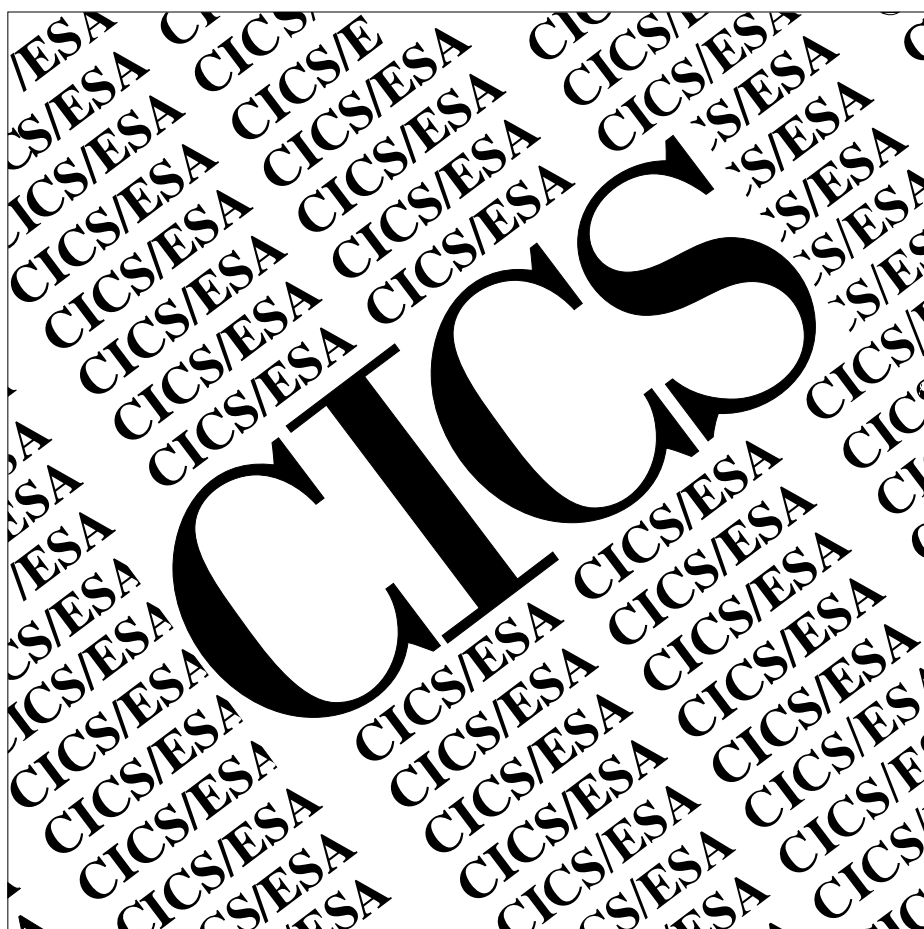


Resource Definition Guide

Version 4 Release 1



CICS for MVS/ESA



Resource Definition Guide

Version 4 Release 1

Note!

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

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 Resource Definition Guide for CICS/ESA 4.1. It is based on the first edition, SC33-1166-00, 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.

© Copyright International Business Machines Corporation 1982, 1997. All rights reserved.

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

Contents

	Notices	ix
	Trademarks and service marks	x
	Preface	xi
	What this book is about	xi
	Who this book is for	xi
	What you need to know to understand this book	xi
	How to use this book	xi
	Notes on terminology	xi
+	CICS syntax notation and conventions used in this book	xii
	Determining if a publication is current	xii
	Bibliography	xiv
	CICS/ESA 4.1 library	xiv
	Other CICS books	xv
	Books from related libraries	xv
	ACF/TCAM books	xv
	MVS books	xv
	Miscellaneous books	xv
	Summary of changes	xvii
+	Changes for this second edition of the CICS/ESA 4.1 book	xvii
	Changes for the first edition of the CICS/ESA 4.1 book	xvii
	Summary of changes to CICS/ESA 3.3	xviii
	Summary of changes to CICS/ESA 3.2.1	xviii
<hr/>		
	Part 1. Introduction	1
	Chapter 1. Resource definition—an introduction	3
	Chapter 2. CICS RDO resources	7
	Chapter 3. Getting started	11
<hr/>		
	Part 2. The CICS system definition file (CSD)	13
	Chapter 4. The CICS system definition file (CSD)	15
	Introducing the CSD	15
	Compatibility mode (CSD sharing)	19
	Installing resource definitions	19
	Dual-purpose resource definition	21
	Duplicate resource definition names	21
	Using lists for initialization	22
	Security and RDO	24
	Dependent default values	26
	Chapter 5. Using CEDA	27
	Entering commands from the command line	39
	Chapter 6. CEDA commands - syntax and examples	43
	ADD a group to a list	44
	ALTER a resource definition	45
	APPEND a list to another list	47
	CHECK related resources	48
	COPY a resource definition	50
	DEFINE a resource definition	52

DELETE a resource definition	53
DISPLAY groups and lists	54
EXPAND one or more groups	56
INSTALL a resource definition or group on an active CICS system	58
LOCK a resource definition	61
MOVE a resource definition	63
REMOVE a group from a list	65
RENAME a resource definition	66
UNLOCK a group or list	67
USERDEFINE	68
VIEW resource definitions	71
Chapter 7. System definition file utility program, DFHCSDUP	73
Sharing the CSD between CICS/ESA 4.1 and earlier releases	73
Input to the DFHCSDUP program	74
Output from the DFHCSDUP program	74
Invoking DFHCSDUP as a batch program	75
Invoking the DFHCSDUP program from a user program	77
Commands for the DFHCSDUP program	78
Chapter 8. DFHCSDUP commands	81
ADD a group to a list	82
ALTER a resource definition	83
APPEND a list to another list	85
COPY a resource definition	87
DEFINE a resource definition	89
DELETE a resource definition, group, or list	91
EXTRACT resource definition data	93
INITIALIZE a new CSD with the required CICS resource definitions	94
LIST resource definitions	95
MIGRATE macro-defined resource definitions from tables to the CSD	97
REMOVE a group from a list	99
SERVICE a CSD	100
UPGRADE a CSD	101
VERIFY a CSD	102

Part 3. Autoinstall	103
Overview of autoinstall	103
Chapter 9. Autoinstall for VTAM terminals	105
Getting started with autoinstall	105
Autoinstall considerations	107
Autoinstall and VTAM	109
Autoinstall and recovery and restart	113
The autoinstall control program	116
Chapter 10. Autoinstall for APPC connections	119
Getting started with APPC connection autoinstall	119
Considerations for connection autoinstall	120
Model definitions for connection autoinstall	120
The autoinstall control program for connection autoinstall	121
Connection autoinstall and recovery and restart	122
Chapter 11. Autoinstall for programs, mapsets, and partitionsets	123
Getting started with program autoinstall	123
Considerations for program autoinstall	124
Model definitions for program autoinstall	125
The autoinstall control program for program autoinstall	125
Program autoinstall and recovery and restart	126

Part 4. RDO resource types and their attributes	127
Chapter 12. CONNECTION	129
Definitions for various links and sessions	130
Defining a CONNECTION	134
Chapter 13. FILE	141
Defining a FILE	142
Chapter 14. LSRPOOL	147
Defining an LSRPOOL	148
Chapter 15. MAPSET	151
Defining a MAPSET	152
Chapter 16. PARTITIONSET	155
Defining a PARTITIONSET	156
Chapter 17. PARTNER	159
Defining a PARTNER	160
Chapter 18. PROFILE	163
Defining a PROFILE	164
Chapter 19. PROGRAM	169
Defining a PROGRAM	170
Chapter 20. SESSIONS	175
Defining a SESSION	176
Chapter 21. TERMINAL	183
Types of terminals	184
Terminals for transaction routing	188
Duplicating definitions	191
Sharing definitions	192
Extended recovery (XRF) for terminals	195
Defining a TERMINAL	197
Chapter 22. TRANCLASS	205
Defining a TRANCLASS	205
Chapter 23. TRANSACTION	207
Defining a TRANSACTION	208
Chapter 24. TYPETERM	217
Devices supported	221
Defining a TYPETERM	225
Part 5. Macro reference information	243
Chapter 25. Introduction to resource definition with macros	245
Introduction to CICS control tables and macros	245
DL/I directories	248
TYPE=INITIAL	248
TYPE=FINAL	249
Format of macros	250
Syntax notation	251

Chapter 26. CLT — command list table	253
Control section — DFHCLT TYPE=INITIAL	253
Specifying alternate systems — DFHCLT TYPE=LISTSTART	254
Specifying takeover commands — DFHCLT TYPE=COMMAND	254
Messages to the operator — DFHCLT TYPE=WTO	254
Closing the command list — DFHCLT TYPE=LISTEND	255
Chapter 27. DCT — destination control table	257
DFHDCT macro types	257
Control section — DFHDCT TYPE=INITIAL	257
Data set control information — DFHDCT TYPE=SDSCI	257
Extrapartition destinations — DFHDCT TYPE=EXTRA	258
Indirect destinations — DFHDCT TYPE=INDIRECT	259
Intrapartition destinations — DFHDCT TYPE=INTRA	260
Remote destinations — DFHDCT TYPE=REMOTE	262
DFHDCT examples	263
Required entries in the destination control table	264
Chapter 28. DL/I directories — DDIR and PDIR	267
DL/I database directory — DDIR	267
Control section — DFHDLDBD TYPE=INITIAL	267
Physical databases — DFHDLDBD TYPE=ENTRY	267
DL/I PSB directory — PDIR	268
Control section — DFHDLPSB TYPE=INITIAL	268
Program specification blocks — DFHDLPSB TYPE=ENTRY	268
Chapter 29. FCT — file control table	269
DFHFCT macro types	269
Control section — DFHFCT TYPE=INITIAL	269
Local files — DFHFCT TYPE=FILE	269
Migrating FCT definitions — DFHFCT TYPE=GROUP	275
Remote files — DFHFCT TYPE=REMOTE	276
Example	277
Chapter 30. JCT — journal control table	279
Elements of DFHJCT	279
Control section — DFHJCT TYPE=INITIAL	280
Journal entries — DFHJCT TYPE=ENTRY	280
DFHJCT example	283
Chapter 31. MCT — monitoring control table	285
Elements of DFHMCT	285
Control section — DFHMCT TYPE=INITIAL	285
User event monitoring points — DFHMCT TYPE=EMP	285
Control data recording — DFHMCT TYPE=RECORD	288
DFHMCT example	290
Chapter 32. PLT — program list table	291
Elements of DFHPLT	291
Control section — DFHPLT TYPE=INITIAL	291
Entries in program list table — DFHPLT TYPE=ENTRY	292
DFHPLT example	293
Chapter 33. RST — recoverable service table	295
DFHRST macro types	295
Control section — DFHRST TYPE=INITIAL	295
Recoverable service elements — DFHRST TYPE=RSE	295
DBCTL subsystems — DFHRST TYPE=SUBSYS	295
DFHRST example	296

Chapter 34. SRT — system recovery table	297
DFHSRT macro types	297
Control section — DFHSRT TYPE=INITIAL	297
Abend codes — DFHSRT TYPE=SYSTEM USER	297
DFHSRT example	298
Chapter 35. TCT — terminal control table	299
How this chapter is organized	299
Introduction to the TCT	299
DFHTCT macro types	299
Control section — DFHTCT TYPE=INITIAL	300
Migrating TCT definitions — DFHTCT TYPE=GROUP	301
Logical device codes	302
Sequential devices	306
TCAM DCB interface	309
Remote terminals for transaction routing	321
CICS terminals list	324
Chapter 36. TLT — terminal list table	329
Elements of DFHTLT	329
Control section — DFHTLT TYPE=INITIAL	329
Entries in terminal list table — DFHTLT TYPE=ENTRY	329
DFHTLT example	330
Chapter 37. TST — temporary storage table	331
Elements of DFHTST	331
Control section — DFHTST TYPE=INITIAL	331
Recoverable temporary storage — DFHTST TYPE=RECOVERY	332
Local temporary storage—DFHTST TYPE=LOCAL	332
Remote temporary storage — DFHTST TYPE=REMOTE	333
Temporary storage security checking — DFHTST TYPE=SECURITY	334
DFHTST example	335
Chapter 38. XLT — transaction list table	337
Elements of DFHXLT	337
Control section — DFHXLT TYPE=INITIAL	337
Entries in transaction list table — DFHXLT TYPE=ENTRY	337
DFHXLT example	338

Part 6. Appendixes	339
---------------------------	-----

Appendix A. Obsolete attributes retained for compatibility	341
Appendix B. CICS-supplied resource definitions, groups, and lists	347
DFHLIST definitions	348
CICS-supplied groups not in DFHLIST	350
CICS-supplied compatibility groups	351
The sample application program groups	353
CICS transactions supplied by IBM	357
TYPETERM definitions in group DFHTYPE	361
Model TERMINAL definitions in group DFHTERM	368
PROFILE definitions in group DFHISC	371
PROFILE definitions in group DFHSTAND	372
Model definitions in group DFHPGAIP	375
Appendix C. Keyword cross-reference tables	377
Macro operand to RDO keyword	377
RDO keyword to macro operand	382

Appendix D. Migrating the TCT to the CSD	389
Summary of implementation	389
Editing your existing DFHTCT source	390
Assembling and link-editing	390
Using the MIGRATE command	391
Checking the output from MIGRATE	392
Using DFHCSDUP to add migrated groups to a list	392
Operations after migration	393
Final modification of the macro source	393
Abandoning the use of DFHTCT macros (if possible)	393
Migrating remote terminal definitions	394
Migrating different macro types	396
Index	403

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
AIX
Application System/400
AS/400
BookManager
C/370
CICS
CICS OS/2
CICS/ESA
CICS/MVS
CICS/VM
CICS/6000
CICSPlex
CUA
DATABASE 2
DB2
Enterprise Systems Architecture/370
Hiperspace
IBM
IBMLink
IMS
IMS/ESA
Language Environment
MVS/DFP
MVS/ESA
MVS/SP
MVS/XA
NETVIEW
OS/2
Personal System/2
PS/2
RACF
S/370
SAA
Selectric
Series/1
System/36
System/370
System/38
System/390
VTAM
3090

Preface

What this book is about

This book describes how to define the characteristics of your data processing resources to your CICS system. All four methods of resource definition are described:

- Macro definition
- Online definition
- Automatic installation (autoinstall)
- Batch definition.

Who this book is for

This book is for those responsible for defining resources to CICS.

What you need to know to understand this book

The book assumes that you have a basic understanding of CICS concepts and facilities, which you can gain by reading the *CICS Family: General Information* and the *CICS/ESA 3.3 Facilities and Planning Guide*. You must also be familiar with your own system and the resources to be defined and maintained.

How to use this book

Part 1, "Introduction" on page 1 is an introduction to resource definition for CICS; read it if you are new to CICS.

Part 2, "The CICS system definition file (CSD)" on page 13 introduces the CICS system definition file (CSD), and provides guidance and reference for using the online transaction CEDA and the offline utility DFHCSDUP.

Part 3, "Autoinstall" on page 103 explains automatic installation (autoinstall); read this if you want to make use of the benefits offered by autoinstall.

Part 4, "RDO resource types and their attributes" on page 127 describes each of the resources definable by CEDA, DFHCSDUP, and autoinstall. It is intended as reference material.

Part 5, "Macro reference information" on page 243 explains how to define resource by macro tables. Use it for reference if you use macro definitions.

Notes on terminology

"CICS/MVS" is used for Customer Information Control System/Multiple Virtual Storage.

"CICS/ESA" is used for Customer Information Control System/Enterprise System Architecture.

Other abbreviations that may be used for CICS releases are as follows:

For CICS/MVS Version 2 Release 1 and subsequent modification levels	– CICS/MVS 2.1
For CICS/ESA Version 3 Release 3	– CICS/ESA 3.3
For CICS/ESA Version 4 Release 1	– CICS/ESA 4.1

The MVS/Enterprise System Architecture (MVS/ESA) operating system, which is a prerequisite for CICS/ESA Version 4, is generally referred to as 'MVS'.

+ CICS syntax notation and conventions used in this book

+ In the CICS books, CICS commands are presented in a standard way.

+ Each command has a syntax box to show you what options there are.

+ You interpret the syntax by following the arrows from left to right. The conventions are:

Symbol	Action
	A set of alternatives—one of which you must code.
	A set of alternatives—one of which you must code. You may code more than one of them, in any sequence.
	A set of alternatives—one of which you may code.
	A set of alternatives — any number (including none) of which you may code once, in any sequence.
	Alternatives where A is the default.
	Use with the named section in place of its name.
Punctuation and uppercase characters	Code exactly as shown.
Lowercase characters	Code your own text, as appropriate (for example, name).

+ For example, with `DEFINE FILE(filename)` you must code `DEFINE FILE` and `()` unchanged, but are free to code any valid text string to mean the name of the file.

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.

Bibliography

CICS/ESA 4.1 library

Evaluation and planning		
<i>Release Guide</i>	GC33-1161	April 1997
<i>Migration Guide</i>	GC33-1162	April 1997
General		
<i>CICS Family: Library Guide</i>	GC33-1226	April 1995
<i>Master Index</i>	SC33-1187	October 1994
<i>User's Handbook</i>	SX33-1188	April 1997
<i>Glossary (softcopy only)</i>	GC33-1189	n/a
Administration		
<i>Installation Guide</i>	GC33-1163	April 1997
<i>System Definition Guide</i>	SC33-1164	April 1997
<i>Customization Guide</i>	SC33-1165	April 1997
<i>Resource Definition Guide</i>	SC33-1166	April 1997
<i>Operations and Utilities Guide</i>	SC33-1167	April 1997
<i>CICS-Supplied Transactions</i>	SC33-1168	April 1997
Programming		
<i>Application Programming Guide</i>	SC33-1169	October 1994
<i>Application Programming Reference</i>	SC33-1170	April 1997
<i>System Programming Reference</i>	SC33-1171	April 1997
<i>Sample Applications Guide</i>	SC33-1173	October 1994
<i>Distributed Transaction Programming Guide</i>	SC33-1174	October 1994
<i>Front End Programming Interface User's Guide</i>	SC33-1175	October 1994
Diagnosis		
<i>Problem Determination Guide</i>	SC33-1176	October 1994
<i>Messages and Codes</i>	GC33-1177	April 1997
<i>Diagnosis Handbook</i>	LX33-6093	October 1994
<i>Diagnosis Reference</i>	LY33-6082	April 1997
<i>Data Areas</i>	LY33-6083	April 1997
<i>Supplementary Data Areas</i>	LY33-6081	October 1994
<i>Closely-Connected Program Interface</i>	LY33-6084	November 1996
Communication		
<i>Intercommunication Guide</i>	SC33-1181	April 1997
<i>Server Support for CICS Clients</i>	SC33-1591	February 1996
<i>CICS Family: Inter-product Communication</i>	SC33-0824	October 1996
<i>CICS Family: Communicating from CICS on System/390</i>	SC33-1697	October 1996
Special topics		
<i>Recovery and Restart Guide</i>	SC33-1182	October 1994
<i>Performance Guide</i>	SC33-1183	October 1994
<i>CICS-IMS Database Control Guide</i>	SC33-1184	October 1994
<i>CICS-RACF Security Guide</i>	SC33-1185	October 1994
<i>Shared Data Tables Guide</i>	SC33-1186	October 1994
<i>External CICS Interface</i>	SC33-1390	April 1997
<i>CICS ONC RPC Feature for MVS/ESA Guide</i>	SC33-1119	February 1996
<i>CICS Web Interface Guide</i>	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 xii. 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
- *CICS/OS/VS IBM 3270 Data Stream Device Guide*, SC33-0232
- *CICS/OS/VS IBM 3650/3680 Guide*, SC33-0234
- *CICS/OS/VS IBM 3767/3770/6670 Guide*, SC33-0235
- *CICS/OS/VS IBM 3790/3730/8100 Guide*, SC33-0236

Books from related libraries

This section lists the non-CICS books that are referred to in this manual.

ACF/TCAM books

- *ACF/TCAM Version 3 Application Programming*, SC30-3233.

MVS books

Short Title	Version 3 book	Version 4 book
<i>Authorized Assembler Programming Reference</i>	<i>MVS/ESA System Programming Library: Application Development Macro Reference</i> , GC28-1857	<i>MVS/ESA Application Development Reference: Services for Authorized Assembler Language Programs Volume 1</i> , GC28-1647
—	<i>MVS/ESA Data Administration: Macro Instruction Reference</i> , SC26-4506	—
JES2 Initialization and Tuning	<i>MVS/ESA System Programming Library: JES2 Initialization and Tuning</i> , SC28-1038	<i>MVS/ESA JES2 Initialization and Tuning Guide</i> , SC23-1082
JES3 Initialization and Tuning	<i>MVS/ESA System Programming Library: JES3 Initialization and Tuning</i> , SC23-0073	—

Miscellaneous books

- *DATABASE 2 Version 2 Administration Guide*, SC26-4374
- *IBM 3290 Information Panel Description and Reference*, GA23-0021
- *Language Environment/370 Programming Guide*, SC26-4818
- *Resource Access Control Facility (RACF) Security Administrator’s Guide*, SC28-1340.
- *DPPX/Distributed Presentation Services Version 2: System Programming Guide*, SC33-0117.

Summary of changes

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

+ This book is the second edition of the *CICS/ESA Resource 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.

+ For this edition, TST definition has been extended by adding the TYPE=LOCAL macro and
+ the null DATAID parameter — DATAID=(). Also, there are various changes resulting from
+ APARs and Readers Comments.

Changes for the first edition of the CICS/ESA 4.1 book

This book is based on the CICS/ESA 3.3 *Resource Definition (Online)* and the *Resource Definition (Macro)* manuals.

Changes for this edition of the manual are marked by vertical bars to the left of the text.

For CICS/ESA 4.1, the following changes have been made:

Changes to resource definition online (RDO)

- There are seven new attributes on the TRANSACTION definition:
 - CONFDATA
 - ISOLATE
 - RUNAWAY
 - SHUTDOWN
 - STORAGECLEAR
 - TASKDATAKEY
 - TRANCLASS.
- The TCLASS attribute on the TRANSACTION definition is now obsolete, and is replaced by the TRANCLASS attribute.
- A new resource type, TRANCLASS, has been added.
- The meaning of EXECKEY on the PROGRAM definition has been changed.
- There are five new attributes on the CONNECTION definition:
 - CONNTYPE
 - MAXQTIME
 - QUEUELIMIT
 - PSRECOVERY
 - REMOTESYSNET.
- A new value, EXCI, on the PROTOCOL attribute of the CONNECTION definition.
- A new value, EXCI, on the PROTOCOL attribute of the SESSIONS definition.
- Removal of the RECOVNOTIFY attribute from the SESSIONS definition.
- Changes to the RECOVOPTION attribute of the SESSIONS definition.
- Changes to the RECOVNOTIFY and RECOVOPTION attributes of the TYPETERM definition.

- CEDA gives the current resource name for the ALTER, DEFINE, and VIEW commands.
- Before-images of a CEDA ALTER command are logged to the CSDL queue.
- New and changed CICS-supplied resource definitions.
- The introduction of autoinstall for programs, mapsets, and partitionsets.
- The introduction of autoinstall for APPC connections.
- The system's SYSID is displayed on all RDO panels.
- The CEDA CHECK LIST command is extended to four lists.

Changes to resource definition at macro level

- DFHDCT now operates over a whole CICSplex
- CPU and CONV are removed from DFHMCT TYPE=INITIAL
- Increase in size of the MOVE option of the PERFORM operand of DFHMCT TYPE=EMP
- Removal of the DFHPCT macro table
- Removal of the DFHPPT macro table
- Removal of VSAM file definition from DFHFCT
- Removal of data tables definition from DFHFCT
- Removal of local shared resource pool definition from DFHFCT.

Summary of changes to CICS/ESA 3.3

The main changes for CICS/ESA 3.3 were:

- the addition of information about the DFHCSDUP generic copy command.
- the addition of the EXECKEY, EXECUTIONSET, REMOTENAME, REMOTESYSTEM, and TRANSID keywords for PROGRAM definition.
- the addition of the TASKDATAKEY keyword for TRANSACTION definition.

Summary of changes to CICS/ESA 3.2.1

The main changes in this book for CICS/ESA 3.2.1 were the addition of information about:

- Compatibility mode
- Dual-purpose resource definition.

Part 1. Introduction

This part introduces resource definition as it applies to CICS/ESA 4.1. You should read it if you are new to CICS or to resource definition. It consists of the following chapters:

- Chapter 1, "Resource definition—an introduction" on page 3
- Chapter 2, "CICS RDO resources" on page 7
- Chapter 3, "Getting started" on page 11

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 minimises 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 1 on page 4 shows you the methods you can use for each resource.

Table 2 on page 5 suggests some of the things you should consider when deciding which definition method to use when there is a choice.

Introduction

Table 1. 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

<i>Table 2. Methods of resource definition</i>			
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.	<ul style="list-style-type: none"> • RDO is used while CICS is running, so allows fast access to resource definitions. 	<ul style="list-style-type: none"> • 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.	<ul style="list-style-type: none"> • 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. 	<ul style="list-style-type: none"> • 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.	<ul style="list-style-type: none"> • 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. 	<ul style="list-style-type: none"> • 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.	<ul style="list-style-type: none"> • Where possible, you should use the other methods. 	<ul style="list-style-type: none"> • 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.

Chapter 2. CICS RDO resources

This chapter introduces the CICS resources that you can define by RDO. For a description of the resources you can define by macro, see Chapter 25, "Introduction to resource definition with macros" on page 245.

FILE	CICS file control uses the FILE definition for information on the physical and operational characteristics of a file, such as types of operations allowed on the file, recovery attributes, and operations to be journaled. Remote files (that is, files on another system) are also defined using this RDO resource type. See Chapter 13, "FILE" on page 141 for further information and keywords.
LSRPOOL	The local shared resources (LSR) pool is a reserve of data buffers, strings, and HiperSpace buffers (see the <i>CICS/ESA Performance Guide</i> for more information on HiperSpace) that VSAM uses when processing access requests for certain files. See Chapter 14, "LSRPOOL" on page 147 for further information and keywords.
MAPSET	Each interactive application using a display device can use specific screen layouts, or maps. Each map can be used by multiple invocations of the same program, or by different programs. You use either basic mapping support (BMS) or Screen Definition Facility (SDF) to create maps. Every map must belong to a mapset. See Chapter 15, "MAPSET" on page 151 for further information and keywords.
PARTITIONSET	The screen areas of some display devices (for example, the 8775 Display Terminal, and the IBM 3290 Information Panel) can be divided into partitions , each of which can be treated as a separate display. Different programs or transactions can write to or receive input from different partitions. See Chapter 16, "PARTITIONSET" on page 155 for further information and keywords.
PARTNER	You use the PARTNER definition to enable CICS application programs to communicate via APPC protocols to a partner application program running on a remote logical unit. This interaction is called a conversation . The PARTNER definition also facilitates the use of the call to the interface with the communications element of the System Application Architecture. See Chapter 17, "PARTNER" on page 159 for further information and keywords.
PROFILE	The PROFILE definitions is used to specify options that control the interactions between transactions and terminals or logical units. It is a means of standardizing the use of, for example, screen size and printer compatibility. Each TRANSACTION definition names the PROFILE to be used. See Chapter 18, "PROFILE" on page 163 for further information and keywords.
PROGRAM	You use the PROGRAM definition to describe the control information for a program that is stored in the program library and used to process a transaction. See Chapter 19, "PROGRAM" on page 169 for further information and keywords.
TERMINAL	CICS needs a definition for each terminal with which it communicates. A terminal's unique properties are in its TERMINAL definition. Properties that it has in common with other terminals (usually static) are in the TYPETERM definition. See Chapter 21, "TERMINAL" on page 183 for further information and keywords.
TRANSACTION	A CICS application consists of one or more programs written to perform a specific function. A particular invocation of such an application is a transaction . In the TRANSACTION definition you specify options related to functions provided by CICS itself, such as transaction priority, security

key, and the length of the transaction work area (TWA). See Chapter 23, “TRANSACTION” on page 207 for further information and keywords.

TRANCLASS	By putting your transactions into transaction classes (TRANCLASSES), you can control how CICS dispatches tasks. For example, you can separate transactions into those that are heavy resource users and those that are of lesser importance, such as the “Good morning” broadcast messages. You can then use the attributes on the TRANCLASS definition to control the number of active and new tasks allowed from each transaction class. See Chapter 22, “TRANCLASS” on page 205 for further information and keywords.
TYPETERM	A TYPETERM is a partial terminal definition which identifies a set of common terminal properties or attributes. Every TERMINAL definition must specify a TYPETERM. TYPETERMS make it easier to define your terminals if you have many terminals of the same kind. See Chapter 24, “TYPETERM” on page 217 for further information and keywords.
CONNECTION	A CONNECTION is the definition of a remote system that your CICS system communicates with using intersystem communication (ISC) or multiregion operation (MRO). See Chapter 12, “CONNECTION” on page 129 for more information.
SESSIONS	Before two systems can communicate using ISC or MRO, they must be logically linked through one or more sessions. The nature of the link determines how they can communicate. You specify the link in the SESSIONS definition. See Chapter 20, “SESSIONS” on page 175 for further information and keywords.

How the resource types relate to each other

Every resource is defined with a set of **attributes**. The attributes are the properties of the resource, telling CICS such things as whether a file can be updated, what security level should be given to a transaction, what remote systems CICS can communicate with, and so on. Some of the attributes relate resources to each other. For example:

- Every TRANSACTION definition must refer to either a PROGRAM definition or a REMOTESYSTEM definition.
- Every SESSIONS definition must refer to a CONNECTION definition.
- Every TERMINAL definition must refer to a TYPETERM definition.

Figure 1 on page 9 shows how the definitions for the RDO resource types relate to each other.

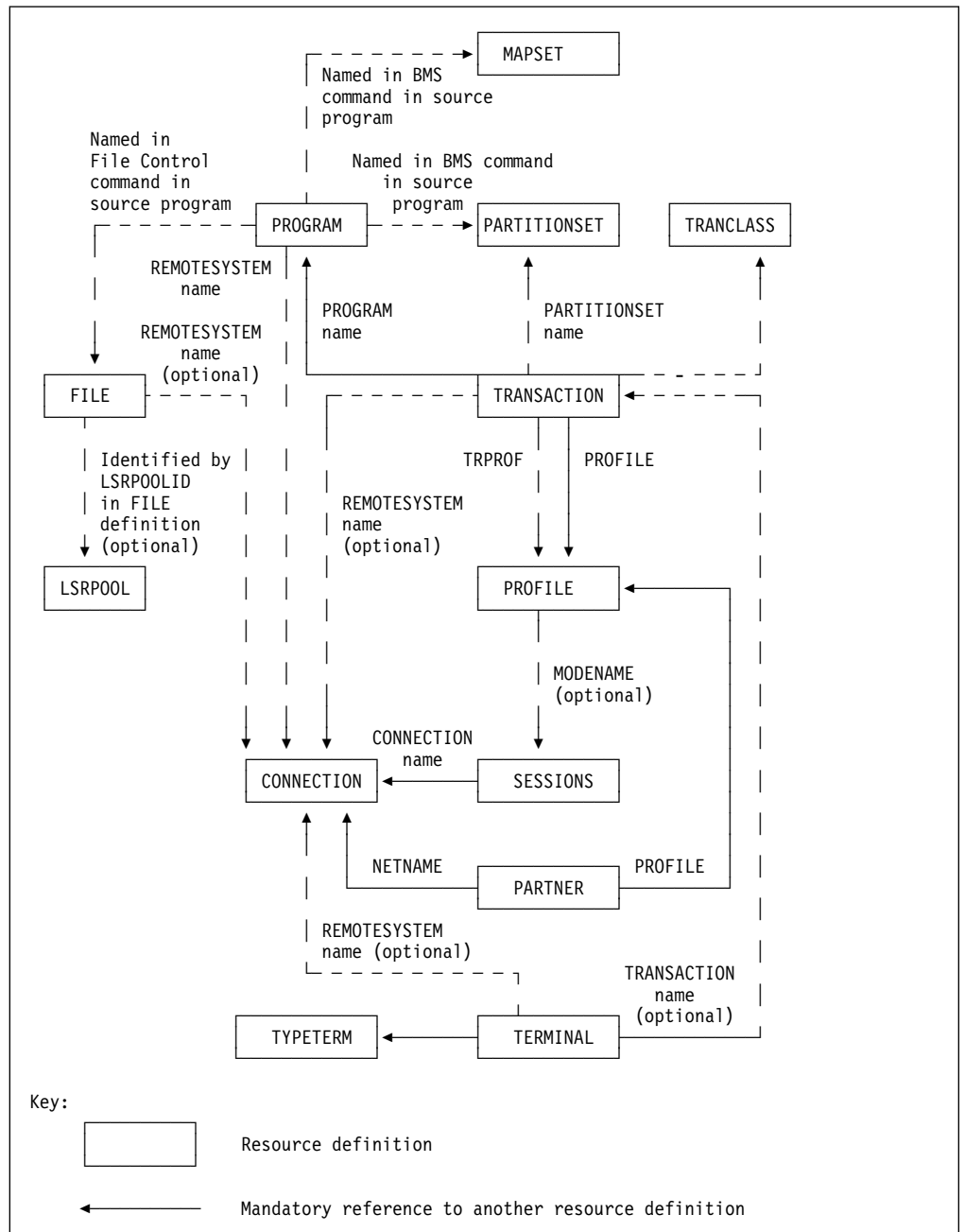


Figure 1. How resource definitions refer to each other

Chapter 3. Getting started

The following list shows you what you must do to begin defining resources. If you are new to CICS, all the items in the list may apply to you; if you are an existing CICS user but are new to RDO, some items may not apply to you.

1. Create and initialize a CSD

If you are a new CICS user, you must create and initialize a CICS system definition file (CSD). The CSD is used to store records of all resources defined by CEDA or DFHCSDUP. Refer to the *CICS/ESA System Definition Guide* for information on how to do this.

