,STARTER={**NO**|YES}] ***(SIT macro only)***
		[,STATRCD={**OFF**|ON}]
		[,STGPROT={**NO**|YES}]
		[,STGRCVY={**NO**|YES}]
		[,STNTR={**1**|(1[,2][,3])|ALL|OFF}]
		[STNTRxx={(**1,2**|1[,2][,3])|ALL|OFF}]
		[,SUBTSKS={**0**|1}]
		[,SUFFIX=xx] ***(SIT macro only)***

Table 28 (Page 5 of 5). The DFHSIT macro parameters

		[,SYDUMAX={**999**|number}] [,SYSIDNT={**CICS**|name}] [,SYSTR={**ON**|OFF}] [,TAKEOVR={**MANUAL**|AUTO|COMMAND}] [,TBEXITS=([name1][,name2][,name3][,name4])] [,TCAM={**NO**|YES}] [,TCP={**YES**|NO}] [,TCSACTN={**NONE**|UNBIND|FORCE}] [,TCSWAIT={**4**|number|NO|NONE|0}] [,TCT={**YES**|xx|NO}] [,TCTUAKEY={**USER**|CICS}] [,TCTUALOC={**BELOW**|ANY}] [,TD=({**3**|number1}[,{**3**|number2}])] [,TRANISO={**NO**|YES}] [,TRAP={**OFF**|ON}] [,TRDUMAX={**999**|number}] [,TRTABSZ={**16**|number}] [,TRTRANSZ={**40**|number}] [,TRTRANTY={**TRAN**|ALL}] [,TS=([COLD][,{0|**3**|value-1}][,{**3**|value-2}])] [,TSMGSET={**4**|number}] [,TST={**NO**|YES|xx}] [,USERTR={**ON**|OFF}] [,USRDELAY={**30**|number}] [,VTAM={**YES**|NO}] [,VTPREFIX={****|character}] [,WRKAREA={**512**|number}] [,XAPPC={**NO**|YES}] [,XCMD={**YES**|name|NO}] [,XDCT={**YES**|name|NO}] [,XFCT={**YES**|name|NO}] [,XJCT={**YES**|name|NO}] [,XLT={**NO**|xx|YES}] [,XPCT={**YES**|name|NO}] [,XPPT={**YES**|name|NO}] [,XPSB={**YES**|name|NO}] [,XRF={**NO**|YES}] [,XRFSOFF={**NOFORCE**|FORCE}] [,XRFSTME={**5**|number}] [,XTRAN={**YES**|name|NO}] [,XTST={**YES**|name|NO}] [,XUSER={**YES**|NO}] You must terminate your macro parameters with the following END statement.
	END	DFHSITBA

DFHSIT, the default system initialization table

The macro source statements used to assemble the default system initialization table are given in Figure 48. This default SIT is in CICS410.SDFHAUTH, and its source, named DFHSIT$$, is in CICS410.SDFHSAMP.

```
* $MOD(DFHSIT$$ COMP(STARTER) PROD(CICS/ESA):
*     5655-018                                                  *
*     COPYRIGHT = NONE                                          *
*                                                               *
*      PN= REASON REL YYMMDD HDXXIII : REMARKS                  *
*                                                               *
* SIT parameters (in alphabetical order)                        *
*                                                               *
SIT      TITLE 'DFHSIT - CICS DEFAULT SYSTEM INITIALIZATION TABLE'
         DFHSIT TYPE=CSECT,                                     *
               ADI=30,           XRF(B) - Alternate delay interval  *
               AIEXIT=DFHZATDX,  Autoinstall user program name      *
               AILDELAY=0,       Delete delay period for AI TCTTEs  *
               AIQMAX=100,       Maximum no. of terminals queued for AI*
               AIRDELAY=700,     Restart delay period for AI TCTTEs *
               AKPFREQ=200,      Activity keypoint frequency        *
               APPLID=DBDCCICS,  VTAM APPL identifier               *
               AUTCONN=0,        Autoconnect delay                  *
               AUXTR=OFF,        Auxiliary trace option             *
               AUXTRSW=NO,       Auxiliary trace autoswitch facility*
               BMS=(FULL,,UNALIGN,DDS), Basic Mapping Support options *
               CICSSVC=216,      The CICS SVC number                *
               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      *
               CSDACC=READWRITE, CSD access                         *
               CSDBKUP=STATIC,   Backuptype 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 datasetname for dynamic allocation*
               CSDFRLOG=NO,      Journal id. for CSD forward recovery *
               CSDJID=NO,        Journal id. for CSD auto. journaling *
               CSDLSRNO=1,       The VSAM LSR pool number for the CSD *
               CSDRECOV=NONE,    CSD recoverable file option        *
               CSDSTRNO=2,       CSD Number of strings              *
               CWAKEY=USER,      CWA storage key                    *
+              DAE=NO,           SDUMPS will not be suppressed by DAE *
               DATFORM=MMDDYY,   CSA date format                    *
               DBP=1$,           Required version of DBP with DLI=NO*
               DBUFSZ=500,       Dynamic backout buffer size        *
               DCT=YES,          Dest. control table option/suffix  *
```

Figure 48 (Part 1 of 6). DFHSIT, the pregenerated default system initialization table

```
                DDIR=YES,         DL/I DMB directory option/suffix      *
                DFLTUSER=CICSUSER, Default user                         *
                DIP=NO,           Batch data interchange program        *
                DISMACP=YES,      Disable macro programs                *
                DLDBRC=NO,        DL/I DBRC support - CICS local DL/I   *
                DLI=NO,           DL/1 option                           *
                DLIOLIM=100,      Number of errors per DL/I data base   *
                DLIRLM=NO,        DL/I IRLM option/name                 *
                DLLPA=NO,         Use IMS/VS modules from LPA option    *
                DLMON=NO,         DL/I Data Base Monitor option         *
                DLTHRED=1,        DL/I number of threads (CICS-DLI)     *
                DLXCPVR=NO,       Page-fix ISAM/OSAM buffers for DL/I   *
                DMBPL=4,          DMB pool size in 1024-byte blocks     *
|               DSALIM=5M,        Upper limit of DSA below 16Mb line    *
|               DSHIPIDL=020000,  Delete shipped idle time      @BA59626*
|               DSHIPINT=120000,  Delete shipped interval       @BA59626*
|               DTRPGM=DFHDYP,    Dynamic transaction routing program   *
                DTRTRAN=CRTX,     Default dynamic tran routing transid  *
                DUMP=YES,         Dump option                           *
                DUMPDS=AUTO,      CICS dump data set opening option     *
                DUMPSW=NO,        Dump data set autoswitch option       *
                DURETRY=30,       SDUMP total retry time (in seconds)   *
                EDSALIM=20M,      Upper limit of DSA above 16MB line    *
                ENQPL=2,          Max.control.blk.space (in 1K blocks)  *
                EODI=E0,          End-of-data indicator for seq. devices*
                ESMEXITS=NOINSTLN, External security manager exits      *
                FCT=YES,          File control table option/suffix      *
                FEPI=NO,          Front-End Programming Interface       *
                FLDSEP='    ',    End-of-field separator characters     *
                FLDSTRT=' ',      Field start character for builtin fn  *
+               FSSTAFF=NO,       Function-shipped START affinity option*
                GMTEXT='WELCOME TO CICS/ESA', Good-morning message text *
                GMTRAN=CSGM,      Initial transaction                   *
+               GNTRAN=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=COLD,         Interval control pgm. start option    *
                ICV=1000,         Region exit interval (milliseconds)   *
                ICVR=5000,        Runaway task interval (milliseconds)  *
```

Figure 48 (Part 2 of 6). DFHSIT, the pregenerated default system initialization table

```
                ICVTSD=500,       Terminal scan delay interval (  "  )  *
                INITPARM=,        Initialisation parms for programs     *
                INTTR=ON,         CICS internal trace option            *
                IRCSTRT=NO,       Interregion communication start       *
                ISC=NO,           Intersystem communication option      *
                JCT=YES,          Journal control table option/suffix   *
                JESDI=30,         JES delay interval for XRF alternate  *
                LGNMSG=NO,        Extract VTAM logon data               *
|               LLACOPY=YES,      Use MVS LLACOPY support               *
                LPA=NO,           Use-LPA option for CICS/user modules  *
                MCT=NO,           Monitoring cntl.table option/suffix   *
                MN=OFF,           CICS monitoring option                *
|               MNCONV=NO,        Monitoring converse recording option  *
                MNEVE=OFF,        Monitoring event class option         *
                MNEXC=OFF,        Monitoring exception class option     *
|               MNFREQ=0,         Monitoring frequency period           *
                MNPER=OFF,        Monitoring performance class option   *
|               MNSUBSYS=,        Monitoring subsystem identification   *
|               MNSYNC=NO,        Monitoring syncpoint recording option *
|               MNTIME=GMT,       Monitoring timestamp (GMT/LOCAL)      *
                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            *
                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=YES,         DL/I PSB directory option/suffix      *
|               PGAICTLG=MODIFY,  PG autoinstall catalog state          *
|               PGAIEXIT=DFHPGADX, PG autoinstall exit program          *
|               PGAIPGM=INACTIVE, PG autoinstall state                  *
                PGCHAIN=,         BMS CHAIN command                     *
                PGCOPY=,          BMS COPY command                      *
                PGPURGE=,         BMS PURGE command                     *
                PGRET=,           BMS RETURN command                    *
                PISCHD=YES,       Program isolation scheduling option   *
                PLTPI=NO,         Program list table PI option/suffix   *
```

Figure 48 (Part 3 of 6). DFHSIT, the pregenerated default system initialization table

```
                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 ageing value      *
                PSBCHK=NO,        PSB resource checking required        *
                PSBPL=4,          PSB pool size in 1024-byte blocks     *
                PSDINT=0,         Persistent Session Delay Interval     *
+               PSTYPE=SNPS,      VTAM Single Node Persistent Sessions  *
                PVDELAY=30,       Timeout value for LUIT Table          *
                RAMAX=256,        Max. I/O area for RECEIVE ANY         *
#               RAPOOL=50,        Max. RECEIVE ANY Request Parm.Lists   *
                RENTPGM=PROTECT,  Reentrant program write protection    *
                RESP=FME,         Logical unit response type            *
                RESSEC=ASIS,      Resource security check               *
                RMTRAN=CSGM,      XRF alternate recovery transaction    *
                RST=NO,           Recovery service table (XRF-DBCTL)    *
                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            *
                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           *
```

Figure 48 (Part 4 of 6). DFHSIT, the pregenerated default system initialization table

```
                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=YES,          System recovery table option/suffix   *
                START=AUTO,       CICS system initialization option     *
                STARTER=YES,      Starter ($ and #) suffixes option     *
                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=,         Transaction backout exit programs     *
                TCAM=NO,          TCAM option                           *
                TCP=YES,          Terminal control program option/suffix*
|               TCSACTN=NONE,     TC Shutdown action            @BA59626*
|               TCSWAIT=4,        TC Shutdown wait              @BA59626*
                TCT=YES,          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    *
                TRAP=OFF,         F.E. global trap exit option          *
|               TRANISO=NO,       Transaction Isolation                 *
+               TRDUMAX=999,      No of TRANDUMPS to be taken           *
                TRTABSZ=16,       Internal trace table size in 1K bytes *
+               TRTRANSZ=40,      Transaction dump trace table size     *
+               TRTRANTY=TRAN,    Transaction dump trace option         *
                TS=(,3,3),        Temporary storage buffers and strings *
                TSMGSET=4,        # of entries for pointers to TS MSGset*
```

Figure 48 (Part 5 of 6). DFHSIT, the pregenerated default system initialization table

```
                TST=NO,           Temporary storage table option/suffix *
                USERTR=ON,        Master user trace flag                *
                USRDELAY=30,      Timeout value for User Dir. Entries   *
                VTAM=YES,         VTAM access method option             *
+               VTPREFIX=\        Client virtual terminal prefix        *
                WRKAREA=512,      Common work area (CWA) size in bytes  *
                XAPPC=NO,         RACF class APPCLU required            *
                XCMD=YES,         SPI use default name for RACF check   *
                XDCT=YES,         DCT use default name for RACF check   *
                XFCT=YES,         FCT use default name for RACF check   *
                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   *
                XRF=NO,           Extended recovery feature (XRF) option*
                XRFSOFF=NOFORCE,  XRF -  Re-sign on after takeover      *
                XRFSTME=5,        XRF -  sign off timeout value         *
|               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
          END   DFHSITBA
```

Figure 48 (Part 6 of 6). DFHSIT, the pregenerated default system initialization table

Assembling the SIT

When you have coded the DFHSIT macro, you must assemble and link-edit the table into an APF-authorized library, such as CICS410.SDFHAUTH, and include this library in the STEPLIB concatenation of the startup job stream. For information about assembling and link-editing CICS control tables, see Chapter 2, “Defining resources in CICS control tables” on page 7. For an explanation of the syntax notation used to describe CICS macros, see the manual.

Initialization parameters that cannot be coded in the DFHSIT macro

There are some CICS system initialization parameters that you cannot define in the DFHSIT macro. These parameters can only be supplied at CICS startup time in any of these 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.

The system initialization parameters that you cannot define in the DFHSIT macro are shown in Figure 49 on page 226.

For information about coding system initialization parameters in PARM, SYSIN, or at the console, see Chapter 22, “Processing system initialization parameters” on page 321.

```
PN70228

The following change was made by APAR PN70228.

+    CDSASZE={0K|number}
     CHKSTRM={CURRENT|NONE}
     CHKSTSK={ALL|CURRENT|NONE}

PN88030

ECDSASZE, ERDSASZE, ESDSASZE and EUDSASZE have been added by
APAR PN88030.

#    ECDSASZE={0K|number}
#    ERDSASZE={0K|number}
#    ESDSASZE={0K|number}
#    EUDSASZE={0K|number}
     JSTATUS=RESET
|    NEWSIT={YES|NO}
     PRVMOD={name|(name,name...name)}

PN70228

The following change was made by APAR PN70228.

+    RDSASZE={0K|number}
+    SDSASZE={0K|number}
     SERIES=PURGE
     SIT=xx
     SPCTRxx={(1[,2][,3])|ALL|OFF}
     START=LOGTERM
     START=(option,ALL)
     STNTRxx={(1[,2][,3])|ALL|OFF}

PN70228

The following change was made by APAR PN70228.

+    UDSASZE={0K|number}
```

Figure 49. System initialization parameters you cannot code in the DFHSIT macro

Notes on CICS resource table and module keywords

Table 29 on page 227 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 29. Summary of resources with a suffix, a dummy load module, or a COLD option

DFHSIT keyword	Default [1]	Suffix [2]	Dummy [3]	COLD start [4]
BMS	FULL	-	-	COLD
CLT	-	xx	-	-
DBP	-	xx	-	-
DCT	YES	xx	-	COLD
DDIR	YES	xx	-	-
DIP	NO	-	program	-
DL1 or DLI	NO	-	-	COLD
FCT	YES	xx	-	-
ICP	-	-	-	COLD
JCT	YES	xx	program	-
MCT	NO	xx	[5]	-
PDIR	YES	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	-	-

Notes:

[1] 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:

```
DBP=, or DBP=YES
FCT=, or FCT=YES
```

blanks are appended to DFHDBP and DFHFCT respectively, 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.

2 The **Suffix** column indicates whether you can code a suffix. (xx indicates that a suffix can be coded.)

The DBP keyword is mandatory; you must code either a suffix or YES for DBP. For more information about the DBP system initialization parameter, see page 244.

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. For example, DBP=2$ causes DFHDBP2$ dynamic backout program to be included in your CICS region.

When the suffix option is specified with other values, the two values must be enclosed within parentheses: for example, DCT=(xx,COLD).

3 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 318.

4 The **COLD start** column indicates whether the resource can be forced to start COLD. (COLD indicates that the resource can be cold started individually).

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 294.

For more information about CICS table and program selection, see “Selecting versions of CICS programs and tables” on page 318.

5 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

Unless otherwise stated, all of the system initialization parameters described here can be defined to CICS by any of these four 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.

Default notation

Default values are **underscored**; for example, TYPE=**CSECT**. This notation applies to the SIT macro parameters only.

TYPE={CSECT|DSECT}

Indicates the type of SIT to be generated.

CSECT

A regular control section that is normally used.

DSECT

A dummy control section.

ADI={30|number}

Code this parameter for an alternate CICS region when you are running CICS with XRF. It specifies the alternate delay interval in seconds. 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: If you are using MVS/ESA SP 4.1 or later, 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.

+ **AIEXIT={DFHZATDX|DFHZATDY|name}**

+ Code this parameter with the name of the autoinstall user-replaceable program that you want CICS to use when autoinstalling local VTAM terminals, APPC connections, and shipped terminals and connections. Autoinstall is the process of installing resource definitions automatically, using VTAM logon or BIND data, model definitions, and an autoinstall program.

+

Important

+ 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/ESA Resource Definition Guide*.

+ **DFHZATDX**

+ A CICS-supplied autoinstall user program. This is the default. It installs definitions for locally-attached VTAM terminals, remote shipped terminals, and remote shipped connections.

+ **DFHZATDY**

+ A CICS-supplied autoinstall user program. It installs definitions for locally-attached VTAM terminals, local APPC connections, remote shipped terminals, and remote shipped connections.

+ **name**

+ The name of your own customized autoinstall program, which may be based on one of the supplied sample programs. For programming information about writing your own autoinstall program, see the *CICS/ESA Customization Guide*.

AILDELAY={0|hhmmss}

Code this parameter with the delay period that elapses after a session between CICS and an autoinstalled terminal is ended, before the terminal entry is deleted. A session is ended when a terminal logs off or when a transaction disconnects a terminal from CICS.

hhmmss

Specify a 1-to 6-digit number. The default is 0, meaning the terminal entry is deleted as soon as the session is ended. If you leave out the leading zeros, they are supplied (for example, 123 becomes 000123, that is, 1 minute 23 seconds).

Note: The AILDELAY parameter does not apply to autoinstall of APPC connections, because they are not deleted.

AIQMAX={100|number}

Code this parameter with the maximum number of VTAM terminals and APPC connections that can be queued concurrently for autoinstall.

number

A number in the range 0 through 999. The default is 100.

A zero value disables the autoinstall function.

You should 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/ESA Performance Guide*.

AIRDELAY={700|hhmmss}

Code this parameter with the delay period that elapses after an emergency restart before autoinstalled terminal entries that are not in session are deleted.

PQ01573

The following change was made by APAR PQ01573.

+ The AIRDELAY parameter also applies when CEMT SET VTAM OPEN is
+ issued 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.

hhmmss

Specify 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 terminal 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/ESA Performance Guide*.

Note: The AIRDELAY parameter does not apply to autoinstall of APPC connections, because they are not cataloged.

XRF restriction

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.

For background information, see the *CICS/ESA 3.3 XRF Guide*.

AKPFREQ={200|number}

If AKPFREQ is a number other than zero, it specifies the number of consecutive blocks, written by DFHJCP to the system log data set, that triggers the activity keypoint function. The minimum number that should be coded is 200 (the default) and the maximum number is 65535. (The CICS region must support activity keypointing: that is, the CSKP transaction and DFHAKP program must be defined. For information about supporting activity keypointing, see the *CICS/ESA Recovery and Restart Guide* and the *CICS/ESA Performance Guide*.)

If AKPFREQ=0 is coded, no activity keypoints are taken and a subsequent emergency restart is not possible. This has serious effects on BWO support, because it prevents tie-up records being written to the forward recovery logs and the data set recovery point from being updated. Therefore, for forward recovery to take place, all forward recovery logs must be kept since the data set was first opened for update. For more information about the effect of AKPFREQ=0 on BWO, see "Effect of disabling activity keypointing" on page 108.

APPLID={DBDCCICS|applid}

The VTAM application identifier for this CICS region.

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/ESA Installation Guide*.

When you define this CICS region to another CICS region, in a CONNECTION definition you specify the applid as the NETNAME. 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)

This specifies the generic and specific XRF applids for the CICS region. Both applids must be 1 through 8 characters.

generic_applid

This is 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 *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 a CONNECTION definition you specify the generic applid as the NETNAME.

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.

specific_applid
This identifies 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. For background information about using this command, see the *CICS/ESA 3.3 XRF Guide*.

AUTCONN={0|hhmmss}
Specify this to delay the reconnection of terminals after an XRF takeover, to allow time for manual switching. The delay is hh hours, mm minutes, 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.

AUXTR={OFF|ON}
Code this parameter to indicate whether the auxiliary trace destination is to be activated at system initialization. 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 264.

AUXTRSW={NO|ALL|NEXT}
Code this parameter to indicate whether you want the auxiliary trace autoswitch facility.

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=({MINIMUM|STANDARD|FULL}[,COLD][,{UNALIGN|ALIGN}]
[,{DDS|NODDS}])

Code this parameter to specify which version of basic mapping support you want to be included. The function included in each version of BMS is shown in Table 30 on page 234. The parameter BMS can be overridden during CICS initialization.

You need 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 is the default in the SIT.

COLD
CICS deletes delayed messages from temporary storage, and destroys their interval control elements (ICEs).

UNALIGN
Code this parameter to indicate 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
Code this to indicate that 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 30. 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 *CICS/ESA Application Programming Guide*	All function of MINIMUM, ***plus*** 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 *CICS/ESA Application Programming Guide*	Same as STANDARD, ***plus*** terminal operator paging, cumulative mapping, page overflow, cumulative text processing, routing, message switching returning BMS-generated data stream to program before output.

PN70228

The following change was made by APAR PN70228.

CDSASZE={0K|number}

PN88030

The description of CDSASZE has been changed by APAR PN88030.

Code this parameter with the size of the CDSA. 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 code the CDSASZE parameter in PARM, SYSIN, or CONSOLE only.

CHKSTRM={CURRENT|NONE}

Activate, or deactivate, terminal storage-violation checking. The operands have the following meanings:

CURRENT
Code CURRENT to check for TIOA storage violations.

NONE
Code NONE to deactivate TIOA storage-violation checking.

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/ESA Problem Determination Guide*.

Restrictions

You can code the CHKSTRM parameter in PARM, SYSIN, or CONSOLE only.

CHKSTSK={ALL|CURRENT|NONE}

Activate, or deactivate, task storage-violation checking at startup. The operands have the following meanings:

ALL
Code ALL to check all storage areas on the transaction storage chains for all tasks.

CURRENT
Code CURRENT to check all storage areas on the transaction storage chain for the current task only.

NONE
Code NONE to deactivate task storage-violation checking.

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/ESA Problem Determination Guide*.

Restrictions

You can code the CHKSTSK parameter in PARM, SYSIN, or CONSOLE only.

CICSSVC={216|number}

Specify the number that you have assigned to the CICS type 3 SVC. 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/ESA Installation Guide*.

CICS/ESA 4.1 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 page 275.

CLSDSTP={NOTIFY|NONOTIFY}

Code this parameter to specify the notification required for an EXEC CICS ISSUE PASS command. 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/ESA 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=xx

Specify the suffix for the command list table (CLT), if this SIT is used by an alternate XRF system. The name of the table is DFHCLTxx.

For information about coding the macros for this table, see the *CICS/ESA Resource Definition Guide* manual.

CMDPROT={YES|NO}

Code this parameter to allow, or inhibit, CICS validation of start addresses of storage referenced as output parameters on EXEC CICS commands.

YES

If you specify 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 31 on page 237.

NO
If you specify 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 31. 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=YES)	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={ASIS|ALWAYS}
Specifies whether or not you want CICS to honor the CMDSEC option specified on a transaction's resource definition.

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
means that CICS overrides the CMDSEC option, and always calls its command security checking routine to issue the appropriate call to the SAF interface.

Notes:

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 code the CMDSEC parameter in the SIT, PARM, or SYSIN only.

CONFDATA={SHOW|HIDETC}

Code this parameter to indicate whether CICS is to suppress (hide) user data that might otherwise appear in CICS trace entries or in dumps that contain the VTAM receive any input area (RAIA.). This option applies to initial input data received on a VTAM RECEIVE ANY operation, the initial input data received on an MRO link, and FEPI screens and RPLAREAs.

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

This specifies that you want CICS to 'hide' user data from CICS trace entries. It also indicates that VTAM RAIAs are to be suppressed from CICS dumps. The action actually taken by CICS is subject to the individual CONFDATA attribute on the transaction resource definition (see Table 32 on page 239).

If you specify CONFDATA=HIDETC, CICS processes VTAM, MRO, 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).

- **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 and MRO data.

Modified data: By waiting until the transaction has been identified to determine the CONFDATA option, VTAM or MRO 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 32.

Table 32. Effect of CONFDATA system initialization and transaction definition parameters

CONFDATA on transaction	CONFDATA system initialization parameter	
	SHOW	**HIDETC**
NO	Data not suppressed	VTAM RAIAs are cleared. Initial input of VTAM and MRO data is suppressed from the normal FC90, FC91, DD16, DD23, and DD25 trace entries. For FC90 and DD16 traces only, suppressed user data is traced separately in an FC92 trace entry. FEPI screens and RPLAREAs are traced as normal.
YES	Data not suppressed	VTAM RAIAs are cleared. All VTAM, MRO, and FEPI user data is suppressed from trace entries.

You cannot modify the CONFDATA option while CICS is running. You must restart CICS to make such a change.

Restrictions

You can code the CONFDATA parameter in the SIT, PARM, and SYSIN only.

CONFTXT={NO|YES}

Code this parameter to indicate whether CICS is to prevent VTAM from tracing user data.

NO

CICS does not prevent VTAM from tracing user data.

YES

CICS prevents VTAM from tracing user data.

Restrictions

You can code the CONFTXT parameter in the SIT, PARM, and SYSIN only.

CSDACC={READWRITE|READONLY}

Code this parameter to specify the type of access to the CSD to be permitted to this CICS region. Note that 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 an 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={STATIC|DYNAMIC}

Code this parameter to specify whether or not the CSD is eligible for BWO. If BWO is wanted then 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 150.

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=number

Code this parameter with the number of buffers to be used for CSD data. 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 start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDBUFNI=number

Code this parameter with the number of buffers to be used for the CSD index. 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 start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDDISP={OLD|SHR}

Code this to set the disposition of the data set to be allocated to the CSD. 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 start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDDSN=name

Code this parameter with 1 to 44 characters to specify the JCL data set name (DSNAME) to be used for the CSD. 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 DSNAME. If a DD statement exists in the JCL of the CICS startup job, it takes precedence over this DSNAME.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDFRLOG=number

Code this parameter with a journal identifier to indicate the journal that you want to use for forward recovery of the CSD. This parameter is used only if CSDRECOV=ALL is specified, otherwise it is ignored. If you omit CSDFRLOG, but specify CSDRECOV=ALL, CSDFRLOG defaults to 1, which indicates that the CICS system log is to be used for forward recovery of the CSD.

| 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 150.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

number
The journal identification of the journal that is to be used for forward recovery. The number must be in the range 1 through 99. The number 1 indicates the CICS system log, and any other number refers to a user journal.

CSDJID={NO|number}
Code this parameter with the journal identifier of the journal that you want CICS to use for automatic journaling of file requests against the CSD.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

NO
Code NO if 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. The number 1 indicates the CICS system log, and any other number refers to another CICS journal.

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/ESA Resource Definition Guide*.

CSDLSRNO={1|number|NONE|NO}
Code this parameter to specify whether the CSD is to be associated with a local shared resource (LSR) pool.

This parameter is effective only on a CICS cold 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
Code NONE (or NO) if the CSD is not to be associated with a local shared resource pool.

CSDRECOV={NONE|ALL|BACKOUTONLY}
Code this parameter to indicate whether the CSD is a recoverable file.

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 150.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

NONE
The default is that the CSD is not recoverable.

ALL
Code ALL to specify that 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. If you do not code the CSDFRLOG parameter, CICS uses the system log for forward recovery.

Note: For backout purposes, CICS always uses the system log. (See the BACKOUTONLY option.)

BACKOUTONLY
Code BACKOUTONLY to limit CSD recovery to file backout only. If you specify backout for the CSD, CICS uses the system log to record before images for backout purposes.

CSDSTRNO={2|number}
Code this parameter with the number of concurrent requests that can be processed against the CSD. 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” on page 144 before you code this parameter.

This parameter is effective only on a CICS cold 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.

2 The minimum number of concurrent requests for the CSD is 2.

number
This number must be a multiple of 2, in the range 2 through 254.

CWAKEY={USER|CICS}
Code this parameter to specify the storage key for the common work area (CWA) if you are operating CICS with storage protection (STGPROT=YES). (You specify how much storage you want for the CWA on the WRKAREA parameter.) The permitted values are USER (the default), or CICS:

USER If you specify USER, or allow this parameter to default, CICS obtains storage for the CWA in user key. This allows a user program executing in any key to modify the CWA.

CICS If you specify 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.

PN61792

The DAE parameter has been added by PN61792.

DAE={NO|YES}
specifies the default DAE action when new system dump table entries are created.

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/ESA System Programming Reference* manual.

DATFORM={MMDDYY|DDMMYY|YYMMDD}
Specifies the external date display standard that you want to use for CICS date displays. 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 0CYYDDD 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.

PQ00059

The following change was made by APAR PQ00059.

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.

DBP={1$|2$|xx|YES}
Code this parameter to indicate which version of the dynamic transaction backout program is to be part of the system. This DBP parameter is mandatory, and has no default. (See page 226 for more information about coding this parameter.)

Note: If the JCT parameter in the SIT has a value of NO, no recovery of any sort is performed.

If you are using local DL/I support, specify DBP=2$. If you have generated a user-defined DFHDBP program for use with local DL/I support, specify either DBP=xx, where xx is the suffix of your program, or DBP=YES if you have defined an unsuffixed version of DFHDBP.

If you are not using local DL/I support, specify DBP=1$. You should also specify DBP=1$ if you are using only remote DL/I support or DBCTL.

For background information about dynamic transaction backout, see the *CICS/ESA Recovery and Restart Guide*.

The dynamic transaction backout program needs a program resource definition entry in the CSD. This entry must be included in your GRPLIST list at CICS initialization. The CICS-supplied programs DFHDBP1$ and DFHDBP2$ have resource definition entries in the CSD group DFHBACK, which is included in the default group list DFHLIST. If you use your own dynamic backout program, you must create a program resource definition entry for it in the CSD, and include the entry in your GRPLIST list at CICS initialization. You are recommended to code RESIDENT(YES) on your program definition. For information about the required program resource definition entry, see the *CICS/ESA Resource Definition Guide*.

Restrictions

You must code a DBP parameter in the system initialization table, although you can override that DBP parameter in PARM, SYSIN, or CONSOLE.

DBUFSZ={500|number}

Code this parameter with the maximum size, in bytes, of the dynamic log buffer (needed for dynamic transaction backout) for each transaction. CICS initially allocates for each buffer half the maximum size that you specify, and subsequently uses the value specified to calculate the actual size of the dynamic buffer. The value can be in the range 6 bytes through 32000 bytes; the default is 500 bytes.

If the resultant dynamic buffer is too small, CICS spills the dynamic log data to a further area of main storage. (CICS allocates all dynamic log storage, including spilled buffers, above the 16MB line.)

For information on the dynamic log, and how you can tune this parameter, see the *CICS/ESA Performance Guide*.

DCT=({YES|xx|NO}[,COLD])

Code this parameter to specify the destination control table suffix. (See page 226.) For information about coding the macros for this table, see the *CICS/ESA Resource Definition Guide*.

DDIR={YES|xx}

Code this parameter with a suffix for the DDIR list. (See page 226.) A DDIR is a list of data management blocks that define for DL/I the physical characteristics of DL/I databases. This is applicable only if the local CICS-DL/I interface is to be used.

For information about coding the macros for a DDIR list, see the *CICS/ESA Resource Definition Guide*.

DFLTUSER={CICSUSER|userid}

Code this parameter to specify the RACF userid with the security attributes to be used for all terminal users who have not explicitly signed on (by the CESN transaction, the EXEC CICS SIGNON command, or by the preset security options of the TERMINAL resource definition).

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 code the DFLTUSER parameter in the SIT, PARM, or SYSIN only.

DIP={NO|YES}

Code YES to include the batch data interchange program, DFHDIP, which 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 page 226.)

DISMACP={YES|NO}

DISMACP=YES allows CICS to disable any transaction that terminates abnormally with an ASRD abend (caused by a user program invoking a CICS macro, or referencing the CSA or the TCA).

Note: DISMACP=YES has no effect if the ASRD abend is handled by an active abend exit.

DLDBRC={YES|NO}

Code this parameter to indicate whether DBRC is present when you are using CICS local DL/I support. You must code DLDBRC=YES if you are using database recovery control, or if you are using data sharing at any level.

Note that this parameter is applicable only if you are have also coded DLI=YES as a system initialization parameter.

Restrictions

You can code **DLDBRC=YES** in the SIT only.

You can code **DLDBRC=NO** in the SIT, PARM, SYSIN, or CONSOLE.

{DLI|DL1}=({NO|YES|REMOTE}[,COLD])

Code this parameter to indicate whether DL/I databases are to be accessed, using CICS local or remote DL/I support, during the execution of CICS.

This parameter selects only local and remote DL/I support that runs within the CICS address space. It does not affect the use of database control (DBCTL).

NO

You can access DBCTL databases ***only***.

YES

You can access local, remote DL/I, and DBCTL databases. You must also specify DBP=2$.

REMOTE

You can access remote DL/I and DBCTL databases. DFHDBP2$ is not required if you code DLI=REMOTE; specify DBP=1$.

COLD

If you code DLI=YES, you can specify the COLD option if you want to cold start the DL/I resources (the COLD option is not applicable with the DLI=REMOTE option). The default is the start option coded on the START parameter. If you specify COLD, DL/I databases are not backed out in the event of an emergency restart.

In a data-sharing environment, because a cold start would free any locks and authorizations held by IRLM and DBRC on behalf of CICS, and hence prevent a correct database recovery, do not specify COLD, either on this or the START system initialization parameter, unless recovery has already taken place.

+ **Note:** On initialization of an XRF alternate region, the DLI COLD option is
+ overridden so that DL/I is emergency restarted, and so DL/I databases are
+ backed out.

DLIOLIM={100|number}

| The maximum number of I/O errors allowed for each DL/I database. The value
| must be in the range 0 through 999999 (1 through 999999 when used as an
| override).

When this number is exceeded for a database, CICS tries to stop that database. Any new schedule requests for the database fail. Any tasks that have the database scheduled are allowed to continue until they issue a DLI TERM call or a CICS SYNCPOINT, or until the task finishes.

DLIRLM={YES|name|NO}

Code this parameter to indicate the name of the IMS resource lock manager.
| This parameter is applicable only if you have also coded DLI=YES. You do not
| need to code this parameter if you are using data sharing at the database level.

If you code DLIRLM=YES, the default name IRLM is assumed. If you are running CICS in an IMS data-sharing environment with share levels 2 or 3, you must have IRLM present.

Restrictions

You can code the DLIRLM parameter in the SIT, PARM, or SYSIN only.

DLLPA ={NO|YES}

Code this parameter to indicate that you want IMS modules that are used by CICS to be used from the link pack area, where possible. This reduces operating system paging.

Note: This parameter is applicable only if you have also coded DLI=YES.

DLMON={NO|YES}

Code this parameter if DL/I database monitoring using the IMS database monitor is to be active for this invocation of CICS. This parameter is applicable only if you have also coded DLI=YES.

For information about the IMS database monitor, see the *IMS System Administration Guide*.

The CICS system log does not accept output from the IMS database monitor. The user must provide a data set with a DD name of IMSMON for the output, as described in the *IMS Utilities Reference Manual*.

The IMS database monitor is designed to run in a batch environment. Unless DL/I is run single thread through CICS, carefully analyze the output produced by the monitor.

DLTHRED={1|number}
Code this parameter to specify the number of threads provided through the CICS local DL/I interface. This is applicable only if the CICS-DL/I interface is to be used. The number may be in the range 1 through 255. An interface thread is held from the time a PSB is scheduled until it is terminated (or the transaction ends). The value specified in this operand must be large enough to accommodate the online local DL/I usage plus the shared database requirements; that is, a number of threads equivalent to the number specified in the RECEIVECOUNT field of the SESSIONS entry for CONNECTION(@BCH). (For details of the CONNECTION-SESSIONS pair of definitions you need for CICS shared database, see “CICS shared database support” on page 73.)

DLXCPVR={NO|YES}
Code this parameter to indicate that you want to page-fix the ISAM or OSAM buffers. Although this uses more storage, it improves performance by reducing the amount of paging.

DMBPL={4|number}
Code this parameter with the data management block (DMB) pool size in 1024-byte blocks for CICS-DL/I interface support. The number of 1024-byte blocks specified must be in the range 1 through 9999.

The DMBPL parameter is applicable only if the CICS local DL/I interface is used. It limits the amount of storage allocated for DMBS at any time (actual allocations/deallocations are made as non-resident DMBs are opened).

For information about coding this parameter, see the BUFPOOLS system definition macro in the *IMS System Definition Reference* manual.

DSALIM={5M|number}
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 16MB boundary.

5M
The default DSA limit is 5MB (5 242 880).

number
Specify *number* as an amount of storage in the range 2MB to 16MB (2 097 152 bytes to 16 777 216 bytes) in multiples of 262 144 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, 4 194 304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

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 16MB 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 16MB boundary, and for CICS control blocks that reside below the 16MB boundary.

Notes:

1. CICS allocates the UDSA in multiples of 1MB when transaction isolation is active, but in multiples of 256KB in CICS regions without transaction isolation. The other DSAs below 16MB are allocated in multiples of 256KB, 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/ESA Performance Guide*.

DSHIPIDL={020000|hhmmss}
Specifies the minimum time, in hours, minutes, and seconds, that an *inactive* shipped terminal definition must remain installed in this region. 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 Specify a 1 to 6 digit number in the range 0–995959. Numbers that have fewer than six digits are padded with leading zeros.

DSHIPINT={120000|0|hhmmss}
Specifies the interval between invocations of the timeout delete mechanism. 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 Specify a 1 to 6 digit number in the range 1–995959. Numbers that have fewer than six digits are padded with leading zeros.

DTRTRAN={CRTX|name|NO}

This is the name of the transaction definition that you want CICS to use for dynamic transaction routing. 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 named on DTRTRAN must be installed in the CICS terminal-owning regions if you want to eliminate the need for resource definitions for individual transactions.

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 you own dynamic transaction routing definition, see the *CICS/ESA Resource Definition Guide*.

DTRPGM={DFHDYP|program-name}

Code this parameter with the name of the dynamic transaction routing program you want to use for routing transactions that are defined with the DYNAMIC attribute. DFHDYP, the default, is the name of the CICS-supplied version.

DUMP={YES|NO}

Code this parameter to specify whether the CICS dump domain is to take SDUMPs.

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 176.

DUMPDS={AUTO|A|B}

Code this parameter to specify the transaction dump data set that is to be opened during CICS initialization.

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={NO|NEXT}

Code this parameter to specify whether you want CICS to switch automatically to the next dump data set when the first is full.

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 177.

DURETRY={30|number-of-seconds|0}

Code this parameter to specify, in seconds, the total time that CICS is to continue trying to obtain a system dump using the SDUMP macro. 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 STIMERM 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.