2. Migrate existing definitions

If you are moving to CICS/ESA 4.1 from an earlier release and you have any of the following definitions, you **must** migrate them to the CSD before using CICS/ESA 4.1:

- All of your PCT
- All of your PPT
- All FCT entries except those for BDAM files
- VTAM TCT entries.

CICS/ESA 4.1 does not support macro definition of these resources.

3. Run a DFHCSDUP UPGRADE job

Do this to bring the definitions in your CSD up to the level of CICS/ESA 4.1 function. See "UPGRADE a CSD" on page 101 for information on how to do this.

4. Work with your resource definitions

Use Table 1 on page 4 and Table 2 on page 5, to help you decide which methods to use to define and manage your resources. If you are an existing CICS user, some of these decisions will have been made for you already.

When you know which methods you want to use, read the following sections:

- If you want to use macro definitions, read Part 5, "Macro reference information" on page 243.
- If you want to use CEDA, read Chapter 5, "Using CEDA" on page 27.
- If you want to use DFHCSDUP, read Chapter 7, "System definition file utility program, DFHCSDUP" on page 73.
- If you want to use autoinstall, read Part 3, "Autoinstall" on page 103.

Part 2. The CICS system definition file (CSD)

This part explains what the CICS system definition file (CSD) is and how it relates to CICS. It contains the following chapters:

- Chapter 4, "The CICS system definition file (CSD)" on page 15
- Chapter 5, "Using CEDA" on page 27
- Chapter 6, "CEDA commands - syntax and examples" on page 43
- Chapter 7, "System definition file utility program, DFHCSDUP" on page 73
- Chapter 8, "DFHCSDUP commands" on page 81

Chapter 4. The CICS system definition file (CSD)

This chapter introduces the CSD. It contains the following information:

- “Introducing the CSD”
- “Compatibility mode (CSD sharing)” on page 19
- “Installing resource definitions” on page 19
- “Dual-purpose resource definition” on page 21
- “Duplicate resource definition names” on page 21
- “Using lists for initialization” on page 22
- “Security and RDO” on page 24
- “Dependent default values” on page 26.

Introducing the CSD

The CICS system definition (CSD) file is a VSAM data set containing a resource definition record for every resource defined to CICS by means of CEDA or DFHCSDUP. It can be defined as recoverable, so that changes that were incomplete when an abend occurred will be backed out.

Creating a CSD

If you do not already have a CSD, you must create one. For detailed information about creating a CSD, see the *CICS/ESA System Definition Guide*.

You can create more than one CSD, depending on your requirements. For example, you can have different CSDs for different systems, so that your test systems and production systems are separate from each other.

You can also share one CSD between CICS releases; see “Compatibility mode (CSD sharing)” on page 19 for information about this.

When the CSD has been initialized it contains a number of groups (all beginning with the letters ‘DFH’) containing related resource definitions, and one list, called DFHLIST. These definitions are supplied by CICS and are necessary for some system functions and for running CICS-supplied transactions.

How the CSD is organized — groups and lists

Information on the CSD is organized into **groups** and **lists**. The main purpose of the **group** is to provide a convenient method of collecting related resources together on the CSD. Every resource that you define must belong to a group; you cannot define a resource without also naming its group.

A **list** contains the names of groups that CICS installs at a cold start. You can add groups to lists if you want them to be installed at a cold start, or if it helps you to manage your groups better, but groups do not need to belong to lists, and can be defined independently.

Creating groups and lists

A group is created when you specify it as the GROUP name in a DEFINE command or as the TO group in a COPY command. For example, the command:

```
CEDA DEFINE PROGRAM(PROG1) GROUP(MYGROUP)
```

defines a program called PROG1, and creates a group called MYGROUP if it does not already exist. See page 50 for information on the DEFINE command, and page 52 for information on the COPY command.

These are the only ways to create a group; a nonexistent group can be named in a list, but naming it in a list does not create it.

A group must not have the same name as an existing group or list.

You can create a list in either of the following ways:

- Use the ADD command to add a group to a list. If the specified list does not exist, it is created. See page 44 for information about the ADD command.
- Use the APPEND command to append the contents of one list to another list. If the appended-to list does not exist, it is created, containing the contents of the first list. See page 47 for information about the APPEND command.

A list must not have the same name as an already existing group or list.

Managing resource definitions

You manage your resource definitions using commands supplied as part of RDO. These commands allow you to work with your resources, defining, deleting, copying, renaming, and so on.

The commands are listed in Table 3 on page 17. For the syntax of these commands and information on how to use them, Chapter 6, “CEDA commands - syntax and examples” on page 43 and Chapter 8, “DFHCSDUP commands” on page 81. To help you use CEDA, there is also a CEDA tutorial in Chapter 5, “Using CEDA” on page 27.

CEDA and DFHCSDUP commands

<i>Table 3. CEDA and DFHCSDUP commands</i>		
COMMAND	FUNCTION	See page
ADD	Adds a group name to a list.	44 (RDO), 82 (DFHCSDUP)
ALTER	Modifies the attributes of an existing resource definition.	45 (RDO), 83 (DFHCSDUP)
APPEND	Copies a list to the end of another list.	47 (RDO), 85 (DFHCSDUP)
CHECK	Cross checks the resource definitions within a group, or within the groups in a list or lists, up to a maximum of four lists. The CHECK command is RDO only.	48
COPY	Copies one or more resource definitions from one group to another, or one resource definition within a group.	50 (RDO), 87 (DFHCSDUP)
DEFINE	Creates a new resource definition.	52 (RDO), 89 (DFHCSDUP)
DELETE	Deletes one or more resource definitions.	53 (RDO), 91 (DFHCSDUP)
DISPLAY	Shows the names of one or more groups, lists, or resource definitions within a group. The DISPLAY command is RDO only.	54
EXPAND	Shows the names of the resource definitions in one or more groups or lists. The EXPAND command is RDO only.	56
EXTRACT	Extracts and processes resource definition data from groups or lists on the CSD. The EXTRACT command is DFHCSDUP only.	93
INITIALIZE	Prepare a newly-defined data set for use as a CSD file. The INITIALIZE command is DFHCSDUP only.	94
INSTALL	Dynamically adds a resource definition or a group of resource definitions to the active CICS system. The INSTALL command is RDO only.	58
LIST	Produce listings of the current status of the CSD. The LIST command is DFHCSDUP only.	95
LOCK	Prevents other operators updating or deleting a group or the groups in a list. The LOCK command is RDO only.	61
MIGRATE	Transfers the contents of a terminal control table (TCT) or file control table (FCT) to the CSD. The MIGRATE command is DFHCSDUP only.	97
MOVE	Moves one or more resource definitions from one group to another. The MOVE command is RDO only.	63
REMOVE	Removes a group name from a list.	65 (RDO), 99 (DFHCSDUP)
RENAME	Renames a resource definition, either within a group, or while simultaneously moving it to another group. The RENAME command is RDO only.	66
SERVICE	Applies corrective maintenance to the CSD. The SERVICE command is DFHCSDUP only.	100
UNLOCK	Releases a lock on a group or list. The UNLOCK command is RDO only.	67
UPGRADE	Upgrades the CICS-supplied resource definitions on the CSD (for example, when you migrate to a higher release of CICS). The UPGRADE command is DFHCSDUP only.	101
USERDEFINE	Creates a new resource definition with your own defaults. The USERDEFINE command is RDO only.	68
VERIFY	Removes internal locks on groups and lists. The VERIFY command is DFHCSDUP only.	102
VIEW	Shows the attributes of an existing resource definition. The VIEW command is CEDA only.	71

How many resource definitions should a group contain?

Try to keep your groups to a manageable size; ideally, there should be no more than about 100 resource definitions in a group. You should allocate your resource definitions between groups in such a way as to obtain optimum performance, in both system and administration terms. The following considerations may help:

- A large group can involve a lot of unnecessary processing time to install. This is particularly true of those containing TERMINAL and SESSIONS definitions, because they take a large amount of dynamic storage.
- A large number of very small groups can also use unnecessary processing time, due to the extra I/O involved as a result of many group names having to be read from the CSD. In theory, you could have one resource definition per group, but this is not recommended; the processing of a large number of single-resource groups can affect DASD space, cold start performance, and the performance of both CEDA and DFHCSDUP.
- Administration is easier if you have smaller groups. For example, the DISPLAY GROUP ALL command involves a lot of scrolling if the resource definitions in the group extend over many screens. You cannot see at a glance what you have in a large group.
- You may find that you have storage problems when you EXPAND, COPY or INSTALL a large group. In particular, if a very large number of CSD records are defined in a region with a small DSA, issuing a

```
CEDA EXPAND GROUP(*)
```

results in the system going SOS (short on storage).

What should be in a group?

Usually, the definitions within a group have something in common. For example:

For application resources:

- You will probably find it most convenient to keep all the resource definitions belonging to one application in one group.
- If you use PARTITIONSET or PROFILE definitions for many applications, keeping them separate in their own groups avoids the possibility of unnecessary duplication.

For communication resources:

- SESSIONS definitions **must** be in the same group as the CONNECTION definition to which they refer. You may have more than one group of definitions for each system and its sessions with other systems, in a single CSD that is shared by all the systems; be careful that you install each group of definitions in the correct system.
- We recommend that you restrict a group to contain only one CONNECTION definition with its associated SESSIONS definitions.
- We recommend that you keep all your TYPETERM definitions in one group. This avoids the possibility of unnecessary duplication. You must put the group of TYPETERMs before the groups of TERMINAL definitions in your lists.
- You will probably find it convenient to group TERMINAL definitions according to departmental function or geographical location.
- You **must** keep all the TERMINAL definitions for one pool of pipeline terminals in the same group.
- You should keep AUTINSTMODEL TERMINAL definitions separately in a group of their own.

Compatibility mode (CSD sharing)

CICS supports the capability for a CSD and its resource definitions to be shared between different CICS systems. The systems might be running the same or different releases of CICS. *Compatibility mode* is intended for use when you want to create or change resource definitions on a CSD which is shared between different releases.

All maintenance should be done under the latest release of CICS. This avoids the risk of earlier releases modifying entries created under more recent releases with new keywords that the older version does not recognize. Ensure this by restricting write access to the CSD to the latest release. See the *CICS/ESA System Definition Guide* for further details on defining CSDs.

Compatibility mode is entered by using PF2 on the CEDA panels where it is available. It gives you access to those keywords which were current at your earlier release, but are obsolete at your later release. However, it is only possible to use *compatibility mode* with commands affecting individual resources: you cannot perform generic commands (VIEW, ALTER and DEFINE) in *compatibility mode*.

You can find further information about issues relating to compatibility mode in the following places:

- For the usage and meaning of keywords and their compatibility with previous releases of CICS, see Appendix A, “Obsolete attributes retained for compatibility” on page 341.
- For information about what compatibility groups you need in your startup group list in order for CSD sharing to work, see Table 90 on page 351 and Table 91 on page 352.
- If you need more detailed information about migration and compatibility, see the *CICS/ESA Migration Guide*.
- For information on defining your CSDs, see the *CICS/ESA System Definition Guide*.

Installing resource definitions

This section explains the installation of resource definitions into an active CICS system.

When a resource definition is installed, the information on it is moved into CICS storage, to become an entry in a “table”. This does not necessarily mean that all the information is stored in a large, contiguous area; the tables are virtual, but it is convenient to refer to them in this way.

+
+

In this section we look at what happens to resource definitions:

- When CICS is initialized
- When you use the INSTALL command.

We also discuss dual-purpose resource definitions which allow resource definitions to be shared between systems.

The section following this deals with particular aspects of installing definitions for specific resource types.

What happens when CICS is initialized

When you initialize CICS, what happens to your resource definitions depends on the type of start. This is defined in the START system initialization operand; START=COLD for a cold start, and START=AUTO for a warm or emergency restart.

Installing resource definitions

Cold start

During cold start, CICS creates system tables by installing groups named in the list or lists named by the GRPLIST system initialization parameter. If you installed a group with the INSTALL command during the previous CICS execution, you must add its name to a list if you want it to be installed during a cold start.

If you usually use START=COLD at CICS initialization, installing by means of a list will probably be your standard way of making resource definitions available to CICS. Use of the INSTALL command is a supplementary method, which you will find very useful when testing a system, or if an unexpected need for a resource arises once CICS is running.

You may not want to use the RDO transactions in a production system, for security or performance reasons. In this case, the CSD is shared by both systems, but is read-only in the production system. You define all your production resources using your development system, and install them in the production CICS system when you cold start it.

Warm or emergency start

During a warm or emergency start, CICS recreates the tables from the resource definitions stored in the restart data set and global catalog.

No reference is made to the CSD file, nor is the GRPLIST name used. So all groups that had been installed by the end of the previous CICS execution are reinstalled automatically at a warm or emergency restart. Thus any CICS system modifications you have introduced using RDO will persist. For autoinstalled resources, see the following sections:

- “What happens at CICS restart” on page 113
- “Connection autoinstall and recovery and restart” on page 122
- “Program autoinstall and recovery and restart” on page 126.

If you have named a different list in the GRPLIST operand, or if you have added new groups to it since the last system initialization, CICS does not install the groups in the new list during a warm or emergency restart, because CICS does not refer to the list.

If you usually use START=AUTO at CICS initialization, using the INSTALL command will be your standard way of making resource definitions available to CICS. You will use a list to define your system only when you need to do a cold start. You can ensure that your list is up to date by adding to it each group installed using the INSTALL command.

What happens when you use the INSTALL command

Resource definitions may be installed in groups or as single definitions.

If the INSTALL command fails part of the way through installing the group, those resource types committed at the group level are backed out, whereas those resource types committed at the resource level are left in a committed state.

The following resource types are committed by individual resource, and not by complete group:

- AUTINSTMODEL
- PROFILE
- PROGRAM
- MAPSET
- PARTITIONSET
- PARTNER
- TRANCLASS
- TRANSACTION
- FILE
- LSRPOOL.

For these resources, the effect of a partially successful INSTALL is to leave the resources that were added in a committed state.

The following resources are committed at the group level:

- CONNECTION
- SESSIONS
- TERMINAL
- TYPETERM.

These resources are backed out after a partially successful INSTALL.

There are basically two reasons why an INSTALL may not be successful, both for INSTALLS of complete groups and for individually committed definitions:

1. One of the resource definitions could not be installed because it was currently in use (an executing transaction, for example).
2. System failure during installation.

If you wish to install only a few new or changed definitions, you are advised to install single resources, as described on page 58. (Note that the single-resource INSTALL of some CONNECTIONS and SESSIONS is not possible.) Use of the single resource INSTALL eliminates the problems of a partial INSTALL caused by a failure. However, if you wish to change or add a larger number of definitions, you might prefer to install a new group. In that case, the following considerations apply.

When you install a group containing an updated definition of an existing resource, the installation will fail if the resource is being used at the time. You should make sure that none of the resources in a group is in use before trying to install the group.

Installation is a two-stage process: any existing definition for the resource must be “deleted” from the system tables before a definition can be installed. This can result in more than one message, if the “deletion” fails, and causes the installation to fail.

If you have several CICS systems that share the same CSD, you must be careful not to install a group of resources in the wrong system.

Dual-purpose resource definition

With the extensions to dual-purpose resource definitions, you may want to consider the way in which resources are organized within your resource definition groups. For example, you might now have two groups; one containing all the resources for a CICS TOR, and one containing all the resources for a CICS AOR. When using shared resource definitions, you can decide to maintain three groups instead, with the first group containing resources specific to the TOR, the second group containing resources specific to the AOR, and the third group containing resources that will be installed in both the TOR and the AOR.

These resources should be defined as either local or remote. When the definition is installed on the TOR, CICS compares the SYSIDNT name with the REMOTESYSTEM name. If they are different, a remote transaction definition is created. When the definition is installed on the AOR, CICS compares the REMOTESYSTEM name with the SYSIDNT name. If they are the same, a local transaction definition is installed.

Duplicate resource definition names

An RDO-defined definition overrides a macro-defined definition of the same name. For example, if you try to install a definition for a VTAM terminal that has the same name as a non-VTAM terminal, the VTAM terminal entry overwrites the non-VTAM terminal entry.

If you INSTALL a group while CICS is active, the resource definitions in the group override any of the same type and name already installed.

When an existing resource definition is replaced in this way, the statistics associated with the old resource definition are transferred to the new definition. If a PROGRAM definition is

Using lists for initialization

replaced, the program is relocated on the library and loaded afresh when the new definition is referenced for the first time. In effect, the new definition implies a NEWCOPY operation. The same rules apply to map sets and partition sets.

+ An exception to these rules can occur with duplicate file definitions. If the file is defined as
+ ENABLED, then the later install of a duplicate will fail. However, if the file is defined as
+ DISABLED, then the later install of a duplicate will succeed.

It is probably unwise to have more than one resource definition of the same name on the CSD, even for different resource types. You must keep PROGRAM, MAPSET, and PARTITIONSET names unique. If you have, for example a PROGRAM and a MAPSET with the same name, only one of them is available to CICS. As far as names are concerned, after installation these definitions are treated as if they were the same resource type.

? If two groups in a list contain resource definitions of the same name, of the same resource
? type, CICS uses the definition in the group that is later in the list, for all resource types
? except TDQUEUES. For TDQUEUE definitions, the first definition is used.

The only reason why you might have more than one resource definition of the same name is if you have alternative definitions of the same real resource, with different attributes. These resource definitions must be in different groups.

Using lists for initialization

| How to set up lists for initialization

| The lists that you name in the GRPLIST system initialization parameter must include all the
| resource definitions required by CICS. These are supplied by CICS and are added to the
| CSD when you initialize it before starting to use RDO. (For further details about this, see the
| *CICS/ESA Operations and Utilities Guide*.)

| You can specify up to four lists, using specific or generic naming, on the GRPLIST system
| initialization parameter. The default list is the CICS-supplied list DFHLIST.

This is how you would go about creating a list containing both CICS-supplied and your own resource definitions.

1. Start to create the list that you will use to initialize CICS, by appending DFHLIST to a new list. For example:

```
CEDA APPEND LIST(DFHLIST) TO(INITLIST)
```

This ensures that all CICS-supplied definitions are installed, whether or not you need to change them.

2. Remove the groups containing definitions for function that you do not require. For example, for local DL/I:

```
CEDA REMOVE GROUP(DFHDLI) LIST(INITLIST)
```

3. Copy all the resource definitions that you need to change into your own groups. For example:

```
CEDA COPY TRANSACTION(CEDF) GROUP(DFHOPER) TO(SECTRANS)  
CEDA COPY PROFILE(DFHCICST) GROUP(DFHSTAND) TO(REQMOD)
```

Do **not** rename the copies. You can now use ALTER to change the attributes as necessary. For example:

```
CEDA ALTER TRANSACTION(CEDF) GROUP(SECTRANS)
```

4. Add these groups to your list for initialization. For example:

```
CEDA ADD GROUP(SECTRANS) LIST(INITLIST)  
CEDA ADD GROUP(REQMOD) LIST(INITLIST)
```

Make sure that you add this group **after** the DFH groups. Although you now have two definitions for the resources that you have altered, the second definition in the list is the one that will be installed, if you name this list as a GRPLIST parameter when you initialize CICS.

5. Add any other groups containing resource definitions of your own that you want to use, or append other lists. Your list might look like this:

```
DFHAKP
DFHBACK
DFHBMS
DFHCONS
:
DFHVTAMP
SECTRANS
REQMOD
ZEMAPPL
ZEMCOMM
ZEMTYPES
ZEMTERMS
```

Note that the group containing the TYPETERMS should come before the groups containing the TERMINAL definitions.

6. Cold start your CICS system, naming the list or lists that you have created in the GRPLIST system initialization parameter. For example:

```
START=COLD,GRPLIST=INITLIST
```

How you can use several lists

You can create lists that contain different sets of groups so that you can initialize different “flavors” of CICS using the GRPLIST initialization override.

Using different lists at different times

We recommend that you initialize your CICS system with the DFHSIT parameter `START=AUTO`, so that the CICS catalog is used to define the system whenever possible, instead of the list or lists named in the GRPLIST operand. However, if you use CICS differently each time you initialize it, you will not want to do this. You will code `START=COLD` in DFHSIT, and you can specify a different list to define your system every time you initialize CICS. For example, you might have:

- A different list for each day of the week, if the pattern of work is different on each day.
- A list for the CICS used for the day shift and a list for the CICS used for the night shift.
- A test only list used only when CICS is started up by the system programmers on a Sunday.
- For security reasons, a special list containing groups of restricted resource definitions. You could append this list to your usual one, when these resources are needed.

Consider how you might use the list and group mechanisms with transactions related to a company’s salary operations.

Let’s assume that some transactions used by the salary administrators will be used every day. For example, a transaction for handling an employee’s tax details may have to be performed at any time. Other transactions, such as minor weekly or monthly payroll adjustments, are run at predefined intervals, or on specific days or dates. You would therefore not want to include the same mixture of transactions and programs every time the system was started up.

By creating a resource definition group for taxation transactions, and another for payroll transactions, you could add them to different lists to produce the required system tables for different days. In the above example, one list would identify only the taxation group; the other would identify both taxation and payroll groups. You would specify the appropriate list in a SIT override.

Clearly, a real system would have many more groups and lists than this.

Using different lists for different CICS systems

If you are running more than one CICS system, in the same MVS image, you may use the same CSD file to define your resources to both systems. This helps you to ensure that each system has the same definition of resources where necessary. You probably don't want to use all the same resources in each system, so you could create a list for each system. You name the appropriate list in the DFHSIT macro for each system.

For example, you might have two production CICS systems sharing a CSD file. Let's assume that one production system runs three applications: customer inquiry, billing, and adjustments. Each application has its own resources (programs, mapsets, and transactions), so you put the resource definitions in three groups: CUSTINQ, CUSTBILL, and CUSTADJ. Then you add these groups to a list called CICS1A.

Another production system runs two more applications in addition to customer inquiry: customer update and customer credit authorization. For these you create two more groups (CUSTCRED and CUSTUPDT) and another list called CICS1B.

CICS1B contains the same CUSTINQ group as CICS1A, and it also contains CUSTCRED and CUSTUPDT. If you decide, for performance reasons, to move one of your applications to a different CICS system, all you need to do is add the appropriate group name to the appropriate list. The next time you initialize CICS with this list specified in the GRPLIST operand, you will install the new group.

Using different lists when you introduce changes

The list with which you initialize CICS is a definition of your system (for RDO resources). When you introduce changes to your resources, it is useful to create a new list, keeping the old list to return to if something goes wrong. Then you can re-initialize CICS with the old list, knowing that everything is as it was previously.

Checking groups and lists of resource definitions for consistency

The CHECK command checks the consistency of definitions within a group or within all of the groups within a list or lists. It does not, however, cross-check every attribute of a resource. You may still get error messages when installing a group although you found no problems when using the CHECK command.

If you use the CHECK GROUP command, CEDDA cross-checks all of the resources in a specified group to ensure that the group is ready to be used. For example, CHECK might warn you that a transaction definition within the group does not name a program within the same group. (Note, however, that this might not be an error. The group might intentionally be paired with a group that does contain the program, or you may want the program to be autoinstalled, in which case it would not have a definition.)

If you use the CHECK LIST command, CEDDA cross-checks every group named in the list. It does not simply check each group in turn, but merges the definitions in all of the listed groups, and checks them all. In this way it warns you if there are duplicate resource definitions, or references to definitions that do not exist.

Security and RDO

This section discusses security for RDO and the CSD.

Resource security checking

Resource security checking ensures that terminal operators are able to access only those resources for which they have been authorized. You can use resource security checking (RESSEC) for the TRANSACTION definition.

Multiple CSD files

You can have different CSD files for different CICS systems. The users of one CICS do not have access to the CSD for another CICS.

You could have a test CSD in a system where the RDO transactions can be used, and a production CSD in a system where the RDO transactions are not available. There would then be no chance of unauthorized users altering resource definitions needed for production work.

Read-only and update definitions for the same CSD file

Having two CSDs means duplicating resource definitions for resources that are shared by more than one system. An advantage of RDO is that you need only one definition for each resource. You can define one CSD data set to be shared among several CICS systems with only one having write access. To do this, you define one CSD data set differently to different systems, by using the CSDACC system initialization parameter. For the system where the CSD can be used, but not updated, you could code:

```
CSDACC=READONLY
```

and for the system where you are planning to update the CSD, you could code:

```
CSDACC=READWRITE
```

You need read-only access to install definitions. This also allows you to use the DISPLAY and VIEW commands. You need READWRITE access to use the DEFINE, ALTER, MOVE, RENAME, DELETE, or COPY commands. For details of defining the CSD, see the *CICS/ESA Operations and Utilities Guide*.

RDO also provides a means of controlling access to any group or list, so that users in the same system can have different types of access. This is done with the LOCK command.

Controlling access to a group or list — LOCK and UNLOCK

The LOCK and UNLOCK commands enable you to control update access to a group or list so that only operators with the same operator identifier can make changes.

The lock is held on the CSD file and remains in effect across restarts of CICS. The lock is owned by the user, who is identified by a combination of the CICS generic applid (specified by the APPLID system initialization parameter), and the user's operator identifier (OPIDENT).

The OPIDENT is the one associated with the user when he or she signs on to the terminal used for RDO. For further information on OPIDENT see the *CICS/ESA CICS-RACF Security Guide*. Any user who is not signed on or who has a different OPIDENT is not allowed to perform any operation that would change the locked group. However, any user is allowed to do the following things to a locked group:

- DISPLAY
- VIEW
- COPY
- CHECK
- INSTALL.

The lock can be removed, using the UNLOCK command, only by a user on the same system and with the same operator identifier.

It would be wise to put a lock on your group of TYPETERMs and on your group of AUTINSTMODEL TERMINALS.

Controlling access to the RDO transactions

Recommended access for the CEDA, CEDB, and CEDC transactions is as follows:

- CEDC can be given fairly wide access, as it allows only read-only commands.
- CEDB should be restricted, as it allows modification of the CSD as well as read-only commands.
- CEDA should be further restricted to the few people allowed to modify both the active CICS system and the CSD file.

Dependent default values

Dependent default values

Some attribute values have default values that differ, depending on the values for the other attributes you have already specified on the command line. This means you don't have to remember so many attribute values.

These default values are known as dependent defaults. They are similar to forced values, but you **can** change them if you want to.

RDO cannot remember that they are defaults, as it can with default names and forced values, so if you change the value of the attribute on which they depend, ***the dependent defaults stay as they were***. For example, if you copy a 3270P TYPETERM and then change its DEVICE attribute to 3270, you do not get the appropriate default values for the other attributes, and your TYPETERM definition is not applicable to a display device or to a printer, unless you change all the values yourself.

It is very important that you should **not** change the value of DEVICE, SESSIONTYPE, or TERMMODEL either using ALTER or by overtyping. If you make a mistake with these attributes, delete the definition and define a new one. For the same reason, you should **not** use COPY and ALTER as a method of creating many TYPETERMs with different DEVICE values.

Dependent and forced values mean that you don't have to remember which attributes are relevant to different devices. (See Table 13 on page 219 for a list of TYPETERM dependent defaults and forced values.)

Chapter 5. Using CEDA

This chapter is in the form of a tutorial to show you how to use the RDO transactions CEDA, CEDB, and CEDC. It assumes that you have read the information on resources, groups and lists in “How the CSD is organized — groups and lists” on page 15 and “Using lists for initialization” on page 22.

At the end of the tutorial, you should be familiar with the CEDA panels, and be able to manage your resources efficiently.

The examples in the tutorial all use CEDA, because if you have CEDA authorization, you can issue all RDO commands. If you have access to only CEDB or CEDC, you can issue the following commands:

CEDB all RDO commands except INSTALL.

CEDC DISPLAY, EXPAND, and VIEW commands only.

The cursor is indicated by the symbol '■'.

CEDA tutorial: defining new resources

Enter CEDA at a CICS terminal. The following panel appears:

```

  ■
  ENTER ONE OF THE FOLLOWING

ADd
ALter
APpend
CHeck
COpy
DEFine
DElete
DISplay
EXpand
Insta11
Lock
Move
REMove
REName
UN1ock
USerdefine
View

                                SYSID=CICA  APPLID=MYCICS

PF 1 HELP      3 END      6 CRSR      9 MSG      12 CNCL
```

Figure 2. CEDA transaction: initial panel

This is a list of all the CEDA commands. CEDB has all of these except INSTALL, and CEDC offers you only DISPLAY, EXPAND, and VIEW.

Abbreviating commands Each command can be abbreviated. The minimum abbreviations are shown in uppercase - for example, you can enter REM for REMOVE, but not just R or RE, because to CEDA that would be ambiguous with RENAME.

So that you have some resources of your own to work with, use the DEFINE command to create a new resource in a new group. At the top of the screen, enter DEFINE. You will see the following screen:

```

DEFINE
ENTER ONE OF THE FOLLOWING

Connection
File
Lsrpool
Mapset
PARTitionset
PARTner
PROFile
PROGram
Sessions
TERminal
TRANClass
TRANSACTION
TYpeterm

                                     SYSID=CICA APPLID=MYCICS

PF 1 HELP      3 END          6 CRSR          9 MSG          12 CNCL

```

Figure 3. CEDA DEFINE panel

This is a list of all the resource types you can define for CICS by means of CEDA or DFHCSDUP. They are explained in detail in Part 4, "RDO resource types and their attributes" on page 127.

SYSID This is your system identifier specified in the SYSIDNT parameter of the system initialization table (SIT).

APPLID This is the VTAM application identifier for the CICS system, as specified in the APPLID parameter of the SIT.

PF keys The PF keys on this screen are:

PF1 HELP Provides help in using CEDA.

PF3 END Takes you out of CEDA. After using PF3, you can clear the screen and continue with another transaction.

PF6 CRSR Puts the cursor in the top left corner of the screen.

PF9 MSG Shows you any error messages produced by CEDA.

PF12 CNCL Takes you back to the previous panel.

After the word DEFINE that you entered, type in MAPSET(NEW1) GROUP(AAA1).

```

DEFINE MAPSET(NEW1) GROUP(AAA1)
OVERTYPE TO MODIFY                                CICS RELEASE = 0410
CEDA DEFine Mapset( NEW1      )
Mapset      : NEW1
Group       : AAA1
Description ==>
REsident    ==> No                No | Yes
USAge       ==> Normal            Normal | Transient
USElpacopy  ==> No                No | Yes
Status      ==> Enabled           Enabled | Disabled
RSL         : 00                  0-24 | Public

I New group AAA1 created.
                                     SYSID=CICA APPLID=MYCICS
DEFINE SUCCESSFUL                    TIME: 14.03.13 DATE: 94.087
PF 1 HELP 2 COM 3 END                6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Figure 4. DEFINE MAPSET(NEW1) GROUP(AAA1)

You can see that your whole command (DEFINE MAPSET(NEW1) GROUP(AAA1)) remains at the top of the screen. You must specify a group name for every resource you want to define or work with; if you do not, CEDA displays a severe error message informing you that you have not supplied a group.

The main part of the screen shows the attributes of the MAPSET that you have just defined. All the attributes and values that you see are described in Chapter 15, "MAPSET" on page 151.

Attributes and values The attributes of the map set are listed in the first column, and their associated values are listed in the second column. For example:

Status ==> Enabled Enabled | Disabled

Status is the attribute.

Enabled is the value it has been defined with.

Enabled | Disabled shows the values it is possible to define it with.

Use of ==> When you see a '==>' symbol between an attribute and its value, it means that you can change the value. These values which can be changed are also highlighted.

Use of colon (:) When you see a colon (:) between an attribute and its value, it means that you cannot change the value. This can be for a number of reasons:

- You do not have authority to change the values (for example, if you are a CEDC user).
- You are using the VIEW command (explained later in this tutorial).
- The attribute is the resource name or the group name (in this example, MAPSET name NEW1 and GROUP name AAA1).
- The attribute is obsolete for the current release of CICS (for example, the RSL attribute).

To change the value of obsolete attributes, you must use the **compatibility mode** option, as described in the PF keys below.

PF keys The PF keys on this screen which were not on the previous screen are:

- PF2 COM** Allows you to use **compatibility mode** to change the value of obsolete attributes. On the above panel, if you press PF2, you will see that the symbol between RSL and its value changes from '==>' to ':', the value is highlighted, and the display at the top right of the screen changes from 'CICS RELEASE=0410' to 'COMPATIBILITY MODE'. You can now overwrite the value to change it, then press PF2 again to get out of compatibility mode. Compatibility mode is described in "Compatibility mode (CSD sharing)" on page 19.
- PF7 SBH** Scrolls back half a screen. Because MAPSET has a small number of attributes, this does not apply to this panel.
- PF8 SFH** Scrolls forward half a screen. Because MAPSET has a small number of attributes, this does not apply to this panel.
- PF9 MSG** Shows you any error messages produced by CEDA. If you press PF9 just now, you will see the message "NEW GROUP AAA1 CREATED". This is the same message as is displayed on the screen, but if you issue a command which generates more than one message, you may have to use PF9 to see them all.
- PF10 SB** Scrolls up one full screen. Because MAPSET has a small number of attributes, this does not apply to this panel.
- PF11 SF** Scrolls down one full screen. Because MAPSET has a small number of attributes, this does not apply to this panel.

Messages. CEDA has produced the informational message 'I New group AAA1 created'. There are four levels of error messages in CEDA, as follows:

- S for severe. CEDA cannot continue until you correct what is wrong.
- E for error. Commands resulting in these messages will be executed when you press enter, but the results may not be what you intended; for example, if you abbreviate a command to the point where it becomes ambiguous, CEDA warns you, and tells you which command it is going to assume when you press enter. It may not be the command you intended.
- W for warning.
- I for information.

You can use PF9 to view CEDA messages. If you try it at this point it will be the same as what is on the screen, as there is only one message. When you have read the messages, use the Enter key to get back to where you were.

Press PF3 to get out of CEDA. The following is displayed:

```
CEDA DEFINE MAPSET(NEW7) GROUP(AAA1)
STATUS: SESSION ENDED
```

Figure 5. SESSION ENDED panel

Clear the screen, and you will be able to enter another transaction.

So far, you have gone through a number of panels to define a resource, but you have the alternative of entering a complete CEDA command on the command line. For example, if you enter:

```
CEDA DEFINE TRANSACTION(XYZ1) PROGRAM(XYZ2) GROUP(AAA2)
```

The screen you see is very similar to Figure 4 on page 30:

```

OVERTYPE TO MODIFY                                CICS RELEASE = 0410
CEDA DEFine TRAnsAction( EEEE )
  TRAnsAction   : EEEE
  Group         : AAA1
  Description   ==>
  PROGram      ==> AAAA
  TWAsize       ==> 00000          0-32767
  PROFile       ==> DFHCICST
  PARTitionset ==>
  STatus        ==> Enabled       Enabled | Disabled
  PRIMedsize    : 00000          0-65520
  TASKDATAloc  ==> Below         Below | Any
  TASKDATAKey  ==> User          User | Cics
  STOrageclear ==> No            No | Yes
  RUNaway      ==> System        System | 0-2700000
  SHutdown     ==> Disabled     Disabled | Enabled
  ISolate      ==> Yes           Yes | No
  REMOTE ATTRIBUTES
+ DYnamic      ==> No            No | Yes
I New group AAA2 created
DEFINE SUCCESSFUL                                SYSID=CICA APPLID=MYCICS
TIME: 16.47.45 DATE: 94.088
PF 1 HELP 2 COM 3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Figure 6. CEDA DEFINE TRANSACTION(XYZ1) PROGRAM(XYZ2) GROUP(AAA2) panel

You must specify a PROGRAM whenever you define a new TRANSACTION; if you do not, CEDA displays a severe error message saying that you must specify either a PROGRAM or REMOTESYSTEM. The attributes of the TRANSACTION definition are described in Chapter 23, "TRANSACTION" on page 207.

You can see that the TRANSACTION definition has many more attributes than the MAPSET definition. The **plus sign (+)** to the left of the last attribute in the list means that there are more attributes, so you can use either PF8 or PF11 to scroll down through them, then PF7 or PF10 to scroll back up.

You now have two new resources to work with, a MAPSET in group AAA1, and a TRANSACTION in group AAA2. These resources will be used throughout the remainder of the tutorial to demonstrate the other CEDA commands.

Press PF3 to get out of CEDA. Instead of clearing the screen as you did before, you have the alternative of typing a new command over the existing one at the top of the screen. Type 'DISPLAY' over 'DEFINE' and delete the rest of the line. You will see a screen like this:

```

DISPLAY
OVERTYPE TO MODIFY
CEDA Display
Group      ==> ■
List       ==>
All        ==> *
Connection ==>
File       ==>
LSrpool   ==>
Mapset    ==>
PARTitionset ==>
PARTNer   ==>
PROFile   ==>
PROGram   ==>
Session   ==>
TERminal  ==>
TRANClass ==>
TRANsAction ==>
TYpeterm  ==>

S No GROUP value has been previously specified so there is no current value
to assume.
                                SYSID=CICA  APPLID=MYCICS

PF 1 HELP      3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Figure 7. CEDA DISPLAY panel

You may not know any group names or list names yet, so type in an asterisk where the cursor is, beside GROUP, then press enter. The asterisk means that you want to display all groups. If your system has a large number of groups in it, this command may take some time to execute. You get a screen like this:

```

ENTER COMMANDS
NAME      TYPE      GROUP      DATE      TIME
NEW1     MAPSET     AAA1      93.328 11.00.03
NEW2     TRANSACTION AAA2      93.176 09.23.56
IGZCPAC  PROGRAM    COBOL2    93.033 08.18.34
IGZCPCC  PROGRAM    COBOL2    93.033 08.18.34
ACCTFIL  FILE       DFH$ACCT  93.349 15.56.46
ACCTIX   FILE       DFH$ACCT  93.349 15.56.46
ACCTSET  MAPSET     DFH$ACCT  93.349 15.56.46
ACCT00   PROGRAM    DFH$ACCT  93.349 15.56.46
ACCT01   PROGRAM    DFH$ACCT  93.349 15.56.46
ACCT02   PROGRAM    DFH$ACCT  93.349 15.56.46
ACCT03   PROGRAM    DFH$ACCT  93.349 15.56.46
ACCT04   PROGRAM    DFH$ACCT  93.349 15.56.46
ACCT     TRANSACTION DFH$ACCT  93.349 15.56.46
ACEL     TRANSACTION DFH$ACCT  93.349 15.56.47
ACLG     TRANSACTION DFH$ACCT  93.349 15.56.47
AC01     TRANSACTION DFH$ACCT  93.349 15.56.46
+AC02    TRANSACTION DFH$ACCT  93.349 15.56.46

                                SYSID=CICA APPLID=MYCICS
RESULTS: 1 TO 17                TIME: 14.29.10 DATE: 94.071
PF 1 HELP      3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
    
```

Figure 8. CEDA DISPLAY GROUP(*) panel

This shows the groups in your system and the resource definitions that they contain.

Plus sign. The plus sign (+) beside the last value means that there are more resources, so you can use PF8 or PF10 to scroll down to see them, then PF7 or PF11 to scroll back up again. PF7 and PF8 enable you to scroll down and up half a screen at a time. PF10 and PF11 enable you to scroll down and up a full screen at a time.

GROUP. This tells you which group each resource definition belongs to. This whole display of resource definitions is ordered by listing the groups alphabetically.

NAME. The list under the NAME heading tells you the name of each individual resource definitions installed on your system. You can have duplicate resource names only if the resource definitions are in different groups. The resource names are listed alphabetically according to their resource types (that is, within any one group, all the transactions are listed alphabetically, then all the programs, and so on).

TYPE. This tells you what resource type each definition is for. The resource types were introduced briefly on page 7, and are described fully in Part 4, "RDO resource types and their attributes" on page 127. Within each group, the similar resource types are shown together (that is, all the transactions, then all the programs, and so on).

DATE and TIME. This shows the date and time when the resource definition was last updated. The time and date immediately above the PF key descriptions at the bottom of the screen are the current time and date.

RESULTS: 1 TO 17. Below the list of groups, there is a line that says "RESULTS: 1 TO 17". This tells you how many group names are displayed on the screen. If there are more than 17, this message changes as you scroll up and down the list of group names. At the beginning of a CEDA DISPLAY GROUP(*) command, it says "RESULTS: 1 TO 17", but as you scroll down, CEDA builds up a count of the total number of groups, and eventually the message changes to be of the form "RESULTS: 52 TO 68 OF 97". If the list is very long, you can use PF5 to go straight to the bottom of it, and PF4 to get back to the top of it.

ENTER COMMANDS. You can enter commands in the area of the screen where the cursor is. If you are using CEDA, valid commands from this screen are:

ALTER
COPY
DELETE
INSTALL
MOVE
RENAME
VIEW
?
=

If you are using CEDB, all of these are valid except INSTALL, and if you are using CEDC, only VIEW, ?, and = are valid.

CEDA tutorial: displaying resources

Move the cursor down to the TRANSACTION XYZ1 that you defined earlier and type in 'VIEW' beside it. A screen like this is displayed:

```
OBJECT CHARACTERISTICS                                CICS RELEASE = 0410
CEDA View TRAnsaction( XYZ1 )
TRANsaction      : XYZ1
Group           : AAA2
DEscription     :
PROGram        : XYZ2
TWAsize         : 00000          0-32767
PROFile        : DFHCICST
PARTitionset    :
STatus         : Enabled        Enabled | Disabled
PRIMedsize     : 00000          0-65520
TASKDATAloc    : Below         Below | Any
TASKDATAkey    : User          User | Cics
STOrageclear   : No            No | Yes
RUnaway        : System        System | 0-2700000
SHUTDOWN       : Disabled      Disabled | Enabled
ISolate        : Yes           Yes | No
REMOTE ATTRIBUTES
+ Dynamic      : No            No | Yes

                                SYSID=CICA APPLID=MYCICS

PF 1 HELP 2 COM 3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Figure 9. CEDA VIEW TRANSACTION(XYZ1) GROUP(AAA2) panel

This is almost identical to the panel you saw when you defined this transaction (shown in Figure 6 on page 32), with the difference that on the VIEW panel there is a colon (:) between each attribute and its value. You cannot change any values from a VIEW panel.

If you notice at this point that you have defined TRANSACTION(XYZ1) incorrectly, you may want to alter one or more of its values. Press PF12 to return to the DISPLAY screen. There is now an asterisk (*) beside the transaction to indicate that you have worked with it. Type 'ALTER' over the asterisk, and you will see a panel which begins like this:

```

OVERTYPE TO MODIFY                                CICS RELEASE = 0410
CEDA ALTER TRANsAction( XYZ1 )
TRANsAction      : XYZ1
Group            : AAA2
DEscription      ==>
PROGram          ==> XYZ2
TWAsize          ==> 00000          0-32767
PROFile         ==> DFHCICST
:
    
```

Figure 10. CEDA ALTER TRANSACTION(XYZ1) GROUP(AAA2) panel

The highlighted text at the top left of the screen, 'OVERTYPE TO MODIFY', means that you can overtype any of the highlighted values. Overtyping the TWASIZE value, changing it from 00000 to 32767. When you press Enter, CEDA displays an 'ALTER SUCCESSFUL' message at the bottom of the screen.

Press PF12 to return to the DISPLAY screen, where the 'ALTER SUCCESSFUL' message is displayed on the same line as the TRANSACTION. Type in VIEW against the TRANSACTION, and you can see that the value of TWASIZE has been changed.

CEDA tutorial: copying a resource

At this point in the tutorial, you still have only two resources that you defined by yourself. You can create more using the DEFINE command as before, or you can copy existing resources, both within their own groups, and to other groups.

Press PF12 to return to the DISPLAY screen. Beside TRANSACTION(XYZ1) type in:

```
COPY TO(AAA1) AS(XYZ2)
```

A 'COPY SUCCESSFUL' message is displayed in the right hand column of the screen.

The COPY command has copied the definition for TRANSACTION(XYZ1) from group AAA2 to group AAA1, renaming it as TRANSACTION(XYZ2). The original transaction, TRANSACTION(XYZ1), still exists in group AAA2. Press PF3 to display the **SESSION ENDED** panel, and overtype the existing 'CEDA DISPLAY' with:

```
CEDA DISPLAY GROUP(AAA1)
```

Group AAA1 now contains two resource definitions, a MAPSET called NEW1 and a TRANSACTION called XYZ2. Because this is a DISPLAY screen similar in format to Figure 8 on page 34, you can enter the same commands against the definitions.

DISPLAY GROUP(AAA1)				
ENTER COMMANDS				
NAME	TYPE	GROUP	DATE	TIME
NEW1	MAPSET	AAA1	94.087	16.22.12
NEW5	MAPSET	AAA1	94.088	15.34.00
:				

Figure 11. CEDA DISPLAY GROUP(AAA1) panel

The next part of the tutorial is about entering commands from the command line. Press PF3 to get out of CEDA and clear the screen.

Entering commands from the command line

So far, this tutorial has taken you through panels to execute CEDA commands, but when you become familiar with your system's resources and with CEDA, you can enter most CEDA commands on the command line.

Here is a sequence of commands; if you follow them, you can see that entering commands on the command line is quicker than going through all the CEDA panels.

You do not need to use PF3 or PF12 at this point. Use PF6 to move the cursor to the top left corner of the screen. Insert a transaction name and remove the group; the command now looks like this:

```
CEDA DEFINE TRANSACTION(BBB1) PROGRAM(CCC1)
```

When you press Enter, the new transaction BBB1 is defined. Note that it has been defined without an associated group name; this is because, as long as you are within the same CEDA session, the current group (that is, the one you have most recently been working with) is used by default. If you had used PF3 before this, your CEDA session would have ended, so there would be no current group name for CEDA to assume.

Rename the transaction by overtyping the command to read:

```
RENAME TRANSACTION(BBB1) AS(DDD1)
```

Move the transaction to group AAA2 by overtyping the command to read:

```
MOVE TRANSACTION(DDD1) TO(AAA2)
```

Before installing groups AAA1 and AAA2, you can check them by using the CHECK command. This ensures that all transactions have access to their related programs, that terminal definitions have access to their related typeterm definitions, and so on. Overtyping the command to read:

```
CHECK GROUP(AAA1)
```

This should result in the message:

```
W PROGRAM CCC1 referenced by TRANSACTION DDD1 in group AAA2
cannot be found
```

This is because when you moved transaction DDD1, you did not also move its related program, CCC1.

? **Note:** You will not see this message if you have autoinstall programs active. If autoinstall
 ? programs is active, CEDA assumes that the program will be installed at execution
 ? time and does not check the program definitions.

You can remedy this by using the COPY command:

```
COPY PROGRAM(CCC1) G(AAA1) TO G(AAA2)
```

If you repeat the CHECK command on both groups, there should be no more error messages.

This is clearly faster and more efficient than using panels. You can now use PF3 to get out of CEDA, then clear the screen.

Removing resource definitions from the CSD

The resource definitions that you have created for the tutorial can be removed; this allows other people to follow the tutorial, and ensures that your CSD space is not being used unnecessarily. Using the DELETE command, you can delete the whole of groups AAA1 and AAA2 from your CSD:

```
CEDA DELETE ALL GROUP(AAA1)
CEDA DELETE ALL GROUP(AAA2)
```

Note that you cannot delete the group itself; it ceases to exist only when it contains no resource definitions. See "DELETE a resource definition, group, or list" on page 91 for more information on the DELETE command.

Generic naming

You used this feature of CEDA on page 34 when you used an asterisk to mean 'all' groups. The asterisk can be used to mean 'all' anything. For example, CEDA VIEW TRANSACTION(DISC) GROUP(*) shows you transactions called DISC in any group where they occur. CEDA DISPLAY GROUP(*) TRANSACTION(*) shows you all the transactions in every group.

Another way to see groups without having to specify their exact names is to use the plus sign (+) as part of the group name to represent one character. For example, whereas CEDA DISPLAY GROUP(DFH*) will display all groups beginning with DFH, CEDA DISPLAY GROUP(DFH++) will display only those groups which begin with DFH and are six characters long, CEDA DISPLAY GROUP(DFH++++) will display those which begin with DFH and are seven characters long, and so on.

Table 4 on page 41 tells you which commands you can use generic naming with.

Which commands accept generic names?

Generic names allow you to use one command to perform the same operation on many objects.

You can use the keyword ALL in some commands, instead of specifying a resource type. The command will operate on all resource definitions that fit the name specified with ALL.

yes means that you may use a generic name, or an actual name.

no means that you must use an actual name.

- means that this keyword is not applicable for this command.

Table 4. Commands accepting generic names

COMMAND	Generic Resource Name	Generic Group Name	Generic List Name	ALL Resource Types
ADD	-	no	no	-
ALTER	yes	yes	-	no
APPEND	-	-	no	-
CHECK GROUP	-	no	-	-
CHECK LIST	-	-	no	-
COPY	yes	no	-	yes
DEFINE	no	no	-	no
DELETE	yes	no	-	yes
DISPLAY GROUP	-	yes	-	-
DISPLAY GROUP ALL	yes	yes	-	yes
DISPLAY LIST	-	-	yes	-
DISPLAY LIST GROUP	-	yes	yes	-
EXPAND GROUP	yes	yes	-	yes
EXPAND LIST	-	yes	yes	-
INSTALL	-	no	-	-
INSTALL GROUP	-	no	-	-
LOCK GROUP	-	no	-	-
LOCK LIST	-	-	no	-
MOVE	yes	no	-	yes
REMOVE	-	yes	no	-
RENAME	no	no	-	yes
UNLOCK GROUP	-	no	-	-
UNLOCK LIST	-	-	no	-
USERDEFINE	no	no	-	no
VIEW	yes	yes	-	no

Chapter 6. CEDA commands - syntax and examples

This chapter explains the syntax and rules for each of the CEDA commands. The commands are:

- “ADD a group to a list” on page 44
- “ALTER a resource definition” on page 45
- “APPEND a list to another list” on page 47
- “CHECK related resources” on page 48
- “COPY a resource definition” on page 50
- “DEFINE a resource definition” on page 52
- “DELETE a resource definition” on page 53
- “DISPLAY groups and lists” on page 54
- “EXPAND one or more groups” on page 56
- “INSTALL a resource definition or group on an active CICS system” on page 58
- “LOCK a resource definition” on page 61
- “MOVE a resource definition” on page 63
- “REMOVE a group from a list” on page 65
- “RENAME a resource definition” on page 66
- “UNLOCK a group or list” on page 67
- “USERDEFINE” on page 68
- “VIEW resource definitions” on page 71

ADD a group to a list

ADD syntax

```

▶—Add—Group(groupname1)—List(listname)—
┌──────────────────────────────────┐
│Before(groupname2)│
│After(groupname3)│
└──────────────────────────────────┘
◀

```

The ADD command adds a group to a list.

Group(*groupname1*)

The name of the group to be added. The name must not already exist in the list. A generic group name is not accepted. If you don't specify a group, the current group name is added.

List(*listname*)

The name of the list to which the group is to be added. If the list does not already exist, a new one is created. If LIST is not specified, the group name is added to the current list if there is one. A generic list name is not accepted.

The list name can be up to 8 characters in length. The characters allowed are A-Z, 0-9, @, #, and \$. Lowercase characters are treated as uppercase. Do not use Grouplist names beginning with DFH, because these characters are reserved for use by CICS.

Before(*groupname2*)

You can use this to control the placing of the new group name. If you don't specify BEFORE or AFTER, the group name is added at the end of the list.

After(*groupname3*)

You can use this to control the placing of the new group name. If you don't specify BEFORE or AFTER, the group name is added at the end of the list.

You can use the ADD command from a DISPLAY screen.

Examples of the ADD command

Create a list LA01, by adding a group to it:

```
ADD GROUP(GA001) LIST(LA01)
```

Add another group to list LA01, without specifying where:

```
ADD GROUP(GA002) LIST(LA01)
```

LA01 now looks like this:

```
GA001
GA002
```

Add another group at the top of the list:

```
ADD GROUP(GA003) LIST(LA01) BEFORE(GA001)
```

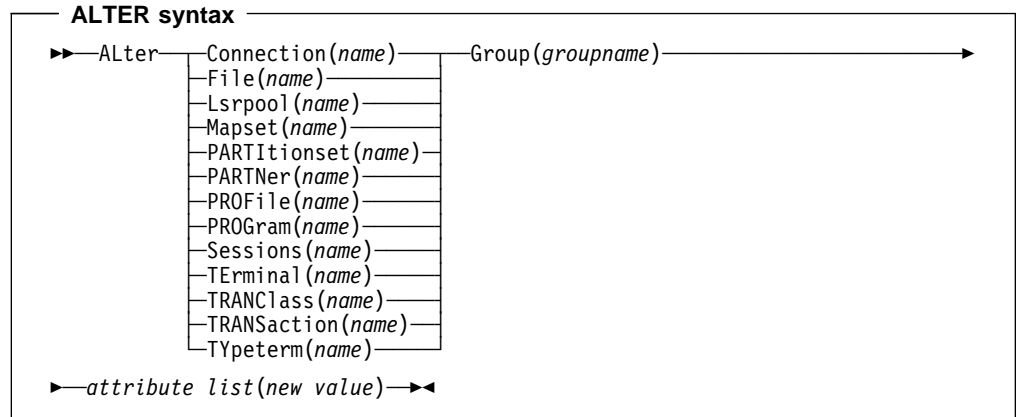
and another group between GA001 and GA002:

```
ADD GROUP(GA004) LIST(LA01) AFTER(GA001)
```

LA01 now looks like this:

```
GA003
GA001
GA004
GA002
```


ALTER a resource definition



The ALTER command changes some or all of the attributes of an **existing** resource definition.

For information about the attributes that you can specify on the ALTER command for the various resource types, see Part 4, “RDO resource types and their attributes” on page 127.

Resource(name)

The resource whose attributes you want to alter.

Group(groupname)

The name of the group containing the resource to be altered.

Attribute list

The attributes to be altered. For information about the attributes that you can specify on the ALTER command for each resource type, see Part 4, “RDO resource types and their attributes” on page 127.

- Do **not** use ALTER to change the value of the attributes of a TYPETERM definition on which other attributes depend. If you make a mistake with DEVICE, SESSIONTYPE, or TERMMODEL, you should delete the definition, and define a new one with the correct values.
- You can specify null operand values, for example:

```
ALTER FILE(TEST) GROUP(ACT1) DESCRIPTION()
```

If an attribute for which you have specified a null value has a default, the value used depends upon the type of field.

- The command:

```
ALTER FILE(TEST) GROUP(ACT1) RLSACCESS()
```

behaves as if RLSACCESS was not specified. The RLSACCESS parameter is ignored.

- The command:

```
ALTER FILE(TEST) GROUP(ACT1) DESCRIPTION()
```

has the effect of blanking out the description, as there is no default value, and it is optional.

- The command:

```
ALTER FILE(TEST) GROUP(ACT1) PROFILE()
```

puts the default value of DFHCICSA into the PROFILE field.

- Changes to resource definitions in the CSD do not affect the running CICS system until you install the definition, or the group in which the resource definition resides.

ALTER

- You can use CEDA ALTER from a DISPLAY panel. If you use PF12 after making your alterations, CEDA gives you the DISPLAY panel again, with an 'ALTER SUCCESSFUL' message in the Date and Time fields. If you do this but do not make any alterations, an asterisk replaces your 'ALTER' command.
- With a generic name, you can use one ALTER command to change the same attributes in the same way on more than one resource definition.

Command-line examples of the CEDA ALTER command

To make a program resident:

```
ALTER PROGRAM(ERR01) GROUP(GENMODS) RESIDENT(YES) DATALOCATION()
```

To change the status of a whole group of programs:

```
ALTER PROGRAM(*) GROUP(GENMODS) STATUS(ENABLED)
```

If you do not specify an attribute list and type in:

```
ALTER PROGRAM(ERR01) GROUP(GENMODS)
```

CEDA gives an "ALTER SUCCESSFUL" message followed by the 'overtime to modify' panel.

APPEND a list to another list

APPEND syntax

```
►►—APpend—LIst(listname1)—To(listname2)—►◄
```

The APPEND command adds the groups in one list to the end of another list.

List(*listname1*)

The source list to be appended. A generic list name is not accepted.

To(*listname2*)

The target list to be appended to. A generic list name is not accepted. If *listname2* already exists, the source list is appended to it. If *listname2* does not exist, it is created.

Example of the APPEND command

A list called LISTA contains the following groups:

```
GB001
GB002
GB003
```

A list called LISTB contains the following groups:

```
G001
G002
G003
```

Append LISTB to LISTA, like this:

```
APPEND LIST(LISTB) TO(LISTA)
```

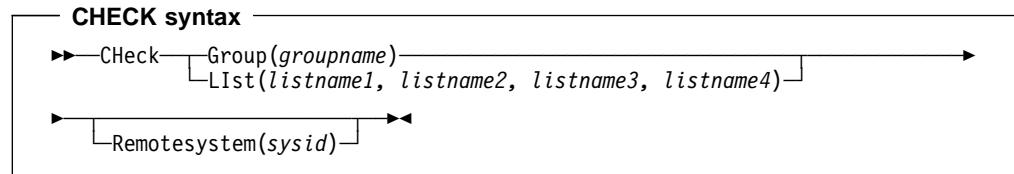
After this, LISTA will contain the following groups, in this order:

```
GB001
GB002
GB003
G001
G002
G003
```

and LISTB will still contain:

```
G001
G002
G003
```

CHECK related resources



The CHECK command checks the consistency of definitions.

Group

The group you want to check. A generic group name is not accepted.

List

The list you want to check. A generic list name is not accepted.

Remotesystem(sysid)

A check is to be run on a group or list in a CICS region with a different sysid from the region that the CHECK command is being issued from. If this option is not used, the CHECK command will use the sysid of the region from which the CHECK command is issued.

Note that:

- The CHECK command performs a cross-check of a group, list, or lists of resource definitions, and should be used before the resource definitions are installed.

It checks that the resource definitions within the group or lists are consistent with one another.

For example: for each TRANSACTION in the list being checked, a check is made that the named PROGRAM definition exists in one of the groups. The success of the check does not necessarily mean that the PROGRAM is available to the running system.

- Lists should be checked before being used to initialize CICS during a cold start (when the list or lists are specified in the GRPLIST operand of the DFHSIT macro).
- A group should be checked before you use the INSTALL command to install it on the running CICS system.
- A group can be checked before you use the ADD command to add the group to a list. (The group might not be self-contained, in which case there is no point in checking it alone. You should put it into a list with the groups containing related resource definitions.)
- You can use the CHECK command from a DISPLAY panel.

Checking groups and lists of resource definitions for consistency

The CHECK command checks the consistency of definitions within a group or within all of the groups within a list. It does not, however, cross-check every attribute of a resource. You may still get error messages when installing a group although you found no problems when using the CHECK command.

If you use the CHECK GROUP command, CEDDA cross-checks all of the resources in a specified group to ensure that the group is ready to be used. For example, CHECK might warn you that a transaction definition within the group does not name a program within the same group. (Note, however, that this might not be an error. The group might intentionally be paired with a group that does contain the program, or you may want the program to be autoinstalled, in which case it would not have a definition.)

If you use the CHECK LIST command, CEDDA cross-checks every group named in the list. It does not simply check each group in turn, but merges the definitions in all of the listed groups, and checks them all. In this way it warns you if there are duplicate resource definitions, or references to definitions that do not exist.

Checking terminal definitions

Although you can use the CHECK command to check a group of TERMINAL definitions (it resolves references from display devices to printers, for instance), it is not very useful in resolving TYPETERM references if these are in a separate group because it would produce many unwanted messages for missing TYPETERMS.

When you add a new cluster of terminal devices, create a new group for them, and then create a list containing your TYPETERM definitions group and the new group of TERMINAL definitions. You can then use the CHECK command to check the whole list, without it being too time-consuming. This list is only needed for the duration of the checking, and is never named as the GRPLIST.

To avoid duplicating TERMINAL names, you could maintain a list of all groups containing TERMINAL definitions. You can use CHECK list to ensure that all new TERMINAL names are unique. If this is too lengthy a process, you can avoid it if TERMINAL names beginning with similar characters are kept in separate groups, for example:

```
TERMINAL(AZ01) GROUP(AZTERMS)
TERMINAL(AZ02) GROUP(AZTERMS)
TERMINAL(AZ03) GROUP(AZTERMS)
```

.

.

.

```
TERMINAL(AZnn) GROUP(AZTERMS)
```

```
TERMINAL(BJ01) GROUP(BJTERMS)
```

```
TERMINAL(BJ02) GROUP(BJTERMS)
```

```
TERMINAL(BJ03) GROUP(BJTERMS)
```

.

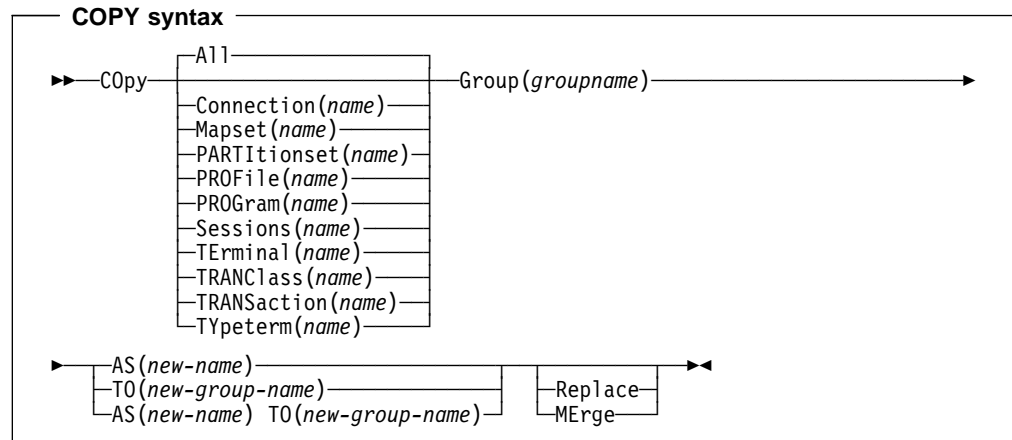
.

.

```
TERMINAL(BJnn) GROUP(BJTERMS)
```

If you are using autoinstall for terminals, you must install the TYPETERM definitions **before** installing the autoinstall model definitions, to ensure that the model is created. The CHECK command does not check the order of such definitions.

COPY a resource definition



The COPY command copies a resource definition, either within the same group or to a different group.

Resource(name)

The resource you want to copy. The default is ALL, which copies all the resource definitions in a group to another group.

Group(groupname)

The group containing the definitions to be copied.

AS If you copy a definition within a group, you must use AS to rename it. You can also use AS if you want to copy a definition to another group and rename it at the same time. You cannot use a generic name when using AS.

TO You can copy definitions to a different group, using TO to specify the new group.

Replace

This applies when there are duplicate definition names in the groups named in the COPY command. If you specify REPLACE, the definitions being copied replace those in the group named in the TO operand.

MERge

This applies when there are duplicate definition names in the groups named in the COPY command. If you specify MERGE, duplicate definitions in the TO group are not replaced.

If you don't specify either **MERGE** or **REPLACE**, a message will warn you that you are attempting to create duplicate resources, and your COPY will be unsuccessful.

Examples of the COPY command

- You can copy a single resource definition into a new group, using the **TO** keyword to specify the new group. For example:

```
COPY SESSIONS(L122) GROUP(CICSC1) TO(CICSC2)
```

- You can copy a resource definition within the same group. If you do this, you must rename it using the **AS** option. For example:

```
COPY TERMINAL(TD12) AS(TD34) GROUP(TERMDU1)
```

- When copying between groups, you can give the copy a new name, using the **AS** keyword to specify the new name.

```
COPY PROGRAM(ABC01) GROUP(XYZ) AS(ABC02) TO(NEWXYZ)
```

(If you leave the copy with the same name as the original definition, be careful that you install the one you want when the time comes.)

- Using the **ALL** keyword, without a name, you can copy all the resource definitions in the group to the new group. For example:

```
COPY ALL GROUP(N21TEST) TO(N21PROD)
```

- You can copy more than one resource definition to a new group, using the **TO** keyword to specify the new group.
- Using a generic resource definition name, you can copy all or some definitions of the same resource type to the new group. For example:

```
COPY CONNECTION(*) GROUP(CICSG1) TO(CICSG2)
COPY PROGRAM(N21++) GROUP(NTEST) TO(NPROD)
```

- Using the **ALL** keyword with a generic name, you can copy all or some of the resource definitions in the group to the new group. For example:

```
COPY ALL(N21*) GROUP(N21OLD) TO(N21NEW)
```

- Using the **ALL** keyword with a specific name, you can copy all the resource definitions of that name (which must necessarily be for different resource types) in the group to the new group. For example:

```
COPY ALL(XMPL) GROUP(EXAMPLE) TO(EX2)
```

- If you are copying a number of definitions into another group, and the groups contain duplicate resource names, you must specify either **MERGE** or **REPLACE**. For example:

```
COPY ALL GROUP(TAX1) TO(TAX2) MERGE
COPY ALL GROUP(TAX1NEW) TO(TAX1) REPLACE
```

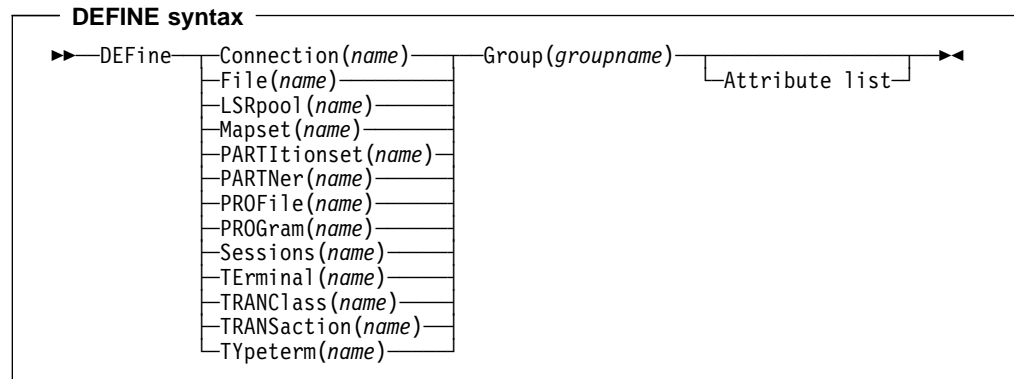
The following example copies a group named GA001 to a group named GA002, which already exists, replacing any duplicate resource definitions by those in group GA001.

```
COPY GROUP(GA001) TO(GA002) REPLACE
```

The following example copies group GA003 to group GA004, but if any duplicate definitions occur, preserves the group GA004 definitions.

```
COPY GROUP(GA003) TO(GA004) MERGE
```

DEFINE a resource definition



The DEFINE command creates a new resource definition.

Resource(name)

The name of the resource you want to define. Do not use a generic resource name.

Group(groupname)

The name of the group containing the resource definition being defined. Do not use a generic group name. If you specify the name of a group which does not already exist, the group is created.

Attribute list

The attribute list depends on the resource type being defined; some resources have attributes that must be included in the definition. See Part 4, "RDO resource types and their attributes" on page 127 for a description of the attributes and default values of each resource type. Attributes that you do not specify are given default values.

You can use the same name for more than one resource definition in a group, if the definitions are for different resource types. For example:

```

DEFINE PROGRAM(N28A) GROUP(N28APPL)
DEFINE TRANSACTION(N28A) GROUP(N28APPL)

DEFINE TERMINAL(USER) GROUP(USERDEF)
DEFINE PROGRAM(USER) GROUP(USERDEF)
  
```

Be careful, when you have any resource definitions with the same name, that you install the one you really want. See "Duplicate resource definition names" on page 21.

You must specify the resource type and name on the command line. You may also specify the group and other attributes on the command line.