PN88030

ECDSASZE has been added by APAR PN88030.

ECDSASZE={0K|number}
Code this parameter with the size of the ECDSA. 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 code the ECDSASZE parameter in PARM, SYSIN, or CONSOLE only.

| **EDSALIM={20M|number}**
| 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 16MB boundary.

| **20M**
| The default EDSA limit is 20MB (20 971 520 bytes).

| **number**
| Specify *number* as a value in the range 10MB to 2047MB, in multiples of 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, 33 554 432), or as a whole number of kilobytes (for example, 32 768K), or a whole number of megabytes (for example, 32M).

| 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 16MB 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 16MB boundary.

The extended read-only DSA (ERDSA)
The key-0 storage area for all reentrant programs and tables above the 16MB 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 16MB 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 16MB boundary, and CICS control blocks that reside above the 16MB boundary.

CICS allocates all the DSAs above the 16MB boundary in multiples of 1MB.

Note: For information about calculating the amount of storage to specify on the EDSALIM parameter, see the *CICS/ESA Performance Guide*. For example, the value that you specify must allow for all temporary storage MAIN requests, for which CICS uses the ECDSA.

ENQPL={2|number}
Code this parameter with the maximum size, in kilobytes, to be allocated by IMS for ENQ control block space when you are using CICS local DL/I support. ENQ control blocks are heavily used when program isolation scheduling (PISCHD=YES) is used for DL/I transactions. The range is 0 through 999.

For information about how to calculate the ENQPL value, see the *IMS System Definition Reference* manual.

EODI={E0|xx}
Code this parameter with the end-of-data indicator for input from sequential devices. 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 (\).

PN88030

ERDSASZE and ESDSASZE have been added by APAR PN88030.

ERDSASZE={0K|number}
Code this parameter with the size of the ERDSA. 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 code the ERDSASZE parameter in PARM, SYSIN, or CONSOLE only.

ESDSASZE={0K|number}
Code this parameter with the size of the ESDSA. 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 code the ESDSASZE parameter in PARM, SYSIN, or CONSOLE only.

ESMEXITS={NOINSTLN|INSTLN}
Code this parameter to specify whether installation data is to be passed via the RACROUTE interface to the external security manager (ESM) for use in exits written for the ESM.

NOINSTLN
Specifies that the INSTLN parameter is not used in RACROUTE macros.

INSTLN
Specifies that 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/ESA Customization Guide*. This data is intended for use in exits written for the ESM.

Restrictions

You can code the ESMEXITS parameter in the SIT only.

PN88030

EUDSASZE has been added by APAR PN88030.

EUDSASZE={0K|number}

Code this parameter with the size of the EUDSA. 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 code the EUDSASZE parameter in PARM, SYSIN, or CONSOLE only.

FCT={YES|xx|NO}

Code this parameter with the suffix of the file control table to be used.

This parameter is effective only on a CICS cold 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/ESA 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 (to allow migration of the FCT to the CSD) are ignored.

FEPI={NO|YES}

Code this parameter to specify whether or not you want to use the Front End Programming Interface feature (FEPI).

NO

Means that 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

Means you require FEPI support, and causes CICS to start the CSZI transaction.

This book does not contain any information about the installation process for the Front End Programming Interface feature. Installation information can be found in the *CICS/ESA Front End Programming Interface User's Guide*.

FLDSEP={'___'|'xxxx'

Code this parameter with one through four field-separator characters, each of which indicates end of field in the terminal input data. 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={‘ ’|‘x’}

Code this parameter with a single character to be the field-name-start character for free-form input for built-in functions. 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:

+ **( ) ' = ,**

+ **FSSTAFF={YES|NO}**

+ Code this parameter in an application-owning region (AOR) to prevent transactions initiated by function-shipped EXEC CICS START requests being started against incorrect terminals.

+ You may 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*, shipped back to a TOR, for routing, *across the link it was received on*,

Apar PQ07579

Documentation for Apar PQ07579 added 03/12/97

unless the request supplies a TOR_NETNAME and a remote terminal with the correct TOR_NETNAME is located.

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/ESA 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.

Notes:

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/ESA Customization Guide*.

GMTEXT={‘WELCOME TO CICS/ESA’|‘text’}

Code this parameter to indicate whether the default logon message text (WELCOME TO CICS/ESA) or your own message text is to be displayed on the screen 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 GMTEXT 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,

```
GMTEXT='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/ESA 4.1 SYSTEM    *
GMTEXT='An Information Development CICS/ESA 4.1 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/ESA 4.1 Terminal-Owning Region (TOR) - CICSIDC.
 This message is to show the use of continuation lines when creating a GMTEXT pa
rameter in the SYSIN data set 09:56:14
```

The CESN transaction displays this as follows:

```
                     Signon for CICS/ESA Release 4.1.0       APPLID CICSHTH1

An Information Development CICS/ESA 4.1 Terminal-Owning Region (TOR) - CICSIDC.
This message is to show the use of continuation lines when creating a GMTEXT
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/ESA Resource Definition Guide*.

GMTRAN={CSGM|CESN|transaction-id}

Apar PQ22632

Documentation for Apar PQ22632 added 23/04/99

Code this parameter to 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.

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.

PN89036

GNTRAN changed by APAR PN89036.

GNTRAN={NO|transaction_id}

Specifies the transaction that you want CICS to invoke when a user's terminal-timeout period expires.

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 sign-off, if the LOGONMSG(YES) option is specified in the TYPETERM resource definition for the terminal, the transaction specified in the GMTRAN system initialization parameter is executed.

transaction_id

The name of a timeout transaction to sign off 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, sign-off. 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 the *CICS/ESA 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.

PN74807

The following note was added by APAR PN74807.

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=name

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.

There is no default for GRNAME. If you don't specify GRNAME, CICS does not register itself with the VTAM generic resources function.

Notes:

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, you should not specify a value for the GRNAME system initialization parameter. Any value specified for GRNAME is set to blanks.
3. If you specify the XRF=NO parameter, and a value for GRNAME, you should 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.

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.

The examples used here are based on a CICS naming convention described in the *MVS Sysplex Application Migration* manual.

Note: There are rules that restrict CICS use of the VTAM generic resources function; for more information see the *CICS/ESA Intercommunication Guide*.

GRPLIST={DFHLIST|name|(name[,name2][,name3][,name4])}

Specifies the names (each 1 through 8 characters) of up to four lists of resource definition groups on the CICS system definition (CSD) file. 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 CEDA 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.

Notes:

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, CICSSD, and CICSHT3V in that order. If the order of concatenation is important (for example, to ensure that a particular resource definition overrides another), you should consider coding real group list names.
2. If a group list contains resource definitions that are needed by another group list, the prequisite group list 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. This may mean that you have 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, if you want to use the four group lists CICSHT#1, CICSHTAP, CICSHT3V, and CICSHTSD, you could specify either of the following system initialization parameters:

```
GRPLIST=(CICSHT#1,CICSHTAP,CICSHT3V,CICSHTSD)
GRPLIST=(CICSHT*)
```

In the first example GRPLIST, the group lists are loaded in the order specified, and resource definitions installed from the CICSHTSD group list will override any duplicate definitions installed by the other groups.

In the second example GRPLIST, the group lists are loaded in the order CICSHT#1, CICSHTAP, CICSHTSD, then CICSHT3V, and resource definitions installed from the CICSHT3V group list will override any duplicate definitions installed by the other groups.

If your SIT contains the parameter:

```
GRPLIST=(CICSHT#1,CICSAP*,CICSHT3V,CICSHTSD)
```

and you want to replace the list CICSHT3V with the list ANOLST05, you should 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/ESA Resource Definition Guide*.

GTFTR={OFF|ON}

Code this parameter to enable CICS to use the MVS generalized trace facility (GTF) as a destination for trace data.

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 *MVS/ESA Service Aids* manual.

HPO={NO|YES}

Code this parameter to indicate whether you want to use the VTAM authorized path feature of the high performance option (HPO). If you code YES, the CICS type 6 SVC must be link-edited in your MVS nucleus, and defined to MVS in an SVCPARM 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/ESA Installation Guide*.

Restrictions

You can code the HPO parameter in the system initialization table only.

ICP=COLD

Code this parameter if you want to cold start the interval control program. See page 226.

ICV={1000|number}

Code this parameter with the region exit time interval in milliseconds. 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/ESA 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={5000|number}

specifies the default runaway task time interval in milliseconds as a decimal number. You can code 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/ESA Resource Definition Guide*). 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/ESA Problem Determination Guide*.

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.

INITPARM=(pgmname_1='parmstring_1'[, ... ,pgmname_n='parmstring_n'])

Code this parameter in order to pass parameters to application programs that use the ASSIGN INITPARM command. For example, 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/ESA Application Programming Reference* manual.

pgmname

Code pgmname with the name of a program. This name must be 1 through 8 alphanumeric or national language characters.

parmstring

Code parmstring with 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 327.

You can specify up to 255 pgmname='parmstring' sets.

Note: You can specify the INITPARM keyword and its parameters more
than once, see 344.

INTTR={ON|OFF}

Code this parameter to specify whether the internal CICS trace destination is to be activated at system initialization.

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={NO|YES}
Code this parameter to indicate whether IRC is started up at system initialization. If IRCSTRT=YES is not coded, IRC can be initialized by issuing a CEMT or EXEC CICS SET IRC OPEN command.

ISC={NO|YES}
Code YES to include the CICS programs required for interregion or intersystem communication.

JCT={YES|xx|NO}
Code this parameter with the suffix of the journal control table to be used. This indicates whether journaling and volume control are to be used. Note that you must have journaling if you code XRF=YES. If you code JCT=NO, a dummy journal control program (DFHJCPDY) and a dummy volume management program (DFHVCPDY) are loaded, and no recovery of any sort is performed. (See also page 226.) If you code JCT=NO with START=AUTO CICS performs a COLD start.

For information about coding the macros for this table, see the *CICS/ESA Resource Definition Guide*.

JESDI={30|number}
Code this, in a SIT for an alternate XRF system, to specify the JES delay interval, in seconds, the minimum being 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. If you are not using MVS/ESA SP 4.1 or later and PR/SM, the JESDI value specifies the interval before the system operator becomes involved, because the cancellation process is not fully automatic (for example, if the primary MVS image fails).

Note: If you are using MVS/ESA SP 4.1 or later, 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.

JSTATUS=RESET
Code JSTATUS=RESET to reset the journal status of all journal data sets that are defined on disk (JTYPE=DISK1|DISK2), but are ***not*** specified with JOUROPT=AUTOARCH.

This parameter resets the journal status held in the global catalog to "ready for use". Before using the JSTATUS option at startup, ensure that any disk journals that contain essential information are archived, either to tape or another disk data set.

Restrictions

You can code the JSTATUS parameter in PARM, SYSIN, or CONSOLE only. It is not applicable if you are using the CICS automatic archiving facility.

LGNMSG={NO|YES}

Code this to indicate whether VTAM logon data is made available to an application program.

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/ESA Application Programming Reference* manual.

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={YES|NO|NEWCOPY}**

| Code this to indicate whether CICS is to use the LLACOPY macro or the | BLDL macro when locating modules in the DFHRPL concatenation.

| **YES**

| CICS always uses the LLACOPY macro when locating modules in the | DFHRPL concatenation.

| **NO**

| CICS always uses the BLDL macro when locating modules in the | DFHRPL 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 concatenation.

| **Notes:**

| 1. If you code LLACOPY=NO or LLACOPY=NEWCOPY you can still | benefit from having LLA managed data sets within your DFHRPL | 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.

LPA={NO|YES}

Code this parameter to indicate whether any CICS or user modules can be # used from the link pack areas.

NO

will not load CICS or user modules from the link pack areas.

YES

CICS or user modules installed in the LPA or 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 CICS410.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 281.

MCT={NO|YES|xx}

Code this parameter to specify the monitoring control table suffix, (see page 226.) 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 ed 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/ESA Operations and Utilities Guide*.

MN={OFF|ON}

Code this parameter to indicate whether monitoring is to be switched on or off at initialization, and use the individual monitoring class parameters to control which monitoring classes are to be active. (See the MNEVE, MNEXC, and MNPER 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. For details of the effect of monitoring status being on or off, in conjunction with the status of the various monitoring classes, see the following notes:

Notes:

1. If the monitoring status is ON, CICS accumulates monitoring data continuously and, depending on the status of each of the monitoring classes, processes the accumulated data as follows:
   - For the performance and exception monitoring classes, CICS writes the monitoring data for each class that is active to a system management facilities (SMF) data set.
   - For the SYSEVENT monitoring class, CICS notifies the MVS system resources manager (SRM) of the completion of each transaction. This data can be reported using the resource measurement facility (RMF), or written to SMF data sets, depending on the RMF options in force.

     For information about the effect of SYSEVENT recording in an MVS workload manager environment, see the *CICS/ESA Performance Guide*.

   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 an MN system initialization parameter in PARM, SYSIN, or through the system console.
- While CICS is running, by either of:
  - 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 operand 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.

For programming information about controlling CICS monitoring, see the *CICS/ESA System Programming Reference* manual.

MNCONV={NO|YES}

Specifies whether or not conversational tasks are to have separate performance class records produced for each pair of terminal control I/O requests.

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.

MNEVE={OFF|ON}

Code this parameter to indicate whether SYSEVENT monitoring is to be made active during CICS initialization. The monitoring SYSEVENT status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Set SYSEVENT monitoring to “not active.”

ON Set SYSEVENT monitoring to “active.”

For information about the implications for SYSEVENT recording in an MVS Workload Manager environment, see the *CICS/ESA Performance Guide*.

MNEXC={OFF|ON}

Code this parameter to indicate whether the monitoring exception class is to be made active during initialization. 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/ESA Customization Guide*.

MNFREQ={0|hhmmss}

Specifies the interval for which CICS automatically produces a transaction performance class record for any long-running transaction. The monitoring frequency value is recorded in the CICS global catalog for use during warm and emergency restarts.

0 means that no frequency monitoring is active.

hhmmss

is 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 001500–240000. Numbers that are fewer than six digits are padded with leading zeroes.

MNPER={OFF|ON}

Code this parameter to indicate whether the monitoring performance class is to be made active during CICS initialization. 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/ESA Customization Guide*.

MNSUBSYS={null|xxxx}

Specifies the 4-character name to be used as the subsystem identification in the monitoring SYSEVENT class records. If you do not specify a name, the subsystem identification defaults to the first four characters of the *name1* operand of the APPLID system initialization parameter. The monitoring subsystem id is recorded in the CICS global catalog for use during warm and emergency restarts.

For background information on the SYSEVENT class of monitoring data and the subsystem identification, and about the implications for SYSEVENT recording in a MVS Workload Manager environment, see the *CICS/ESA Performance Guide*.

MNSYNC={NO|YES}

Specifies whether or not you want CICS to produce a transaction performance class record when a transaction takes an implicit or explicit syncpoint (unit-of-work). 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={GMT|LOCAL}

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. 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/ESA System Programming Reference*.

MROBTCH={1|number}

Code this parameter to specify the number of events that must occur before CICS is posted for dispatch due to the batching mechanism. 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. 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/ESA Performance Guide*.

Apar pq20752

Documentation for Apar pq20751 added 24/11/98

MROFSE={NO|YES}

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.

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 is 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 long running mirror sections of the *CICS/ESA*
Intercommunication Guide and the *CICS/ESA Performance Guide*.

MROLRM={NO|YES}

Code this parameter to specify whether you want to establish an MRO long-running mirror task.

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/ESA Intercommunication Guide*.

MSGCASE={MIXED|UPPER}

CICS messages handled by the CICS message domain are in mixed case. Code the MSGCASE parameter to indicate how you want the message domain to display these mixed case messages.

MIXED This is the default in the SIT; all messages displayed by the CICS message domain remain in mixed case.

UPPER The message domain displays all mixed case messages in uppercase only.

Note: 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.

MSGLVL={1|0}

Code this parameter with the message level that controls the generation of messages to the console.

1 All messages are to be printed.

0 Only critical errors or interactive messages are to be printed.

| **MXT={5|number}**

| Specifies the maximum number, in the range 1 through 999, of ***user*** tasks
| CICS allows to exist at any time. 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:** The MXT value does **not** include CICS system 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 actually 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/ESA 4.1 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.

NATLANG=(E,x,y,z,...)

Specify on this parameter the single-character codes for the languages to be supported in this CICS run, selected from the codes in Table 33 on page 273.

E Code E for English, which is the *system* default (that is, is provided even if you do not specifically code E).

x,y,z,... Code 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 translation utility, DFHMEU, to create your own DFHMEyyx modules, see the *CICS/ESA 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 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 33. 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 DBCS	CHT	Traditional Chinese
C DBCS	CHS	Simplified Chinese
2	CSY	Czech
D	DAN	Danish
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 DBCS	JPN	Japanese
H DBCS	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

Note:

DBCS denotes Double-Byte Character Set languages.

The following language module suffixes are not supported by the DFHMEU utility program:

- E - English Master data sets
- K - Japanese data sets, where translation is performed by IBM.

The NATLANG code is used as the suffix of the message modules for the associated language.

| **NEWSIT={YES|NO}**

Specify NEWSIT=YES to cause CICS to load the specified SIT, and enforce the use of all system initialization parameters, modified by any system initialization parameters provided via PARM, SYSIN, or the system console, even in a warm start. 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 via PARM, SYSIN, or the console:

CSDACC
CSDBUFND
CSDBUFNI
CSDDISP
CSDDSN
CSDFRLOG
CSDJID
CSDLSRNO
CSDRECOV
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 "Classes of start and restart" on page 329.

Restrictions

You can code the NEWSIT parameter in PARM, SYSIN, or CONSOLE only.

OPERTIM={120|number}

Code this parameter with the write-to-operator timeout value, in the range 0 through 86400 seconds (24 hours). If an application program issues an EXEC CICS WRITE OPERATOR command that omits the TIMEOUT option, the OPERTIM system default applies. This is the maximum time (in seconds) that CICS waits for a reply before returning control to this transaction. For information about using the write-to-operator timeout value, see the *CICS/ESA Application Programming Reference*.

PN36044

The following change was made by APAR PN36044.

OPNDLIM={10|number} (Not required for currently supported releases of VTAM.)

Code this parameter with the open destination and close destination request limit. 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/ESA Performance Guide*.

PARMERR={INTERACT|IGNORE|ABEND}
Code this parameter to specify what action you want to follow if CICS detects incorrect system initialization parameter overrides during initialization.

Note: 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 via 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={30|decimal-value}
Use this parameter in a SIT for an active CICS region. It specifies the XRF primary delay interval, in seconds. 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={YES|xx}
Code this parameter with a suffix for the PDIR list. A PDIR is a list of program specification blocks (PSBs) that define, for DL/I, the use of databases by application programs. This is applicable only if the local CICS-DL/I interface, or DL/I remote support, is being used. (See also page 226.)

For information about coding the macros for this table, see the manual.

PGAICTLG={MODIFY|NONE|ALL}
Specifies whether autoinstalled program definitions should be cataloged. 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.

PGAIEXIT={DFHPGADX|name}
Specifies the name of the program autoinstall exit program. 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={INACTIVE|ACTIVE}
Specifies the state of the program autoinstall function at initialization. While CICS is running, you can set the status of program autoinstall dynamically, by using either the EXEC CICS SET SYSTEM or CEMT SET SYSTEM command.

INACTIVE
The program autoinstall function is disabled.

ACTIVE
The program autoinstall function is enabled.

PGCHAIN=character(s)
Code this parameter with the character string that is identified by terminal control as a BMS terminal page-chaining command. It can be 1 through 7 characters. For more information about the character string, see the notes on page 277.

PGCOPY=character(s)
Code this parameter with the character string that is identified by terminal control as a BMS command to copy output from one terminal to another. It can be 1 through 7 characters. For more information about the character string, see the notes on page 277.

PGPURGE=character(s)
Code this parameter with the character string that is identified by terminal control as a BMS terminal page-purge command. It can be 1 through 7 characters. For more information about the character string, see the notes on page 277.

PGRET=character(s)
The character string that is recognized by terminal control as a BMS terminal page-retrieval command. It can be 1 through 7 characters.

Notes:

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 the *CICS/ESA CICS-Supplied Transactions* manual.
5. CICS always processes a paging command entered by the operator before initiating a transaction in response to a macro request, that consists of either DFHBMS or DFHPC with the TRANSID operand coded.

PISCHD={YES|NO}

Code this parameter to indicate whether program isolation scheduling (PISCHD=YES) or segment intent scheduling (PISCHD=NO) is to be performed for transactions that access CICS local DL/I databases. The default is PISCHD=YES. If you have generated the CICS local DL/I interface to support IMS/ESA 4.1 and you are not using the IRLM, the PISCHD option is overridden, and program isolation scheduling is forced. It is therefore unnecessary to specify the PISCHD option if you are using IMS/ESA 4.1. If you use local DL/I and change from intent scheduling to program isolation scheduling you must specify the size of the ENQPOOL that you need, by using the system initialization parameter ENQPL.

The CSD entry for transactions that use program isolation scheduling may optionally specify RESTART(YES) so that a transaction that fails can be backed out dynamically and restarted automatically after such conditions as a transaction deadlock. For information about program isolation scheduling, see the *CICS/ESA Recovery and Restart Guide*.

PLTPI={NO|xx|YES}

Code this parameter with a program list table, which contains a list of programs to be executed in the final stages of system initialization. (See page 226.) You can use the system initialization parameter INITPARM to pass parameters to those programs.

For information about coding the macros for this table, see the manual.

PLTPISEC={NONE|CMDSEC|RESSEC|ALL}

Specifies whether or not you want CICS to perform command security or resource security checking for PLT programs during CICS initialization. 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

Specifies that you do not want any security checking on on PLT initialization programs.

CMDSEC

Specifies that you want CICS to perform command security checking only.

RESSEC

Specifies that you want CICS to perform resource security checking only.

ALL

Specifies that you want CICS to perform both command and resource security checking.

Restrictions

You can code the PLTPISEC parameter in the SIT, PARM, or SYSIN only.

PLTPIUSR=userid

Specifies the userid that CICS is to use for security checking for PLT programs that run during CICS initialization. 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 code the PLTPIUSR parameter in the SIT, PARM, or SYSIN only.

PLTSD={NO|xx|YES}

Code this parameter with a program list table that contains a list of programs to be executed during system termination. (See page 226.)

PRGDLAY={0|hhmm}

Code this parameter with 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. This time interval is coded 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 reinitialized.

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. For programming information about notification of undelivered messages, see the *CICS/ESA Application Programming Reference* manual.

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={NO|YES|PA1|PA2|PA3}

Code this parameter with the method of requesting printout of the contents of a 3270 screen.

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. This facility is not supported for TCAM devices.

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/ESA CICS-Supplied Transactions* manual.

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 information, refer to the *3270 Information Display System Component Description* manual.

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={32768|value}

Code this parameter with the number of milliseconds to be used in the priority aging algorithm for incrementing the priority of a task. 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={name|(name,name...name)}

Code the PRVMOD parameter with the name of those modules that are not to be used from the LPA.

The operand is a list of 1- to 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/ESA Installation Guide*.

Restrictions

You can code the PRVMOD parameter in PARM, SYSIN, or CONSOLE only.

PSBCHK={NO|YES}

Code this parameter if you want CICS 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).

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 code 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 313.

PSBPL={4|number}

Code this parameter with the program specification block (PSB) pool size in 1024-byte blocks for local CICS-DL/I interface support. The number of 1024-byte blocks must be in the range 1 through 9999.

PSBPL is applicable only if the local CICS-DL/I interface is being used. It limits the amount of storage allocated for PSBs at any given time (actual allocations/deallocations are made as non-resident PSBs are scheduled).

For information about coding PSBPL, see the *IMS System Definition Reference* manual.

PSDINT={0|hhmmss}

Specifies the persistent session delay interval. This delay interval specifies if, and for how long, VTAM is to hold sessions in a recovery-pending state if CICS fails. The value for hours can be in the range 0 through 23; the minutes and seconds in the range 00 through 59 inclusive.

This value can be overridden during CICS execution (and hence change the action taken by VTAM if CICS fails).

0 A zero value specifies that, if CICS fails, sessions are terminated. This is the default.

hhmmss

Specifies a persistent session delay interval from 1 second up to the maximum of 23 hours 59 minutes and 59 seconds. If CICS fails, VTAM holds sessions in recovery pending state for up to the interval specified on the PSDINT system initialization parameter.

Specify a 1-to-6 digit time in hours, minutes and seconds, up to the maximum time. 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.

The interval you specify must cover the time from when CICS fails to when the VTAM ACB is opened by CICS during the subsequent emergency restart.

VTAM holds all sessions in 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 PSDINT value used must take account of the types and numbers of sessions involved.

You must exercise care when specifying large PSDINT values because of the problems they may give in some environments, in particular:

- Dial-up sessions—real costs may be incurred
- LU6.2 sessions to other host systems—such systems may become stressed.

Notes:

1. When specifying a PSDINT value, you must consider the number and, more particularly, the nature of the sessions involved. If LU6.2 sessions to other host systems are retained in recovery pending state, the other host systems may experience excessive queuing delays. This point applies to LU6.1 sessions which are retained until restart (when they are unbound).
2. The PSDINT parameter is incompatible with the XRF=YES parameter. If XRF=YES is specified, the PSDINT parameter is ignored.

PQ01573

The following parameter was added by APAR PQ01573.

+ **PSTYPE={SNPS|MNPS}**

+ specifies whether CICS is running with VTAM Single Node Persistent
+ Sessions (SNPS) or Multi Node Persistent Sessions (MNPS). Code this

+ parameter if you are using VTAM MNPS and you wish to recover sessions
+ when the VTAM ACB is opened after a VTAM crash. APPC sync level 2
+ persisting sessions are unbound when the VTAM ACB is opened.

PVDELAY={30|number}

| Code this with the persistent verification delay as a value in the range 0
| through 10080 minutes (up to 7 days). 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/ESA*
| *Performance Guide*.

RAMAX={256|value}

Code this parameter with the size in bytes of the I/O area allocated for each RECEIVE ANY issued by CICS, in the range 0 through 32767 bytes.

| **Notes:**

| 1. If you are using APPC, do not code a value less than 256; otherwise,
| the results are unpredictable.
| 2. If you are using pipeline terminals, do not code a value that is less than
| the RUSIZE (from the CINIT), because pipelines cannot handle data
| longer than this.

For information about coding this parameter, see the *CICS/ESA Performance Guide*.

Apar PQ15635

Documentation for Apar PQ15635 added 09/10/98

RAPOOL={50|value1|(value1,value2)},FORCE

Code this parameter to determine the size of the CICS receive any pool. 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.

Apar PQ15635

Documentation for Apar PQ15635 added 09/10/98

The default for value1 in the DFHSIT macro is 50. 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.

Apar PQ15635

Documentation for Apar PQ15635 added 09/10/98

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

The number of RECEIVE ANYs needed depends on the expected activity of the system, the average transaction lifetime, and the MAXTASK value specified.

Apar PQ15635

Documentation for Apar PQ15635 added 09/10/98

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 or 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.

For information about coding this parameter, see the *CICS/ESA Performance Guide*.

PN70228

The following change was made by APAR PN70228.

RDSASZE={0K|number}

PN88030

The description of RDSASZE has been changed by APAR PN88030.

Code this parameter with the size of the RDSA. 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 code the RDSASZE parameter in PARM, SYSIN, or CONSOLE only.

RENTPGM={PROTECT|NOPROTECT}

| Code this parameter to specify whether you want CICS to allocate the read-only DSAs, RDSA and ERDSA, from read-only key-0 protected storage. 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={FME|RRN}

Code this parameter to specify the type of request that CICS terminal control receives from logical units.

FME Function management end is the default.

RRN Reached recovery node.

| **RESSEC={ASIS|ALWAYS}**

| Specifies whether or not you want CICS to honor the RESSEC option specified on a transaction's resource definition.

| **ASIS**

| means that 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
means that 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. You should be aware that using this option can significantly degrade performance.

Restrictions

You can code the RESSEC parameter in the SIT, PARM, or SYSIN only.

RMTRAN={{CSGM|name1}[,{CSGM|name2}])
Code this parameter with 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. 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.

RST={NO|xx|YES}
Recoverable service table suffix. (See page 226.) For information about coding the macros for this table, see the *CICS/ESA Resource Definition Guide*.

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/ESA CICS-IMS Database Control Guide*.

PN70228

The following change was made by APAR PN70228.

Apar 16844

Documentation for Apar 16844 added 17/09/98

RUWAPOOL={NO|YES}
specifies that CICS is to create a run-unit work area pool for each task.

NO CICS does not create a pool of storage at task initialization.

YES CICS creates a pool of storage at task initialization that can be reused by Language Environment-conforming application programs.

SDSASZE={0K|number}

+ **PN88030**

+ **The description of SDSASZE has been changed by APAR PN88030.**

Code this parameter with the size of the SDSA. 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 code the SDSASZE parameter in PARM, SYSIN, or
CONSOLE only.

| **SEC={YES|NO}**
Code this parameter to indicate what level of external security you want CICS to use.

YES
Code YES if 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

| For the results of the interaction between the access intent of the
| user application, and the permission defined to RACF, see
| Table 34 on page 288.

| **NO**
| Code NO if 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/ESA CICS-RACF Security Guide*.

Define whether to use RACF for resource level checking by using the XDCT, XFCT, XJCT, XPCT, XPPT, XPSB, and XTST system initialization parameters. Define whether to use RACF for transaction-attach security checking by using the XTRAN system initialization parameter. Define whether RACF 1.9 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/ESA CICS-RACF Security Guide*.

For programming information about the use of external security for CICS system commands, see the *CICS/ESA System Programming Reference* manual.

Table 34. Results of RACF authorization requests (with SEC=YES)

Access Permission defined to RACF for CICS user	Access intent in application	
	READ	UPDATE
NONE	Refused	Refused
READ	Permitted	Refused
UPDATE	Permitted	Permitted

Restrictions

You can code the SEC parameter in the SIT, PARM, or SYSIN only.

SECPRFX={NO|YES}

Specifies whether or not CICS prefixes the resource names in any authorization requests to RACF with a prefix corresponding to the RACF userid for the CICS region. The prefix to be used (the userid for the CICS region) is obtained by the DFHIRP module.

NO

CICS does not prefix the resource names in any authorization requests to RACF.

YES

The RACF userid is used as the prefix for CICS resources defined to RACF. CICS prefixes the resource name in any authorization requests to RACF with a prefix corresponding to the RACF userid of the CICS region, obtaining the userid as follows:

- If you start CICS as a job, the prefix corresponds to the USER operand coded on the JOB statement of the CICS startup job stream.
- If you start CICS as a started task, the prefix corresponds to the RACF userid associated with the name of the start procedure in the RACF ICHRIN03 table.

- If you start a CICS job without an associated RACF userid, the prefix defaults to CICS.

For information about using the PREFIX option, see the *CICS/ESA CICS-RACF Security Guide*.

Restrictions

You can code 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.

SERIES=PURGE

Code this parameter to purge from the global catalog all information in the tape volume descriptor lists for standard-labeled tape journals. The CICS volume manager maintains a tape-volume descriptor list for each journal that is defined on standard-labeled tape, each list containing the volume serial identifiers of the series of tapes that constitute the journal or system log.

For information about tape-volume descriptor lists, see the *CICS/ESA Recovery and Restart Guide*.

Restrictions

You can code the SERIES parameter in PARM, SYSIN, or CONSOLE only.

SIT=xx

Code this parameter to specify the suffix, if any, of the system initialization table that you want CICS to load at the start of initialization. If you omit this parameter, CICS loads the unsuffixed table, DFHSIT, which is pregenerated with all the default values. This default SIT (shown in "DFHSIT, the default system initialization table" on page 220) is in CICS410.SDFHAUTH, and its source, named DFHSIT$$, is in CICS410.SDFHSAMP.

Restrictions

You can code the system initialization parameter anywhere in PARM or SYSIN, or as the ***first*** parameter entry at the CONSOLE.

SKRxxxx='page-retrieval-command'

Code this parameter if a single-keystroke-retrieval operation is required. '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 are PA1 through PA3, and PF1 through PF36. Thus up to 39 keys can be specified in this way (each by a separate command).

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={NONE|CICS|MVSIMAGE|SYSPLEX}

Specifies whether or not a userid can be signed on to CICS more than once, within the scope of:

- A single CICS region
- A single MVS image
- A sysplex.

Apar PQ12891

Documentation for Apar PQ12891 added 06/03/98

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. See the *OS/390: MVS Programming: Authorized Assembler Services Guide* for guidance on increasing the MVS ENQ limit.

NONE
Each userid 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/ESA 4.1.

CICS
Each userid 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 userid 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 userid 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 userids 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 userids, specified in the USERID option of the DEFINE TERMINAL command.
- Userids for remote users, received in attach headers.
- Userids for link security. For information about which userid is used for link security on a specific connection, see the *CICS/ESA CICS-RACF Security Guide*.
- The userid specified on the PLTPIUSR system initialization parameter.
- The CICS region userid.