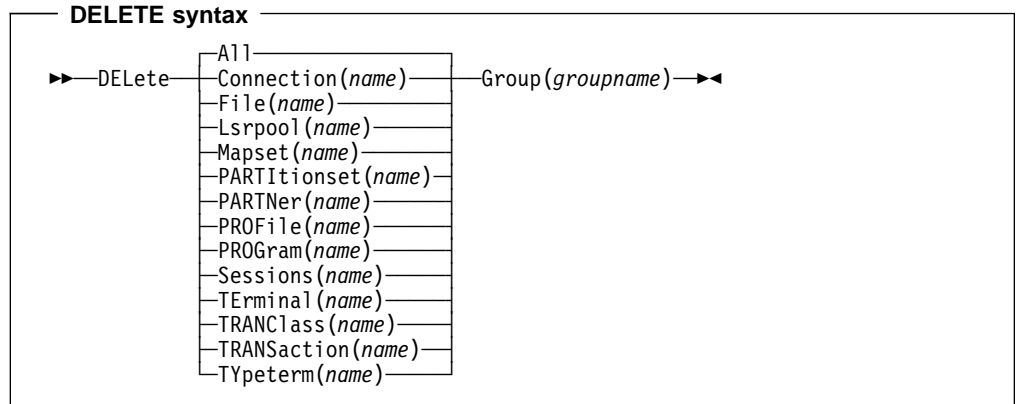
The whole definition will be displayed on an overtyping-to-modify panel, and you can change any of the attributes there, unless you entered a complete DEFINE command on the command line, in which case you can not change the *name* or *groupname* attributes.

The definition was created when you entered the DEFINE... command. If you do not wish to further modify it, you may enter another command from the command line.

If a resource definition of the same name and type already exists in the group, any attributes specified on the command line will be ignored, and the **existing** resource definition will be displayed. You can then overtype and modify any of the existing values if you wish. If you do not wish to modify the definition, you may enter another command from the command line.

Beware of overtyping values on which other attribute values are dependent. See "Dependent default values" on page 26.

DELETE a resource definition



Deletes one or more resource definitions from the CSD.

Resource(name)

The resource you want to delete. You can use a generic resource name. To delete all the resource definitions in the group, specify ALL(*).

Group(groupname)

The group containing the resource. Do not use a generic group name.

Deleting a resource definition is different from removing a group from a list (see "REMOVE a group from a list" on page 65). A deleted resource definition really does disappear from the CSD.

Note:

When you DELETE the last resource in a group, the group is automatically deleted. An empty group cannot exist.

You should ensure that if a group is deleted any LIST which refers to that group is also updated.

This command does **not** affect definitions installed on the active CICS system. To remove installed definitions from the active CICS system, you can use either the CEMT DISCARD transaction or the EXEC CICS DISCARD command in an application program. You can discard AUTINSTMODEL, FILE, PARTNER, PROFILE, PROGRAM, TRANCLASS, and TRANSACTION resources. For information on the CEMT transaction see the *CICS/ESA CICS-Supplied Transactions* manual. For programming information on the EXEC CICS DISCARD command see the *CICS/ESA System Programming Reference* manual.

Examples of the DELETE command

Delete all resources in a group:

```
DELETE ALL(*) GROUP(TOPS)
```

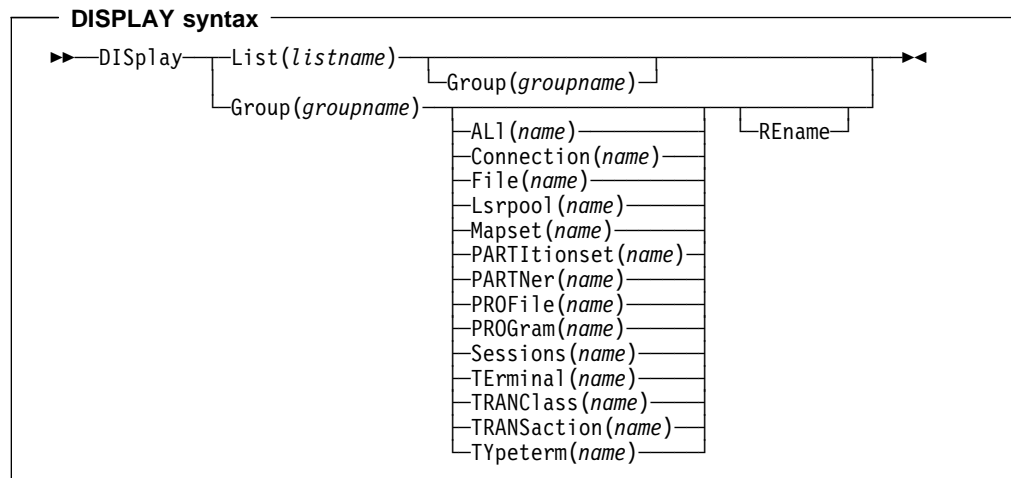
Delete all programs in a group:

```
DELETE PROGRAM(*) GROUP(NSOS)
```

Delete all resources with names beginning D0 in a group:

```
DELETE ALL(D0*) GROUP(REINDEER)
```

DISPLAY groups and lists



The DISPLAY command displays one or more group names, list names, or resources on a full screen panel.

List(listname)

The name of the list to be displayed. You can use a generic list name.

Group(groupname)

The name of the group to be displayed. You can use a generic group name.

REname

This option applies only to GROUP. Specifying RENAME allows you to rename resource definitions within the group.

Resource(name)

The name of the resource to be displayed.

DISPLAY GROUP command

You can enter the following commands next to the names on the panel:

```

CHECK
DISPLAY ALL
EXPAND1
INSTALL2
LOCK
UNLOCK
  
```

On this panel, all these commands can be abbreviated down to their initial letter. It is unnecessary to type the group name. For the syntax of each command, see that command in this section.

To DISPLAY the group names, you must use a generic name. For more information about using the DISPLAY panel, see Figure 7 on page 33 and Figure 8 on page 34.

¹ Only for compatibility with previous release.

² You cannot install CONNECTION and SESSIONS resources from a DISPLAY panel. They have to be installed in groups.

Examples of the DISPLAY GROUP command

Display all groups on the CSD

```
DISPLAY GROUP(*)
```

Display all groups whose names begin with PROD and have 7 characters.

```
DISPLAY GROUP(PROD+++)
```

DISPLAY LIST command

- Displays one or more list names on a full screen panel.
- You can enter the following commands next to the names on the panel:

```
CHECK
DISPLAY GROUP
EXPAND3
LOCK
UNLOCK
```

On this panel, all these commands can be abbreviated down to their initial letter. It is not necessary to type the list name. For the syntax of each command, see that command in this section.

- To DISPLAY the list names, you must use a generic list name.
- For more information about using the DISPLAY panel, see Figure 7 on page 33 and Figure 8 on page 34.

Examples of the DISPLAY LIST command

Display all lists on the CSD:

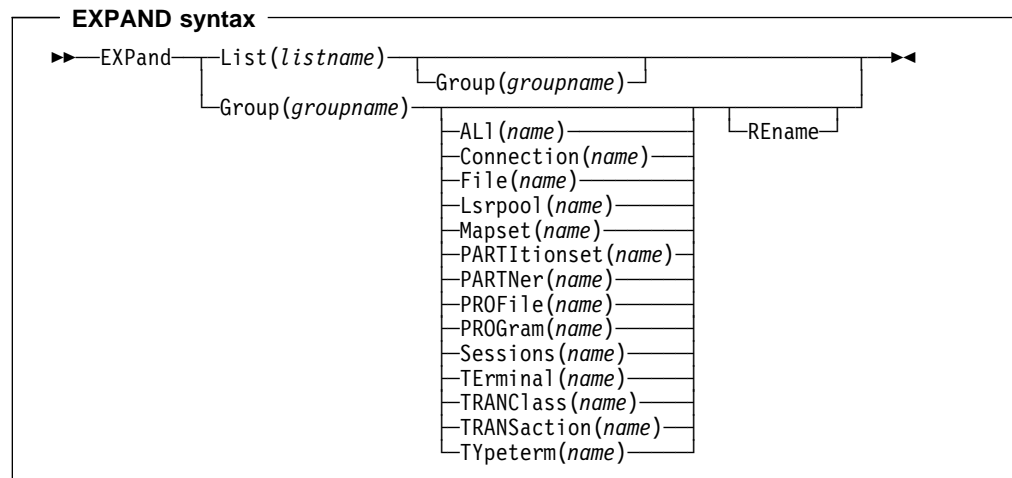
```
DISPLAY LIST(*)
```

Display all lists whose names begin with PROD and have five characters:

```
DISPLAY LIST(PROD+)
```

³ Only for compatibility with previous release.

EXPAND one or more groups



The EXPAND command shows the resource definitions in one or more groups or lists. It has been retained in CICS/ESA 4.1 for compatibility with previous releases.

Group(groupname)

The group to be expanded.

List(listname)

The list to be expanded.

REname

This option applies only to GROUP. Specifying RENAME allows you to rename resource definitions within the group.

Resource(name)

The resource you want to see within the expanded group.

EXPAND GROUP command

- Shows the resource definitions in one or more groups on a full screen panel.
- You can enter the following commands next to the names on the panel:

```

ALTER
COPY
DELETE
INSTALL
MOVE
RENAME
VIEW
  
```

All these commands can be abbreviated down to their initial letter. It is unnecessary to type the group or list name. For the syntax of each command, see that command in this section.

- You may EXPAND a generic group name. For example:

```

Show all the resource definitions in all groups on the CSD:
EXPAND GROUP(*)
  
```

```

Show all the resource definitions in groups whose names begin with PROD and are
7 characters long:
  
```

```

EXPAND GROUP(PROD+++)
```

- You may EXPAND a group to show only one resource type. The name you specify as the resource name may be a generic name. For example:

Show all PROFILE definitions in all groups on the CSD:

```
EXPAND GROUP(*) PROFILE(*)
```

Show all TERMINAL definitions that begin with SZ in a group:

```
EXPAND GROUP(ZEMGROUP) TERMINAL(SZ++)
```

- You may EXPAND a group or groups, limiting the resource definitions to those that share a generic name. For example:

Show all resource definitions, of all types, ending in A1:

```
EXPAND GROUP(REINDEER) ALL(*A1)
```

- If you use the RENAME keyword, you get a special panel on which you can overtype the resource definition names. This is a quick way of renaming many resource definitions.

EXPAND LIST command

- Shows the group names in one or more lists on a full screen panel.
- When expanding a list, you can enter the following commands next to the names on the panel:

```
ADD
REMOVE
```

- You may EXPAND part of a list, by using a generic group name, for example:

```
EXPAND LIST(INITLIST) GROUP(DFH*)
```

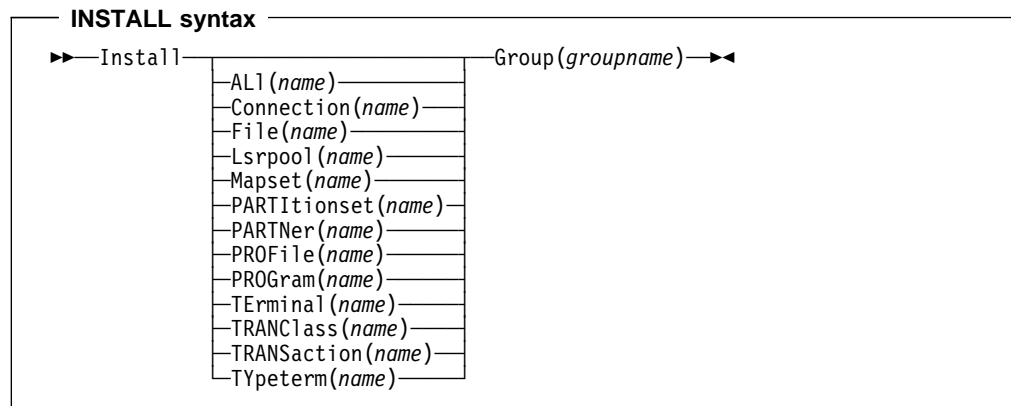
- You may EXPAND more than one list, by using a generic list name. For example, to show the groups in all lists on the CSD:

```
EXPAND LIST(*)
```

Show the groups in lists whose names begin with PROD and are five characters long:

```
EXPAND LIST(PROD+)
```

INSTALL a resource definition or group on an active CICS system



The INSTALL command dynamically makes the resource definitions in the named group available to the active CICS system.

Resource(name)

The resource to be installed. Using single-resource INSTALL avoids the problem of installing a group in which one of the resources is busy. If you INSTALL a group, that install will fail if **any** of the resource definitions in the group is busy. By using single resource install, you install only the resources you require.

When applied to telecommunication and intercommunication resources there are restrictions on the single resource INSTALL command. The restrictions apply to resource definitions that are linked: CONNECTION and SESSIONS definitions, or TERMINAL and related TYPETERM definitions.

You can install the following resource types **only** as part of a group:

- CONNECTION definitions, except a CONNECTION with ACCESSMETHOD(INDIRECT)
- SESSIONS definitions.

If you want to use single resource INSTALL for TERMINAL and related TYPETERM definitions, you must install the TYPETERM that is referred to by a TERMINAL definition **before** you install the TERMINAL definition.

The same applies when installing groups containing TERMINAL and TYPETERM definitions; you should install the group containing the TYPETERM definition before you install the group containing the TERMINAL definition.

If the named resource already exists in the active CICS system, the existing definition is replaced by the new one.

Group(groupname)

The group to be installed, or containing the resource to be installed. A generic group name is not accepted.

Replacement of an existing definition will only occur if it is not actually in use.

If the install of one or more of the resources in the group being installed fails because the resource is in use or for any other reason, the following takes place:

1. The install process continues with the next resource in the group.
2. When the group install is complete, the TERMINAL, TYPETERM, CONNECTION and SESSION resource types, (that is, those committed at the group level) are backed out. The other resource types; FILE, LSRPOOL, PROGRAM, MAPSET, AUTINSTMODEL, PARTITIONSET, TRANCLASS, TRANSACTION, PARTNER and PROFILE (that is, those committed at the resource level) are not backed out.

3. A message is displayed to indicate that the install of the group has been unsuccessful. A message is also produced on the message panel for each of the resources that failed to install stating the reason for each failure. No messages are produced for those resources that have been backed out.

CEDA INSTALL can be performed from a console, using the MVS MODIFY command.

What happens when you use the INSTALL command

Resource definitions may be installed in groups or as single definitions.

If the INSTALL command fails part of the way through installing the group, those resource types committed at the group level are backed out, whereas those resource types committed at the resource level are left in a committed state.

The following resource types are committed by individual resource, and not by complete group:

- AUTINSTMODEL
- PROFILE
- PROGRAM
- MAPSET
- PARTITIONSET
- PARTNER
- TRANCLASS
- TRANSACTION
- FILE
- LSRPOOL

For these resources, the effect of a partially successful INSTALL is to leave the resources that were added in a committed state.

The following resources are committed at the group level:

- CONNECTION
- SESSIONS
- TERMINAL
- TYPETERM

These resources are backed out after a partially successful INSTALL.

There are basically two reasons why an INSTALL may not be successful, both for INSTALLS of complete groups and for individually committed definitions:

1. One of the resource definitions could not be installed because it was currently in use (an executing transaction, for example).
2. System failure during installation.

If you wish to install only a few new or changed definitions, you are advised to install single resources. (Note that the single-resource INSTALL of some CONNECTIONs and SESSIONS is not possible.) Use of the single resource INSTALL eliminates the problems of a partial INSTALL caused by a failure. However, if you wish to change or add a larger number of definitions, you might prefer to install a new group. In that case, the following considerations apply.

When you install a group containing an updated definition of an existing resource, the installation will fail if the resource is being used at the time. You should make sure that none of the resources in a group is in use before trying to install the group.

Installation is a two-stage process: any existing definition for the resource must be “deleted” from the system tables before a definition can be installed. This can result in more than one message, if the “deletion” fails, and causes the installation to fail.

INSTALL

If you have several CICS systems that share the same CSD, you must be careful not to install a group of resources in the wrong system.

LOCK a resource definition



The LOCK command assures exclusive access to a group or list by restricting update and delete access to a single operator identifier.

Group(groupname)

The group to be locked.

List(listname)

The list to be locked.

The group or list can be used, looked at, and copied by other users of RDO, but cannot be changed or deleted by them.

You can LOCK a nonexistent group or list, thereby reserving the named group or list for your own future use.

The only RDO command which releases a lock is the UNLOCK command. No other RDO commands can unlock a group or list. For example, if you DELETE all the resources in a group, or all the groups in a list, the lock remains.

You must specify either GROUP or LIST, even if you are locking the current group or list.

A generic group or list name is not accepted.

Controlling access to a group or list — LOCK and UNLOCK

The LOCK and UNLOCK commands enable you to control update access to a group or list so that only operators with the same operator identifier can make changes.

The lock is held on the CSD file and remains in effect across restarts of CICS. The lock is owned by the user, who is identified by a combination of the CICS generic applid (specified by the APPLID system initialization parameter), and the user's operator identifier (OPIDENT).

The OPIDENT is the one associated with the user when he or she signs on to the terminal used for RDO. For further information on OPIDENT see the *CICS/ESA CICS-RACF Security Guide*. Any user who is not signed on or who has a different OPIDENT is not allowed to perform any operation that would change the locked group. However, any user is allowed to do the following things to a locked group:

- DISPLAY
- VIEW
- COPY
- CHECK
- INSTALL

The lock can be removed, using the UNLOCK command, only by a user on the same system and with the same operator identifier.

It would be wise to put a lock on your group of TYPETERMs and on your group of AUTINSTMODEL TERMINALs.

LOCK

Examples of the LOCK command

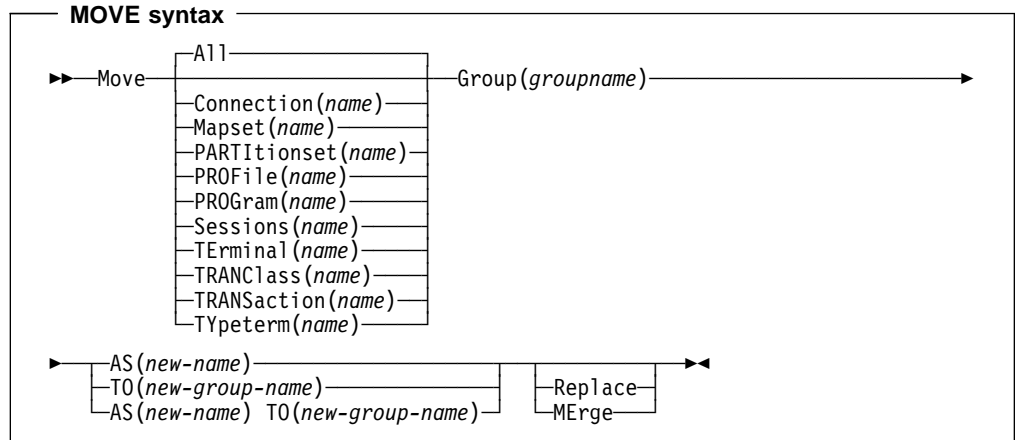
Lock a list L1:

```
LOCK LIST(L1)
```

Lock a group G1:

```
LOCK GROUP(G1)
```

MOVE a resource definition



The MOVE command moves one or more resource definitions from the group named by the GROUP keyword to the group named by the TO keyword. This command has the effect of copying the resource definitions from the first group into the second, followed by the deletion of the resource definitions from the first group.

Resource(name)

The resource you want to move. The default is ALL, which moves all the resource definitions in a group to another group.

Group(groupname)

The group containing the definitions to be moved.

AS If you move a definition within a group, you must use AS to rename it. You can also use AS if you want to move a definition to another group and rename it at the same time. You cannot use a generic name when using AS.

TO You can move definitions to a different group, using TO to specify the new group.

Replace

This applies when there are duplicate definition names in the groups named in the MOVE command. If you specify REPLACE, the definitions being moved replace those in the group named in the TO operand.

MErge

This applies when there are duplicate definition names in the groups named in the MOVE command. If you specify MERGE, duplicate definitions in the TO group are not replaced.

If you don't specify either **MERGE** or **REPLACE**, a message warns you that you are attempting to create duplicate resource definitions. The definitions will not be moved.

Note:

When you MOVE the last resource in a group TO a different group, the group is automatically deleted. An empty group cannot exist.

You should ensure that if a group is deleted any LIST which refers to that group is also updated.

#

Examples of the MOVE command

When you move a single resource definition, you can simultaneously rename it, using the AS keyword to specify the new name. For example:

```
MOVE PARTITIONSET(PSETQ1) GROUP(PSET1) AS(PSETQ4) TO(PSET2)
```

A generic resource definition name can be specified, to move all or some definitions of the same resource type. For example:

```
MOVE TRANSACTION(*) GROUP(DENTRY) TO(TEST1)  
MOVE MAPSET(ACCT+++) GROUP(ACCOUNTS1) TO(ACCOUNTS2)
```

To move all the resource definitions in a group to the new group, you can use ALL. For example:

```
MOVE ALL GROUP(N21TEST) TO(N21PROD)
```

You can use ALL with a generic name, to move all qualifying resource definitions in the group to the new group. For example:

```
MOVE ALL(N21*) GROUP(N21OLD) TO(N21NEW)
```

You can use ALL with a specific name, to move all the resource definitions of that name (which must be for different resource types) in the group to the new group. For example:

```
MOVE ALL(XMPL) GROUP(EXAMPLE) TO(EXAMPLE2)
```

To merge definitions from group X in with the definitions in group Y, keeping the Y version of any duplicates:

```
MOVE GROUP(X) TO(Y) MERGE
```

To combine definitions from group X in with definitions in group Y, keeping the X version of any duplicates:

```
MOVE GROUP(X) TO(Y) REPLACE
```

REMOVE a group from a list

REMOVE syntax

```
►►—REMOve—Group(groupname)—List(listname)—►◄
```

The REMOVE command removes a group name from a list.

Group(*groupname*)

The group to be removed.

List(*listname*)

The list from which the group is to be removed.

The group, and all its resource definitions, still exists on the CSD.

When the last group is removed from a list, the list no longer exists on the CSD.

A generic list name is not accepted.

A generic group name can be specified to remove many or all groups from a list with one command.

Examples of the REMOVE Command

A list LL02 contains the following groups:

```
G001
X001
XG001
G002
G003
X002
G004
```

To remove all groups beginning with G:

```
REMOVE GROUP(G*) LIST(LL02)
```

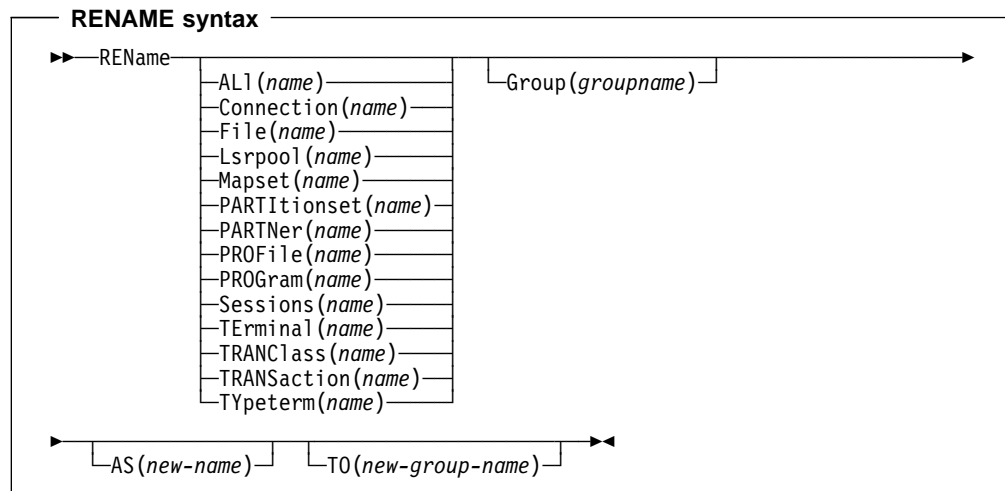
This leaves:

```
X001
XG001
X002
```

To remove the list completely:

```
REMOVE GROUP(*) LIST(LL02)
```

RENAME a resource definition



The RENAME command renames a resource definition to the new name specified in the AS option.

Resource(name)

The resource you want to copy. The default is ALL, which copies all the resource definitions in a group to another group.

Group(groupname)

The group containing the definitions to be copied.

AS If you copy a definition within a group, you must use AS to rename it. You can also use AS if you want to copy a definition to another group and rename it at the same time. You cannot use a generic name when using AS.

TO You can copy definitions to a different group, using TO to specify the new group.

You can also rename a resource definition by using the DISPLAY command or the EXPAND command with the RENAME option. See “DISPLAY groups and lists” on page 54 and “EXPAND one or more groups” on page 56 for information about these commands.

Examples of the RENAME command

- To rename a resource and keep it in its current group:

```
RENAME PROFILE(PROF1) AS(NEWPROF) GROUP(PROFS)
```
- You can rename all resource definitions which share the same name, to a new name, using the ALL keyword instead of a resource type. For example:

```
RENAME ALL(TVA) AS(XTVA) GROUP(XTVA1)
RENAME ALL(USER) AS(OLDU) GROUP(USERDEF)
```
- You can move a resource definition to a new group, which you specify in the TO option, at the same time as renaming it. (You can also do this with the MOVE command.) For example:

```
RENAME PROGRAM(N20ZA) AS($SOSERR) GROUP(N20) TO($MODULES)
```
- You can move all the resource definitions of the same name from one group to another, using the TO option, at the same time as renaming them. For example:

```
RENAME ALL(USER) GROUP(USERDEF) AS(TEMP) TO(TEMPGRP)
```
- You cannot rename a resource definition to a name that already exists in the target group.
- A generic resource definition name or group name is not accepted.

UNLOCK a group or list

UNLOCK syntax

```

UNLOCK Group(groupname)
        List(listname)
  
```

UNLOCK removes the lock from a group or a list of definitions.

- The UNLOCK command is the only RDO command which can remove a lock on a list or group put there by use of the RDO LOCK command.
- You can UNLOCK a nonexistent group or list.
- You must specify either the GROUP or the LIST keyword, even if you are unlocking the current group or list, because the UNLOCK command can be used with both.
- A generic group name or list name is not accepted.
- For more information about UNLOCK, see “Controlling access to a group or list — LOCK and UNLOCK” on page 25.

Group(groupname)

The group to be unlocked.

List(listname)

The list to be unlocked.

Controlling access to a group or list — LOCK and UNLOCK

The LOCK and UNLOCK commands enable you to control update access to a group or list so that only operators with the same operator identifier can make changes.

The lock is held on the CSD file and remains in effect across restarts of CICS. The lock is owned by the user, who is identified by a combination of the CICS generic applid (specified by the APPLID system initialization parameter), and the user’s operator identifier (OPIDENT).

The OPIDENT is the one associated with the user when he or she signs on to the terminal used for RDO. For further information on OPIDENT see the *CICS/ESA CICS-RACF Security Guide*. Any user who is not signed on or who has a different OPIDENT is not allowed to perform any operation that would change the locked group. However, any user is allowed to do the following things to a locked group:

- DISPLAY
- VIEW
- COPY
- CHECK
- INSTALL

The lock can be removed, using the UNLOCK command, only by a user on the same system and with the same operator identifier.

It would be wise to put a lock on your group of TYPETERMs and on your group of AUTINSTMODEL TERMINALS.

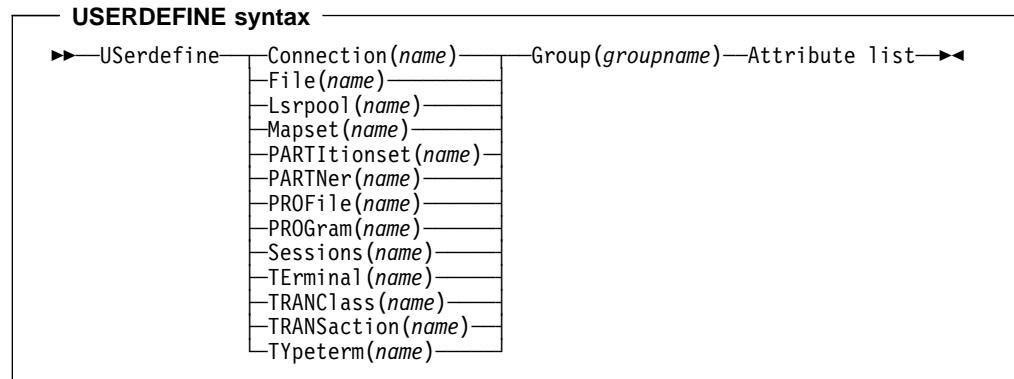
Example of the UNLOCK command

```

Unlock a group G1:
UNLOCK GROUP(G1)

Unlock a LIST L1:
UNLOCK LIST(L1)
  
```

USERDEFINE



USERDEFINE is an alternative to the DEFINE command: Instead of using CICS-supplied default values, USERDEFINE uses your own defaults. Otherwise it operates in exactly the same way as DEFINE.

Group(groupname)

The name of the group to be defined.

To set up your own defaults, use DEFINE to create a dummy resource definition named USER in a group named USERDEF. Each dummy resource definition must be complete (for example, a transaction definition must name a program definition, even though you will always supply a program name when you USERDEFINE a transaction). You need not install the dummy resource definitions before using USERDEFINE.

Do this for each type of resource for which you want to set default values. Each of them will be named USER, but this does not matter because the fact that they are definitions of different resource types makes them unique.

So you could have the following resources in your USERDEF group:

- FILE(USER)
- PROGRAM(USER)
- MAPSET(USER)
- PARTITIONSET(USER)
- TRANSACTION(USER)
- TRANCLASS(USER)
- PROFILE(USER)
- TYPETERM(USER)
- TERMINAL(USER)
- CONNECTION(USER)
- SESSIONS(USER)
- LSRPOOL(USER)
- PARTNER(USER)

Let's use this example and look at the panels you would see.

Example of the USERDEFINE process

Assembler programmers at an installation have created a dummy program definition called USER with Assembler as the default language. They use USERDEFINE to define their programs to CICS.