Apar PQ08190

Documentation for Apar PQ08190 added 07/07/98

The signon SCOPE is enforced with the MVS ENQ macro. MVS places 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 messages which describe why the signon attempt is being rejected. See the OS/390: MVS Programming: Authorized Assembler Services Guide for guidance on increasing the MVS ENQ limit.

Restrictions

You can code the SNSCOPE parameter in the SIT, PARM, or SYSIN only.

SPCTR={(1,2|1[,2][,3])|ALL|OFF}

Code this parameter to set the level of tracing for all CICS components used by a transaction, terminal, or both, selected for special tracing. If you want to set different tracing levels for an individual component of CICS, use the SPCTRxx system initialization parameter. You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels. For a list of all the available trace points and their level numbers, see the *CICS/ESA User's Handbook*. For information about the differences between special and standard CICS tracing, see the *CICS/ESA Problem Determination Guide*.

number Code the level numbers for the level of special tracing you want for all CICS components. The options are: 1, (1,2), or (1,2,3). The default, (1,2), specifies special tracing for levels 1 and 2 for all CICS components.

ALL Enables the special tracing facility for all available levels.

OFF Disables the special tracing facility.

SPCTRxx={(1,2|1[,2][,3])|ALL|OFF}

Code this parameter to set the level of tracing for a particular CICS component used by a transaction, terminal, or both, selected for special tracing. 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)). 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 code for xx on the SPCTRxx keyword are shown in Table 35 on page 292:

Table 35. CICS component names and abbreviations

Code	Component name	Code	Component name
AP	Application domain	BM	Basic mapping support
BF	Built-in function	CP	Common programming interface
DC	Dump compatibility layer	DD	Directory manager domain
DI	Batch data interchange	DM	Domain manager domain
DS	Dispatcher domain	DU	Dump domain
EI	Exec interface	FC	File control
GC	Global catalog domain	IC	Interval control
IS	Inter-system communication	JC	Journal control
KC	Task control	KE	Kernel
LC	Local catalog domain	LD	Loader domain
LM	Lock domain	ME	Message domain
MN	Monitoring domain	PA	Parameter domain
PC	Program control	PG	Program manager domain
SC	Storage control	SM	Storage domain
SP	Sync point	ST	Statistics domain
SZ	Front end programming interface	TC	Terminal control
TD	Transient data	TI	Timer domain
TR	Trace domain	TS	Temporary storage
UE	User exit interface	US	User domain
XM	Transaction manager domain	XS	Security manager domain

Note: The component codes BF, BM, CP, DC, DI, EI, FC, IC, IS, JC, KC, PC, SC, SP, TC, TD, TS, and UE are sub-components of the AP domain. As such, the corresponding trace entries will be produced with a point ID of AP nnnn.

For details of using trace, see the *CICS/ESA Problem Determination Guide*.

number Code the level numbers for the level of special tracing you want for the CICS component indicated by xx. The options are: 1, (1,2), or (1,2,3).

ALL Code ALL to indicate that you want all the available levels of special CICS tracing switched on for the specified component.

OFF Code OFF to switch off all levels of special CICS tracing for the CICS component indicated by xx.

Restrictions

You can code the SPCTRxx parameter in PARM, SYSIN, or CONSOLE only.

SPOOL={NO|YES}

Code this parameter to specify whether the system spooling interface is required.

NO

The system spooling interface is not required.

YES

The system spooling interface is required.

+ **Note:** If you use the CICS spool interface, this makes use of the MVS exit
+ IDFDOIXT, which is provided in the SYS1.LINKLIB library. If you have a
+ high volume spool output, you should install the IDFDOIXT exit in a library
+ in the CICS STEPLIB concatenation, and consider having a PLT startup
+ program MVS load the exit during CICS initialization. This will help
+ optimize the performance of the CICS spool interface.

+ For further information about the MVS exit IEFDOIXT, see the *MVS/ESA*
+ *Installation Exits* manual, GC28-1637.

SRBSVC={215|number}

Specify the number that you have assigned to the CICS type 6 SVC. The default number is 215.

For information on changing the SVC number, see the *CICS/ESA Installation Guide*. A CICS type 6 SVC with the specified (or default) number must have been link-edited with the system nucleus.

SRT={YES|NO|xx}

Code this parameter with the system recovery table suffix. (See page 226.) For information about coding the macros for this table, see the *CICS/ESA Resource Definition Guide* manual.

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 clean-up operations before CICS terminates. Therefore, an SRT must be provided if recovery from either program checks or abnormal terminations, or both, is required.

START={AUTO|COLD|STANDBY|LOGTERM|}[,ALL])

Code this parameter with the type of start for the system initialization program. The value specified for START, or the default of AUTO, becomes the default value for each resource.

AUTO

CICS performs either a warm or an emergency restart, according to the status of the control record written to the global catalog by the previous execution of CICS. If the global catalog is newly initialized, it does not contain a control record, and CICS forces a cold start for START=AUTO.

If you code START=AUTO, you must provide a restart data set for use in case CICS has to perform an emergency restart, and the system log from the previous execution of CICS must be available. You must also have coded an activity keypoint value (see the AKPFREQ parameter on page 231) on the previous execution of CICS for an emergency restart to be successful.

COLD

A cold start. 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. However, some information saved in the global catalog and local catalog is preserved across cold starts. For information about how CICS uses information saved in the global catalog and local catalog, see the *CICS/ESA Recovery and Restart Guide*.

LOGTERM

LOGTERM is a special option for use as an alternative to a full emergency restart when the previous run terminated in an uncontrolled shutdown, and is available only if the CICS system log is defined on disk data sets. If you specify START=LOGTERM, CICS initializes up to the point where it writes an end of file on the system log. After it has written the end of file, CICS ends the emergency restart process and shuts down without doing any backout processing.

If you are using DBRC, DBRC is notified of the log closure.

For background information about this restart process, see the *CICS/ESA Recovery and Restart Guide*.

Restriction

START=LOGTERM is applicable only if XRF=NO is coded. You can code START=LOGTERM in PARM, SYSIN, or CONSOLE only.

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.)

+ If you have specified a COLD start for other CICS resources, for
+ example, DCT=(xx,COLD), they are cold started when the alternate
+ CICS region (with START=STANDBY specified) takes over. This may
+ cause CICS to lose data on an XRF takeover; for example, coding
+ ICP=COLD results in outstanding STARTs being lost. You are
+ recommended to code START=(STANDBY,ALL) to ensure a full
+ emergency restart during takeover, unless you wish to specifically cold
+ start individual resources.

(option,ALL)

The ALL operand 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, or cold). The ,ALL option overrides any individual settings in other system initialization parameters (for example, DCT=(xx,COLD)).

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 Table 29 on page 227.

Restrictions

You can code START=(option,ALL) in PARM, SYSIN, or CONSOLE only.

For more information about the types of CICS startup, see “Classes of start and restart” on page 329.

STARTER={NO|YES}

Code YES to indicate that the generation of starter system modules (with $ and # suffixes) is permitted, and various MNOTES are to be suppressed. This parameter should only be used when service is being performed on starter system modules.

Restrictions

You can code the STARTER parameter in the SIT only.

STATRCD=OFF|ON

+ Code this parameter to set the interval statistics recording status at CICS initialization. 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 colected 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 the *CICS/ESA CICS-Supplied Transactions* manual. For programming information about the EXEC CICS PERFORM commands, see the *CICS/ESA System Programming Reference* manual.

For information about the statistics utility program DFHSTUP, see the *CICS/ESA Operations and Utilities Guide*.

STGPROT={NO|YES}

Code this parameter to specify whether you want storage protection in the CICS region. 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 39 on page 362 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 39 on page 362 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.

STGRCVY={NO|YES}

Code this parameter to indicate whether CICS should try to recover from a storage violation.

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 the STGRCVY(YES) system initialization parameter, CICS resets the check zones correctly and the task continues running.

STNTR={ 1|(1[,2][,3])|ALL|OFF}

Code this parameter to indicate the level of standard tracing required for CICS as a whole. You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels.

Note: Before globally activating tracing levels 3 and ALL for the storage manager (SM) component, read the warning 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/ESA Problem Determination Guide*.

STNTRxx={(1|(1[,2][,3])|ALL|OFF}

Specifies the level of standard tracing you require for a particular CICS component. 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 code for xx on this STNTRxx keyword are shown in Table 35 on page 292.

number Code the level number(s) for the level of standard tracing you want for the CICS component indicated by xx. The options are: 1, (1,2), or (1,2,3).

ALL Code ALL to indicate that you want all the available levels of standard tracing switched on for the specified component.

OFF Code OFF to switch off all levels of standard CICS tracing for the CICS component indicated by xx.

Warning: If you select tracing levels 3 or ALL for the storage manager (SM) component, the performance of your CICS system will be degraded. This is because options 3 and ALL switch on levels of trace that are also used for field engineering purposes. See the *CICS/ESA Problem Determination Guide* for information about the effects of trace levels 3 and ALL.

Restrictions

You can code the STNTRxx parameter in PARM, SYSIN, or CONSOLE only.

SUBTSKS={0|1}

Code this parameter to define the number of task control blocks (TCBs) you want CICS to use for running tasks in concurrent mode. 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=xx

Code this parameter to define the last two characters of the name of this system initialization table.

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 code the SUFFIX parameter in the SIT only.

+ **SYDUMAX={999|number}**

+ Code this parameter to specify the limit on the number of system dumps
+ that may be taken for each dump table entry. If this number is exceeded,
+ subsequent system 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.

SYSIDNT={CICS|name}

Code this parameter with a 1- to 4-character name that is known only to your CICS region. 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/ESA Intercommunication Guide*.

SYSTR={ON|OFF}

Code this parameter to control the master system trace flag.

Code ON to obtain trace entries of CICS system activity. Entries are written to all the trace destinations that are active.

TAKEOVR={MANUAL|AUTO|COMMAND}

Use this parameter in the SIT for an alternate CICS region. It specifies the action to be taken by the alternate CICS region, following the (apparent) loss of the surveillance signal in the active CICS region. In doing this, it also specifies the level of operator involvement.

If both active and alternate CICS regions are running under different MVS/ESA SP 4.1 or later 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

This ensures that 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

This specifies that no operator approval, or intervention, is needed for a takeover.

COMMAND

This specifies that 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=([name1][,name2][,name3][,name4])

Code the names of your transaction backout exit programs. These exits are used for resource backout during emergency restart processing. For programming information about transaction backout exit programs, see the *CICS/ESA Customization Guide*. For background information about transaction backout, see the *CICS/ESA Recovery and Restart Guide*.

The order in which you code the names is critical. If you do not want to use all four exits, code commas in place of the ones you miss out. For example:

TBEXITS=(,,EXITF,EXITV)

name1 The name of your initialization/termination exit program.

name2 The name of your input exit program.

name3 The name of your file error exit program.

name4 The name of your open error exit program.

| If no transaction backout exit programs are required, you can do one of the following:

- Omit the whole parameter from the SIT
- Code TBEXITS=(,,,) as a SIT parameter
- Code TBEXITS=(,,,) as a SIT override.

TCAM={NO|YES}

Code this parameter to include TCAM support.

NO TCAM support is not to be included.

YES TCAM support is to be included.

TCP={YES|NO}

Code TCP=YES to include the pregenerated non-VTAM terminal control program, DFHTCP.

| You must code TCP=YES if you intend using card reader/line printer (sequential) devices.

Apar PQ15611

Documentation for Apar PQ15611 added 09/07/98

TCSACTN={NONE|UNBIND|FORCE}

Code this parameter with the required action that CICS terminal control should take if the terminal control shutdown wait threshold expires. 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 parameter only applies to VTAM terminals (including LU Type 6.2 single-session APPC terminals), not VTAM intersystem connections (LU Type 6.1 and LU Type 6.2 parallel connections). 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.

UNBIND
CICS terminal control attempts to force-close the session by issuing a VTAM CLSDST and sending an SNA UNBIND command to the hung terminal. This can be overridden by DFHZNEP.

Apar PQ15611

Documentation for Apar PQ15611 added 09/07/98

FORCE
CICS terminal control attempts to forceclose the CICS VTAM ACB if there are any hung terminals. All CICS VTAM terminals and sessions are released and CICS normal shutdown continues.

TCSWAIT={4|number|NO|NONE|0}
Code this parameter with the required CICS terminal control shutdown wait threshold. 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 session 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)).

TCT={YES|xx|NO}
Code this parameter to indicate which terminal control table, if any, is to be loaded. (See page 226.) For guidance about coding the macros for this table, see the *CICS/ESA 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 code TCT=NO. If you do this, note that a dummy TCT, called DFHTCTDY, is loaded during system initialization. For more information about DFHTCTDY, see page 320. (If you code TCT=NO, you must specify a CSD group list in the GRPLIST parameter.)

TCTUAKEY={USER|CICS}
Code this parameter to specify the storage key for the terminal control table user areas (TCTUAs) if you are operating CICS with storage protection (STGPROT=YES). The permitted values are USER (the default), or CICS:

USER If you specify USER, or allow this parameter to default, 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 If you specify 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 359 for more information about TCTUAs.

TCTUALOC={BELOW|ANY}

Code this parameter to indicate where terminal user areas (TCTUA) are to be stored.

BELOW

The TCTUAs are stored below the 16MB line.

ANY

The TCTUAs are stored anywhere in virtual storage. CICS stores TCTUAs above the 16MB line if possible.

For more information about TCTUAs, see “The terminal control table user areas” on page 359.

For details about defining terminals using RDO, see the *CICS/ESA Resource Definition Guide*.

| **TD=({3|decimal-value-1}[,{3|decimal-value-2}])**

| Specifies the number of VSAM buffers and strings to be used for intrapartition transient data (TD).

| **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 32767. The default value is 3.

| CICS obtains, above the 16MB line, storage for the TD buffers in units of the page size (4KB). 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 actually 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.

| **TRANISO={NO|YES}**
| Code this parameter, together with the STGPROT system initialization
| parameter, to specify whether you want transaction isolation in the CICS
| region. 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 as in earlier releases. If you
| specify STGPROT=YES and TRANISO=NO, CICS storage protection is
| active without transaction isolation.

| **YES**
| Specify YES for transaction isolation. 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).

TRAP={OFF|ON}
Code this parameter to indicate whether the FE global trap exit is to be activated at system initialization. This exit is for diagnostic use under the guidance of service personnel. For background information about this exit, see the *CICS/ESA Problem Determination Guide*.

+ **TRDUMAX={999|number}**
+ Code this parameter to specify the limit on the number of transaction
+ dumps that may be taken for each dump table entry. 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={16|number-of-kilobytes}
Code this parameter to specify the size in kilobytes of the internal trace table. (1KB = 1024 bytes.) The CICS trace table is allocated in virtual storage above the 16MB 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.

16 16KB is the default size of the trace table, and also the minimum size.

number The number of kilobytes of storage to be allocated for the internal trace table, in the range 16KB through 1048576KB. 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 (4KB), it is rounded up to the next multiple of 4KB.

Trace entries are of variable lengths, but the average length is approximately 100 bytes.

Note: To switch on internal tracing, use the INTTR parameter; for a description of INTTR, see page 264.

PN60636

SIT parameters TRTRANSZ and TRTRANTY added by PN60636.

+ **TRTRANSZ={40|number-of-kilobytes}**
+ specifies the size in kilobytes of the transaction dump trace table. (1KB = 1024 bytes.)

+ When a transaction dump is taken, CICS performs an MVS GETMAIN for storage above the 16MB line for the transaction dump trace table.

+ **40** 40KB is the default size of the transaction dump trace table. The minimum size is 16KB.

+ **number**
+ The number of kilobytes of storage to be allocated for the transaction dump trace table, in the range 16KB through 1048576KB.

+ **TRTRANTY={TRAN|ALL}**
+ specifies which trace entries should be copied from the internal trace table to the transaction dump trace table.

+ **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=([COLD][,{0|3|decimal-value-1}][,{3|decimal-value-2}])**
| Specifies:

| - Whether or not you want to cold start temporary storage
| - The number of VSAM buffers to be used for auxiliary temporary storage
| - The number of VSAM strings to be used for auxiliary temporary storage.

COLD
The type of start for the temporary storage program. If you do not want a cold start, code a comma before the second operand.

Note: IF ICP is warm-started (that is no ICP=COLD) and TS is cold-started, any ICEs and AIDs with data attached are lost.

0 No buffers are required; that is, only MAIN temporary storage is required. Code this as TS=(,0).

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.

CICS obtains, above the 16MB line, storage for the auxiliary temporary storage buffers in units of the page size (4KB). Because CICS optimizes the use of the storage obtained, TS may allocate more buffers than you specify, depending on the control interval (CI) size you have defined for the auxiliary temporary storage data set.

For example, if the CI size is 2048, and you specify 3 buffers (the default number), CICS actually allocates 4 buffers. This is because 2 pages (8192 bytes) are required to obtain sufficient storage for three 2048-byte buffers, a total of 6144 bytes, which would leave 2048 bytes of spare storage in the second page. In this case, CICS allocates another 2048-byte buffer to use all of the 8192 bytes obtained. 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 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=(COLD,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.

TSMGSET={4|number}
Code this parameter with the number of entries for which dynamic storage is allocated for storing pointers to records put to a temporary storage message set. When the entries are used, space is acquired for the same number of entries as many times as required to accommodate the total number of records in the queue. The range is 4 through 100.

PN70228

The following change was made by APAR PN70228.

TST={NO|YES|xx}
Code this parameter with the temporary storage table suffix. (See page 226.)

For information about coding the macros for this table, see the *CICS/ESA Resource Definition Guide*

+ **UDSASZE={0K|number}**
+

PN88030

+ **The description of UDSASZE has been changed by APAR**
+ **PN88030.**

Code this parameter with the size of the UDSA. 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 code the UDSASZE parameter in PARM, SYSIN, or
CONSOLE only.

USERTR={ON|OFF}

Code this parameter to set the master user trace flag on or 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/ESA Application Programming Reference* manual.

| **USRDELAY={30|number}**
| Specify the maximum time, in the range 0 through 10080 minutes (up to 7
| days), that an eligible userid and its associated attributes are to be retained
| in the user table if the userid is unused. An entry in the user table for a
| userid that is retained during the delay period can be reused.

| The userids eligible for reuse within the USRDELAY period are any that
| are:

| • Received from remote systems
| • Specified on SECURITYNAME in CONNECTION definitions
| • Specified on USERID in SESSIONS definitions
| • Specified on USERID on DFHDCT TYPE=INTRA definitions
• Specified or inherited on non-terminal STARTed transactions.

| Within the USRDELAY period, a userid in any one of these categories can
| be reused in one of the other categories, provided the request for reuse is
qualified with the same qualifiers. If a userid 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 userid 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 userids 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 definitions
+ for the connections used
+ - 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
+ sign-on for each incoming request (with I/O to RACF) and a full sign-off at
+ the end of each transaction. For function shipping, there may be multiple
+ sign-ons/sign-offs driven on a data-owning region for one task on an AOR.

+ **Note:** If a value, other than 0, is specified for USRDELAY, the ability to
+ change the user's attributes or revoke the userid becomes more difficult
+ because the userid and its attributes are retained in the region until the
+ USRDELAY value has expired. For example, if you have specified
+ USRDELAY=30 for a userid, but that userid continues to run transactions
+ every 25 minutes, the USRDELAY value will never expire and any changes
+ made to the userid will never come into effect.

| When running a remote transaction, a userid 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 userid
| has completed. When this event occurs, the userid is removed from the
| remote CICS region.

| For more information about the use of USRDELAY, see the *CICS/ESA
| Performance Guide*.

VTAM={YES|NO}
Code VTAM=NO only if you do not use the VTAM access method. The
default is VTAM=YES.

+ **VTPREFIX={\|character}**
+ Code this parameter with the first character to be used for the terminal
+ identifiers (termids) of autoinstalled virtual terminals. 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/ESA 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.

+ **Notes:**

+ 1. The autoinstall user program is not called at install of Client terminals,
+ so cannot be used to specify termids.

+ 2. 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.

+ 3. 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.

+ 4. Clients can override CICS/ESA-generated termids.

+ For further information about Client virtual terminals, see the *CICS/ESA
+ Intercommunication Guide*.

WRKAREA={512|number}
Code this parameter with the number of bytes to be allocated to the common work area (CWA). 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={NO|YES}
Code this parameter to specify whether RACF 1.9 session security can be used when establishing APPC sessions.

NO
RACF 1.9 session security cannot be used.

YES
RACF 1.9 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 code the XAPPC parameter in the SIT, PARM, or SYSIN only.

XCMD={YES|name|NO}

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. 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.

For information about preparing for and using security with CICS, see the *CICS/ESA CICS-RACF Security Guide*.

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 C*name* and the grouping class name is V*name*.

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 code the XCMD parameter in the SIT, PARM, or SYSIN only.

XDCT={YES|name|NO}

specifies whether you want CICS to perform transient data resource security checking. 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.

For information about preparing for and using security with CICS, see the *CICS/ESA CICS-RACF Security Guide*.

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 D*name* and the grouping class name is E*name*.

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 code the XDCT parameter in the SIT, PARM, or SYSIN only.

XFCT={YES|name|NO}

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. 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.

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.

For information about preparing for and using security with CICS, see the *CICS/ESA CICS-RACF Security Guide*.

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 F*name* and the grouping class name is H*name*.

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.

Restrictions

You can code the XFCT parameter in the SIT, PARM, or SYSIN only.

XJCT={YES|name|NO}
specifies whether you want CICS to perform journal resource security checking. 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.

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.

For information about preparing for and using security with CICS, see the *CICS/ESA CICS-RACF Security Guide*.

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.

Restrictions

You can code the XJCT parameter in the SIT, PARM, or SYSIN only.

XLT={NO|xx|YES}

Code this parameter with a suffix for the transaction list table. (See page 226.) The table contains a list of transactions that can be attached during the first quiesce stage of system termination.

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/ESA Resource Definition Guide*

XPCT={YES|name|NO}

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. 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.

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.

For information about preparing for and using security with CICS, see the *CICS/ESA CICS-RACF Security Guide*.

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.

Restrictions

You can code the XPCT parameter in the SIT, PARM, or SYSIN only.

XPPT={YES|name|NO}

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. Such 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.

For information about preparing for and using security with CICS, see the *CICS/ESA CICS-RACF Security Guide*.

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 M*name* and the grouping class name is N*name*.

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.

Restrictions

You can code the XPPT parameter in the SIT, PARM, or SYSIN only.

XPSB={YES|name|NO}

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. 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.

Notes:

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 further information about the PSBCHK system initialization parameter, see page 281.

For information about preparing for and using security with CICS, see the *CICS/ESA CICS-RACF Security Guide*.

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 P*name* and the grouping class name is Q*name*.

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.

Restrictions

You can code the XPSB parameter in the SIT, PARM, or SYSIN only.

XRF={NO|YES}
You must specify YES if you want XRF support to be included in the CICS region. 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 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. For background information about XRF, see the *CICS/ESA 3.3 XRF Guide*.

Note: You must code JCT=xx|YES, if you are using XRF.

XRFSOFF={NOFORCE|FORCE}
Code this parameter to control whether all users signed-on to the active CICS region are to remain signed-on following a takeover. This parameter is only applicable if you also code XRF=YES as a system initialization parameter.

NOFORCE
Code NOFORCE to allow CICS to determine sign-off according to the option set in either of:

- The CICS segment of the RACF database
- The TYPETERM definition for the user's terminal.

Note: For a terminal to remain signed-on after an XRF takeover, NOFORCE must be specified in the SIT, RACF database, and the terminal's TYPETERM definition.

FORCE
Code FORCE if you want ***all*** terminal users to be signed off in the event of a takeover by an alternate CICS region, regardless of individual options set in the RACF database or in the terminals' TYPETERM definitions.

For information about the XRFSOFF option on the CEDA DEFINE TYPETERM command, see the *CICS/ESA Resource Definition Guide*.

XRFSTME={5|decimal-value}
Code this parameter to define, in minutes, a time-out delay interval for users who are still signed-on when an XRF takeover occurs.

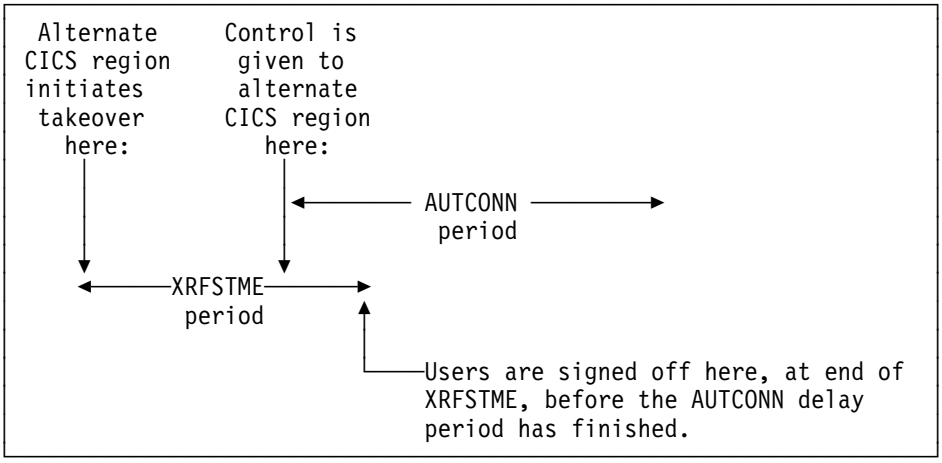
If you have specified NOFORCE in the RACF database, and in the terminals' TYPETERM definitions, and the takeover takes longer than the time specified in the XRFSTME, all users who are still signed-on after takeover are signed off.

5 Five minutes is the default value in the DFHSIT macro.

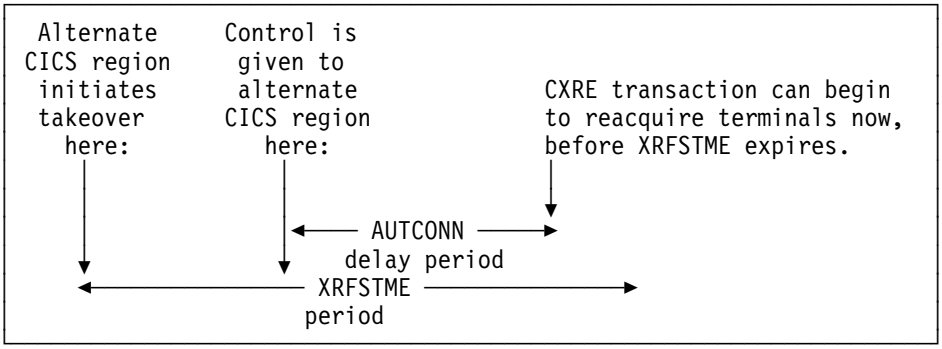
decimal-value
Code a value in the range 0 through 60 for the number of minutes CICS permits users to remain signed on during the takeover period. The takeover period is the time from when the takeover is initiated to the time at which CICS is ready to process user transactions. If the takeover takes longer than the specified period, all users signed-on at the time the takeover was initiated are signed-off.

A value of 0 specifies that there is no time-out delay, and terminals are signed off as soon as takeover commences, which means that XRFSTME=0 has the same effect as coding XRFSOFF=FORCE.

For non-XRF-capable terminals, take into account any AUTCONN delay period when setting the value for XRFSTME. (See the description of the AUTCONN parameter on page 232.) You may need to increase the XRFSTME value to allow for the delay to the start of the CXRE transaction imposed by the AUTCONN parameter; otherwise, terminals may be signed-off too early. For example:



You can avoid this situation by extending the XRFSTME period to exceed the AUTCONN period. For example:



XTRAN={YES|name|NO}

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. 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 and specified the RESSEC(YES)
| option on the resource definitions.

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 T*name* and the corresponding grouping class name is G*name*.

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.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter.

Restrictions

You can code the XTRAN parameter in the SIT, PARM, or SYSIN only.

XTST={YES|name|NO}
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. 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.

Restrictions

You can code the XTST parameter in the SIT, PARM, or SYSIN only.

XUSER={YES|NO}
Specifies whether or not CICS is to perform surrogate user checks.

YES
specifies that 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). For

information about the various circumstances in which CICS performs surrogate user checks, see the *CICS/ESA CICS-RACF Security Guide*.

NO
Specifies that CICS is not to perform any surrogate user checking.

Restrictions

You can code the XUSER parameter in the SIT, PARM, or SYSIN only.

Assembler errors from undefined keywords

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  --  OPENC/aaaaaaa
```