First you must define a program called USER in group USERDEF. You could do this with the command:

```
CEDA DEFINE PROGRAM(USER) GROUP(USERDEF)
```

The following figure shows the panel you would see as a result of this command:

```

DEFINE PROGRAM(USER) GROUP(USERDEF)                CICS RELEASE = 0410
OVERTYPE TO MODIFY
CEDA DEFine
  PROGram      : USER
  Group        : USERDEF
  DDescription  ==>
  Language     ==>
  RELoad       ==> No
  RESident     ==> No
  USAge        ==> Normal
  USE1pacopy   ==> No
  Status       ==> Enabled
  RS1          : 00
  Cedf         ==> Yes
  DAtalocation ==> Below

  CObol | Assembler | Le370 | C | Pli | Rpg
  No | Yes
  No | Yes
  Normal | Transient
  No | Yes
  Enabled | Disabled
  0-24 | Public
  Yes | No
  Below | Any

I New group USERDEF created

                                SYSID=ABCD  APPLID=DBDCCICS
DEFINE SUCCESSFUL                TIME: 11.24.39  DATE: 93.359
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Type in ASSEMBLER as the LANGUAGE keyword and press ENTER:

```

OVERTYPE TO MODIFY
CEDA USerdefine
  PROGram      : USER
  Group        : USERDEF
  DDescription  ==>
  Language     ==> Assembler
  RELoad       ==> No
  RESident     ==> No
  USAge        ==> Normal
  USE1pacopy   ==> No
  Status       ==> Enabled
  RS1          : 00
  Cedf         ==> Yes
  DAtalocation ==> Below

  CObol | Assembler | Le370 | C | Pli | Rpg
  No | Yes
  No | Yes
  Normal | Transient
  No | Yes
  Enabled | Disabled
  0-24 | Public
  Yes | No
  Below | Any

                                SYSID=ABCD  APPLID=DBDCCICS
DEFINE SUCCESSFUL                TIME: 11.24.41  DATE: 93.359
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Now, each time you want to define a new program, you can use the USERDEFINE command to get the default value ASSEMBLER automatically. So, if you want to define a new program P2 in group GRP you enter the command:

```
CEDA USERDEFINE PROGRAM(P2) GROUP(GRP)
```

USERDEFINE

The following figure shows the panel resulting from this command.

```
USERDEFINE PROGRAM(P2) GROUP(GRP)
OVERTYPE TO MODIFY
CEDA USerdefine
PROGram      : P2
Group        : GRP
DEscription  ==>
Language     ==> Assembler      CObol | Assembler | Le370 | C | Pl1 | Rpg
RELoad      ==> No              No | Yes
RESident     ==> No              No | Yes
USAge       ==> Normal          Normal | Transient
USElpacopy   ==> No              No | Yes
Status       ==> Enabled        Enabled | Disabled
RS1          : 00                0-24 | Public
Cedf         ==> Yes            Yes | No
DATAlocation ==> Below          Below | Any

                                     APPLID=DBDCCICS
USERDEFINE SUCCESSFUL                TIME: 11.25.48  DATE: 89.359
PF 1 HELP 2 COM 3 END                6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

You will see that the value ASSEMBLER has appeared for the LANGUAGE keyword. You can overwrite the keyword values on this panel to complete the definition just as you can with the DEFINE command panel.

Once you have set up your own defaults in a USER resource definition, anyone using the USERDEFINE command for that resource type will get those default values.

By renaming your USER to something else and defining your own, you can use your own default values. Normally, though, your installation will probably agree upon default values for standardization reasons, and put a LOCK on the USERDEF GROUP.

VIEW resource definitions

VIEW syntax

```

▶▶ View Group(groupname)
  -ALl(name)
  -ConneCtion(name)
  -File(name)
  -Lsrpool(name)
  -Mapset(name)
  -PARTItionset(name)
  -PARTNer(name)
  -PROFile(name)
  -PROGram(name)
  -Sessions(name)
  -TERminal(name)
  -TRANClass(name)
  -TRANSaction(name)
  -TYpeterm(name)
  ▶▶

```

The VIEW command lets you look at the resource definition attributes in the same way as the ALTER command. However, you cannot update any of the definitions.

Group(*groupname*)

The group to be viewed. If no name is given, the current group is assumed.

Examples of the VIEW Command

```
VIEW TERMINAL(SZT1) GROUP(ZEMTERMS)
```

```
VIEW MAPSET(N20MAP01) GROUP(N20)
```

See Figure 9 on page 36 for an example of the VIEW panel.

VIEW

Chapter 7. System definition file utility program, DFHCSDUP

The CICS system definition utility program, DFHCSDUP, is a component of resource definition online (RDO). DFHCSDUP is an offline utility program that allows you to read from and write to a CICS system definition (CSD) file, either while CICS is running or while it is inactive.

You can use the DFHCSDUP program to:

- ADD a group to the end of a named list in a CSD file
- ALTER attributes of an existing resource definition
- APPEND a group list from one CSD file to a group list in another, or in the same, CSD file
- COPY all of the resource definitions in one group or several generically named groups to another group or several other generically named groups in the same, or in a different, CSD file
- DEFINE a single resource, or a group of resources, on the CSD
- DELETE from the CSD a single resource definition, all of the resource definitions in a group, or all of the group names in a list
- EXTRACT data from the CSD and pass it to a user program for processing
- INITIALIZE a new CSD file, and add to it CICS-supplied resource definitions
- LIST selected resource definitions, groups, and lists
- MIGRATE the contents of a table from a CICS load library to a CSD file
- REMOVE a single group from a list on the CSD file
- SERVICE a CSD file when necessary
- UPGRADE the CICS-supplied resource definitions in a primary CSD file for a new release of CICS
- VERIFY a CSD file by removing internal locks on groups and lists.

You can invoke the DFHCSDUP program in two ways:

1. As a batch program (see page 75)
2. From a user program running either in batch mode or in a TSO environment (see page 77).

Sharing the CSD between CICS/ESA 4.1 and earlier releases

If you want to share the CSD between CICS regions at different release levels, to enable you to share common resource definitions, you must update the CSD from the higher level region – CICS/ESA 4.1.

In CICS/ESA Version 4, some attributes are obsolete, and are removed from the CSD definitions. 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/MVS 2.1.2 or a CICS/OS/VS 1.7 region, you can use the CICS/ESA 4.1 CSD utility, 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. (See Figure 13 on page 75.)

The *CICS/ESA Migration Guide* discusses these obsolete attributes and their compatibility with earlier releases.

DFHCSDUP utility program

Note: You cannot use the EXTRACT command of the CICS/ESA 4.1 DFHCSDUP utility when the COMPAT option is specified.

Input to the DFHCSDUP program

Input to the DFHCSDUP program (see Figure 12) is from:

- A **primary CSD file**, which must be present, and have a ddname of DFHCSD
- Optionally, a **secondary CSD file**, for which you can specify any ddname
- A CICS table, as specified on the MIGRATE command.

Output from the DFHCSDUP program

The result of running the DFHCSDUP program (see Figure 12) may be an updated primary file, or a print file.

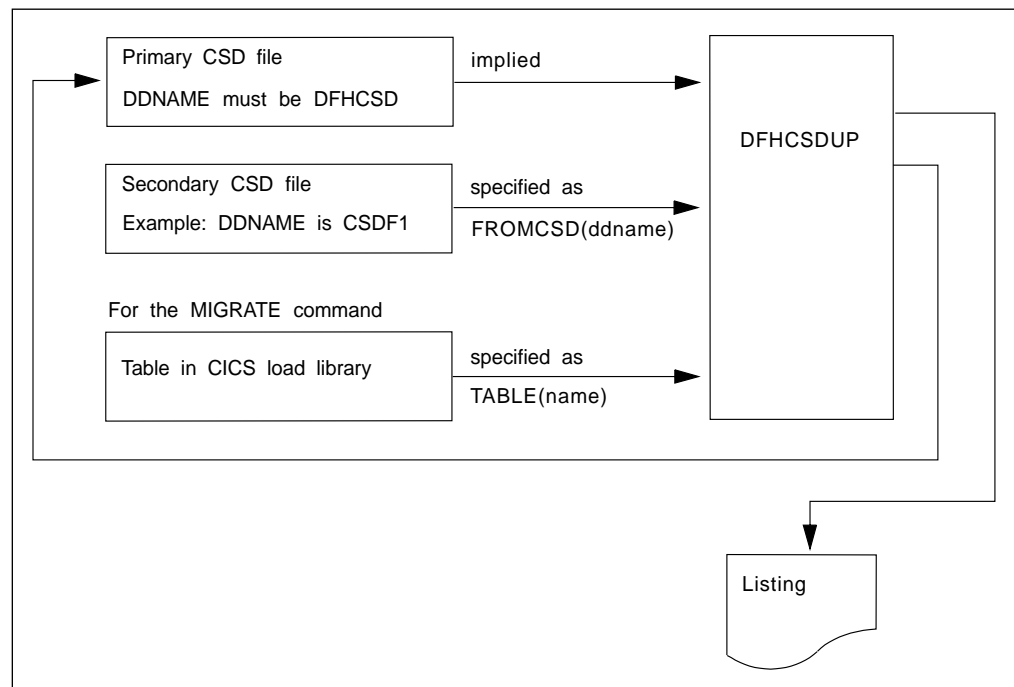


Figure 12. The DFHCSDUP offline utility program

Invoking DFHCSDUP as a batch program

The job in Figure 13 shows you an example of the job control statements you can use to invoke DFHCSDUP as a batch program.

```
//CSDJOB JOB accounting info,name,MSGLEVEL=1
//STEP1 EXEC PGM=DFHCSDUP,REGION=512K, 1
// PARM='CSD(READWRITE),PAGESIZE(60),NOCOMPAT'
//STEPLIB DD DSN=CICS410.SDFHLOAD,DISP=SHR
//*****
//* If you are running DFHCSDUP with the MIGRATE command,
//* and your CICS load tables are not in CICS410.SDFHLOAD,
//* concatenate your own private library here:
//*****
// DD DSN=CICS410.userlib.tables,DISP=SHR
//DFHCSD DD UNIT=SYSDA,DISP=SHR,DSN=CICS410.DFHCSD
//SECNDCSD DD UNIT=SYSDA,DISP=SHR,DSN=CICS410.SECNDCSD 2
//indd DD UNIT=SYSDA,DISP=SHR,DSN=extract.input.dataset 3
//outdd DD UNIT=SYSDA,DISP=SHR,DSN=extract.output.dataset 4 5
//* or
//outdd DD SYSOUT=A 4 5
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
:
: DFHCSDUP commands 6
:
/*
//
```

Figure 13. Sample job to run DFHCSDUP

Notes:

- 1** The EXEC statement should specify a suitable REGION size and a PARM parameter:
- **The REGION size.** A region size of 512KB is generally recommended for the execution of the DFHCSDUP program. However, for the MIGRATE command, the table to be migrated is loaded into main storage, so the region size should be at least 512KB plus the size of the largest table.
 - **The PARM parameter.** Use this to specify any of the following options:
 - UPPERCASE** specifies that you want all output from DFHCSDUP to be in uppercase. If you want all output to be in mixed case (the default), do not code this option.
 - CSD({READWRITE|READONLY})** specifies whether you want read/write or read-only access to the CSD from this batch job. The default value is READWRITE.
 - PAGESIZE(nnnn)** specifies the number of lines per page on output listings. Values for nnnn are 4 through 9999. The default value is 60.
 - NOCOMPAT** or **COMPAT** specifies whether the DFHCSDUP utility program is to run in compatibility mode (that is, whether it can update definitions that are obsolete in CICS/ESA 4.1). The default is NOCOMPAT, which means that you cannot update obsolete attributes. For further information about this option, see “Sharing the CSD between CICS/ESA 4.1 and earlier releases” on page 73.
- 2** You need a DD statement for a secondary CSD if you specify the FROMCSD parameter on an APPEND, COPY, or SERVICE command. The ddname for this DD statement is the name you specify on the FROMCSD parameter. The secondary CSD must be a different data set from the primary; you must not define primary and secondary DD statements that reference the same data set.

3 If you specify the EXTRACT command, you may need to:

- Concatenate with STEPLIB the libraries that contain your USERPROGRAM programs.
- Include a DD statement for any input data set that is defined in your user program. For example, the CICS-supplied user program, DFH\$CRFA, needs a DD statement with a ddname of CRFINPT.

The input file specified by CRFINPT is needed by the user programs DFH\$CRFx (where x=A for Assembler or x=P for PL/I) and DFH0CRFC (for COBOL) to supply the list of resource types or attributes for which you want a cross reference listing. You can specify (in uppercase) any resource type known to CEDA, one resource type per line (starting in column 1). For example, your CRFINPT file may contain the following resource types (one per line) to be cross referenced:

```
PROGRAM
TRANSACTION
TYPETERM
XTPNAME
DSNAME
```

For programming information about the use of the CRFINPT file by the programs DFH\$CRFx or DFH0CRFC (for COBOL), see the *CICS/ESA Customization Guide*.

4 If you specify the EXTRACT command, you need to include the DD statements for any data sets that receive output from your extract program. The ddname is whatever ddname you define in the user program. The CICS-supplied sample programs need DD statements for the following ddnames:

Table 5. DD statements for the CICS-supplied sample programs

program name	ddname	example DD statement
DFH\$CRFx or DFH0CRFC (VS COBOL II)	CRFOUT	//CRFOUT DD SYSOUT=A
DFH\$FORx or DFH0FORC (VS COBOL II)	FOROUT	//FOROUT DD SYSOUT=output.dataset
DFH0CBDC	CBDOUT SYSABOUT	//CBDOUT DD SYSOUT=A //SYSABOUT DD SYSOUT=A

5 The output data sets in these examples are opened and closed for each EXTRACT command specified in SYSIN. If you are writing the output to a sequential disk data set, specify DISP=MOD to ensure that data is not overwritten by successive EXTRACT commands. Alternatively, provided you do not specify SYSOUT on the DD statement, you can change the OPEN statement in the program (for example, in the VS COBOL II versions, to OPEN EXTEND). For programming information about the CICS-supplied user programs, see the *CICS/ESA Customization Guide*.

6 Syntax

You can code commands and keywords using abbreviations and mixed case, as given in the syntax box in the description of each command. If you enter an ambiguous command or keyword, the DFHCSDUP program issues a message indicating the ambiguity.

You can specify keyword values longer than one line, if you use the continuation character (an asterisk) at the end of a line (in column 72). Subsequent lines start in column 1. For example, you can use this facility to specify XTPNAME values of up to 128 hexadecimal characters.

Invoking the DFHCSDUP program from a user program

Invoking the DFHCSDUP program from a user program enables you to create a flexible interface to the utility. By specifying the appropriate entry parameters, your program can cause the DFHCSDUP program to pass control to an exit routine at any of five exit points. The exits can be used, for example, to pass commands to the DFHCSDUP program, or to respond to messages produced by its processing.

You can run your user program:

- In batch mode
- Under TSO.

Notes:

1. In a TSO environment, it is normally possible for the terminal user to interrupt processing at any time by means of an ATTENTION interrupt. In order to protect the integrity of the CSD file, the DFHCSDUP program does not respond to such an interrupt until after it has completed the processing associated with the current command. It then writes message number DFH5618 to the put-message exit, where this is available, and also to the default output file:

```
AN ATTENTION INTERRUPT HAS BEEN REQUESTED
DURING DFHCSDUP PROCESSING
```

Your put-message exit routine can terminate the DFHCSDUP program, if desired. (You **must** supply a put-message routine if you want your operators to regain control after an ATTENTION interrupt.)

2. Suitably authorized TSO users can use the CEDA INSTALL transaction to install resources that have previously been defined with the DFHCSDUP program.

The CICS-supplied sample program, DFH\$CUS1, illustrates how the DFHCSDUP program can be invoked from a user program. It is written as a command processor (CP) for execution under the TSO/E operating system.

The following sections outline the entry parameters of the DFHCSDUP program and the responsibilities of the user program. For programming information about invoking the DFHCSDUP program from a user program, see the *CICS/ESA Customization Guide*.

Entry parameters for the DFHCSDUP program

When invoking the DFHCSDUP program, your program passes a list of up to five parameters, as described below:

OPTIONS

A list of character strings, separated by commas. (The information passed here is that which would otherwise be passed on the PARM keyword of the EXEC statement of JCL.)

Note: A maximum of three options may be specified:

UPPERCASE specifies that you want all output from DFHCSDUP to be in uppercase. If you want all output to be in mixed case (the default), do not code this option.

CSD({READWRITE|READONLY}) specifies whether you require read/write or read-only access to the CSD. The default value is READWRITE.

PAGESIZE(nnnn) specifies the number of lines per page on output listings. Valid values for nnnn are 4 through 9999. The default value is 60.

NOCOMPAT|COMPAT specifies whether the DFHCSDUP utility program is to run in compatibility mode (that is, whether it can update definitions that are obsolete in CICS/ESA 4.1). The default is NOCOMPAT, which means that you cannot update obsolete attributes. For further information

about this option, see “Sharing the CSD between CICS/ESA 4.1 and earlier releases” on page 73.

DDNAMES

A list of ddnames that, if specified, are substituted for those normally used by the DFHCSDUP program.

HDING

The starting page number of any listing produced by the DFHCSDUP program. You can use this parameter to ensure that subsequent invocations produce logically numbered listings. If this parameter is not specified, the starting page number is set to 1.

The page number, if supplied, must be four numeric EBCDIC characters.

DCBs

The addresses of a set of data control blocks for use internally by the DFHCSDUP program. Any DCBs (or ACBs) that you specify are used internally, instead of those normally used by the DFHCSDUP program.

Note that if you specify both replacement DDNAMES and replacement DCBs, the alternative DCBs are used, but the alternative DDNAMES are disregarded.

EXITS

The addresses of a set of user exit routines to be invoked during processing of the DFHCSDUP program.

Responsibilities of the user program

Before invoking the DFHCSDUP program, your calling program must ensure that:

- AMODE(24) and RMODE(24) are in force
- S/370 register conventions are obeyed
- If the EXITS parameter is passed, any programming environment needed by the exit routines has been initialized
- Any ACBs or DCBs passed for use by the DFHCSDUP program are OPEN.

Commands for the DFHCSDUP program

This section describes the commands available with the DFHCSDUP utility program. Commands can be abbreviated, but the minimum abbreviation allowed differs from some of the CEDA command abbreviations.

Rules for the syntax and preparation of commands

Enter the commands in columns 1 through 71 of 80-character input records. You can specify keyword values longer than one line, if you use the continuation character (an asterisk) at the end of a line (in column 72). Subsequent lines start in column 1. For example, you can use this facility to specify XTPNAME values of up to 128 hexadecimal characters.

The command keywords can be specified by abbreviations and in mixed case, as shown in the command syntax under each command description. The minimum abbreviation is given in uppercase in the command syntax, with the optional characters given in lower case; for example:

```
Alter Connection(name) Group(groupname)
```

Leading blanks are ignored, and blanks between keywords and operands are permitted.

Comment records are permitted; they must have an asterisk (*) in column 1. Comment material is not permitted on a record that contains a command.

Blank records between commands are ignored.

Follow the conventions for the names of groups and lists when coding the GROUP, LIST, TO, and TYPESGROUP parameters. If you use a generic specification for the GROUP or LIST parameter in the LIST command, you can use the symbols * and + in the same way as for CEDA.

The FROMCSD parameter must contain a valid ddname conforming to the rules for the JCL of the operating system.

An example of a valid sequence of commands is shown in Figure 14. Other examples of commands are given in the command descriptions that follow.

```

*                               SET UP INITIAL CSD FILE
INITialize
*
LIst LIst(DFHLIST) Objects
*                               UPGRADE FROM EARLIER RELEASE
UPgrade
*                               MIGRATE MAIN TABLES
MIgrate TABle(DFHFCTF1)
*
MIgrate TABle(DFHTCTT1)
*
LI Group(PPTM1)
LI G(SETM*)
*                               CREATE GROUP PCTZ4
Copy G(PCTM1) To(PCTZ4)
C G(SETMP3) T(PCTZ4) ReplacE
LI G(P++M+)
*                               CREATE LIST MODLIST
APpend LIst(TESTLIST) TO(MODLIST) FRomcsd(CSDF1)
AP LI(SECLIST) To(MODLIST) FR(CSDF1)
AP LI(DFHLIST) To(MODLIST)
*
LI ALL OBJECTS

```

Figure 14. Sample commands of the DFHCSDUP program

Command processing following internal error detection

If you have provided a put-message-exit routine for the DFHCSDUP program, it is invoked whenever a message is issued. You can use this exit to respond to error messages produced by DFHCSDUP processing, when the DFHCSDUP program is invoked from a user program. The put-message-exit routine is not used if the DFHCSDUP program is running as a batch program. For programming information about the DFHCSDUP exits, see *C/CS/ESA Customization Guide*.

The reaction of the DFHCSDUP program to an error (with return code 8 or greater) depends on the nature of the error and on how the DFHCSDUP program is invoked.

If an error is detected while the DFHCSDUP program is running as a batch program, one of the following two reactions occurs:

1. If the error occurs during connection of the CSD, no subsequent commands are completed.
2. If the error occurs elsewhere, no subsequent commands are executed other than LIST commands.

If an error is detected while the DFHCSDUP program is receiving commands from a get-command exit, all subsequent commands are processed if possible.

Chapter 8. DFHCSDUP commands

This chapter explains the syntax and rules for each of the DFHCSDUP commands. The commands are:

- “ADD a group to a list” on page 82
- “ALTER a resource definition” on page 83
- “APPEND a list to another list” on page 85
- “COPY a resource definition” on page 87
- “DEFINE a resource definition” on page 89
- “DELETE a resource definition, group, or list” on page 91
- “EXTRACT resource definition data” on page 93
- “INITIALIZE a new CSD with the required CICS resource definitions” on page 94
- “LIST resource definitions” on page 95
- “MIGRATE macro-defined resource definitions from tables to the CSD” on page 97
- “REMOVE a group from a list” on page 99
- “SERVICE a CSD” on page 100
- “UPGRADE a CSD” on page 101
- “VERIFY a CSD” on page 102

+

APAR PN74913

+

added the following note.

+

Note: CICS may add new resource types and attributes across different releases. For similar named fields, such as Transaction and Tranclass, the minimum abbreviation may have to change from release to release. For example, at CICS/ESA R330 the minimum abbreviation for Transaction was TR, whereas at CICS/ESA R410 it is now TRANS, to differentiate between that and TRANC for Tranclass.

+

+

+

+

+

+

+

+

+

+

+

RDO commands are validated online by the CEDA/B/C transactions, and confirmation for such ambiguities is required before the transaction can continue. This is not possible for input to DFHCSDUP however. To avoid batch operations failing due to the minimum abbreviation altering across CICS releases, it is recommended that the full resource type and attribute names are used when coding input to DFHCSDUP. While CICS cannot guarantee upward compatibility of the abbreviations, it always provides such compatibility for the full names of the fields.

ADD a group to a list

ADD syntax

```
▶—Add—Group(groupname1)—List(listname)—▶◀
```

The ADD command adds a group to a list.

Group(*groupname1*)

The name of the group to be added. The name must not already exist in the list. A generic group name is not accepted. If you don't specify a group, the current group name is added.

List(*listname*)

The name of the list to which the group is to be added. If the list does not already exist, a new one is created. If LIST is not specified, the group name is added to the current list if there is one. A generic list name is not accepted.

The list name can be up to 8 characters in length. The characters allowed are A-Z, 0-9, @, #, and \$. Lowercase characters are treated as uppercase. Do not use Grouplist names beginning with DFH, because these characters are reserved for use by CICS.

+
+
+

Examples of the ADD command

Create a list LA01, by adding a group to it:

```
ADD GROUP(GA001) LIST(LA01)
```

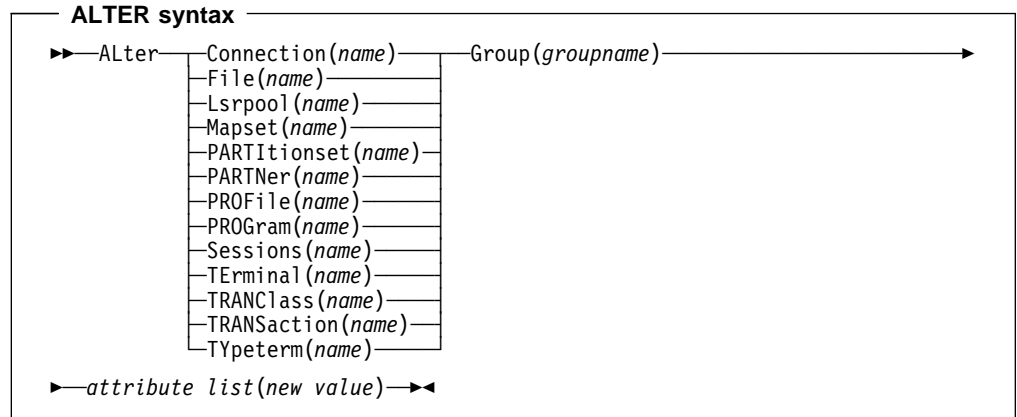
Add another group to list LA01:

```
ADD GROUP(GA002) LIST(LA01)
```

LA01 now looks like this:

```
GA001
GA002
```

ALTER a resource definition



The ALTER command changes some or all of the attributes of an **existing** resource definition.

APAR PN66642

added the next sentence.

A generic resource name or group name is **not** accepted.

Resource(name)

The resource whose attributes you want to alter.

For information about the attributes that you can specify on the ALTER command for the various resource types, and for a description of the attributes and default values of each resource type, see Part 4, "RDO resource types and their attributes" on page 127.

Group(groupname)

The name of the group containing the resource to be altered.

Attribute list

The attributes to be altered. For information about the attributes that you can specify on the ALTER command for each resource type, see Part 4, "RDO resource types and their attributes" on page 127.

- Use the ALTER command to change some or all of the attributes of an **existing** resource definition.
- Do **not** use ALTER to change the value of the attributes of a TYPETERM definition on which other attributes depend. If you make a mistake with DEVICE, SESSIONTYPE, or TERMMODEL, you should delete the definition, and define a new one with the correct values.
- You can specify null operand values, for example:

```
ALTER FILE(TEST) GROUP(ACT1) DESCRIPTION()
```

If an option for which you have specified a null value has a default, the value used depends upon the type of field.

– The command:

```
ALTER FILE(TEST) GROUP(ACT1) RLSACCESS DESCRIPTION()
```

uses the default value of NO for RLSACCESS and the description is blanked out.

– The command:

```
ALTER FILE(TEST) GROUP(ACT1) PROFILE()
```

uses the default value DFHCICSA for the PROFILE field.

ALTER

- Changes to resource definitions in the CSD do not take effect until you install the group in which the resource definition resides.

+
+
+

APAR PN66642

deleted an item about the use of generic names which previously appeared here and had been found to be invalid.

Example of the DFHCSDUP ALTER command

To make a program resident:

```
ALTER PROGRAM(ERR01) GROUP(GENMODS) RESIDENT(YES) DATALOCATION()
```

+
+

APAR PN66642

deleted a second example, which had been found to be invalid

APPEND a list to another list

APPEND syntax

```
►—APpend—List(listname1)—To(listname2)—Fromcsd(ddname)—◄
```

The APPEND command adds the groups in one list to the end of another list.

List(*listname1*)

Specifies the name of the list that is appended. Do not use a generic list name.

The list being appended can be on the primary CSD, or on another CSD. If you are appending from another CSD, you must identify it by specifying the FROMCSD parameter.

+ The list name can be up to 8 characters in length. The characters allowed are A-Z, 0-9,
 + @, #, and \$. Lowercase characters are treated as uppercase. Do not use Grouplist
 + names beginning with DFH, because these characters are reserved for use by CICS.

To(*listname2*)

Specifies the name of the list to which you want the group names appended. If you are appending from another CSD, you can give this list the same name as the one you are appending from. Do not use a generic list name.

If this target list already exists, the source list is appended to the end of it. If the target list does not exist, it is created. (In effect, you are copying the source list.)

FRomcsd(*ddname*)

Identifies the *ddname* of the secondary CSD from which you are appending *listname1*.

No duplicate group names are allowed in a list. If DFHCSDUP finds any duplicate names during the APPEND operation it ignores them, and they are not appended. The DFHCSDUP output listing contains a warning message if this happens.

Example of the APPEND command

A list called LISTA contains the following groups:

GB001
GB002
GB003

A list called LISTB contains the following groups:

G001
G002
G003

Append LISTB to LISTA, like this:

```
APPEND LIST(LISTB) TO(LISTA)
```

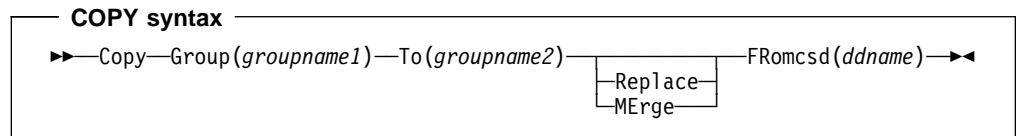
After this, LISTA contains the following groups, in this order:

GB001
GB002
GB003
G001
G002
G003

and LISTB still contains:

G001
G002
G003

COPY a resource definition



The COPY command copies all the resource definitions in one group to another.

Group(groupname1)

Specifies the name of the group to be copied. You can specify a generic name by using an asterisk (*). See “Generic naming in the COPY command” for details.

To(groupname2)

Specifies the name of the group to which the definitions are copied. If you are copying from another CSD, you can give this group the same name as the one you are copying from. You can specify a generic name by using an asterisk (*). See “Generic naming in the COPY command” for details.

Replace

If groupname2 already exists and duplicate definitions occur, the definitions in groupname1 replace those in groupname2.

MErge

If groupname2 already exists and duplicate definitions occur, the original definitions in groupname2 are preserved.

FRomcsd(ddname)

Identifies the ddname of the secondary CSD from which you are copying groupname1.

The COPY command copies all the resource definitions in *groupname1* to *groupname2*. The group to be copied (*groupname1*) can be on the primary CSD, or it can be on the CSD specified by the FROMCSD parameter. Unlike the CEDA version of the COPY command, all the resource definitions in the group are copied. Individual resources can not be identified and selected for copying.

The group is copied to the group named on the TO parameter (*groupname2*) in the primary file. If this group already exists, the definitions from the source group (*groupname1*) are added to those already in the *groupname2* group. If the group specified on the TO parameter does not already exist, a new group of that name is created. However, if duplicate definitions exist in the two groups, the whole copy operation fails unless you specify REPLACE or MERGE to indicate how duplicates should be handled.

Generic naming in the COPY command

The COPY command accepts generic group names, both on the GROUP keyword and on the TO keyword, subject to the following rules:

- The only generic character permitted on the COPY command is the asterisk (*) symbol.
- The prefix length of *groupname1* must be equal to or greater than the prefix length of *groupname2*. Thus COPY GROUP(DFHCOMP*) TO(USRCMP*) is valid, but COPY GROUP(DFHCO*) TO(USRCOMP*) is not.

You can use the asterisk (*) symbol to copy from generically-named groups to other generically-named groups or from generically-named groups to a specific group, as shown in Figure 15 on page 88.

Note: There is no AS parameter as in the CEDA version of the COPY command.

The DFHCSDUP output listing tells you which definitions were copied, and what happened if duplicates were found.

Examples of the COPY command

The following example copies a group named GA001 to a group named GA002, which already exists, replacing any duplicate resource definitions with those in group GA001.

```
COPY GROUP(GA001) TO(GA002) REPLACE
```

The following example copies group GA003 to group GA004, but if any duplicate definitions occur, preserves the group GA004 definitions.

```
COPY GROUP(GA003) TO(GA004) MERGE
```

The following example copies all the CICS-supplied groups to user-named groups with a prefix of USR, with the result that DFHOPER becomes USROPER, DFHSTAND becomes USRSTAND, and so on.

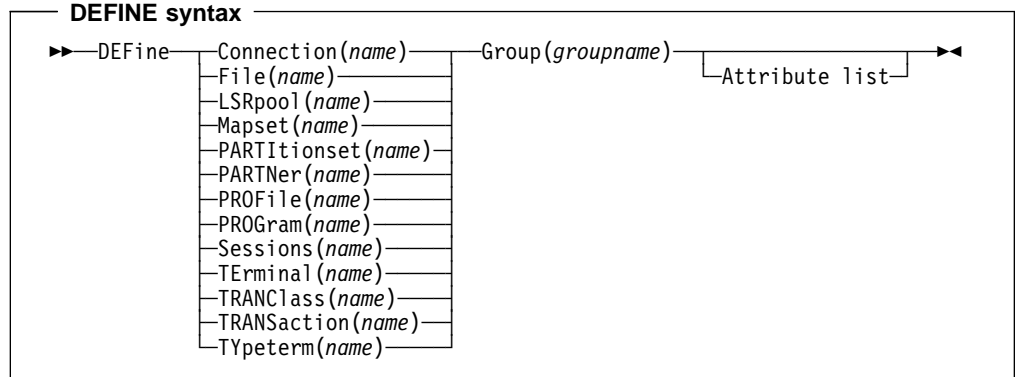
```
COPY GROUP(DFH*) TO(USR*)
```

The following example copies every group starting with ABCD to the group called NEWGROUP:

```
COPY GROUP(ABCD*) TO(NEWGROUP)
```

Figure 15. Examples of the COPY command

DEFINE a resource definition



The DEFINE command creates new resource definitions.

Resource(name)

The name of the resource you want to define. Do not use a generic resource name. The resource keyword must always be the first operand of the DEFINE command.

Group(groupname)

The name of the group containing the resource definition to be altered. Do not use a generic group name. If you specify the name of a group which does not already exist, the group is created.

Attribute list

The attribute list depends on the resource type being defined; some resources have attributes that must be included in the definition. For a description of the attributes and default values of each resource type, see Part 4, "RDO resource types and their attributes" on page 127. Attributes that you do not specify are given default values.

You can use the same name for more than one resource definition in a group, if the definitions are for different resource types. For example:

```
DEFINE PROGRAM(N28A) GROUP(N28APPL)
DEFINE TRANSACTION(N28A) GROUP(N28APPL)

DEFINE TERMINAL(USER) GROUP(USERDEF)
DEFINE PROGRAM(USER) GROUP(USERDEF)
```

DEFINE

Examples of the DEFINE command

This example defines two consoles to CICS.
(You do not need continuation symbols if a definition spans several lines).

```
DEFINE TERMINAL(CON0)    GROUP(CONTERMS)
      CONSOLE(00)        TYPETERM(DFHCONS)
      DESCRIPTION(MVS CONSOLE ID 00, FOR ISSUING JCL COMMANDS)
```

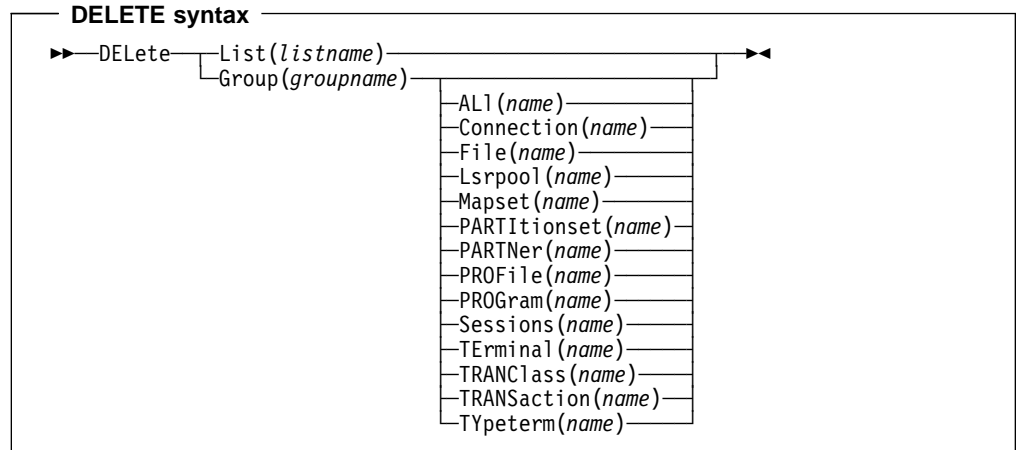
```
DEFINE TERMINAL(CON1)    GROUP(CONTERMS)
      CONSOLE(01)        TYPETERM(DFHCONS)
      DESCRIPTION(MVS CONSOLE ID 01, MVS MASTER CONSOLE)
```

The INITIALIZE command generates a TYPETERM definition, but not a TERMINAL definition, for a console. You must have at least one console defined in order to issue MVS MODIFY commands to CICS.

Console id 00 is used to issue commands using MVS job control language, and by authorized programs that use that MGCR macro to issue MVS commands.

Figure 16. Examples of the DEFINE command

DELETE a resource definition, group, or list



The DELETE command deletes a single resource definition in a group, all the resource definitions in a group, or all the group names in a group list.

List(listname)

Specifies the name of the list to be deleted. Do not use a generic list name.

Group(groupname)

If this is specified alone, it indicates the name of the group to be deleted. If a resource is also specified, it indicates the group to which the resource belongs. Do not use a generic group name.

Resource(name)

The name of the resource to be deleted. Do not use a generic resource name.

This parameter can be used only with the GROUP keyword.

Deleting a resource definition is different from removing a group from a list (see “REMOVE a group from a list” on page 99). A deleted resource definition really does disappear from the CSD.

You can not delete the definitions of groups and lists supplied by IBM.

If you delete a list, the definitions of the resources within the groups contained in the list are not deleted. To do this, you must also delete each group individually.

Note:

When you DELETE the last resource in a group, the group is automatically deleted. An empty group cannot exist.

You should ensure that if a group is deleted any LIST which refers to that group is also updated.

#

#

#

#

#

DELETE

Example of the DFHCSDUP DELETE command

A list in the primary CSD called LISTA contains the following groups:

```
GB001  
GB002
```

Group GB001 contains the following resource definitions:

```
TERMINAL(CON0)  
TERMINAL(CON1)  
TERMINAL(TEST)
```

The following command deletes the resource definition for the terminal TEST from group GB001:

```
DELETE TERMINAL(TEST) GROUP(GB001)
```

The following command deletes all the resource definitions in group GB002:

```
DELETE GROUP(GB002)
```

This leaves only group GB001 in the group list LISTA.

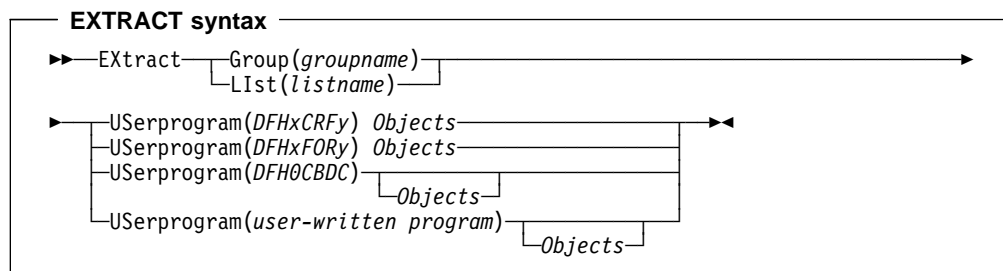
The following command deletes all group names in the group list LISTA:

```
DELETE LIST(LISTA)
```

Note: The resource definitions in the groups in LISTA are not deleted.

Figure 17. Examples of the DELETE command

EXTRACT resource definition data



You can use the EXTRACT command to extract resource definition data from the CSD, either from a list or from a group, and invoke a user program to process the extracted data. You specify the user program on the USERPROGRAM parameter.

Note: For programming information about coding user programs for the EXTRACT command, see the *CICS/ESA Customization Guide*.

Group(groupname)

Selects only those resource definitions within the named group. You can specify a generic group name.

List(listname)

Selects only those resource definitions within the groups contained in the named list. You can use a generic list name only if you are not using the OBJECTS option.

Objects

Returns the detail of each resource definition. You can extract resource definition data at two levels of detail:

- Without the OBJECTS option, the command extracts either the names of all the groups within a specified list, or the names of all the resource definitions within a specified group.
- With the OBJECTS option, all the resource definition attributes are also extracted.

You must specify OBJECTS for the CICS-supplied sample user programs DFHxCRFy and DFHxFORy. It is optional for DFH0CBDC and user-written user programs.

Userprogram(user-written program)

Is the name of the user-written program that is to process the data retrieved by the EXTRACT command. You must supply a USERPROGRAM value.

CICS supplies three types of sample user program: DFHxCRFy, DFHxFORy, and DFH0CBDC. The letter x in the program name is \$ for assembler or PL/I and 0 for COBOL. The letter y in the program name denotes the programming language, where y=A is the assembler version, y=C is the COBOL version, and y=P is the PL/I version. Note that DFH0CBDC is supplied as VS COBOL II only. All other user programs are available in source form, in CICS41.SDFHSAMP, and the assembler versions are also available in pregenerated form in CICS41.SDFHLOAD.

Example of the EXTRACT command

The following command uses the CICS-supplied user program, DFH0CBDC, to extract the resource definitions in group DFHTYPE and create the DEFINE commands needed to create them. It stores these commands in the file specified by the CBDOUT DLBL statement.

```
EXTRACT GROUP(DFHTYPE) USERPROGRAM(DFH0CBDC) OBJECTS
```

Figure 18. Example of the EXTRACT command

INITIALIZE a new CSD with the required CICS resource definitions

INITIALIZE syntax

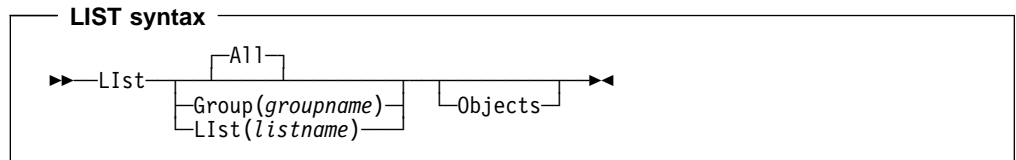
```
▶▶—INITialize—◀◀
```

You can use the INITIALIZE command to prepare a newly defined data set for use as a CSD file. You must initialize your CSD before you can use any of the other DFHCSDUP commands, or the RDO transactions. After you have initialized your CSD, you do not need to execute this function again.

The standard entries for the CICS-supplied resource definitions are created on the CSD. The INITIALIZE command arranges these definitions into groups, and defines these groups in a group list named DFHLIST. This list contains only the CICS-supplied groups that are required by a CICS system.

INITIALIZE also creates a control record at the start of the CSD. This record contains fields identifying the CICS release and the current level of service applied to the CSD. It also has fields containing the date and time of creation of the CSD file, and the date and time the file was last updated. Both these fields appear on the hard copy listing of the CSD produced by the LIST command.

LIST resource definitions



The LIST command produces listings of the current status of the CSD. The listings are output to the SYSOUT data set, along with the messages issued by the command processing. The result is to print the contents of all the qualifying groups or lists.

Group(groupname)

Selects only those resource definitions within the named group. You can specify a generic group name.

List(listname)

Selects only those resource definitions within the groups contained in the named list. You can use a generic list name only if you are not using the OBJECTS option (the only command where a generic list name is not acceptable is LIST LIST(listname) OBJECTS).

Objects

This specifies the level of detail required for each resource definition. You can extract resource definition data at two levels of detail:

- Without the OBJECTS option, the command extracts either the names of all the groups within a specified list, or the names of all the resource definitions within a specified group.
- With the OBJECTS option, all the resource definition attributes are also extracted.

Examples. The listings produced by the various commands are as follows:

- LIST ALL
 - Names of defined lists and groups
 - Summary of lists
 - Summary of groups.

This prints summaries of all the definitions of lists and groups that exist on the CSD.

- LIST ALL OBJECTS
 - Names of defined lists and groups
 - Summary of lists
 - Summary of groups
 - Objects in groups.

This prints summaries of all the definitions of lists and groups that exist on the CSD, together with the properties of the resources in all the groups.

- LIST GROUP(groupname) (group name may be generic)
 - Summary of groups.

This summarizes the names of all the resources in one or more groups. They are organized within each group into resource type categories (for example, map sets, programs, and so on).

- LIST GROUP(groupname) OBJECTS (group name may be generic)
 - Summary of groups (see above)
 - Objects in groups.

This enables you to tabulate the properties of the resources, again organized according to resource type. The creation time for each resource is given, together with all its attributes, as originally set up by using DEFINE and ALTER commands, or by migrating it from a CICS table. The properties of transactions and profiles are arranged in the same subcategories that appear on the CEDA DEFINE screen.

- LIST LIST(listname) (list name may be generic)
 - Summary of lists.

The contents of one or more group lists are tabulated. The groups appear in the same sequence as their position in the list. This order is set by the commands ADD and APPEND, which were used in the CEDA transaction to build the list.

- LIST LIST(listname) OBJECTS (generic list name not allowed)
 - Summary of lists (see above)
 - Objects of groups in list.

This enables you to tabulate the properties of all the resources to be defined in a CICS system at startup time. These are identified by the list name or names specified in the GRPLIST=(list1,list2,list3,list4) parameter in the system initialization table. The names of all the groups in the list appear in the summary of lists. Then, for each group contained in the list, the properties of the individual resources in the group are tabulated.

The 'Objects in Groups in Lists' tabulation arranges the groups in the same order as they were added to the group list. This order matters if duplication occurs, when definitions of the same resource may exist in more than one group. If a list of this type is used at system startup time, the resource definitions used when there is duplication are those belonging to the group that is latest in the list.

MIGRATE macro-defined resource definitions from tables to the CSD

MIGRATE syntax

```

▶—MIgrate—TAbLe(tablename)—┐
                               └─TYpesgroup(typesgroupname)—▶
  
```

The MIGRATE command transfers the contents of an FCT or a TCT from a CICS load library to the CSD.

Table(*tablename*)

Identifies the name in the load library of the table you want to migrate (that is, DFHFCTxx or DFHTCTxx).

TYpesgroup(*typesgroupname*)

For use with TCT migration only, this is the name of the group to which the TYPETERM definitions are to be migrated.

The contents of a table are transferred as one group, or as a set of several groups, containing definitions.

When migrating large tables, make sure you allocate a sufficiently large region for the largest table to be loaded.

If you are migrating a table from an earlier release of CICS, you must specify the COMPAT option on the DFHCSDUP PARM of the EXEC statement to prevent migration errors.

- **To transfer an FCT**, the format is:

```
MIgrate TAbLe(tablename)
```

The result is a set of groups containing file and LSR pool definitions. You can define each group by the macro:

```
DFHFCT TYPE=GROUP, GROUP=xxxxxxxx
```

which you insert in the FCT source instructions before you assemble the FCT for migration. Any file or LSR pool definitions that come before the first such TYPE=GROUP macro are migrated into a group named after the table name: for example, if the table name is DFHFCTxx, the group name is FCTxx.

- **To transfer a TCT**, the format is:

```
MIgrate TAbLe(tablename) [TYpesgroup(typesgroupname)]
```

where TYpesgroup(*typesgroupname*) specifies the name of the group to contain the TYPETERM definitions obtained from the TCT.

If this parameter is not specified, the TYPETERM definitions are put in the GROUP currently being created, with the TERMINAL definitions.

The result is:

1. A set of groups containing terminal definitions. You can define each group by the macro:

```
DFHTCT TYPE=GROUP, GROUP=xxxxxxxx
```

which you insert in the TCT source instructions before you assemble the TCT for migration. Any terminal definitions that come before the first TYPE=GROUP macro are migrated into a group named after the table name. If the table name is DFHTCTxx, the group name is TCTxx.

2. A group of TYPETERM definitions. These are derived from attributes of TYPE=TERMINAL macros which are often identical for many terminals. They are put into the CSD GROUP named in the TYPESGROUP parameter.

The typeterm attributes of each TYPE=TERMINAL table macro are checked with existing TYPETERM definitions and if they don't match with any of these, a new TYPETERM is added to the CSD.

+
+

MIGRATE

The existing TYPETERMs checked are:

- TYPETERMs in the GROUP currently being created
- TYPETERMs in the group specified in the TYPESGROUP parameter of the MIGRATE command.

However, the scope of the checking is never extended to include any other TYPETERMs in other groups already on the CSD. (Such groups may have been created using RDO or by a previous MIGRATE command.) For this reason, it is a good idea to use the TYPESGROUP parameter to avoid creating duplicate TYPETERMs in different groups. It is convenient to keep the TYPETERMs in a separate group anyway.

TYPETERMs created on the CSD during the migration are named systematically, in a way related to the TRMTYPE parameter of the original terminal definition. The name consists of a prefix (3–5 characters) with a 3-character suffix. For example, a TYPETERM defining attributes for a 3270 printer is named 3270P001. Variants with the same TRMTYPE are named 3270P002, and so on. The migration process ensures that this name is used as the TYPETERM parameter of every terminal definition that references it.

Note: Migrating your TCT does not cause an error if the destination group already exists. Only definitions that already exist are flagged by an error message; any new or additional definitions are added to the existing group.

Example of the MIGRATE command

An FCT, DFHFCT\$\$, from CICS/ESA 3.3 contains the following definitions:

```
DFHFCT TYPE=FILE,ACCMETH=VSAM,FILE=DDNAME1,
      DISP=SHR,JREQ=ALL
DFHFCT TYPE=FILE,ACCMETH=VSAM,FILE=DDNAME2,
      DISP=SHR,JREQ=ALL
DFHFCT TYPE=FILE,ACCMETH=VSAM,FILE=DDNAME3,
      DISP=SHR,JREQ=ALL
```

To migrate the first and third definitions to the group SAMPLES, and the second definition to the default group, you could add the following DFHFCT macros before the definitions in DFHFCT\$\$:

```
DFHFCT TYPE=FILE,GROUP=SAMPLES before the first definition
DFHFCT TYPE=FILE,GROUP=*DEFAULT before the second definition
DFHFCT TYPE=FILE,GROUP=SAMPLES before the third definition
```

To name the default group STDGROUP, you could use the MIGRATE command:

```
MIGRATE TABLE(DFHFCT$$)
```

Figure 19. Examples of the MIGRATE command

REMOVE a group from a list

REMOVE syntax

```
►►—Remove—List(listname)—Group(groupname)—►◄
```

The REMOVE command removes a group name from a list. The group, and all its resource definitions, still exists on the CSD.

#

When the last group is removed from a list, the list no longer exists on the CSD.

List(*listname*)

The name of the list from which a group is to be removed. Do not use a generic list name. When the last group is removed from a list, the list no longer exists on the CSD.

Group(*groupname*)

The name of the group to be removed. Do not use a generic group name.

Example of the REMOVE command

A list LL02 contains the following groups:

```
G001 G002 G003 G004
```

To remove group G003:

```
REMOVE GROUP(G003) LIST(LL02)
```

This leaves:

```
G001 G0023 G004
```

Figure 20. Examples of the REMOVE command

SERVICE a CSD**SERVICE syntax**