where: `aaaaaaa` is the unidentified keyword.

However, be aware that because of work space limitations there is a limit to the number of undefined keyword errors that the assembler can generate. This means that if your SIT contains more undefined keywords than the assembler can generate messages for, some are not flagged until a second (or even later) assembly, and as you correct flagged errors, other errors (previously unflagged) may appear during reassembly.

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. 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 following two programs, however, you can select from different versions, by specifying the version you require at system initialization:

1. The dynamic backout program (DBP).
2. The basic mapping support (BMS) program suite.

There are two ways of selecting the versions you need:

1. Using a suffix. Use this method for DBP.
2. Explicitly selecting the level of function needed. Use this method for BMS.

You can also specify that a program is ***not*** needed (see "Excluding unwanted programs" on page 319 for details).

You can use these methods ***only*** for the programs referred to in this section and in "Excluding unwanted programs" on page 319, by coding system initialization parameters.

Using a suffix to select the dynamic backout program

Suffixes are used to distinguish different versions of the dynamic backout program (DBP), a CICS management program. Two versions of DBP are supplied with CICS, one in source form only, the other pregenerated. These two suffixed versions are:

Suffix	Description
1$	CICS local DL/I is not supported. This version of the program is supplied with the pregenerated system. Although this version does not support the CICS local DL/I interface, you can use it with remote DL/I and DBCTL.
2$	CICS local DL/I is supported. This version of the program is generated when you generate CICS local DL/I support. For information about generating support for CICS local DL/I, see the *CICS/ESA Installation Guide*.

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

The three ways of excluding programs that are not required are by specifying:

1. Specifying programname=NO
2. Specifying tablename=NO
3. Specifying function=NO

Specifying programname=NO

If you code programname=NO in your system initialization table (for example, DIP=NO), or as a SIT override parameter, 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 program's control table

Not all of the CICS programs have a programname parameter in the SIT. The alternative method of excluding them is to code NO against the table name for these programs. 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 tables that can be used in this way, and their associated management programs, are shown in Table 36 on page 320.

Table 36. SIT control tables with a NO option	
Control tables	***Associated management modules***
Journal control table (JCT)	Journal control program (JCP)
System recovery table (SRT)	System recovery program (SRP)

The function for which you get a dummy version of the management program (which is supplied on the distribution tape with a suffix of **DY**), is journal control. The dummy program is DFHJCPDY. (You also get a dummy volume management program, DFHVCPDY, for use with JCT=NO, because you don't need DFHVCP if you specify no journaling.)

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 of your terminal resource definitions are in the CSD.

When you specify TCT=NO, CICS loads a dummy TCT named DFHTCTDY. A pregenerated dummy table of this name is supplied in CICS410.SDFHLOAD, and the source statements of DFHTCTDY are supplied in CICS410.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 CICS local DL/I support, intersystem communication (ISC), the 3270 print-request facility, the system spooling interface, TCAM support, or the extended recovery facility (XRF), in this way.

Chapter 22. Processing system initialization parameters

This chapter describes the CICS system initialization process as follows:

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.

Methods of supplying system initialization parameters to CICS

The CICS parameter manager domain loads a system initialization table (SIT) at the start of the initialization process. 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 unsuffixed 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, parameter manager domain 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 page 323)
3. The console (but only if CONSOLE is coded in either the PARM parameter or in the SYSIN data set; see page 323).

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 DCT=1$ in the PARM parameter, DCT=2$ in the SYSIN data set, and finally enter DCT=3$ through the console, CICS loads DFHDCT3$.

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, cold or warm. (For this purpose, the parameter manager domain treats an emergency restart as a warm start.)

Startup is cold
If startup is 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 via 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 274.

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.

System initialization control keywords

There are three special control keywords that you can use at startup to control how CICS is to read system initialization parameters. These are:

1. SYSIN (or SI for short)
2. CONSOLE (or CN for short)
3. .END

The purpose of these special keywords, and where you can code them, are described as follows:

SYSIN (SI)

Code this keyword to tell 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.

PN83058

The following change was made by APAR PN83058.

+ 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 325.)

Examples:

PN83058

The following change was made by APAR PN83058.

```
+ //stepname   EXEC  PGM=DFHSIP,PARM='SIT=6$,SYSIN,.END'
  //stepname   EXEC  PGM=DFHSIP,PARM='SIT=6$,DLI=YES,SYSIN,.END'
```

CONSOLE (CN)

You can use this keyword to tell 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.

PN83058

The following change was made by APAR PN83058.

+ 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:

PN83058

The following change was made by APAR PN83058.

```
+ //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'
```

+ **Note:**

+ 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, depending on its context:

Context	Explanation
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:

   PN83058

   The following change was made by APAR PN83058.

```
+  //stepname   EXEC  PGM=DFHSIP,PARM='SIT=6$,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$,START=COLD,
* XRF=NO,            ( XRF this run – SIT defines XRF=YES
DLI=YES,DLIRLM=RLMA, ( Start of CICS local DL/I parameters
DLTHRED=5            ( Number of DL/I interface threads
DDIR=1$,             ( SUFFIX of DMB directory
PDIR=1$,             ( SUFFIX of PSB directory
DLLPA=NO,            ( Don't use DL/I modules from the LPA
DLMON=NO,            ( IMS Data Base Monitor not required
  ⋮
.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 override or other initialization parameters, and tries to load an unsuffixed module named DFHSIT. As a general rule, this is unlikely to be your intention, so the PARM parameter should 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 can 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 PARM parameters

The rules for coding the PARM parameter on an EXEC job control statement are described fully in the *MVS/ESA JCL Reference* manual. Briefly, 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.

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 simple 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'
  ```

  In CICS/ESA 4.1, 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 following message (where nn is the reply ID):

```
nn DFHPA1104 applid - SPECIFY ALTERNATIVE SIT PARAMETERS, IF ANY,
                                                    AND THEN TYPE '.END'.
```

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:

```
DFHPA1912 'applid' SIT OVERRIDE 'keyword' IS NOT RECOGNIZED.
                   SPECIFY CORRECT SIT OVERRIDE.
DFHPA1915 'applid' INVALID DATA HAS BEEN DETECTED FOR SIT OVERRIDE 'keyword'.
                   RESPECIFY THE OVERRIDE.
```

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.

Classes of 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 at the beginning of this chapter.

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 are installed at CICS shutdown. These are:

- Programs
- Transactions and transaction profiles
- Terminals and typeterms
- Connections and sessions
- BMS maps sets and partition sets
- Files.

The resource definitions that CICS saves at its shutdown may have been installed during a cold start (from a list of groups specified by a group list system initialization parameter), or during CICS execution (by RDO commands).

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. The general rule is that you cannot alter installed resource definitions during a restart except by coding START=COLD.

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. For example, CICS monitoring uses the cataloged status at a restart, but modified by any system initialization parameters you provide. In other cases the domain information saved in the catalog is always used in a restart. For example, 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 this 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 need 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:

- Initialize the local catalog with an initial set of domain records.

Apar PQ07674

Documentation for Apar PQ07674 added 26/08/98

- 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/ESA Operations and Utilities Guide*.
- Also delete and reinitialize the global catalog.

For more information about initializing the local catalog, see "The local catalog" on page 166. 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, by inspecting the control record in the global catalog, which of the following three types of start to perform.

1. **WARM**

   If the control record in the global catalog indicates that the previous run of CICS terminated normally with a successful warm keypoint, CICS performs a warm restart. The local catalog must also contain the information saved by the CICS domains during the previous execution for the warm restart to be successful. A warm start restores CICS to the state it was in at the previous shutdown.

   If you are using disk journaling, the status of the disk journals is also saved in the global catalog. This information is used by CICS at startup to determine which journal data set is to be opened. You can use the JSTATUS=RESET startup parameter to cause the status in the global catalog to be ignored. During CICS startup, the status of all journal data sets is set to "ready for use". For more information about using JSTATUS=RESET, see "Resetting the journal status" on page 125 and the description of the JSTATUS parameter on page 265.

   If you are using standard-labeled tape journaling, and you supply the SERIES=PURGE startup parameter, CICS deletes any journal series information that is stored in the global catalog. In the absence of the SERIES startup parameter, CICS reconstructs all the journal series from the information saved in the global catalog.

   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 are the system initialization parameters 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.

2. **COLD**
   If there is no control record in the global catalog, CICS assumes that the catalog has been newly initialized, and forces a cold start. If you have recreated the global catalog for some reason, you must also reinitialize the local catalog.

3. **EMERGENCY**
   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.

   If the system log is on standard-labeled tape, the control record also contains the volume identifier of the system log volume being used at the time of termination. The emergency restart procedure uses the system log at the time of the failure to return recoverable resources to their committed states.

START=AUTO should be the normal mode of operation, with the choice of start being made by CICS automatically.

START=COLD

If you code COLD as the START operand, 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.

Be aware, however, that a start initiated by START=COLD is not entirely without reference to the previous run of a CICS system using the same global catalog. For example, if you are running CICS with local DL/I support, the global catalog may contain extended error queue elements (EEQEs) relating to DL/I write failures. This information is preserved across a cold start. For background information about EEQEs, and about other information that is referenced even in a cold start, see the *CICS/ESA Recovery and Restart Guide*. You can perform a fully cold start of CICS, without reference to any previous execution, only by reinitializing both CICS catalogs. Generally, do this only when you are starting your CICS system for the first time.

There may be times when it is necessary to restart CICS with START=COLD, irrespective of the type of system termination that has been recorded in the global catalog.

START=STANDBY

The STANDBY option is for use only with XRF=YES. START=STANDBY initializes an alternate CICS region. If you code START=STANDBY with XRF=NO, initialization fails with message DFHXA6530, and CICS terminates abnormally with a dump.

CICS initializes as an alternate CICS region by beginning to perform an emergency restart, which is then suspended until it needs to perform a takeover. During the period when initialization is suspended, the alternate CICS region is in standby mode and monitors the active CICS region. When it takes over, the alternate CICS region completes the emergency restart and becomes the active CICS region.

+ If you have specified a COLD start for other CICS resources, (DCT=(xx,COLD), for example), they are cold started when the alternate CICS region (with START=STANDBY specified) takes over. This may cause CICS to lose data on an XRF takeover; for example, coding ICP=COLD results in outstanding STARTs being lost. You are recommended to code START=(STANDBY,ALL) to ensure a full emergency restart during takeover, unless you wish to specifically cold start individual resources.

For information about operating a CICS region with XRF, see the *CICS/ESA Operations and Utilities Guide*.

START=LOGTERM

The LOGTERM operand on the START parameter is a special restart option. It is for use only when the system log is defined on ***disk*** data sets, to cater for an abnormal termination of a CICS system that does not close the system log. The LOGTERM option causes a CICS restart to put only an end of file label on the log, and then terminate ***before*** doing any backout processing. You can use START=LOGTERM, for example, if you need to close the system log to perform offline file recovery, particularly in those cases when you know (or suspect) that emergency restart won't work.

If you are using DBRC with CICS local DL/I support, DBRC is notified of the log closure, and you can then perform offline recovery.

LOGTERM is available only as a system initialization parameter at run time, and cannot be coded in the system initialization table. It is also intended for use only when you are running CICS with XRF=NO.

Table 37 (Page 1 of 2). Effect of the START= parameter in conjunction with the catalogs

START= parameter	State of the CICS catalogs	Result at restart
COLD	Local and global catalogs are both newly initialized.	CICS performs a fully cold start. All domains are initialized using system default values, modified by any system initialization parameters. All resources are installed as specified by system initialization parameters.
COLD	The global catalog contains a successful warm keypoint from previous run, and the local catalog contains information saved by the CICS domains.	CICS performs a cold restart, installing the resource definitions specified by system initialization parameters. The domains initialize according to system initialization parameters, or using system default values where there are no parameters (for example the statistics domain). If you are running CICS with local DL/I support, CICS recovers any EEQEs from the global catalog. CICS also checks the global catalog for any data set name block entries for VSAM data sets for which backout failures were recorded in the system log. For background information about how CICS uses the DL/I EEQEs and the CICS file control backout information, see the *CICS/ESA Recovery and Restart Guide*.

Table 37 (Page 2 of 2). Effect of the START= parameter in conjunction with the catalogs

START= parameter	State of the CICS catalogs	Result at restart
AUTO	Local and global are both newly initialized.	CICS enforces a fully cold start. All domains are initialized with system default values, modified by any system initialization parameters. All resources are installed as specified by system initialization parameters.
AUTO	The global catalog contains a successful warm keypoint from the previous run. The local catalog is newly initialized.	CICS begins to perform a warm restart, but fails during the initialization process because of absence of expected records in the local catalog. (***Do not reinitialize only one of the catalogs***.)
AUTO	The global catalog contains a successful warm keypoint from the previous run, and the local catalog contains information saved by the domains.	CICS performs a warm restart, restoring all of your CICS system except the trace domain to the same status it was in at CICS shutdown. (CICS trace domain does not save the status of the various trace options at CICS shutdown, and always uses the system initialization parameters.) Only those resources that have a COLD option on their system initialization parameter can be cold started (for example, the destination control table or auxiliary temporary storage).

Table 38 shows the effect of the various START options, combined with system initialization parameters where applicable, on the CICS trace, monitoring, statistics, and dump domains:

Table 38 (Page 1 of 2). Effect of the START= parameter on the CICS domains at initialization

Domain	State of the CICS catalogs	Result at startup if START=AUTO	Result at startup if START=COLD
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.

Table 38 (Page 2 of 2). Effect of the START= parameter on the CICS domains at initialization

Domain	State of the CICS catalogs	Result at startup if START=AUTO	Result at startup if START=COLD
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, but the dump status is 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=xxxxxxxx, 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/ESA Messages and Codes* manual.

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=xxxxxxxx, ACB CODE=yy.
DFHSI1590 'applid'  XRF alternate cannot proceed without VTAM.
```

CICS abends with a dump (abend code 1590).

End of CICS startup

Whichever type of startup is performed, when the message:

```
DFHSI1517 - 'applid':  Control is being given to CICS.
```

is displayed on the operating system console, CICS is ready to process terminal requests. (*applid* is the value of the specific APPLID system initialization parameter.)

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 the *CICS/ESA CICS-Supplied Transactions* manual.

Chapter 23. 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. 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 339. “A sample CICS startup procedure” on page 365 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 (except for START=STANDBY, when most data sets are not opened until after takeover)
- 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 START=STANDBY and XRF=YES, and a takeover occurs.

For background information about backout, and recovery and restart, see the *CICS/ESA 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. For more information about storage protection, see "Storage protection" on page 358. For programming information about writing PLT programs, see the *CICS/ESA Customization Guide*.

If you are running CICS with DB2, you can specify the resource control table suffix and DB2 subsystem ID to be used at startup by the INITPARM system initialization parameter, as follows:

```
INITPARM=(DSN2STRT='xx,yyyy')
```

where xx is the 2-character resource control table suffix and yyyy is the 4-character DB2 subsytem ID. Both values must conform to MVS JCL rules about special characters. If you specify a DB2 subsystem ID, it is used at PLT startup (with the resource control table suffix specified).

For information about setting up the RCT, see the *DB2 Administration Guide*.

Sample startup job stream

You can use the sample job control statements shown in Figure 50 on the following pages to initialize a CICS region. The sample job stream shown in Figure 50 specifies START=AUTO and XRF=YES to start an active CICS region. The notes that follow Figure 50 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$, but reassembled to specify DLDBRC=YES. You must specify DLDBRC=YES if you want to use DBRC with CICS local DL/I support, and you can code this option only in the SIT. For details of all the system initialization options specified in the sample table, see the source in DFHSIT6$, which is a member of the CICS410.SDFHSAMP library.

IMS library names

The IMS libraries referred to in the job streams are identified by IMS.libnam (for example IMS.PGMLIB). If you use your own naming convention for IMS libraries, please rename the IMS libraries accordingly.

For more information about the DD statements in this job stream that are needed by CICS and IMS, see the appropriate chapter in Part 2, "Defining data sets" on page 95.

JCL similar to that in Figure 50 on page 339 is supplied as a sample startup procedure, DFHSTART, in the CICS410.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/ESA Installation Guide*.

A sample CICS startup job

```
//*
//* 1    The JOB statement
//CICSRUN  JOB accounting info,name,CLASS=A,
//            MSGCLASS=A,MSGLEVEL=(1,1)
//*
//* 2    The JOBPARM statement
/*JOBPARM SYSAFF=sysid
//*
//*
//*************************************************************
//*******************  EXECUTE CICS  ************************
//*************************************************************
//*
//* 3    The EXEC CICS=DFHSIP statement
//CICS    EXEC PGM=DFHSIP,REGION=240M,
//* 4    SIT parameters specified on PARM parameter
//             PARM=('SIT=6$',
//             'DSALIM=6M,EDSALIM=120M',
//             'DLI=(YES,COLD),DBP=2$',
//             'RENTPGM=PROTECT,STGPROT=YES',
//             'START=AUTO,SI')
//*
//* 5    SIT parameters specified on the SYSIN data set
//SYSIN    DD *
GRPLIST=(DFHLIST,userlist1,userlist2),
LPA=YES,
APPLID=CICSHTH1,
*
CICSSVC=245,         The CICS Type 3 SVC number
DFLTUSER=CICSUSER,   The default userid
MXT=30,              Maximum number of user tasks is 40
INITPARM=(DFHDBCON='01',DSN2STRT=('01,MYDB')),
                     Pass DFSPZP01 suffix to DBCTL connect program
                     Use RCT DSN2CT01 with DB2 subsystem MYDB
ISC=YES,             Include intersystem communication program
IRCSTRT=YES,         Start interregion communication
```

Figure 50 (Part 1 of 4). CICS startup job stream

```
* The following parameters are for CICS local DL/I support
DDIR=1$,             Suffix of DMB directory list
PDIR=1$,             Suffix of PSB directory list
DLLPA=NO,            No DL/I modules to be used from the LPA
DLMON=NO,            No database monitoring
DLTHRED=5,           Number of interface threads
DLXCPVR=NO,          No page fixing
DMBPL=10,            Maximum size of DMB pool in 1024-byte blocks
ENQPL=2,             ENQ control space in KB
PISCHD=YES,          Program isolation scheduling required
PSBCHK=NO,           No remote terminal checking
PSBPL=10,            PSB pool size in 1024-byte blocks
XPSB=YES,            PSB checking by RACF is required
DLIRLM=LRHS,         IRLM support required for IMS data sharing
.END
/*
//*
//*  6     The STEPLIB library
//STEPLIB  DD  DSN=CICS410.SDFHAUTH,DISP=SHR
//         DD  DSN=IMS.RESLIB,DISP=SHR
//*
//*  7     CICS local DL/I-required libraries
//DFSRESLB DD  DSN=IMS.RESLIB,DISP=SHR
//IMSACB   DD  DSN=IMS.ACBLIB,DISP=SHR
//*
//*  8     Database recovery control (DBRC) data sets
//* This is required with CICS local DL/I only
//RECON1   DD  DSN=IMS.RECON1,DISP=SHR
//RECON2   DD  DSN=IMS.RECON2,DISP=SHR
//RECON3   DD  DSN=IMS.RECON3,DISP=SHR
//*
//*  9     The CBRC transaction data sets
//* This is required with CICS local DL/I only
//JCLPDS   DD  DSN=IMS.JOBS,DISP=SHR
//JCLOUT   DD  SYSOUT=(A,INTRDR)
//IMS      DD  DSN=IMS.DBDLIB,DISP=SHR
//*
//*  10    The CICS library (DFHRPL) concatenation
//DFHRPL   DD DSN=CICS410.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 PL/I library data set
//*        DD DSN=SYS1.PLILINK,DISP=SHR
//*        DD DSN=IMS.RESLIB,DISP=SHR
//*        VS COBOL II library data sets
//*        DD DSN=SYS1.COB2CICS,DISP=SHR
//*        DD DSN=SYS1.COB2LIB,DISP=SHR
//*        C/370 run-time library
//         DD  DSN=EDC.V1R2M0.SEDCLINK,DISP=SHR
//*
//*  11    Auxiliary temporary storage data sets
//DFHTEMP  DD DSN=CICS410.CNTL.CICSHTH1.DFHTEMP,DISP=SHR
//*
```

Figure 50 (Part 2 of 4). CICS startup job stream

```
//* 12    Intrapartition data sets
//DFHINTRA DD DSN=CICS410.CNTL.CICSHTH1.DFHINTRA,DISP=SHR
//*
//* 13    The auxiliary trace data sets
//DFHAUXT  DD DSN=CICS410.CICSHTH1.DFHAUXT,DISP=SHR
//DFHBUXT  DD DSN=CICS410.CICSHTH1.DFHBUXT,DISP=SHR
//*
//* 14    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=136)
//PLIMSG   DD SYSOUT=*,DCB=(DSORG=PS,RECFM=V,BLKSIZE=137)
//COUT     DD SYSOUT=*,DCB=(DSORG=PS,RECFM=V,BLKSIZE=137)
//CEEMSG   DD SYSOUT=A
//CEEOUT   DD SYSOUT=A
//*
//* 15    The CICS system log data sets
//DFHJ01A  DD DSN=CICS410.CICSHTH1.DFHJ01A,DISP=SHR
//DFHJ01B  DD DSN=CICS410.CICSHTH1.DFHJ01B,DISP=SHR
//DFHJ01X  DD DSN=CICS410.CICSHTH1.DFHJ01X,DISP=SHR
//*
//* 16    User journal data set
//DFHJ02A  DD DSN=CICS410.CICSHTH1.DFHJ02A,DISP=SHR
//*
//* 17    Automatic journal archiving data sets
//DFHJACD  DD DSN=CICS410.CICSHTH1.DFHJACD,DISP=SHR
//DFHJPDS  DD DSN=CICS410.DFHJPDS,DISP=SHR
//DFHJOUT  DD SYSOUT=(A,INTRDR)
//*
//* 18    The restart data set
//DFHRSD   DD DSN=CICS410.CICSHTH1.DFHRSD,DISP=SHR
//*
//* 19    The CICS local catalog data set
//DFHLCD   DD DSN=CICS410.CICSHTH1.DFHLCD,DISP=SHR
//* 20    The CICS global catalog data set
//DFHGCD   DD DSN=CICS410.CICSHTH1.DFHGCD,DISP=SHR,
//            AMP=('BUFND=33,BUFNI=32,BUFSP=303104')
//*
//* 21    The CAVM data sets - XRF
//DFHXRMSG DD DSN=CICS410.CNTL.CICSHTH1.DFHXRMSG,DISP=SHR
//DFHXRCTL DD DSN=CICS410.CNTL.CICSHTH1.DFHXRCTL,DISP=SHR
//*
//* 22    The transient data destination data set (CXRF)
//DFHCXRF  DD SYSOUT=A
//*
//* 23    The CICS transaction dump data sets
//DFHDMPA  DD DSN=CICS410.CICSHTH1.DFHDMPA,DISP=SHR
//DFHDMPB  DD DSN=CICS410.CICSHTH1.DFHDMPB,DISP=SHR
//*
```

Figure 50 (Part 3 of 4). CICS startup job stream

```
//* 24   MVS system dump data sets
//SYSABEND DD SYSOUT=*
//SYSMDUMP DD  DSN=SYS1.SYSMDP00,VOL=SER=volid,SPACE=(CYL,(1,1)),
//             DISP=OLD,UNIT=3380
//* SYSUDUMP DD SYSOUT=*
//*
//* 25   The CICS system definition data set
//DFHCSD   DD DSN=CICS410.DFHCSD,DISP=SHR
//*
//* 26   The CMAC data file
//DFHCMACD DD DSN=CICS410.DFHCMACD,DISP=SHR
//*
//* 27   FILEA & other permanently allocated data sets
//* (The FILEA DD statement below overrides the CSD definition in
//*  group DFHMROFD)
//FILEA    DD DISP=SHR,
// DSN=CICS410.CICSHTH1.FILEA
//*
//APPLICA  DD  DSN=CICS410.CICSHTH1.APPLICA,DISP=SHR
//APPLICB  DD  DSN=CICS410.CICSHTH1.APPLICB,DISP=SHR
//*
//* 28   Your user-defined DL/I database
//DLIDB    DD  DSN=CICS410.DLIDB,DISP=SHR
//*
//* 29   DFSVSAMP
//DFSVSAMP DD  *
IOBF=(2048,4)
4096,4
/*
//*
//* 30  and  31  Sequential (BSAM) devices
//PRINTER  DD  SYSOUT=A,DCB=BLKSIZE=125,OUTLIM=0
//CARDIN   DD  *
  ⋮
                                                    \
user transactions input from sequential terminal    \
  ⋮
                                                    \
CESF GOODNIGHT\
/*
```

Figure 50 (Part 4 of 4). CICS startup job stream

Notes:

1 The JOB statement

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 JOBPARM statement

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 The EXEC PGM=DFHSIP statement

DFHSIP is the program that starts CICS initialization.

+

Warning–CICS supports only one EXEC PGM=DFHSIP job step

+ CICS does not support more than one EXEC PGM=DFHSIP job step in the
+ same MVS job.

| The EXEC statement contains the REGION parameter to define the size of CICS
| MVS region. In this example, the value is set to 240, requesting MVS to allocate to
| the job all 16MB of private storage below the 16MB line, and an extended region
| size of 240MB.

| 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 **4** on page 343.)

| For more details about the REGION parameter and CICS storage, see “Storage
| requirements for a CICS region” on page 356.

| If you are running CICS with RACF support, see the *CICS/ESA CICS-RACF
| Security Guide* for information about RACF-related parameters.

4 Options of the PARM parameter

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 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 run-time system initialization parameters are in the PARM parameter, you can end the PARM field simply 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:

DLI CICS local DL/I support is included, and the COLD operand specifies that, even if CICS performs a warm start, the DL/I resources are to be cold started.

| **DBP** Because CICS local DL/I support is being added to this run of CICS, the
| version of the dynamic backout program (DBP) that includes DL/I
| support (DFHDBP2$) is also specified.

| **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 page 285.

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* that you have the required hardware and software. See page 296 for details about the STGPROT parameter itself.

START START=AUTO is normally the type of start you would 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/ESA Operations and Utilities Guide*.

SI The PARM statement is terminated with SI (short for SYSIN), to tell CICS to continue reading overrides from the SYSIN data set.

5 SYSIN data stream

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=(DSN2STRT= '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, as follows:

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) should contain all the resource definitions generated by your installation for your applications that are required for this CICS run. In addition, your group lists should contain any definitions required for IBM program products that you are using, such as VS COBOL II 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, this applid must be defined to VTAM.

If you want this CICS region to use XRF, you must define both a generic and specific applids; 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 necessary, it can be used 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 SVCPARM statement. For more information about the CICSSVC parameter, see page 235.

For guidance information about installing the CICS SVC in the LPA, and defining it to MVS, see the *CICS/ESA Installation Guide*.

DFLTUSER
CICSUSER is the default userid specified to RACF. During startup, CICS tries to sign on the default userid. If it cannot be signed on (for example, if not defined), CICS issues a message and terminates CICS initialization. After the valid default userid 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 userid 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 what tasks are included in the MXT parameter, see page 271.

INITPARM
Passes parameters to programs. In this example the DBCTL suffix (01) of DFSPZPxx is passed to DFHDBCON, and the resource table DSN2CT01 is used with the DB2 subsystem MYDB.

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.

The following parameters are for CICS local DL/I support:

DDIR Suffix of the DMB directory list.
PDIR Suffix of the PSB directory list.
DLLPA No DL/I modules are to be used from the LPA in this run.
DLMON No database monitoring in this run.
DLTHRED The number of DL/I interface threads.
DLXCPVR No page fixing required.
DMBPL A data management block (DMB) pool size of 10 blocks for this run.
ENQPL The size in kilobytes of the ENQ control block space.
PISCHD Program isolation scheduling is required.
PSBCHK Remote terminal checking is not required.
PSBPL A program specification block (PSB) pool size of 10 blocks for this run.
XPSB PSB checking by RACF is required.
DLIRLM IRLM support is required for IMS data sharing in this run.

6 STEPLIB library

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 CICS410.SDFHAUTH.

Placing user-written modules into an APF-authorized library may violate your security and integrity rules. The CICS410.SDFHAUTH library should not be included in the MVS link list, so that you can protect the library and thereby ensure that only approved users are able to execute the DFHSIP module. When you define your STEPLIB DD statement, remember that all other libraries concatenated with the CICS410.SDFHAUTH library must also be APF-authorized (for example, IMS.RESLIB). This is 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/ESA CICS-RACF Security Guide*.

If you are running CICS with DL/I support, you must include the IMS APF-authorized library, RESLIB, and the APF-authorized library into which you generated the CICS-DLI initialization module, DFHDLQ, in your STEPLIB concatenation. (For about generating DFHDLQ, see the *CICS/ESA Installation Guide* .)

To use application programs written in VS COBOL II, you must have installed the following four VS COBOL II library routines in an APF-authorized library in the STEPLIB concatenation (for example, in the CICS410.SDFHAUTH library), or in an APF-authorized library in the MVS LNKLSTnn concatenation:

- IGZ9CIC (and its alias IGZECIC)
- IGZ9WTO (and its alias IGZEWTO)
- IGZ9OPD (and its alias IGZEOPD)
- IGZCMTxx, where xx represents the first two letters of the language specified on the MVS LANGUAGE option for your site (for example, IGZCMTEN).

For more information, see “Installing CICS support for VS COBOL II, COBOL/370 and IBM COBOL for MVS and VM” on page 32.

If you are running CICS with C/370 support, copy the CICS-C/370 interface module, EDCCICS, from EDC.V1R2M0.SEDCLINK to one of the authorized libraries in the STEPLIB concatenation.

The pregenerated DFHSIP module, which has been link-edited with the authorized attribute (SETCODE AC(1)), is supplied in CICS410.SDFHAUTH.

7 CICS local DL/I data sets

Include these libraries only if you are running CICS with local DL/I support; they are not needed for a CICS system using DBCTL.

8 DBRC recovery control data sets

If you are using DBRC with CICS local DL/I (DLDBRC=YES), you can include these statements. DBRC uses RECON1 and RECON2 as its recovery control data sets; it uses RECON3 only if an error is detected in either RECON1 or RECON2.

If you omit these statements, the data sets are allocated dynamically when needed. However, it is advisable to be consistent across subsystems, and either always code these statements, or always use dynamic allocation. If you do use dynamic allocation, make sure that IMS.RESLIB contains the necessary dynamic allocation members described in the *IMS Utilities Reference* manual.

9 CBRC transactions for DBRC (with CICS local DL/I support)

You must include these JCLPDS and JCLOUT DD statements if you intend using the CBRC transaction to issue the DBRC command, GENJCL (which creates and can submit batch jobs). For more information about GENJCL, see the relevant IMS manual. The IMS statement must be included if you intend using CBRC to issue any of the following DBRC commands:

CHANGE.DB
CHANGE.DBS
INIT.DB
INIT.DBS
NOTIFY.REORG

10 CICS module load library (DFHRPL) concatenation

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 our example as “your.prog.library”, and your CICS control tables, shown in our example as “your.table.library”.

In this sample job stream, the CICS-supplied library is CICS410.SDFHLOAD, which must be included in the CICS DFHRPL library concatenation. The CICS410.SDFHLOAD library contains only programs that run in problem state and should not be authorized.

PL/I requirements

The PL/I transient library. For PL/I support, you must have the PL/I transient library installed, and included in the DFHRPL library concatenation. It contains the module IBMESAP, and the PL/I transient modules. If you do not concatenate the PL/I library in DFHRPL, you get the message:

```
DFHSI1596 - Nucleus module IBMESAP cannot be located
```

and you are not able to run your PL/I applications.

PL/I shared library support. For PL/I shared library support, you must ensure that you have the CICS version of the PL/I shared library module, PLISHRE, installed in CICS410.SDFHLOAD. The linkage editor job to generate the CICS PLISHRE module is described under “Generating PL/I shared library support for CICS” on page 36.

You must also ensure that two of the PL/I shared library modules, IBMBPSLA and IBMBPSMA, are installed in a library in the DFHRPL concatenation or in the LPA, otherwise CICS initialization fails.

For information about generating PL/I shared library support, see the *OS PL/I Version 2 Installation and Customization under MVS Guide*.

CICS run-time support for PL/I CICS provides run-time support for PL/I version 2.3. Programs compiled against earlier releases of PL/I run with the PL/I version 2.3 run-time support.

The CSD definitions required for this support are supplied by PL/I. For information about the definitions required and how to install them, see the *OS PL/I Version 2 Installation and Customization under MVS Guide*.

The CICS-supplied group of CSD definitions, DFHPLI, is not needed in CICS/ESA 4.1, and has been deleted. It has been added with other obsolete CSD definitions in the compatibility group DFHCOMP2, which enables you to share your CSD between CICS/ESA 4.1 and your earlier CICS releases. For information about the CSD compatibility groups, see “CICS supplied compatibility groups” on page 149, and the *CICS/ESA Resource Definition Guide*.

VS COBOL II requirements

If you want to use VS COBOL II, you must have the CICS-VS COBOL II interface module, IGZECIC, in an authorized library in the CICS STEPLIB concatenation (for example, in CICS410.SDFHAUTH). IGZECIC is normally in SYS1.COB2CICS following the installation of VS COBOL II. IGZECIC does ***not*** have to be defined to CICS as a program resource in the CSD.

You must also include the libraries containing the VS COBOL II library routines in the DFHRPL concatenation. VS COBOL II requires two packages of subroutines, known as COBPACKs. These library subroutines are in two categories (1) general and (2) environment-specific, containing system-specific logic. The COBPACKs you require for CICS are:

IGZCPCC This module contains the CICS environment-specific modules (ESMs), and is supplied in SYS1.COB2CICS.

IGZCPAC This module contains the general VS COBOL II subroutines, and is supplied in SYS1.COB2LIB.

Ensure that SYS1.COB2CICS is concatenated in front of SYS1.COB2LIB. Unlike IGZECIC, both of these COBPACKs must be defined as programs in the CSD, and the program resource definitions must be included in the group list used for CICS initialization. The following CEDA commands are an example of how you can define these VS COBOL II subroutines to CICS:

```
CEDA DEFINE PROGRAM(IGZCPCC) GROUP(cob2grp) LANGUAGE(ASSEMBLER)
CEDA DEFINE PROGRAM(IGZCPAC) GROUP(cob2grp) LANGUAGE(ASSEMBLER)
CEDA ADD GROUP(cob2grp) LIST(listname)
```

For information about running VS COBOL II applications with CICS, see the *VS COBOL II Installation and Customization* manual, SC26-4048.

C/370 requirements

If you want to use CICS-C/370 support, you must have the CICS-C/370 interface module, EDCCICS, in an authorized library in the CICS STEPLIB concatenation (for example, in CICS410.SDFHAUTH). EDCCICS is normally in EDC.V1R2M0.SEDCLINK following the installation of C/370. EDCCICS does ***not*** have to be defined to CICS as a program resource in the CSD.

You must define another C/370 run-time module, EDCXV, as an assembler program in the CSD, and the program resource definitions must be included in the group list used for CICS initialization. For example, you could use the following CEDA commands:

```
CEDA DEFINE PROGRAM(EDCXV) GROUP(C370) LANGUAGE(ASSEMBLER)
CEDA ADD GROUP(C370) LIST(listname)
```

You may also define the C/370-provided locales in the same way as EDCXV. These locales, EDC$GERM, EDC$USA, EDC$FRAN, EDC$ITAL, and EDC$SPAI, are provided in the EDC.V1R2M0.SEDCLINK data set.

For information about running C/370 applications with CICS, see the *IBM C/370 Programming Guide*.

11 Auxiliary temporary storage (DFHTEMP) data sets

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 9, “Defining the temporary storage data set” on page 111.

12 Intrapartition transient data (DFHINTRA) data sets

The transient data intrapartition data set is used for queuing messages and data within the CICS region.

For information about how to define these data sets, and about space calculations, see “Defining the intrapartition data set” on page 116.

13 Auxiliary trace (DFHAUXT and DFHBUXT) data sets

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 15, “Defining and using auxiliary trace data sets” on page 171.

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 could 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 41 on page 173, you can define them to CICS in the startup job stream using the following DD statements:

```
//DFHAUXT  DD  DSN=CICS410.applid.DFHAUXT,DCB=BUFNO=n,DISP=SHR
//DFHBUXT  DD  DSN=CICS410.applid.DFHBUXT,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.

14 Extrapartition transient data destinations

LOGUSR, MSGUSR, and PLIMSG are examples of data sets for extrapartition transient data destinations. You can choose the DD names of these data sets, but they must agree with the DSCNAMEs on the corresponding DFHDCT TYPE=SDSCI definitions in the destination control table (DCT). In this example, based on the sample table DCT=2$:

- LOGUSR defines a user data set used by the CICS sample programs (DESTID=LOGA)
- MSGUSR defines the data set used by a number of CICS services (DESTID=CSSL)

- PLIMSG is used only when you are running PL/I application programs. Here, PLIMSG defines the data set to which both statistics and messages (DESTID=CPLI), and PL/I dumps (DESTID=CPLD) are directed.
- COUT is used as an output queue only when you are running C/370 application programs.
- CEEMSG is used as an error queue only when you are running application programs under Language Environment.
- CEEOUT is used as an output queue only when you are running application programs under Language Environment.

For details of all the destinations used by CICS, and included in the sample table DFHDCT2$, see the copy member DFH$DCTR.

15 The CICS system log data sets

Journals on disk data sets
You must include a DD statement for the emergency log data set, DFHJ01X, in your CICS startup job stream if all of the following apply:

- You are using CICS local DL/I support
- The system log is defined on disk data sets
- The START parameter is START=AUTO or START=STANDBY.

The sample job stream illustrates the DD statements you need when the CICS system log is defined on disk data sets.

Journals on standard-labeled tapes
If the system log is on standard-labeled tapes, the system log DD statements could be of the following form:

```
//DFHJ01A   DD UNIT=(TAPE,,DEFER),LABEL=(,SUL),DISP=SHR,
//             VOL=SER=serial-number,DSN=CICS410.applid.DFHJ01A
//DFHJ01B   DD UNIT=(TAPE,,DEFER),LABEL=(,SUL),DISP=SHR,
//             VOL=SER=DUMMY,DSN=CICS410.applid.DFHJ01B
```

Journals on unlabeled tapes
When you are not using DBRC, you can define the system log on unlabeled tapes, but this is not recommended; the preferred medium is disk or standard-labeled tape. On unlabeled tapes the DD statements would be of the form:

```
//DFHJ01A   DD  UNIT=(TAPE,,DEFER),LABEL=(,NL),DISP=SHR,
//          VOL=SER=CICS410.applid.DFHJ01A
```

The volume identifiers in these statements are used by the operating system to prompt you to mount the required tape volume, identified by an external (visual) label. If the system log is defined on tape (labeled or unlabeled), and emergency restart is being performed, CICS prompts you to mount the same log tape that was in use when CICS terminated abnormally. The message specifies that the tape should be mounted on the magnetic tape unit (address) referred to in the DFHJ01A DD statement.

Warning: If your log is on ***unlabeled*** tapes, CICS acts as if any tape mounted is the correct one. You must be careful, therefore, when mounting unlabeled tapes on the DFHJ01A drive.

16 User journal data set

This is an example of how to specify a user journal data set.

17 The automatic journal archiving data sets

If you specify JOUROPT=(AUTOARCH,.,.) in the JCT, you must provide DD statements for DFHJACD, DFHJPDS, and DFHJOUT. These data sets are used as follows:

DFHJACD provides control information for automatic journal archiving.

DFHJPDS provides the jobs that are submitted by the journal archiving submission program (DFHJASP) to archive journals that are ready to be archived.

DFHJOUT receives the automatic journal archive job submitted by DFHJASP. It is normally coded as a SYSOUT data set, directed to the internal reader. You should increase the number of internal readers defined by your JES parameters to allow for the number of readers assigned by DFHJOUT DD statements in CICS job streams.

18 The restart data set

The restart data set is used during emergency restarts only, to hold temporarily the backout information read from the CICS log.

For details of how to create and initialize this data set, see Chapter 13, “Defining and initializing the restart data set” on page 159.

19 The CICS local catalog data set

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 14, “Defining and using catalog data sets” on page 161.

20 The CICS global catalog data set

The CICS global catalog has a variety of uses, including:

- During the running of CICS, holding resource definitions that are installed, DL/I status information, and disk journal status information.
- During a controlled shutdown, recording warm keypoint information for a later warm start. 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 14, “Defining and using catalog data sets” on page 161.

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 39 and the associated notes given under

4 on page 163. The values given are the minimum values suitable for these parameters and should not be reduced.

21 The CAVM data sets

These 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 17, “Defining the CICS availability manager data sets” on page 181.

22 The DFHCXRF transient data set (CXRF)

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 119.

23 CICS transaction dump data sets

CICS records transaction dumps on a sequential data set, or 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 16, “Defining dump data sets” on page 175.

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 could be identified by using the specific applid of the alternate CICS region (DBDCCIC2) as a second-level qualifier.

24 MVS system dump data sets

Use a SYSABEND, SYSMDUMP, or SYSUDUMP DD statement to direct MVS to produce a dump.

In the sample job stream (Figure 50 on page 339), 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 terminates abnormally
- CICS starts to terminate abnormally, but system recovery procedures enable CICS to terminate normally.

The dump DD statements for requesting dumps are:

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 input/output 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's 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:

- DSNAME=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 off-load 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 the *MVS/ESA 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 *MVS/ESA 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 *MVS/ESA JCL Reference*. For information about how to interpret dumps, see the *MVS/ESA Diagnosis: Using Dumps and Traces*.

25 The CICS system definition file

The system definition file (CSD) is required by CICS to hold some resource definitions.

You may 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 may 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 SIT (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 12, “Defining the CICS system definition data set” on page 139.

If you are running CICS with XRF, particularly in a multi-MVS environment, there are special considerations concerning CSD sharing; these considerations are discussed under “Sharing and availability of the CSD” on page 143.

26 CMAC data file (DFHCMACD)

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 20, “Defining the CMAC messages data set” on page 207 describes DFHCMACD in greater detail.

27 Sample program file (FILEA) and other permanently allocated data sets

You may 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 our startup job stream as an example of direct allocation by job control statement.

On the other hand, you may 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 18, “Defining user files” on page 189. For information about the parameters that you can code on file resource definitions, see the *CICS/ESA Resource Definition Guide*.

28 User-defined local DL/I database

DLIDB represents a user-defined DL/I database for access by the CICS local DL/I interface. However, DD statements are not necessary if you are using IMS dynamic allocation and deallocation facilities. If you are running CICS with XRF, dynamic allocation is the recommended method. For information about dynamic allocation and deallocation of local DL/I databases, see the *CICS/ESA Recovery and Restart Guide*.

29 DFSVSAMP

You need the DFSVSAMP DD statement only when you are running with CICS local DL/I support. It can be used to describe the size and number of buffers and various other DL/I options. (For information about coding DFSVSAMP, see the *IMS System Definition Reference* manual.)

30 Sequential (BSAM) devices as simulated terminals

You can use a pair of input and output sequential data sets to simulate a terminal to CICS. For example, you might want to do this to test an application program before the intended (real) terminal becomes available. You must provide DD statements for BSAM sequential devices. For more information about these devices, see "Sequential (BSAM) devices" on page 83.

The card reader/line printer (CRLP) simulated terminals shown in our sample job stream are defined in the sample TCT (not used in this startup job). See the copy member DFH$TCTS, in CICS410.SDFHSAMP, for the source statements you need for such devices. For information about defining these devices in a TCT, see the *CICS/ESA Resource Definition Guide*.

31 Quiescing sequential devices

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. This means that the terminal is still in RECEIVE status in a subsequent warm start, and CICS does not then read the input file. For more information about how to restart a device that has been closed in a previous run of CICS by means of a CESF GOODNIGHT transaction, see page 86.

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. For information about CICS storage requirements, and the effect on CICS performance, see the *CICS/ESA Performance Guide*.

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 *MVS/ESA Installation Exits* manual.

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 actually 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/ESA 4.1 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 *MVS/ESA 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 running CICS with local DL/I support, the amount of storage left after deducting DSA requirements must be sufficient for IMS code; IMS modules are not loaded in the DSAs.
- 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 guidance information about virtual storage management in an MVS environment, see the *MVS/ESA Initialization and Tuning Guide* manual.

Storage protection

CICS releases before CICS/ESA 3.3 run mainly in single storage key (key 8) for the whole of their execution. Running a single storage key means that user application programs have the same access to CICS code and control blocks as CICS itself, and there is no protection against “rogue” applications overwriting parts of storage inadvertently.

CICS releases from CICS/ESA 3.3 onward use the extensions added to ESA/390 storage protection facilities, available under MVS/ESA Version 4 Release 2.2, to prevent CICS code and control blocks from being overwritten accidentally by your own 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/ESA 4.1 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 ERDSA 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$$, a default system initialization table. This default table is supplied in the CICS410.SDFHSAMP library in source form, and you can modify this to suit your own requirements. When assembled and link-edited, DFHSIT$$ becomes the unsuffixed DFHSIT, which is supplied in pregenerated form in CICS410.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 243 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 parameter on the TYPETERM resource definition. The USERAREALEN parameter on a typeterm definition determines the TCTUA sizes for all terminals that reference the typeterm definition.

For TCAM and 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/ESA 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 page 301 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 page 296 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 the ISOLATE option of the TRANSACTION or TRANCLASS resource definition.

For an overview of transaction isolation, and CICS' use of MVS subspaces, see the *CICS/ESA 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 page 285 for details of how to specify the RENTPGM parameter.

The dynamic storage areas and associated storage cushions

CICS/ESA 4.1 supports eight dynamic storage areas (DSAs), each with its own "storage cushion". CICS always allocates eight separate DSAs, 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.

The type of storage for the read-only DSAs is controlled by the RENTPGM system initialization parameter.

The eight DSAs 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 39 on page 362 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 39. 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]

Notes:

1. Not applicable. The RENTPGM option has no effect on these DSAs, for which the storage key is determined only by the STGPROT parameter.
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.

The storage cushions

CICS reserves amounts of storage in the dynamic storage areas (DSAs) 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.

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.

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/ESA 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.

Recommendations

To help CICS optimize its use of the DSAs and their storage cushions, you are recommended to:

- Avoid using large GETMAIN requests.

  The storage cushion is a contiguous block of storage of fixed size, and therefore may 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 need to find out why. Reduce the maximum number of user tasks (the MXT system initialization parameter) to reduce the number of tasks using main storage. You may need either 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 248 and 252 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 whole number of kilobytes
- A whole number 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, 2MB can be coded as either 2048K or 2M. (1KB = 1024 bytes; 1MB = 1024KB = 1048576 bytes.)

If the value you specify is not a multiple of 256KB for DSALIM, or 1MB 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 40:

Table 40. 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/ESA 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.

The sample program consists of the following resources:

DFH0STAT Statistics program, VS COBOL II release 2 or later.
DFH$STAS Assembler language program called by DFH0STAT.
DFH$STCN Assembler language program called by DFH0STAT.
DFH0STM Mapset used by STAT transaction.
STAT Transaction used to invoke DFH0STAT.

You can invoke the sample program during the PLT stage of CICS initialization or as a conversational transaction.

To enable you to use the sample program, you must:

1. Assemble and link-edit the BMS mapset DFH0STM. Include the physical mapset in a library that is in the DFHRPL concatenation. You can either include the symbolic map set in a user copy library or insert it directly into the source for DFH0STAT.

   For more information about installing and using map sets, see Chapter 3, “Installing map sets and partition sets” on page 15 and “Using BMS map sets in application programs” on page 37.

2. Translate the DFH0STAT program source code, turning CICS commands into code understood by the compiler. The program source code is provided in CICS410.SDFHSAMP.

   Note: You must use the translator option SP when translating DFH0STAT.

3. Compile the translator output for DFH0STAT to produce object code.
4. Link-edit the object module to produce a load module, which you store in an application load library that is concatenated to the DFHRPL DD statement of the CICS startup job stream.
5. Create resource definition entries, in the CSD, for the programs, the mapset, and the STAT transaction.
6. Define the system initialization parameter SPOOL=YES. This specifies that you need support for the system spooling interface.

For more information about installing programs, see Chapter 4, “Installing application programs” on page 25.

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.

Note that 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.

For information, see the *MVS Systems Modification* manual for your version of MVS.

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] . . .]
```

procname
The name of the cataloged procedure that defines the job to be started.

identifier
The name you choose to identify the task.

SUB=subystemname
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 *MVS/ESA 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, CICS410.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, CICS410.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 *CICS/ESA 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/ESA CICS-RACF Security Guide*.

Part 4. Appendix

Appendix. System initialization parameters grouped by functional area

The following table is provided as a guide to system initialization parameters, related by function (for example, XRF or intersystem communication). By using this table as a guide, you should find it easier to ensure that you code ***all*** the parameters that are needed for a particular function you require in CICS.

Note: A check mark (√) in column two indicates that the parameters are read from the SIT directly by the CICS component that uses them, and are not obtained through the parameter manager domain interface. For more information about the parameter manager domain, see "The CICS parameter manager domain" on page 321.

Table 41 (Page 1 of 3). System initialization parameters grouped by functional area

Functional group		System initialization keywords
Application considerations	√	CMDPROT, CWAKEY, INITPARM, LGNMSG, OPERTIM, TCTUALOC, TCTUAKEY
Autoinstall for VTAM terminals and APPC connections	√	AIEXIT, AILDELAY, AIQMAX, AIRDELAY
Autoinstall for programs		PGAICTLG, PGAIEXIT, PGAIPGM
Basic mapping support	√	BMS, PGCHAIN, PGCOPY, PGPURGE, PGRET, PRGDLAY, SKRxxxx
Data interchange	√	DIP
DL/I: CICS local DL/I support and IMS data sharing	√	DDIR, DLDBRC, DLI|DL1, DLIOLIM, DLIRLM, DLLPA, DLMON, DLTHRED, DLXCPVR, DMBPL, ENQPL, PDIR, PISCHD, PSBCHK, PSBPL
Dispatcher functions		ICV, ICVTSD, MXT, PRTYAGE, SUBTSKS
Dump functions		DAE, DUMP, DUMPDS, DUMPSW, DURETRY, SYDUMAX, TRDUMAX TRTRANSZ, TRTRANTY
Dynamic transaction backout	√	DBP, DBUFSZ
Exits	√	TBEXITS, TRAP
Extended recovery facility	√	ADI, AUTCONN, CLT, JESDI, PDI, RMTRAN, RST, TAKEOVR, XRF, XRFSOFF, XRFSTME
Files (user)		FCT
Front end programming interface		FEPI
Intersystem communication and multiregion operation	√	APPLID, DTRPGM, DTRTRAN, DTRPGM, IRCSTRT, ISC, MROBTCH, MROLRM, SYSIDNT, (For ISC, see also VTAM group.)

Table 41 (Page 2 of 3). System initialization parameters grouped by functional area

Functional group		System initialization keywords
Journaling	√	AKPFREQ, JCT, JSTATUS, SERIES
Loading programs		LLACOPY, LPA, PGAICTLG, PGAIPGM, PGAIEXIT, PLTPI, PLTPISEC, PLTPIUSR, PRVMOD
Miscellaneous	√	DATFORM, FLDSEP, FLDSTRT, ISRDELAY, MSGCASE, MSGLVL, NATLANG, PRINT, SPOOL, STATRCD
Monitoring		MCT, MN, MNCONV, MNFREQ, MNSUBSYS, MNSYNC, MNTIME, MNEVE, MNEXC, MNPER
RDO: file control attributes for the CICS system definition data set		CSDACC, CSDBKUP, CSDBUFND, CSDBUFNI, CSDDISP, CSDDSN, CSDFRLOG, CSDJID, CSDLSRNO, CSDRECOV, CSDSTRNO
Security	√	CMDSEC, CMDPROT, CONFDATA, CONFTXT, DFLTUSER, ESMEXITS, PLTPISEC, PLTPIUSR, PSBCHK, RESSEC, SEC, SECPRFX, XAPPC, XCMD, XDCT, XFCT, XJCT, XPCT, XPPT, XPSB, XTRAN, XTST, XUSER
Signon		SNSCOPE, USRDELAY (See also Security, and Terminal and LU management.)
Storage management		CDSASZE, CHKSTRM, CHKSTSK, CMDPROT, CWAKEY, DSASLIM, ECDSASZE, EDSALIM, ERDSASZE, ESDSASZE, EUDSASZE, RDSASZE, RENTPGM, SDSASZE, STGPROT, STGRCVY, TCTUALOC, TCTUAKEY, TRANISO, UDSASZE
Supervisor calls	√	CICSSVC, SRBSVC
System initialization		GRPLIST, INITPARM, NEWSIT, PARMERR, PLTPI, PLTPIUSR, PLTPISEC, SIT, START, STARTER, SUFFIX, WRKAREA
System recovery	√	SRT
System termination	√	PLTSD, XLT
Temporary storage	√	TS, TSMGSET, TST

Table 41 (Page 3 of 3). System initialization parameters grouped by functional area

Functional group		System initialization keywords
Terminal and LU management	√	APPLID, CLSDSTP, DSHIPIDL, DSHIPINT, EODI, FLDSEP, FLDSTRT, GMTEXT, GMTRAN, GNTRAN, GRNAME, HPO, ICVTSD, LGNMSG, OPERTIM, OPNDLIM, PRINT, PSDINT, PSTYPE, PVDELAY, RAMAX, RAPOOL, RESP, TCAM, TCP, TCSACTN, TCSWAIT, TCT, TCTUALOC, TCTUAKEY, VTAM (See also autoinstall for VTAM terminals.)
Trace		AUXTR, AUXTRSW, INTTR, GTFTR, TRTABSZ, SPCTR, SPCTRxx, STNTR, STNTRxx, SYSTR, USERTR
Transient data	√	DCT, TD
Timer		ICP, ICVR, OPERTIM

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Sending your comments to IBM

CICS for MVS/ESA

System Definition Guide

SC33-1164-03

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book. Please limit your comments to the information in this book and the way in which the information is presented.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM 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.

You can send your comments to IBM in any of the following ways:

- By mail, use the Readers' Comment Form
- By fax:
  - From outside the U.K., after your international access code use 44 962 870229 (after 16 April 1995, use 44 1962 870229)
  - From within the U.K., use 0962 870229 (after 16 April 1995, use 01962 870229)
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink: WINVMJ(IDRCF)
  - Internet: idrcf@winvmj.vnet.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The page number or topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Readers' Comments

CICS for MVS/ESA

System Definition Guide

SC33-1164-03

Use this form to tell us what you think about this manual. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer. This form is provided for comments about the information in this manual and the way it is presented.

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.

Be sure to print your name and address below if you would like a reply.

Name	Address
Company or Organization	
Telephone	Email

You can send your comments POST FREE on this form from any one of these countries:

Australia	Finland	Iceland	Netherlands	Singapore	United States
Belgium	France	Israel	New Zealand	Spain	of America
Bermuda	Germany	Italy	Norway	Sweden	
Cyprus	Greece	Luxembourg	Portugal	Switzerland	
Denmark	Hong Kong	Monaco	Republic of Ireland	United Arab Emirates	

If your country is not listed here, your local IBM representative will be pleased to forward your comments to us. Or you can pay the postage and send the form direct to IBM (this includes mailing in the U.K.).

1 Cut along this line

2 Fold along this line

By air mail
Par avion

IBRS/CCRI NUMBER: PHQ - D/1348/SO

NE PAS AFFRANCHIR

NO STAMP REQUIRED

REPONSE PAYEE
GRANDE-BRETAGNE

IBM United Kingdom Laboratories
Information Development Department (MP095)
Hursley Park,
WINCHESTER, Hants
SO21 2ZZ United Kingdom

3 Fold along this line

From: Name _____

Company or Organization _____

Address _____

_____

EMAIL _____

Telephone _____

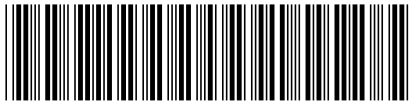
1 Cut along this line

4 Fasten here with adhesive tape

Program Number: 5655-018

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC33-1164-03



Spine information:

CICS for MVS/ESA System Definition Guide *Version 4 Release 1*