```
►►—Service—FRomcsd(ddname)—LEve1(nnn)—►◄
```

You might occasionally (between CICS releases) have to apply a service routine to carry out preventive or corrective maintenance to your CSD. You do this by loading and running a special service program (DFHCUS1), which is supplied with CICS as a separately loadable module.

You can use the SERVICE command to create a new copy of the CSD, from the existing CSD. All the definitions are preserved, with the corrections (if any) applied.

FRomcsd(*ddname*)

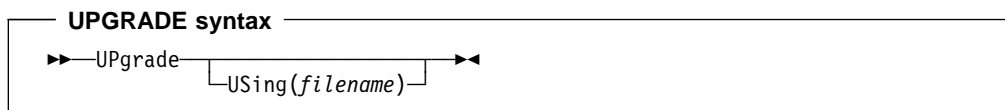
Specifies the *ddname* of the current CSD, which for the purposes of the command is treated as the secondary CSD.

LEve1(*nnn*)

Associated with your CSD is a current service level, initially set to 000 when the file was initialized. Applying the service routine causes the service level to be incremented in steps of one, from a “current level” to a “target level”.

This operand specifies the target service level to which the CSD is to be upgraded, and must be 1 higher than the current level of FROMCSD. Specify it as a 3-character integer; for example, LEVEL(001).

UPGRADE a CSD



You can use the UPGRADE command to change the CICS-supplied resource definitions in a primary CSD. If your existing CSD was initialized at CICS/ESA 3.3 or earlier, you must UPGRADE it using DFHCSDUP to obtain the RDO extensions in CICS/ESA 4.1. Upgrading ensures that the definitions in the DFH-groups are brought up to the level of CICS/ESA 4.1 function.

The upgrade command can also be used to apply any package of IBM-supplied resource definitions to the CSD. For example, the definitions for the CICS sample programs and transactions can be transferred to the CSD with the UPGRADE statement.

Note: Any CICS-supplied definitions which you have copied to your own groups do **not** have any changes applied to them as a result of this command.

USing(filename)

Upgrading a CSD for CICS/ESA 4.1 does not require you to use the USING operand. All IBM-supplied definition from **any** release are deleted and then the CSD is initialized, so you do not need to say which release you came from. However, UPGRADE USING(filename) is used to install IBM features onto CICS. For example, UPGRADE USING(DFHRDJPN) is used to place the double-byte character set feature definitions onto the CSD.

#

VERIFY a CSD

VERIFY syntax

▶▶—VERIFY—◀◀

The VERIFY command to remove internal locks on groups and lists. Only use it when the CSD is not in use and no backout processing is pending on the CSD; preferably use it only when no CICS systems that may use the CSD are running.

VERIFY acts upon the whole CSD, and is for use in the extreme condition where internal lock records have been left behind. These records are normally removed when a function that changes the CSD has been completed. However, this may not have happened if there was a system failure when the CEDA transaction was running, or if an offline utility failed to finish. The locks may prevent CEDA users from accessing certain groups and lists on the CSD.

Note that VERIFY removes only the internal locks. It does not affect the normal user locks applied by the LOCK command in the CEDA transaction.

Part 3. Autoinstall

This part discusses the automatic installation of resources in CICS. It consists of the following chapters:

- Chapter 9, “Autoinstall for VTAM terminals” on page 105
- Chapter 10, “Autoinstall for APPC connections” on page 119
- Chapter 11, “Autoinstall for programs, mapsets, and partitionsets” on page 123

Overview of autoinstall

To use a resource without autoinstall, you must have a definition for that resource installed in your CICS system, created using either CEDA, DFHCSDUP, or a macro table. This clearly uses up time (to define and install every resource), and storage, as every definition occupies storage whether the resource is being used or not.

With autoinstall, you do not need to define and install every resource that you intend to use. Instead, CICS dynamically creates and installs a definition for you when a resource is requested. CICS bases the new definition on a “model” definition provided by you.

Autoinstall can be used for the following resources:

- VTAM terminals
- APPC (LU6.2) connections
- Programs
- Mapsets
- Partitionsets

Autoinstall models

You must provide at least one *model* resource definition for each type of resource to be autoinstalled. When a resource is requested which does not have an installed definition, CICS creates a definition based on what you have specified in the model. You can have more than one model depending on what properties you want the autoinstalled resources to have; for example, if you had 500 terminals all with the same properties and another 500 with a different set of properties, you would have two model terminal definitions, one for each of the two sets of properties.

Autoinstall control program

You control autoinstall by means of an *autoinstall control program*, either user-written or supplied by CICS. This program is responsible for, among other things, providing the model name or names to CICS, providing VTAM information to CICS for autoinstall for terminals, and so on.

CICS provides three autoinstall control programs; one for terminals; one which allows both terminals and connections; and one for programs. You can use the CICS-supplied ones or you can customize them to suit your installation.

These chapters also describe in detail what you must do to use autoinstall, and tell you about other considerations, such as what happens at installation time, and what happens at CICS restart.

Chapter 9. Autoinstall for VTAM terminals

This chapter explains how to implement autoinstall for VTAM terminals. It contains the following information:

- “Getting started with autoinstall”
- “Autoinstall considerations” on page 107
- “Autoinstall and VTAM” on page 109
- “Autoinstall and recovery and restart” on page 113
- “The autoinstall control program” on page 116

If you have not used autoinstall for any resources before, you should read “Overview of autoinstall” on page 103.

For information on autoinstalling connections and parallel sessions, see Chapter 10, “Autoinstall for APPC connections” on page 119.

Getting started with autoinstall

This list explains how to set up autoinstall for your VTAM terminals.

1. Are your terminals eligible for autoinstall?

The terminals that **can** be autoinstalled are:

- VTAM locally attached 3270 terminals (non-SNA), both displays and printers
- VTAM logical unit type 0 terminals
- VTAM logical unit type 1 terminals, including SCS printers
- VTAM logical unit type 2 terminals
- VTAM logical unit type 3 terminals
- VTAM logical unit type 4 terminals
- VTAM logical unit type 6.2 single-session terminals
- TLX or TWX terminals using NTO.

Terminals that **cannot** be autoinstalled are:

- Pipeline terminals
- Automatic teller machines (3614 and 3624)
- MVS consoles
- Non-VTAM resources
- VTAM logical unit type 6.1 ISC and MRO sessions.

2. Decide whether to use autoinstall

You are likely to benefit from autoinstall if any of the following apply to your system:

- A significant number of VTAM terminals
- Frequent changes to your network
- Many VTAM terminals logged off much of the time
- Many VTAM terminals using other applications much of the time
- Many VTAM terminals that need access to multiple, but unconnected, CICS systems.

Autoinstall might be less beneficial if you have:

- A small, static network
- Many terminals more or less permanently logged on to one CICS system
- Many terminals logging on and off frequently
- Many terminals logging on and off at the same time.

3. Decide which devices to autoinstall

This decision depends on how you use your VTAM terminals. For example, a terminal that is logged on all the time can be autoinstalled, but you might choose to define it individually.

An autoinstall logon is slower than a logon to a terminal individually defined to CICS, so if you switch continually between applications and have to log on to CICS frequently, you may require individual definitions for some terminals.

You should also consider your use of automatic transaction initiation (ATI), terminal list tables (TLTs), and the intercommunication methods in use in your installation. "Autoinstall considerations" on page 107 discusses these and other relevant issues.

4. Create your TYPETERM and model TERMINAL definitions

CICS supplies some TERMINAL and TYPETERM definitions; these are listed in "TYPETERM definitions in group DFHTYPE" on page 361 and "Model TERMINAL definitions in group DFHTERM" on page 368. You can use these definitions if they are suitable; if not, create your own using CEDA or DFHCSDUP.

Define an autoinstall model for each different kind of terminal to be autoinstalled. Try to keep the number of definitions to a minimum, so that the autoinstall control program can be as simple as possible.

When you create your definitions, consider whether you want to use the QUERY structured field (see page 217). It can help the autoinstall control program to choose which model on which to base a definition, and so speed up the autoinstall process.

5. Redefine DFHZCQ

For every region using autoinstall, you should redefine DFHZCQ to be RESIDENT(YES). (DFHZCQ is in the CICS-supplied group DFHSPI). See the *CICS/ESA Performance Guide* for guidance on why you should consider making programs resident.

6. Ensure that your VTAM LOGMODE table entries are correct

"Autoinstall and VTAM" on page 109 explains the relationship between CICS autoinstall and VTAM. For programming information, including a list of VTAM LOGMODE table entries, see the *CICS/ESA Customization Guide*.

7. Design and write an autoinstall control program

The terminal autoinstall control program is invoked by CICS every time there is a valid request for a TCT entry to be autoinstalled, and every time an autoinstalled TCT entry is deleted.

For programming information about the autoinstall control program, see the *CICS/ESA Customization Guide*. "The autoinstall control program" on page 116 provides a summary of what the program is about.

Before beginning your program, look at the CICS-supplied autoinstall control program DFHZATDX in group DFHSPI to see if it is suitable for what you want to do with autoinstall.

8. Enable terminal autoinstall

You can enable autoinstall for terminals either by using the system initialization table (SIT) or by using the EXEC CICS or CEMT INQUIRE|SET SYSTEM command.

There are five system initialization parameters relating to terminal autoinstall:

GRPLIST specifies the list or lists containing the group(s) of autoinstall models created.

AIEXIT specifies the name of the autoinstall program to be used. It defaults to DFHZATDX, the name of the IBM-supplied autoinstall control program.

AIQMAX specifies the maximum number of terminals that can be queued concurrently for autoinstall. When this limit is reached, CICS will request VTAM to stop passing LOGON and BIND requests to CICS until CICS has processed one more autoinstall request.

The purpose of the limit is to protect the system from uncontrolled consumption of operating system storage by the autoinstall process, as a result of some other abnormal event. Normally, in the process of autoinstall, the principal consumer of CICS storage is the autoinstall task (CATA) itself. The amount of CICS storage consumed by the autoinstall process during normal operation can therefore be controlled by creating an appropriate TRANCLASS definition to limit the number of autoinstall tasks that can exist concurrently.

AILDELAY specifies the time interval, expressed as hours, minutes and seconds (hhmmss), that will elapse after an autoinstall terminal logs off before its TCTTE is deleted. The default value is 0, indicating that TCTTEs will be deleted at logoff time and at warm shutdown as CLSDST is issued for the autoinstall terminals still in session. Specifying an AILDELAY interval will permit the TCTTE to be reused should the terminal log back on before the interval has expired.

AIRDELAY is the time interval, expressed as hours, minutes and seconds (hhmmss), that will elapse after emergency restart before terminal entries will be deleted if they are not in session. The default value is 700, indicating a restart delay of 7 minutes.

For information on how to specify these parameters in the SIT, see the *CICS/ESA System Definition Guide*.

There are three options relating to terminal autoinstall on the INQUIRE|SET AUTOINSTALL command:

CUR(value) The number of autoinstall logon requests that are currently being processed.

MAXREQS(value) The largest number of autoinstall requests that are allowed to queue at one time, in the range 0-999.

You can prevent more terminals from logging on through autoinstall by setting this value to 0. This allows autoinstalled entries for terminals currently logged on to be deleted by the autoinstall program when they log off.

PROGRAM(pgrmid) The name of the user program that is controlling the autoinstall process. The default is the CICS-supplied program DFHZATDX.

Autoinstall considerations

This section offers some considerations which should help you to decide which devices to autoinstall. It covers the following subjects:

- “Automatic transaction initiation”
- “The TCT user area (TCTUA)” on page 108
- “The terminal list table (TLT)” on page 109
- “Transaction routing” on page 109
- “Autoinstall and output-only devices” on page 109

Automatic transaction initiation

If a BMS ROUTE message is sent to a terminal that has a TCT entry but is not logged on, CICS saves the message for subsequent delivery. In addition, if the TCT entry so specifies, CICS attempts to acquire the terminal and log it on for that purpose. CICS attempts to satisfy other ATI requests (EXEC CICS START or a transient data trigger level) in the same way.

The use of autoinstall for printers is severely limited because almost all transactions for printers are initiated automatically. It is possible to autoinstall printers, however, if you arrange for them to be logged on. Entering VARY NET,...,LOGON as a console command does this.

For an autoinstalled terminal, a TCT entry *may* be available even when the terminal is logged off. This happens only if the terminal has been logged on earlier in the CICS run and depends on the TYPETERM definition, the system initialization parameters, and the SNT entry for the last user of the terminal. For details, see "Automatic signoff, logoff, and TCTTE deletion" on page 113. If a TCT entry exists, an autoinstalled terminal can accept an ATI request just like an individually defined terminal.

If you do choose autoinstall for terminals that might receive ATI requests, you should make use of the AUTOCONNECT attribute on the TYPETERM definition for your models. AUTOCONNECT(YES) means that the terminal is logged on to CICS automatically at an emergency restart (see Figure 24 on page 115), by CICS requesting a VTAM SIMLOGON (simulated logon). (Because this requires a TCT entry, it does not happen at cold or warm start.)

You may find that setting up a printer-owning region is the best approach, especially if you have distributed printing with many small printers.

Whether or not you autoinstall your printers, you can associate a printer and an alternate printer with a display device. The association is made when the display device is autoinstalled. Note that definitions for these printers need not have been installed at the time the display device is autoinstalled, but they must exist at the time of use.

The TCT user area (TCTUA)

The TCT user area is an optional extension to the TCT entry. The TCTUA is available for application use (the rest of the TCT entry belongs to CICS). It has traditionally been used for two purposes:

- To pass data from one transaction of a pseudo-conversational sequence to the next
- To maintain user profile information and statistics during a terminal session. (This is not necessarily a VTAM session, but a period of access to a particular application, as defined by the application.)

The first use has gradually been supplanted by COMMAREA and other CICS facilities, but the second is still fairly common. An application may store statistics, such as teller totals, in the TCTUA, which is initialized by a PLTPI program at the beginning of execution and retrieved by a PLTSD program at termination (shutdown). Autoinstall does not provide the ability to save this information between logoff and logon, because the TCTUA does not exist then. In addition, the TCTUA is not available to PLTPI and PLTSD programs at system initialization and termination. A new technique must be devised to allow the initialization of TCTUA and user data when the user logs on or logs off.

As noted earlier, the autoinstall process creates the TCT entry (including the TCTUA) before the first transaction is executed at the terminal, but after the autoinstall control program has done its initial execution. Thus you cannot access the TCTUA in the autoinstall control program and so any TCTUA initialization must be done later. You could write your own good morning transaction to do this, or use the first transaction of the application in question.

Moreover, the autoinstall control program does not have access to the TCTUA at logoff either, because CICS deletes the TCT entry (including the TCTUA) before invoking this program. Therefore, if an application needs to capture statistics or other information from such a TCTUA, it must get access before CICS does this. The place to do this is in the **node error program (NEP)**, the user-written component of the terminal error processing routines, because CICS drives the NEP exit before it deletes the TCT entry.

The terminal list table (TLT)

A terminal list table is a list of terminals, defined either by four-character CICS terminal names or three-character CICS operator identifiers. It is used principally for routing messages to multiple destinations and giving limited operational control of a group of terminals to a supervisor. Both of these uses must be rethought in an autoinstall environment. If a TLT lists terminals that do not have TCT entries, because they are not logged on at the time the TLT is used, supervisory operations against those terminals will fail. For example, you cannot put a nonexistent TCT entry into or out of service.

Similarly, message routing works differently for individually defined and for autoinstalled terminals. When a message is sent to an individually defined terminal that is not logged on, the message is saved in temporary storage, and delivered when the terminal logs on. When a message is sent to a terminal that is not defined because it is an autoinstall terminal and is not logged on, CICS gives a route fail condition, indicating that it knows nothing of that terminal. Indeed, if terminal names are generated and assigned randomly, as they may be in an autoinstall environment, the whole TLT mechanism breaks down.

Transaction routing

An autoinstall request to a local system overrides an autoinstall request for the same definition shipped from a different system.

If transaction routing can occur between two CICS systems, any terminal that can logon to both should be installed in both in the same way. Such a terminal should either be:

- Autoinstalled in both systems, or
- Defined to each system in its TCT setup at initialization.

Autoinstall and output-only devices

Most of the benefits of autoinstall apply more to **display devices** than to printers. Displays initiate transactions in CICS; usually any transaction output is returned to the input terminal, so that neither CICS nor the application needs to know any other NETNAME than that of the display, which identifies itself in the process of logging on.

On the other hand, when output is directed to output-only **printers**, either CICS or the application must know what NETNAME to use, and this implies that some knowledge of the network be maintained somewhere. The primary and alternate printer names in an individually-defined TCT entry constitute this kind of information, as maintained by CICS. In the case of autoinstalled terminals, corresponding information, if it is required, must either be maintained in tables or files, embedded in the application, supplied by VTAM ASLTAB and ASLENT model terminal support (MTS), or supplied dynamically by the user.

Autoinstall and VTAM

This section explains how autoinstall works with VTAM. It is intended to help you understand the processing that takes place when you use autoinstall. It consists of the following:

- “The process of logging on to CICS using autoinstall”
- “What happens when the user logs off” on page 112

The process of logging on to CICS using autoinstall

To help you understand the process, consider what takes place when you log on to CICS through VTAM. (See also Figure 21 on page 111 and Figure 22 on page 112.) CICS/ESA supports the model terminal support (MTS) function of VTAM 3.3 and above. Using MTS, you can define the model name, the printer (PRINTER), and the alternate printer (ALTPRINTER) for each terminal in a VTAM table. CICS captures this information as part of autoinstall processing at logon, and uses it to create a TCTTE for the terminal. If you are using MTS, you must use a version of DFHZATDX that is suitable for use on CICS/ESA 3.3 or CICS/ESA 4.1. See the *CICS/ESA Customization Guide* for programming information about the user-replaceable autoinstall program.

Autoinstall for VTAM terminals

1. VTAM receives your request, determines that you want to use CICS, and passes your request to CICS.
2. CICS extracts your terminal's NETNAME name from the logon data. CICS searches the TCT for an entry with the same NETNAME.
3. If it finds such an entry, CICS issues an OPNDST to VTAM to establish a session between CICS and the terminal. This is the normal CICS logon process.
4. If it fails to find a matching entry, CICS checks the system initialization parameters that were coded in the SIT, or reset using CEMT, to check whether it can allow an autoinstall.
5. If the system initialization parameters allow an autoinstall, CICS checks the terminal data passed by VTAM, to check whether the terminal is eligible for autoinstall.
6. If the terminal is eligible, CICS examines the bind image to see if it carries sufficient information.
7. If the VTAM bind image data proves sufficient, CICS searches the autoinstall model table (AMT) and autoinstalls the terminal in one of the following ways:
 - If VTAM has supplied CICS with a valid model name, CICS passes this name to the **autoinstall control program**. (If the logon request has come to CICS through VTAM 3.3, and if you have supplied VTAM with names of model terminals, CICS can obtain the name of the model terminal from the logon data.)
 - If VTAM has not supplied CICS with a valid model name, CICS searches the AMT for suitable autoinstall models and passes these to an **autoinstall control program**, together with VTAM logon data. If VTAM has supplied CICS with an invalid model name, message DFHZC6936 results.
8. The autoinstall control program (either the CICS-supplied program or one written by you) selects one of the models and provides the rest of the information necessary to complete a TCT entry for the terminal.
9. When the autoinstall control program returns control, CICS builds a TCT entry using the autoinstall model, the data returned by the autoinstall control program, and the VTAM logon data for the terminal. CICS then adds the new entry to the TCT and issues an OPNDST to VTAM to establish a session between CICS and the terminal.
10. If the TYPETERM definition so specifies, CICS uses the QUERY function to find out about some of the features of the terminal. (These features are listed on page 217.)

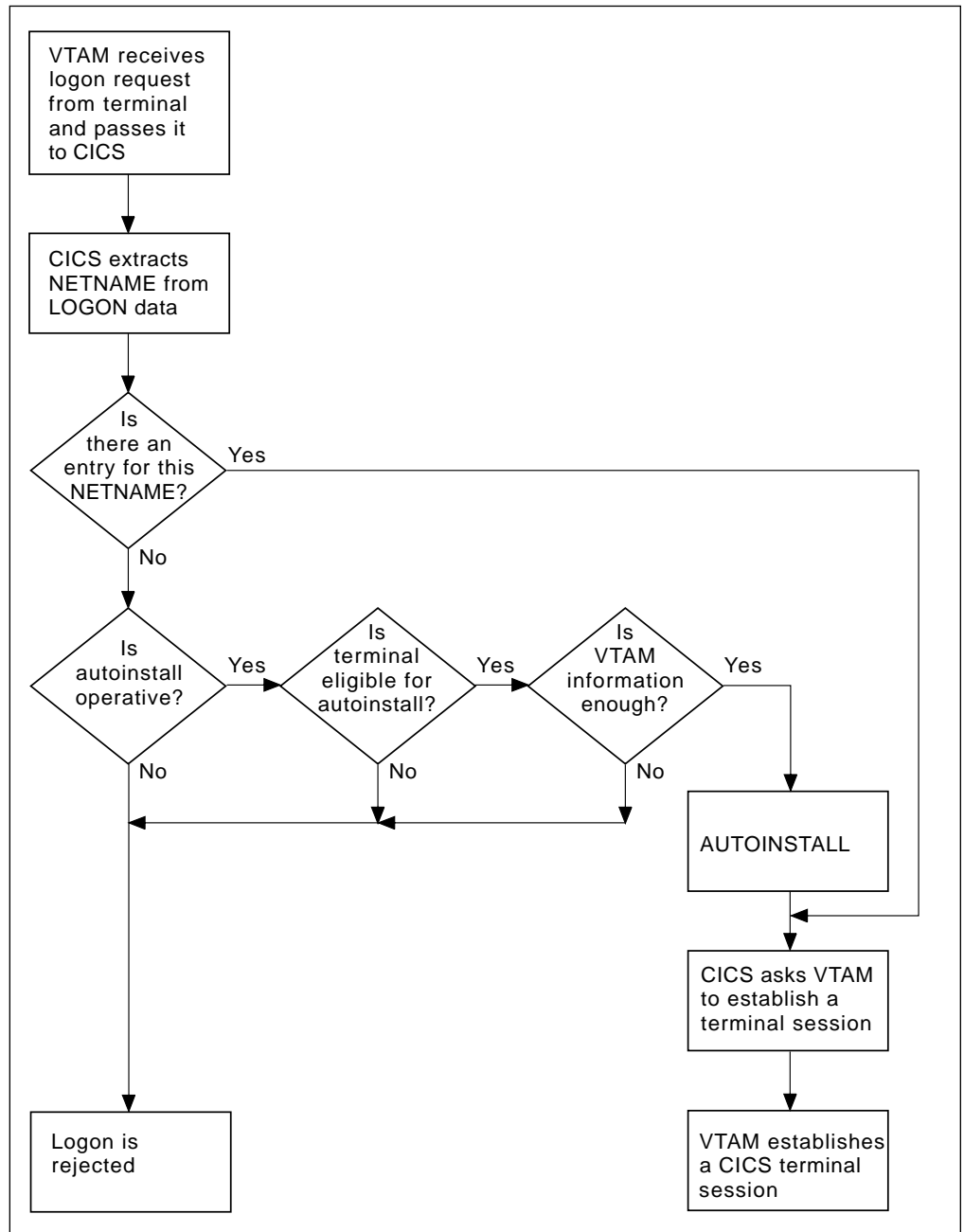


Figure 21. The process of logging on to CICS using autoinstall. Note that the autoinstall process itself is shown as a single box. What happens inside this box is depicted in Figure 22 on page 112.

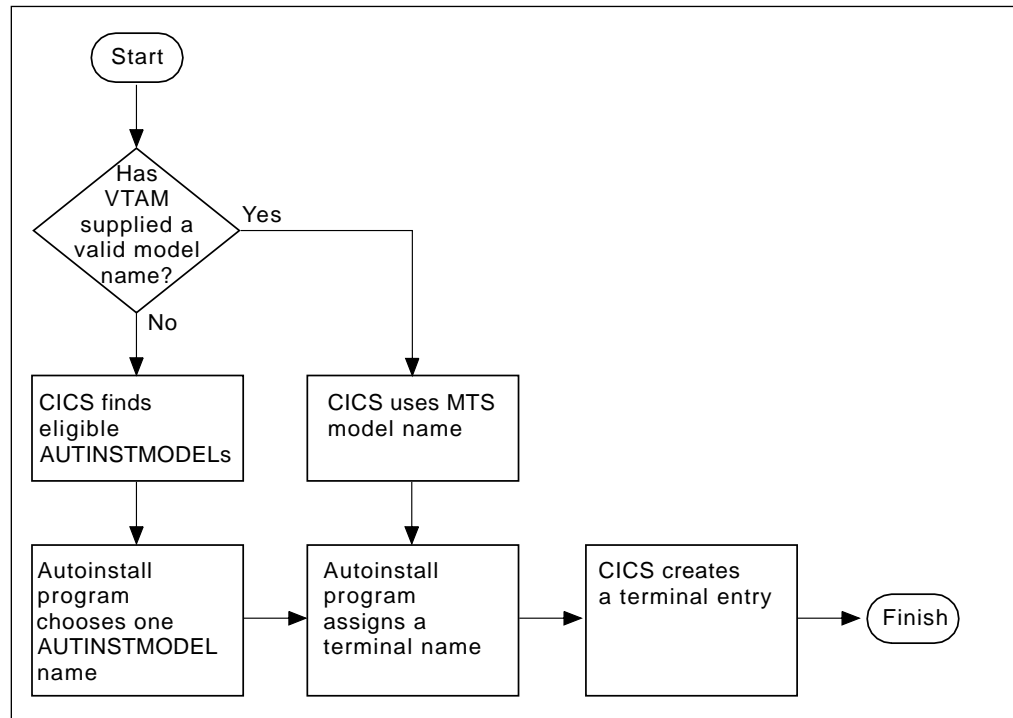


Figure 22. How CICS autoinstalls a terminal. Note that the overall process in which this fits is shown in Figure 21 on page 111.

What happens when the user logs off

When the terminal user finishes work and logs off:

1. CICS issues a CLSDST to ask VTAM to end the session.
2. CICS attempts to delete the TCT entry after a delay specified in the AILDELAY system initialization parameter.

Note that the TCT entry cannot be deleted if there is a lock effective at the time. For instance, CEMT INQUIRE TERMINAL or an outstanding ATI request locks the TCT entry.

3. CICS gives control to the autoinstall control program in case it has any further processing to do, for example, to free the TERMINAL name it gave to the terminal.

If the terminal's TYPETERM definition specifies SIGNOFF(LOGOFF) **and** the RACF segment specifies timeout, expiry of the timeout period causes the terminal to be logged off and the user to be signed off. After this automatic logoff, the delay period commences, leading to deletion of the TCT entry unless the terminal logs on again before the period ends. For details, see "Automatic signoff, logoff, and TCTTE deletion" on page 113.

Autoinstall and recovery and restart

This section explains what happens to autoinstalled terminal definitions at logoff and system restart times. It consists of:

- “What happens at CICS restart”
- “Automatic signoff, logoff, and TCTTE deletion”

What happens at CICS restart

At an **emergency restart**, autoinstalled TCT entries are recovered unless you specify a restart delay period of zero in the AIRDELAY system initialization parameter. This means that users can log on again after an emergency restart without going through the autoinstall process. Those terminals with AUTOCONNECT(YES) specified in their TYPETERM definition are automatically logged on during the restart process, without the need for operator intervention. The recovery of autoinstalled TCT entries avoids the performance impact of many concurrent autoinstall requests following a CICS restart. What happens is that a terminal user logging on after restart uses the TCT entry created for that terminal's NETNAME during the previous CICS run. It is just as if that terminal had an individual TERMINAL definition installed in the TCT.

Because this could pose a threat to security, CICS checks for operator activity after recovery. After a delay, all autoinstalled TCT entries that were recovered but are not in session again are deleted. As well as improving security, this ensures that CICS storage is not wasted by unused TCT entries. You can specify the length of the delay using the system initialization parameters.

If **persistent sessions** is in use and AIRDELAY is not equal to zero, autoinstalled TCT entries are treated exactly like other TCT entries. See the *CICS/ESA Recovery and Restart Guide* for more information about persistent sessions.

If **XRF** is in use, autoinstalled TCT entries are treated exactly like other TCT entries. That is, any TCT entry installed in the active CICS can be tracked, and a corresponding TCT entry is then installed in the alternate CICS. At an XRF takeover, for a terminal that is tracked, a TCT entry is already present in the new active CICS and the terminal is logged on. Therefore, the user does not have to go through the autoinstall process again.

At a **warm start**, TCT entries that were previously autoinstalled are lost, unless you logoff and CICS is shut down before the AIRDELAY time has expired.

If a TCTTE is recovered during emergency restart, a specification of AUTOCONNECT(YES) prevents deletion of the TCTTE by AIRDELAY. If you want the TCTTE storage to be deleted in this case, specify AUTOCONNECT(NO).

Automatic signoff, logoff, and TCTTE deletion

If a session ends through expiry of the user's TIMEOUT period, the terminal entry is deleted as described in the preceding section only if SIGNOFF(LOGOFF) is specified in the TYPETERM definition of the model. Table 6 on page 114 summarizes the automatic deletion and recovery of TCTTEs for autoinstalled terminals.

Table 9 on page 115 shows how automatic signoff, logoff, and TCTTE deletion occur if a session is timed out. Figure 24 on page 115 shows how logon and TCTTE deletion occur if, at a warm start or emergency restart, a TCTTE exists for an autoinstalled terminal.

Table 10 on page 115 shows how automatic TCTTE deletion occurs if a session ends for any reason other than timeout.

Autoinstall for VTAM terminals

CICS RUNNING	Restart delay = 0	TCTTE entries are not cataloged and therefore can never be recovered in a subsequent run.
	Restart delay > 0	TCTTE entries are cataloged. If they are not deleted during this run or at shutdown, they can be recovered in a subsequent emergency restart.
SHUTDOWN	Warm	Terminals are logged off and their TCTTEs are deleted, except for those terminals which were logged off before shutdown but whose AILDELAY has not expired.
	Immediate	No TCTTEs are deleted. All can be recovered in a subsequent emergency restart.
	Abnormal Termination	No TCTTEs are deleted. All can be recovered in a subsequent emergency restart.
STARTUP	Cold	No TCTTEs are recovered.
	Warm	No TCTTEs are recovered.
	Emergency Restart	<p>Restart Delay = 0. No TCTTEs are recovered (but see note in Table 9 on page 115). This session is unbound if it persists.</p> <p>Restart Delay > 0. TCTTEs are recovered. For details see Table 9 on page 115. This session is recovered if it persists.</p>

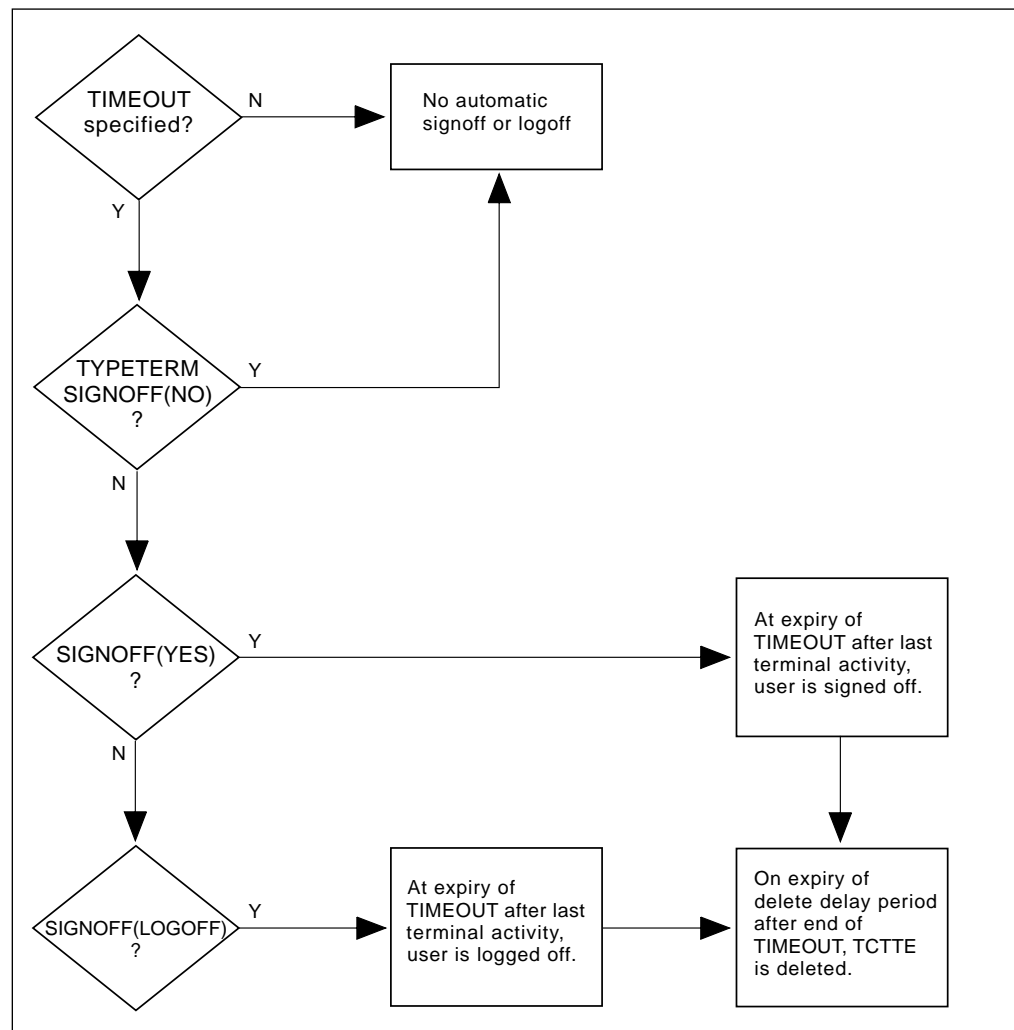


Figure 23. AUTOINSTALL - automatic signoff, logoff, and TCTTE deletion

<i>Table 7. Chronological sequence for SIGNOFF(YES)</i>		
SIGNOFF(YES) Last activity at terminal.	<----TIMEOUT period----> User can resume session at any time in this period.	User signed off.

<i>Table 8. Chronological sequence for SIGNOFF(LOGOFF)</i>				
SIGNOFF (LOGOFF) Last activity at terminal.	<---TIMEOUT period---> User can resume session at any time in this period.	User signed off, terminal logged off.	<--Delete delay period--> The terminal can log on during this period without repeating the autoinstall process.	TCTTE entry deleted

<i>Table 9. Chronological sequence for AUTOCONNECT(NO)</i>		
Emergency restart	<-----Restart delay period-----> For recovered autoinstalled TCTTE entries, the terminal can log on during this period without repeating the autoinstall process.	TCTTE entry deleted

<i>Table 10. AUTOINSTALL—automatic TCTTE deletion after non-automatic logoff</i>		
Non-automatic LOGOFF VTAM informs CICS of a session outage, terminal logs off, or transaction disconnects terminal	<-----Delete delay period-----> The terminal can log on during this period without repeating the autoinstall process.	TCTTE entry deleted

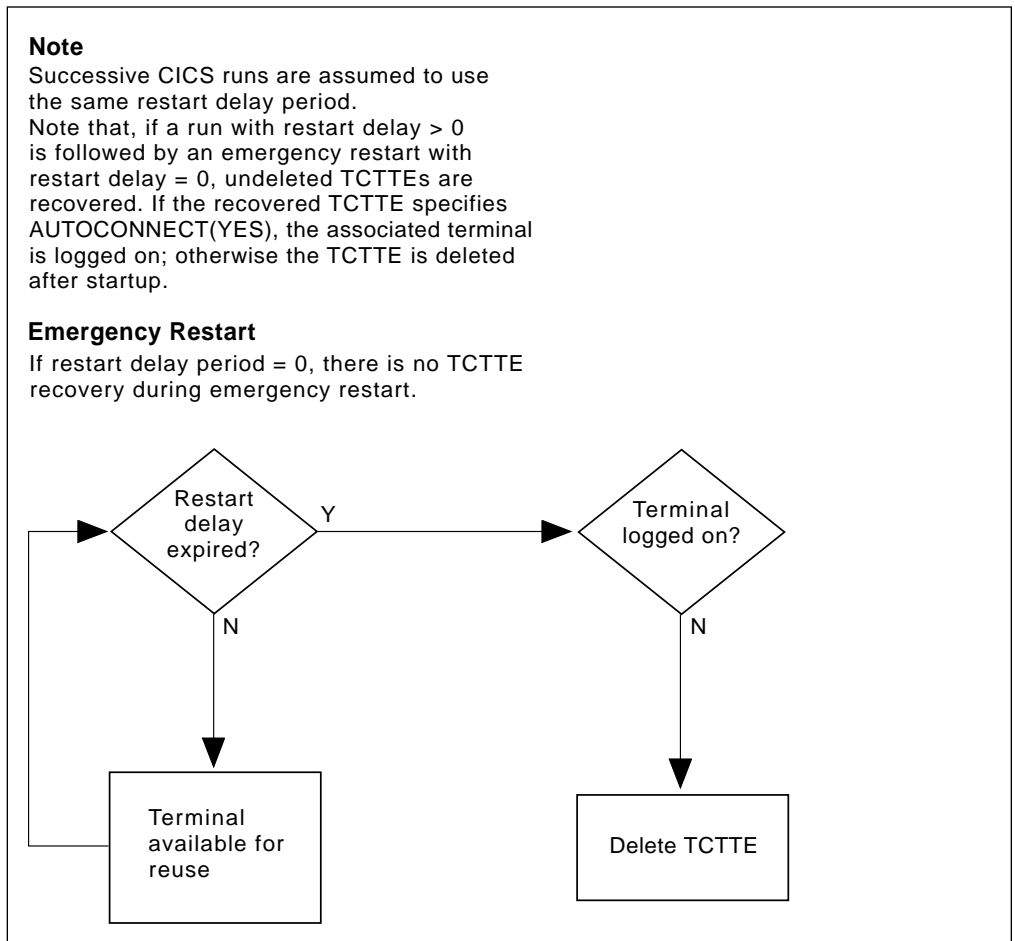


Figure 24. AUTOINSTALL - automatic logoff and TCTTE deletion after an emergency restart

The autoinstall control program

The autoinstall control program is invoked by CICS every time there is a valid request for a TCT entry to be autoinstalled, and every time an autoinstalled TCT entry is deleted.

For programming information about the autoinstall control program, see the *CICS/ESA Customization Guide*; this section is a summary of that information. It consists of:

- “Autoinstall with model terminal support”
- “Autoinstall program functions”

Autoinstall with model terminal support

If you are using VTAM 3.3 or later, you can define, in a VTAM table, the model name as well as other information for each terminal. During logon, VTAM sends this information to CICS, which may use it to create a TCTTE for the terminal. Therefore you do not need to code routines to select model terminal and optional printer and alternate printer, but your autoinstall control program still needs to provide a TERMINAL name. However, if you are already using an autoinstall control program that selects model terminal, optional printer, and alternate printer, CICS can continue to use this program.

Autoinstall program functions

If you are using a level of VTAM prior to VTAM 3.3, the NETNAME and other VTAM data about the terminal are not sufficient to build a TCT entry for the terminal. This program must create a CICS identifier for the TERMINAL and must choose a suitable model from among those passed to it by CICS.

IBM supplies a program (DFHZATDX) that performs the basic functions, but it may not perform all the functions that you require. For example, you may have your own conventions for TERMINAL names and their relationship to NETNAMEs. (Note that TERMINAL names are up to four characters long, and NETNAMEs are up to eight characters long, so that it is often not possible to derive one from the other.)

In addition, you could code your program to perform other functions associated with terminal logon and logoff. For example:

- Security checking
- Providing associated printer names
- Monitoring the number of terminals currently logged on through autoinstall.

The autoinstall control program runs in a transaction environment, rather than as a CICS exit. This means that you can read files and issue other CICS commands, to determine the TERMINAL name, or the associated PRINTER and ALTPRINTER names. The TCT entry, however, does not exist at either of the times that this program is invoked, because at logon it has not yet been created, and at logoff it has already been deleted. The program therefore runs in transaction-without-terminal mode.

You can write the autoinstall control program in any CICS language: assembler language, C/370, COBOL II, or PL/I. The CICS-supplied autoinstall program is available in all four languages; the assembler version is used by default. If you decide to write your own program, you can use one of the CICS-supplied programs as a pattern. Note that for COBOL II and C/370, you will need extra program definitions in the CSD. See the *CICS/ESA System Definition Guide* for details of defining the CSD.

You specify the name of the program you want to use in the AIXIT system initialization parameter.

When you test your autoinstall control program, you will find that the transient data destination CADL records each installation and each deletion of TCT entries. Message DFHZC6987 is useful for indicating which model came closest to being chosen, when a null list of models is passed to the autoinstall control program.

|
|
|
|
|
|
|

Note: You can have only one autoinstall control program active at one time for terminals and connections. The active program is specified on the AIEXIT system initialization parameter. The DFHZATDY program described in Chapter 10, “Autoinstall for APPC connections” on page 119 provides the same function for terminal autoinstall as DFHZATDX, but also provides function to autoinstall APPC connections initiated by BIND requests. Therefore, if you want to autoinstall APPC connections as well as terminals, you should use a customized version of DFHZATDY rather than DFHZATDX.

For programming information on implementing the CICS-supplied autoinstall control program, or designing and writing your own program, see the *CICS/ESA Customization Guide*.

Autoinstall for VTAM terminals

Chapter 10. Autoinstall for APPC connections

This chapter describes how to implement autoinstall for APPC parallel-session or single-session connections when a BIND is received from a primary. It consists of the following:

- “Getting started with APPC connection autoinstall”
- “Considerations for connection autoinstall” on page 120
- “Model definitions for connection autoinstall” on page 120
- “The autoinstall control program for connection autoinstall” on page 121
- “Connection autoinstall and recovery and restart” on page 122

If you have not used autoinstall for any resources before, read “Overview of autoinstall” on page 103 before going any further.

If autoinstall is enabled, and an APPC BIND request is received for an APPC service manager (SNASVCMG) session that does not have a matching CICS CONNECTION definition, or a BIND is received for a single session, a new connection is created and installed automatically.

Getting started with APPC connection autoinstall

1. Decide whether to use autoinstall for connections

You are most likely to benefit from autoinstall for connections if you have large numbers of workstations all with the same characteristics. The main **benefits** of using autoinstall for connections are that COLD, WARM, and EMERGENCY restarts are faster, you have fewer CSD definitions to manage, and less storage is taken up by unused definitions.

However, there are some possible **restrictions** that you should be aware of; these are discussed in “Security” on page 120 and “Connection autoinstall and recovery and restart” on page 122.

2. Decide which sessions to autoinstall

You can use autoinstall for CICS to CICS connections, but it is intended primarily for workstations.

3. Create your model connection definitions

The purpose of a model definition is to provide CICS with one definition that can be used for all connections with the same properties. See “Model definitions for connection autoinstall” on page 120 for information.

4. Design and write an autoinstall control program

The purpose of the autoinstall control program is to provide CICS with the extra information it needs to complete an autoinstall request, such as the autoinstall model name.

For programming information about the autoinstall control program, see the *CICS/ESA Customization Guide*. “The autoinstall control program for connection autoinstall” on page 121 provides a summary of that information.

5. Enable autoinstall for connections

Autoinstall for connections is enabled when you enable autoinstall for terminals; see page 106 for information about the system initialization (SIT) parameters and the CEMT and EXEC CICS INQUIRE and SET options used.

If terminal autoinstall has been enabled but you want to prevent autoinstall for connections, set the model connection out of service by using the CEMT or EXEC CICS SET CONNECTION(connection-name) OUTSERVICE command. (See the *CICS/ESA System Programming Reference* for programming information about this command.)

When the model connection is out of service, the autoinstall control program cannot access it and copy it, so the autoinstall function for connections is effectively disabled.

Considerations for connection autoinstall

This section offers some considerations which should help you in deciding whether to use autoinstall for connections. It covers the following subjects:

- “Security”
- “Model terminal support (MTS)”
- “Deletion of autoinstalled connections”

Security

Before using autoinstall for connections, you should consider whether the security considerations are suitable for your purposes.

The autoinstalled connection inherits the security attributes specified in the model. The security attributes from the CONNECTION definition are:

```
SECURITYNAME
ATTACHSEC
BINDSECURITY
```

If you are using compatibility mode to share a CSD with an earlier release of CICS, the BINDPASSWORD attribute is also inherited. See Chapter 12, “CONNECTION” on page 129 for information on these attributes.

From the SESSIONS definition, the autoinstalled connection inherits the preset security attribute USERID. See page 182 for a description of this attribute. * If you are attempting to autoinstall an attachsec identify connection from a CICS system prior to CICS/ESA 4.1 or you are attempting to autoinstall an attachsec verify connection from a non-EBCDIC based system, then you should refer to 'Attach Time Security and the USEDFTUSER option' in chapter 12 of the *CICS/ESA CICS-RACF Security Guide*.

Model terminal support (MTS)

MTS is not supported for connection autoinstall. This means that you must code the routines yourself to select the model definition name; you cannot get VTAM to do it for you. MTS is described on page 116.

Deletion of autoinstalled connections

Unlike autoinstalled terminal definitions, autoinstalled connection definitions are **not** deleted when they are no longer in use. This means that they continue to occupy storage.

Model definitions for connection autoinstall

Model definitions for connections autoinstall are different from those for terminal autoinstall in that they do not have to be defined explicitly as models. Any installed connection definition can be used as a “template” for an autoinstalled connection.

For performance reasons you should use an installed connection definition that is not otherwise in use. The definition is locked while CICS copies it, and if you have a very large number of sessions autoinstalling, the delay may be noticeable.

You can set the model connection definition out of service by using the CEMT or EXEC CICS SET CONNECTION OUTSERVICE command. This effectively disables autoinstall for connections, since the autoinstall control program cannot access and copy the model definition.

For CICS-to-CICS connections autoinstall, the MAXIMUM attribute in the SESSIONS definition of the model connection should be large enough to accommodate the largest device using the model. If not, the lower of the two is used.

When creating model connections, you must ensure that the usergroup modenames specified in the session's MODENAME field match the usergroup modenames in the connection to be autoinstalled.

The autoinstall control program for connection autoinstall

The autoinstall control program is invoked at installation for:

- APPC parallel-session connections initiated by a BIND
- APPC single-session connections initiated by a BIND.

The purpose of the autoinstall control program is to provide CICS with any extra information it needs to complete an autoinstall request. For APPC parallel sessions, the control program provides a SYSID for the new definition.

When an APPC BIND request is received by CICS, CICS receives the partner's VTAM NETNAME and passes it to the autoinstall control program. The control program uses the information contained in the partner's NETNAME and in the VTAM BIND to select the most appropriate model on which to base a new connection. In order to return the name of the most suitable model to CICS, the control program must know the NETNAME or SYSID of every model.

CICS supplies a sample control program, DFHZATDY, for connections autoinstall. You can use DFHZATDY unchanged if both of the following conditions are met:

- Your model connections are called CCPS, CBPS, or CBSS
- You use the last four characters of the NETNAME as the SYSID or terminal name.

If not, you will have to change DFHZATDY to suit your installation. Its source is supplied in CICS410.SDFHSAMP. DFHZATDY is defined as follows:

```
DEFINE PROGRAM(DFHZATDY) GROUP(DFHAI62) LANGUAGE(ASSEMBLER)
    RELOAD(NO) RESIDENT(NO) STATUS(ENABLED) CEDF(NO)
    DATALOCATION(ANY) EXECKEY(CICS)
```

The definitions for the supplied model connections and sessions are:

```
DEFINE CONNECTION(CBPS) GROUP(DFHAI62) NETNAME(TMPLATE1)
    ACCESSMETHOD(VTAM) PROTOCOL(APPC) SINGLESESS(NO)
```

```
DEFINE SESSION(CBPS) GROUP(DFHAI62) CONNECTION(CBPS)
    MODENAME(LU62PS) PROTOCOL(APPC) MAXIMUM(10,5)
```

```
DEFINE CONNECTION(CBSS) GROUP(DFHAI62) NETNAME(TMPLATE2)
    ACCESSMETHOD(VTAM) PROTOCOL(APPC) SINGLESESS(YES)
```

```
DEFINE SESSION(CBSS) GROUP(DFHAI62) CONNECTION(CBSS)
    MODENAME(LU62SS) PROTOCOL(APPC) MAXIMUM(1,0)
```

```
DEFINE CONNECTION(CCPS) GROUP(DFHAI62) NETNAME(TMPLATE3)
    ACCESSMETHOD(VTAM) PROTOCOL(APPC) SINGLESESS(NO)
```

```
DEFINE SESSION(CCPS) GROUP(DFHAI62) CONNECTION(CCPS)
    MODENAME(LU62PS) PROTOCOL(APPC) MAXIMUM(10,5)
```

If you want to use these definitions, you must add group DFHAI62 to your group list.

Warning: Do not try to use the terminal autoinstall exit DFHZATDX to autoinstall connections; any sessions installed by DFHZATDX are terminated and message DFHZC6921 is issued.

For programming information on customizing the autoinstall control program, see the *CICS/ESA Customization Guide*.

Connection autoinstall and recovery and restart

This section explains how autoinstalled connections are handled in different restart situations.

Persistent sessions support: Because autoinstalled connections are not cataloged, they can not benefit from persistent sessions support. This means that when a session does persist, any autoinstalled connections relating to it are unbound.

XRF support: Autoinstalled connections are tracked to XRF.

Autoinstalled connections are recovered only at a COLD start of CICS; they are *not* recovered at WARM and EMERGENCY restarts.

Unit-of-recovery descriptors (URDs) for autoinstalled connections are recovered after COLD, WARM, and EMERGENCY restarts, as long as the SYSIDs of the connections are consistent.

The SYSIDs of the autoinstalled connections must be consistent if you are using recoverable resources.

Chapter 11. Autoinstall for programs, mapsets, and partitionsets

This chapter describes how to implement autoinstall for programs, mapsets, and partitionsets. It consists of the following:

- “Getting started with program autoinstall”
- “Considerations for program autoinstall” on page 124
- “Model definitions for program autoinstall” on page 125
- “The autoinstall control program for program autoinstall” on page 125
- “Program autoinstall and recovery and restart” on page 126

If you have not used autoinstall for any resources before, read “Overview of autoinstall” on page 103 before going any further.

If autoinstall for programs is active, and an implicit or explicit load request is issued for a previously undefined program, mapset, or partitionset, CICS dynamically creates a definition and installs it and catalogs it as appropriate. An implicit or explicit load occurs when:

- CICS starts a transaction
- An application program issues one of the following commands:
 - EXEC CICS LINK
 - EXEC CICS XCTL
 - EXEC CICS LOAD
 - EXEC CICS ENABLE (for global user exit, or task-related user exit, program)
- When, after an EXEC CICS HANDLE ABEND PROGRAM(...), a condition is raised and CICS transfers control to the named program
- CICS calls a user-replaceable module for the first time
- An application program issues one of the following commands:
 - EXEC CICS RECEIVE or SEND MAP
 - EXEC CICS SEND PARTNSET
 - EXEC CICS RECEIVE PARTN

Getting started with program autoinstall

1. Decide whether your programs eligible for autoinstall

Programs that **cannot** be autoinstalled are:

- The program autoinstall control program
- The terminal autoinstall control program
- The connection autoinstall control program.

All other programs can be autoinstalled, including:

- All other user-replaceable programs
- Global user exits (GLUEs) and task-related user exits (TRUEs).
- PLT programs.

User-replaceable programs and PLT programs are autoinstalled on first reference. GLUEs and TRUEs are autoinstalled when enabled.

2. Decide whether to use autoinstall for programs

Using autoinstall for programs can save time spent on defining individual programs, mapsets, and partitionsets. Savings can also be made on storage, as the definitions of autoinstalled resources do not occupy space until they are referenced.

Use of autoinstall for programs can reduce the number of definitions to be installed on a COLD start, thereby reducing the time taken.

3. Decide which programs to autoinstall

Depending on your programs, you can choose to use a mixture of RDO and autoinstall. A suggested way to manage program autoinstall is to continue to use RDO for existing program definitions and for installing groups containing related programs. Use autoinstall as you develop and install new applications, and in test environments, where you might otherwise install large numbers of programs at CICS startup.

4. Enable autoinstall for programs

You can enable autoinstall for programs either by using the system initialization table (SIT) or by using the EXEC CICS or CEMT INQUIRE|SET SYSTEM command.

There are three system initialization parameters relating to program autoinstall:

PGAICTLG This specifies whether autoinstalled program definitions should be cataloged. See “Considerations for program autoinstall” for guidance on using this parameter.

PGAIPGM This specifies whether the program autoinstall function is active or inactive.

PGAEXIT This specifies the name of the program autoinstall exit.

For information on how to specify these parameters in the SIT, see the *CICS/ESA System Definition Guide*.

There are three options relating to program autoinstall on the INQUIRE|SET SYSTEM command:

PROGAUTOCTLG This specifies whether autoinstalled program definitions are to be cataloged. See “Considerations for program autoinstall” for guidance on using this option.

PROGAUTOINST This specifies whether the program autoinstall function is active or inactive.

PROGAUTOEXIT This specifies the name of the program autoinstall exit.

For programming information on how to code EXEC CICS commands, see the *CICS/ESA System Programming Reference*, and for information on how to use CEMT, see the *CICS/ESA CICS-Supplied Transactions*.

5. Create your model program definitions

The purpose of a model definition is to provide CICS with one definition that can be used for all programs with the same properties. See “Model definitions for program autoinstall” on page 125 for further information.

6. Design and write an autoinstall control program

The purpose of the autoinstall control program is to provide CICS with the extra information it needs to complete an autoinstall request, such as the autoinstall model name. You can write your autoinstall program in any language supported by CICS, with full access to the CICS application programming interface. See “The autoinstall control program for program autoinstall” on page 125 for further information.

Considerations for program autoinstall

Cataloging You can specify whether an autoinstalled program definition is cataloged or not (that is, whether the definition is retained over a warm or emergency start) by using either the PGAICTLG option on the system initialization table or by using the PROGAUTOCTLG option on the CEMT INQUIRE|SET SYSTEM command. The values you can specify are as follows:

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.

The effects of specifying cataloging for program autoinstall apply mainly to recovery and restart. See "Program autoinstall and recovery and restart" on page 126 for information.

Model definitions for program autoinstall

Model definitions for program autoinstall are different from those for terminal autoinstall in that they do not have to be defined explicitly as models. Any installed program, mapset, or partitionset definition can be used as a "template" for program autoinstall.

If you do not want to use your own definitions, you can use the CICS-supplied model definitions. These are:

- DFHPGAPG for programs
- DFHPGAMP for mapsets
- DFHPGAPT for partitionsets.

They are listed in Appendix B, "CICS-supplied resource definitions, groups, and lists" on page 347.

APAR PN76605

changed the following note to read:

Note: Although the DFHPGAPG definition does not specify a program language, the CICS autoinstall routine detects the program language from the program being loaded.

The autoinstall control program for program autoinstall

You specify the name of the control program you want to use in the PGAEXIT system initialization parameter, or use the CEMT or EXEC CICS INQUIRE|SET SYSTEM command.

For detailed programming information about the autoinstall control program for programs, see the *CICS/ESA Customization Guide*; this section is a summary of that information.

When it is invoked

For **programs**, the autoinstall control program is invoked when:

- Any of these commands references a previously undefined program:
 - EXEC CICS LINK
 - EXEC CICS XCTL
 - EXEC CICS LOAD
- The program is the first program in a transaction
- An EXEC CICS ENABLE is issued for a GLUE or a TRUE
- An abend occurs after an EXEC CICS HANDLE ABEND PROGRAM command is issued and CICS invokes the named program
- CICS calls a user-replaceable module.

For **mapsets**, the autoinstall control program is invoked when an EXEC CICS SEND MAP or EXEC CICS RECEIVE MAP refers to a previously undefined mapset.

Autoinstall for programs, mapsets, and partitionsets

For **partitionsets**, the autoinstall control program is invoked when an EXEC CICS SEND PARTNSET or EXEC CICS RECEIVE PARTN command refers to a previously undefined partitionset.

Sample programs

The following sample programs are supplied by CICS:

- DFHPGADX — assembler program for program autoinstall exit
- DFHPGAHX — C program for program autoinstall exit
- DFHPGALX — PL/I program for program autoinstall exit
- DFHPGAOX — COBOL definition for program autoinstall exit.

The source for these programs is supplied in the CICS410.SDFHSAMP sample library, but only the assembler version is supplied in executable form, in the CICS410.SDFHLOAD load library.

Autoinstall program functions

The program autoinstall facility uses model definitions, together with a user-replaceable module, to create explicit definitions for programs, mapsets, and partitionsets that need to be autoinstalled. CICS calls the user-replaceable module, with a parameter list that gives the name of the appropriate model definition. On return from the user-replaceable module (depending on the return code), CICS creates a resource definition from information in the model and from parameters returned by the user-replaceable module.

Note: CICS does not call the URM for any CICS programs, mapsets, or partitionsets (that is, any objects beginning with the letters DFH).

For programming information on how to code a program autoinstall URM, see the *CICS/ESA Customization Guide*.

Program autoinstall and recovery and restart

There is a difference in performance between warm and emergency restarts using program autoinstall without cataloging, and warm and emergency restarts using autoinstall with cataloging. (See the *CICS/ESA Recovery and Restart Guide* for information on cataloging.)

If you are using autoinstall with cataloging, restart times are similar to those of restarting a CICS region that is not using program autoinstall. This is because, in both cases, resource definitions are reinstalled from the catalog during the restart. The definitions after the restart are those that existed before the system was terminated.

If you are using autoinstall without cataloging, CICS restart times are improved because CICS does not install definitions from the CICS global catalog. Instead, definitions are autoinstalled as required whenever programs, mapsets, and partitionsets are referenced following the restart.

Part 4. RDO resource types and their attributes

This section describes the resource types that you can define, manage, and install using RDO commands. The resource types are in alphabetical order.

For each resource type, we show the panel(s) that you get when you type a **DEFINE** command correctly, followed by a resource type, and with no resulting error messages. For example:

```
DEFINE FILE(FILEB) GROUP(MYGROUP)
```

We have cut out everything but the information area, and joined the panels together when there is more than one.

- Attributes that you **must** fill in are highlighted.
- Capital letters indicate the minimum command line abbreviation for each keyword.

APAR PN74913

added the following note.

Note: CICS may add new resource types and attributes across different releases. For similar named fields, such as Transaction and Tranclass, the minimum abbreviation may have to change from release to release. For example, at CICS/ESA R330 the minimum abbreviation for Transaction was TR, whereas at CICS/ESA R410 it is now TRANS, to differentiate between that and TRANC for Tranclass.

RDO commands are validated online by the CEDA/B/C transactions, and confirmation for such ambiguities is required before the transaction can continue. This is not possible for input to DFHCSDUP however. To avoid batch operations failing due to the minimum abbreviation altering across CICS releases, it is recommended that the full resource type and attribute names are used when coding input to DFHCSDUP. While CICS cannot guarantee upward compatibility of the abbreviations, it always provides such compatibility for the full names of the fields.

- Dots indicate the maximum length of each name.
- The choice of possible values is shown on the right. (The basic default values are shown following the ==>, but note that some default values may differ, depending on other values that you type in.)
- Where you see a colon (:) instead of the ==>, this indicates that the keyword is for *compatibility mode*. See "Compatibility mode (CSD sharing)" on page 19 for further information.

Following the panel for each resource type, there is an alphabetical list of the attributes, with an explanation of each attribute and the possible values that it can take.

If you are familiar with resource definition macro operands, you may also want to refer to Appendix C, "Keyword cross-reference tables" on page 377.

Resource types and their attributes

Chapter 12. CONNECTION

A CONNECTION is the definition of a remote system that your CICS system will communicate with using ISC or MRO. When the CONNECTION definition is installed in the CICS system, this information is stored in the terminal control table (TCT) as a system entry (TCTSE).

When you define a CONNECTION, you give enough information to identify the system and specify its basic attributes. You put details about the sessions you will use to communicate with the system in the SESSIONS definition. CICS uses the CONNECTION name, to identify the other system when the definition has been installed. For other CICS systems connected via MRO, this name is typically the same as that specified in the other CICS system as the SYSIDNT system initialization (SIT) parameter. For other systems connected via ISC, this name is typically based on an acronym that describes the location of or the organization that owns the system (for example, USA1 or IBMC).

The REMOTESYSTEM name on a TRANSACTION definition, or on a TERMINAL definition, refers to a CONNECTION definition through its CONNECTION name. These attributes are used for transaction routing.

The REMOTESYSTEM name on a PROGRAM definition refers to a CONNECTION definition through its CONNECTION name. This attribute is used for Distributed Program Link.

The CONNECTION definition does **not** name associated SESSIONS.

| Installing CONNECTION definitions

| To install **new** CONNECTION definitions, put them in a group of their own which does not
| contain CONNECTION definitions that have already been installed, then use CEDA INSTALL
+ to install the whole group. You cannot install single CONNECTION definitions.

| To modify and reinstall existing MRO CONNECTION definitions, or if you want new and
| existing MRO CONNECTION definitions to be in the same group, you must close down all
| interregion communication (IRC) and open it again, before you can use the definition. If you
| do not close IRC and open it again, you get message DFHIR3788 when you try to bring up
| the region with the new connection. You should adopt the following procedure:

1. Close IRC down:

```
CEMT SET IRC CLOSED
```

2. Install the resource definitions:

```
CEDA INSTALL GROUP(groupname)
```

3. When you have successfully installed the group containing the definitions, open IRC again:

```
CEMT SET IRC OPEN
```

Definitions for various links and sessions

Here is some guidance on creating resource definitions for:

- MRO links and sessions
- APPC (LUTYPE6.2) links and parallel sessions
- APPC (LUTYPE6.2) single session terminals
- LUTYPE6.1 links and sessions
- LUTYPE6.1 CICS-IMS links and sessions
- INDIRECT connections

MRO links and sessions

You define an MRO link using one CONNECTION definition, and its associated parallel sessions using one SESSIONS definition.

ACCESSMETHOD

On the CONNECTION definition, you specify this as IRC (for interregion communication). You can specify XM (for cross-memory services) instead. IRC is used to open and close the links.

PROTOCOL

On the SESSIONS definition you specify LU61 as the PROTOCOL. On the CONNECTION definition you leave the PROTOCOL value blank.

SENDPFX

SENDCOUNT

RECEIVEPFX

RECEIVECOUNT

In one SESSIONS definition you specify a number of send sessions and a number of receive sessions. The values that you specify in these attributes are used to determine the names of the TCT entries created when the definition is installed. (See the section on installing "MRO links and sessions" on page 175.) *

#

USEDFTUSER

If you have a connection or APPC terminal definition to an earlier CICS release or CICS on another platform then you may need to code this as USEDFTUSER(YES), See 'Attach Time Security and the USEDFTUSER option' in chapter 12 of the *CICS/ESA CICS-RACF Security Guide*.

#

#

#

#

#

APPC links and parallel sessions

For APPC, the sessions are grouped into modesets. You define each modeset with a SESSIONS definition, so you will have as many SESSIONS definitions as you require modesets. You define the link as a CONNECTION definition.

The following attributes are significant:

ACCESSMETHOD

On the CONNECTION definition, you specify this as VTAM.

PROTOCOL

On both the CONNECTION and SESSIONS definitions, you specify APPC as the protocol.

MODENAME

On the SESSIONS definition for each modeset, you name the modeset with the MODENAME. This is the name by which the modeset will be known to CICS when the definition is installed in the active system.

MAXIMUM
 # You use this to control the number of sessions in the modeset. *

USEDFLTUSER
 # If you have a connection or APPC terminal definition to an earlier CICS release or CICS
 # on another platform then you may need to code this as USEDFLTUSER(YES), See
 # 'Attach Time Security and the USEDFLTUSER option' in chapter 12 of the *CICS/ESA*
 # *CICS-RACF Security Guide*.

APPC (LUTYPE6.2) single session terminal

| You can define an APPC terminal as a CONNECTION-SESSIONS pair or as a
 | TERMINAL-TYPETERM pair. The TERMINAL-TYPETERM method is described on page
 | 188.

| If you want to use the CONNECTION-SESSIONS method, the following attributes are
 | significant:

ACCESSMETHOD

On the CONNECTION definition, you specify this as VTAM.

PROTOCOL

On both the CONNECTION and SESSIONS definitions, you specify APPC as the
 protocol.

SINGLESESS

YES indicates that the CONNECTION definition is for a single session terminal.

MODENAME

On the SESSIONS definition, you specify the MODENAME. This is the name that CICS
 uses to identify the session when the definition is installed in the active system.

MAXIMUM

| For a single session terminal, specifying 1,0 or 1,1 has the same effect. (For further
 | information, see page 178.) *

USEDFLTUSER
 # If you have a connection or APPC terminal definition to an earlier CICS release or CICS
 # on another platform then you may need to code this as USEDFLTUSER(YES), See
 # 'Attach Time Security and the USEDFLTUSER option' in chapter 12 of the *CICS/ESA*
 # *CICS-RACF Security Guide*.

LUTYPE6.1 links and sessions

LUTYPE6.1 links and sessions can be defined in one of two ways:

- In one CONNECTION and one SESSIONS definition
- In one CONNECTION and a number of SESSIONS definitions: one for each session needed.

If your sessions are all to have **identical** attributes, define each link in one CONNECTION
 definition and all its associated sessions in one SESSIONS definition.

ACCESSMETHOD

On the CONNECTION definition, you specify this as VTAM.

PROTOCOL

On the SESSIONS definition and on the CONNECTION definition, you specify this as
 LU61.

CONNECTION

SENDPFX
SENDCOUNT
RECEIVEPFX
RECEIVECOUNT

These attributes are used as for MRO links and sessions.

If your sessions are to have *different* attributes from each other, you must create a separate SESSIONS definition for each one. With the exception of NETNAMEQ, this method is the same as that for CICS-IMS sessions, described below.

Note: For CICS-CICS ISC links and sessions, you are recommended to use APPC rather than LUTYPE6.1.

LUTYPE6.1 CICS-IMS links and sessions

IMS needs each session to be defined in a separate SESSIONS definition, because each session must have a different NETNAMEQ.

You define the link as a CONNECTION definition, and create a number of SESSIONS definitions: one for each SEND session and one for each RECEIVE session.

ACCESSMETHOD

On the CONNECTION definition, you specify this as VTAM.

PROTOCOL

On both the CONNECTION and SESSIONS definitions, you specify LU61 as the protocol.

SESSNAME

This is the name that CICS uses to identify the session when the definition is installed in the active system.

NETNAMEQ

This is the name that the remote IMS system uses to identify the session.

SENDCOUNT

RECEIVECOUNT

You use these attributes to specify whether a session is a SEND session or a RECEIVE session.

A SEND session is one in which the local CICS is the secondary and is the contention winner. Specify it by defining SENDCOUNT(1) and leaving RECEIVECOUNT to default to blank.

A RECEIVE session is one in which the local CICS is the primary and is the contention loser. It is specified by defining RECEIVECOUNT(1) and leaving SENDCOUNT to default to blank. (You do not need to specify a SENDPFX or a RECEIVEPFX.)

INDIRECT connections

An INDIRECT connection is a remote system for which you have not defined a direct link with the local system. Instead, the two systems communicate with each other by way of one or more intermediate systems. You can use this method for transaction routing: the remote system, indirectly connected, is always the terminal-owning region; the local system is always the application-owning region or an intermediate region on the transaction routing path.

Important

In releases of CICS before 4.1, indirect connections were required for transaction routing across intermediate systems. They were needed on the application-owning region and on intermediate regions, to:

- Identify the NETNAME of the terminal-owning region
- Identify the next system in the path to the terminal-owning region.

In CICS/ESA 4.1, indirect connections are required only if you use non-VTAM terminals for transaction routing across intermediate systems. Optionally, you can use them with VTAM terminals, where several transaction routing paths are possible, to identify the preferred path to the terminal-owning region. For information about why you might want to define indirect connections, and about the resource definitions required for transaction routing, see the *CICS/ESA Intercommunication Guide*.

Defining indirect connections

In the local system, you must have ordinary CONNECTION and SESSIONS definitions for the intermediate systems to which you are directly connected. The ACCESSMETHOD should be IRC or XM with PROTOCOL(LU61), or VTAM with PROTOCOL(APPC).

For the INDIRECT connection (also known as an indirect link or an indirect system) you need, in the local system, a CONNECTION definition only. You do not need a SESSIONS definition: the sessions that are used are those of the intermediate system. The following attributes of the CONNECTION definition are significant:

ACCESSMETHOD

You should specify this as INDIRECT.

NETNAME

You should name the APPLID of the terminal-owning system.

INDSYS

You should name the CONNECTION definition for the MRO or APPC link that is the start of a path to the terminal-owning system.

More information about defining intercommunication resources

Before you start creating definitions for these resources, see the *CICS/ESA Intercommunication Guide* for further guidance. There you will find many useful examples of the attributes you must specify for different types of links and sessions.

Defining a CONNECTION

```

Connection      :
Group           :
Description     ==>
CONNECTION IDENTIFIERS
Netname        ==>
INDsys         ==>
REMOTE ATTRIBUTES
REMOTESystem   ==>
REMOTESysnet   ==>
REMOTENAME     ==>
CONNECTION PROPERTIES
Accessmethod   ==> Vtam          Vtam | IRc | INdirect | Xm
PRotocol       ==> Appc          Appc | Lu61 | Exci
Conntype       ==>                Generic | Specific
SInglesess     ==> No           No | Yes
DATastream     ==> User         User | 3270 | SCs | STRfield | Lms
RECORDformat   ==> U           U | Vb
Queuelimit     ==> No           No | 0-9999
OPERATIONAL PROPERTIES
AUtoconnect    ==> No           No | Yes | All
INService      ==> Yes         Yes | No
SECURITY
SEcurityname   ==>
ATTachsec      ==> Local       Local | Identify | Verify | Persistent
| Mixidpe
BINDPasswd     :
BINDSecurity   ==> No           No | Yes
*
Usedfltuser    ==> No           No | Yes

RECOVERY
PSrecovery     ==> Sysdefault  Sysdefault | None
    
```

Figure 25. The DEFINE panel for CONNECTION

ACCESSMETHOD({VTAM|IRC|INDIRECT|XM})

Indicates the access method to be used for this connection.

VTAM

Specifies that communication between the local CICS region and the system defined by this CONNECTION is through VTAM. You can use VTAM intersystem communication (ISC) for systems that are in different MVS images or in different address spaces in the same MVS image.

IRC

Specifies that communication between the local CICS region and the region defined by this CONNECTION is through the interregion communication (IRC) program DFHIRP. You can use IRC for multiregion operation (MRO) for regions that are in the same MVS image or in different MVS images **within a sysplex**.

INDIRECT

Communication between the local CICS system and the system defined by this CONNECTION will be through the system named in the INDSYS operand.

XM

Specifies that MRO communication between the local CICS region and the region defined by its CONNECTION definition uses MVS cross-memory services. You can use XM for multiregion operation for regions that are in the same MVS image, or in different MVS images **within a sysplex**.

Note: The CICS type 3 SVC is still required with XM because DFHIRP is used when the link is opened. For further information about SVCs, see the *CICS/ESA Installation Guide*.

MVS cross-memory services are used only if the ACCESSMETHOD of the other end of the link is also defined as XM.

MRO in a sysplex

If the MRO partners reside in different MVS images within a sysplex, and the CONNECTION specifies IRC or XM, CICS automatically uses XCF as the access method, and ignores the IRC or XM specification.

Note that you cannot define XCF explicitly; if you want to use XCF, you must specify IRC or XM. See the *CICS/ESA Intercommunication Guide* for more information about XCF.

ATTACHSEC({LOCAL|IDENTIFY|VERIFY|PERSISTENT|MIXIDPE})

Indicates the level of attach-time user security required for the connection.

LOCAL

The authority of the user is taken to be that of the link itself, and you will be relying on link security alone to protect your resource. If the PROTOCOL attribute on the CONNECTION definition is LU6.1, you must specify LOCAL.

IDENTIFY

Incoming attach requests must specify a user identifier. You should enter IDENTIFY when the connecting system has a security manager, for example, if it is another CICS system.

VERIFY

Incoming attach requests must specify a user identifier and a user password. You should enter VERIFY when the connecting system has no security manager and hence cannot be trusted.

You should not specify VERIFY for CICS to CICS communication, because CICS does not send passwords.

PERSISTENT

Incoming attach requests must specify a user identifier and a user password on the first attach request. Subsequent attach requests require only the user identifier. This should be used only between a PWS (programmable workstation, for example, an IBM Personal System/2) and CICS.

MIXIDPE

Supports incoming attach requests which may be using either or both IDENTIFY or PERSISTENT security types. The security type actually used will depend on the incoming attach request.

AUTOCONNECT({NO|YES|ALL})

For systems using ACCESSMETHOD(VTAM), you specify with AUTOCONNECT(YES) or (ALL) that sessions are to be established (that is, BIND is to be performed). Such sessions are set up during CICS initialization, or when you use the CEMT or EXEC CICS SET VTAM OPEN command to start communication with VTAM. If the connection cannot be made at these times

because the remote system is unavailable, you must subsequently acquire the link by using the CEMT or EXEC CICS SET CONNECTION(sysid) INSERVICE ACQUIRED command, unless the remote system becomes available in the meantime and itself initiates communications.

What you have to specify for APPC and LU6.1 connections is discussed below:

APPC

For CONNECTIONS with SINGLESESS(NO) specified, CICS tries to bind, on system start-up, the LU services manager sessions in mode group SNASVCMG.

For CONNECTIONS with SINGLESESS(YES) specified, the AUTOCONNECT operand is ignored. You should instead use the AUTOCONNECT operand of the SESSIONS definition.

NO

CICS will not attempt to bind sessions when the connection is established.

YES

CICS will attempt to bind only contention winning sessions when the connection is established.

ALL

On this definition, ALL is equivalent to YES, but you can specify ALL to be consistent with the SESSIONS definition.

+ You should not specify ALL for connections to
 + other CICS systems. This can lead to a bind
 + race condition, and is discussed in the
 + *CICS/ESA Intercommunication Guide*.

LU6.1

This option is not applicable on an LU6.1 CONNECTION.

Specify AUTOCONNECT(YES) on the SESSIONS if you want the connection to be established at initialization or CEDA install.

Specify AUTOCONNECT(NO) on the SESSIONS if you don't want the connection to be established at initialization or CEDA install.

BINDPASSWORD(password) (APPC only)

The BINDPASSWORD keyword is not valid for CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

BINDSECURITY({NO|YES}) (APPC only)

This indicates whether an ESM is being used for bind-time security.

No No external bind-time security is required. However, if you define a password on the BINDPASSWORD parameter, CICS defaults to using its own bind security checking.

CONNECTION

Yes

If security is active and the XAPPC system initialization parameter is set to YES, then CICS will attempt to extract the session key from RACF in order to perform bind-time security. If no RACF profile is available, the bind fails.

CONNECTION(name)

The name of this CONNECTION definition. The name can be up to four characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. The name should be unique within Connections and Terminals (for more detail, see page 201).

This is the name specified as REMOTESYSTEM on FILE, TERMINAL, TRANSACTION, and PROGRAM definitions.

CONNTYPE({SPECIFIC|GENERIC})

For external CICS interface connections, indicates the nature of the connection.

SPECIFIC

The connection is for communication from a non-CICS client program to the CICS region, and is specific. A specific connection is an MRO link with one or more sessions dedicated to a single user in a client program. For a specific connection, NETNAME is mandatory.

GENERIC

The connection is for communication from a non-CICS client program to the CICS system, and is generic. A generic connection is an MRO link with a number of sessions to be shared by multiple EXCI users. For a generic connection you cannot specify the NETNAME attribute.

DATASTREAM({USER|3270|SCS|STRFIELD|LMS})

The type of data stream.

USER

Let DATASTREAM default to USER, if the data stream is user-defined. If you are communicating between multiple CICS systems, you should always let DATASTREAM default to USER.

3270

The data stream is a 3270 data stream as defined in the type 6.1 logical unit (LUTYPE6.1) architecture.

SCS

The data stream is an SCS data stream as defined in the LUTYPE6.1 architecture.

STRFIELD

The data stream is a structured field data stream as defined in the LUTYPE6.1 architecture.

LMS

The data stream is a Logical Message Services (LMS) data stream consisting of FMH4s and FMH8s as defined in the LUTYPE6.1 architecture.

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see "How the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

INDSYS(name)

The name of an intermediate system that will be used to relay communications between this system and the remote system. The name can be up to four characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

You may name an intermediate system only if you specify ACCESSMETHOD(INDIRECT). The name must be the name of a CONNECTION definition.

INSERVICE({YES|NO})

The status of the connection that is being defined.

YES Indicates that transactions may be initiated and messages may automatically be sent across the connection.

NO Indicates that the connection can neither receive messages nor transmit input.

MAXQTIME

A time control on the wait time of queued allocate requests that are waiting for free sessions on a connection that appears to be unresponsive. The maximum queue time is used only if a queue limit is specified on QUEUELIMIT, and then the time limit is applied only when the queue length has reached the QUEUELIMIT value.

You can specify either NO, or a number:

NO

CICS maintains the queue of allocate requests that are waiting for a free session. No time limit is set

for the length of time requests can remain queued (though the DTIMOUT mechanisms can apply to individual requests). In this case, a value of X'FFFF' is passed on the XZIQUE parameter list (in field UEPeMXQT).

number

The approximate upper limit on the time that allocate requests can be queued for a connection that appears to be unresponsive. The number represents seconds in the range 0 through 9999.

CICS uses the maximum queue time parameter to control a queue of allocate requests waiting as follows:

- When the number of queued allocate requests reaches the queue limit (QUEUELIMIT), and
- A new allocate request is received for the connection, and
- The rate of processing for the queue indicates that, on average, the new allocate will take more than the maximum queue time, then
- The queue is purged, and message DFHZC2300 is issued.

No further queuing takes place until the connection has successfully freed a session. At this point, CICS issues DFHZC2301 and resumes normal queuing.

You can also control the queuing of allocate requests through an XZIQUE global user exit program. This allows you more flexibility to use statistics provided by CICS, which report the state of the link. You can use these statistics, in combination with the queue limit and maximum queue time values you specify, to make more specialized decisions about queues.

The MAXQTIME value is passed to an XZIQUE global user exit program on the XZIQUE parameter list, if the exit is enabled. See the *CICS/ESA Customization Guide* for programming information about writing an XZIQUE global user exit program.

You can also specify the NOQUEUE|NOSUSPEND option on the ALLOCATE command to prevent an explicit request being queued. See the *CICS/ESA Application Programming Reference* for programming information about these API options.

NETNAME(name)

The network name that identifies the remote system. The name can be up to eight characters in length. The name follows assembler language rules. It must start with an alphabetic character. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

The NETNAME is the APPLID of the remote system or region, unless you are defining an APPC or LUTYPE6.1

link to a terminal-owning region that is a member of a VTAM generic resource group, in which case NETNAME must specify the TOR's *generic resource name*, **not** its APPLID.

For VTAM, the APPLID is the label of the remote ACF/VTAM VBUILD TYPE=APPL statement.

If you do not supply a NETNAME, the CONNECTION name is used by default.

There are some rules about duplicate NETNAMEs. You **cannot** have:

- Two or more APPC links with the same NETNAME
- An APPC link and an LUTYPE6.1 link with the same NETNAME
- Two or more IRC connections with the same NETNAME.

You **can** have:

- An IRC connection and an LUTYPE6.1 connection with the same NETNAME
- An IRC connection and an APPC connection with the same NETNAME
- Two or more LUTYPE6.1 connections with the same NETNAME.

+ You cannot use an ALIAS when naming an APPC or
+ LUTYPE6.1 link. CICS requires the real NETNAME of
+ the connecting partner. Any attempt to use an alias
+ causes the BIND to be rejected.

PROTOCOL({APPC|LU61|EXCI|blank})

The type of protocol that is to be used for the link.

blank

For MRO between CICS regions. You must leave the PROTOCOL blank for MRO, and on the SESSIONS definition you must specify LU6.1 as the PROTOCOL.

APPC (LUTYPE6.2 protocol)

Advanced program-to-program communication, or APPC protocol. This is the default value for ACCESSMETHOD(VTAM). Specify this for CICS-CICS ISC.

LU61

LUTYPE6.1 protocol. Specify this for CICS-CICS ISC or CICS-IMS ISC, but not for MRO.

EXCI

The external CICS interface. Specify this to indicate that this connection is for use by a non-CICS client program using the external CICS interface.

PSRECOVERY({SYSDEFAULT|NONE})

In a CICS region running with persistent sessions support, this specifies whether, and how, LU6.2 sessions are recovered on system restart within the persistent session delay interval.

CONNECTION

SYSDEFAULT

If a failed CICS system is restarted within the persistent session delay interval, the following actions occur:

- User modegroups are recovered to the SESSIONS RECOVOPTION value.
- The SNASVCMG modegroup is recovered.
- The connection is returned in ACQUIRED state and the last negotiated CNOS state is returned.

NONE

All sessions are unbound as out-of-service with no CNOS recovery.

QUEUELIMIT

The maximum number of allocate requests that CICS is to queue while waiting for free sessions. You can specify either NO, or a number:

NO

There is no limit set to the number of allocate requests that CICS can queue while waiting for a free session. In this case, a value of X'FFFF' is passed on the XZIQUE parameter list (in field UEPQUELM).

number

The maximum number of allocate requests, in the range 0 through 9999, that CICS can queue on the connection while waiting for a free session. When the number of queued allocate requests reaches this limit, CICS rejects subsequent allocate requests until the queue drops below the limit.

This queue limit is passed to an XZIQUE global user exit program on the XZIQUE parameter list if the exit is enabled.

You can also control the queuing of allocate requests through the MAXQTIME parameter, and through an XZIQUE global user exit program. See the MAXQTIME parameter for more information about controlling queues.

Notes:

1. The queuing of requests applies to Allocate requests only. On an LU6.2 session, function shipping requests fail immediately with a SYSIDERR condition if the value specified in the MAXIMUM attribute of the SESSIONS definition is reached. This option should be used with caution on a link where both Allocate and function shipping requests occur.
2. BIND re-negotiation is not triggered, even if there are unused secondary sessions. Unless the CEMT SET MODE command is used to force re-negotiation, the QUEUELIMIT value is used as soon as all the primary sessions are in use.

RECORDFORMAT({U|VB})

The type of SNA chain.

U Let RECORDFORMAT default to U if the SNA chain is a single, unblocked stream of data. You can have private block algorithms within the SNA chain.

You should let RECORDFORMAT default to U if you are communicating between multiple CICS systems.

VB The SNA chain is formatted according to the VLVB standard as defined in the LUTYPE6.1 architecture.

REMOTENAME(name)

The name by which the APPC connection for transaction routing is known in the system or region that owns the connection. The name can be up to four characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. The remote system or region could be an APPC device (see "APPC devices for transaction routing" on page 189).

REMOTESYSNET(name)

The network name (APPLID) of the system that owns the connection. The name can be up to eight characters in length. It follows assembler language rules, and must start with an alphabetic character. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

Use REMOTESYSNET when transaction routing to remote APPC systems or devices and there is no direct link between the region in which this definition is installed and the system that owns the connection to the remote device. You do not need to specify REMOTESYSNET if either of the following is true:

- You are defining a local connection (that is, REMOTESYSTEM is not specified, or specifies the name of the local system).
- REMOTESYSTEM names a direct link to the system that owns the connection. However, there is one special case: if the connection-owning region is a member of a VTAM generic resources group and the direct link to it is an APPC connection, you **do** need to specify REMOTESYSNET.

REMOTESYSTEM(name)

The name that identifies the intercommunication link to the system that owns the connection. The name can be up to four characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

This is the CONNECTION name on the CONNECTION definition for the intercommunication link.

REMOTESYSTEM is used for transaction routing to remote APPC systems or devices. If it is not specified, or if it is specified as the name of the local system, then this connection will be local to this system. If the name is that of another system, the connection will be remote. You can therefore use the same definition for the connection in both the local system and a remote system.

| If there are intermediate systems between this CICS and
 | the region that owns the (connection to) the device,
 | REMOTESYSTEM should specify the first link in the
 | path to the device-owning region. If there is more than
 | one possible path, it should specify the first link in the
 | preferred path.

SECURITYNAME(name)

| For APPC and LU6.1 links only, this is the security name
 | of the remote system.

In a CICS system with security initialized (SEC=YES or
 MIGRATE), the security name is used to establish the
 authority of the remote system.

Note: If USERID is specified in the session definition
 (DEFINE SESSIONS command) associated with the
 connection definition, it overrides the userid specified in
 the SECURITYNAME parameter, and is used for link
 security.

The security name (or USERID on the sessions
 definition) must be a valid RACF userid on your system.
 Access to protected resources on your system is based
 on the RACF user profile and its group membership.

For further information on defining connections for MRO,
 LUTYPE6.1, and APPC, see the *CICS/ESA CICS-RACF
 Security Guide*.

SINGLESESS({NO|YES})

YES indicates that the definition is for an APPC terminal
 on a single session APPC link to CICS. The
 MODENAME attribute of the SESSIONS definition can
 be used to supply a modename for the single session
 mode set.

| An APPC single session terminal can also be defined as
 | a TERMINAL-TYPETERM definition. Both the
 | TERMINAL-TYPETERM definition and the
 | CONNECTION definition can be autoinstalled. If you are
 | considering using autoinstall, see Chapter 9, "Autoinstall
 | for VTAM terminals" on page 105 and Chapter 10,
 | "Autoinstall for APPC connections" on page 119. *

USEDFLTUSER({NO|YES}) (APPC and MRO only)

No Indicates that each inbound attach FMH will be
 # checked for the presence of those fields required
 # by the ATTACHSEC option and if the required
 # fields are not present a protocol violation
 # message will be issued and the attach will fail.
 # NO is the default.

Yes Indicates that some checks on the validity of the
 # attach FMH are bypassed. This provides the
 # same level of security as in previous releases of
 # CICS. See 'Attach Time Security and the
 # USEDFLTUSER option' in chapter 12 of the
 # *CICS/ESA CICS-RACF Security Guide*.

CONNECTION

Chapter 13. FILE

The following resources associated with CICS files can be managed using RDO:

- VSAM files
- Remote VSAM files
- Remote BDAM files
- VSAM local shared resource (LSR) pools.

You use the FILE definition to describe to CICS file control the physical and operational characteristics of a file. The definition includes keywords that provide information about record characteristics, types of operations allowed on the file, recovery attributes, and the operations that are to be journaled. This information is used to generate control information used by CICS as well as an access control block (ACB). CICS files correspond to physical data sets that must have been defined to VSAM before they are used.

For the file to be used by an active CICS system, its definition must have been installed on to the system. CICS file control uses the definition to find the file when it needs to access it, to keep count of the number of tasks using the file, to maintain a record of the address of the associated ACB, and to capture processing statistics.

Remote files

When multiple CICS systems are connected, they can share each other's files; all such files must be defined to CICS, including those belonging to another system. Files on other systems are defined as 'remote'.

The resource attributes needed by CICS for remote files are not specific to the access method used. You can therefore define both remote BDAM files and remote VSAM files using RDO.

If you name a REMOTESYSTEM, you may also supply a REMOTENAME, which is the name of the *file* used in the remote system.

If you specify a REMOTESYSTEM name that corresponds to the system in which the install has been made, a local FCT entry is created. Otherwise a remote FCT entry is created.

Installing FILE definitions

Adopt the following procedure to install a FILE definition.

1. If the file already exists, ensure that it is closed and disabled:

```
CEMT SET FILE(filename) CLOSED DISABLED
```

2. Terminate the CEMT command.

3. Install the file definition:

```
CEDA INSTALL GROUP(groupname) FILE(filename)
```

4. When you have successfully installed the group containing the file, use CEMT to open and enable the file if it is not already defined as open and enabled:

```
CEMT SET FILE(filename) OPEN ENABLED
```

Defining a FILE

```

File      ==>
Group     ==>
Description ==>

VSAM PARAMETERS
DSName    ==>
Password  ==>          PASSWORD NOT SPECIFIED
Lsrpoolid ==> 1          1-8 | None
DSNSharing ==> Allreqs  Allreqs | Modifyreqs
STRings   ==> 001       1 - 255
Nsrgrupp  ==>

REMOTE ATTRIBUTES
REMOTESystem ==>
REMOTENAME ==>
RECORDSize ==>          1 - 32767
Keylength ==>          1 - 255

INITIAL STATUS
STATus    ==> Enabled   Enabled | Disabled | Unenabled
Opentime  ==> Firstref  Firstref | Startup
DISposition ==> Share   Share | Old

BUFFERS
DATABuffers ==> 00002    2 - 32767
INDEXbuffers ==> 00001  1 - 32767

DATATABLE PARAMETERS
Table     ==> No        No | Cics | User
Maxnumrecs ==>          16 - 16777215

DATA FORMAT
RECORDFormat ==> V      V | F

OPERATIONS
Add       ==> No        No | Yes
BRowse    ==> No        No | Yes
DELeTe    ==> No        No | Yes
REAd      ==> Yes       Yes | No
Update    ==> No        No | Yes

AUTO JOURNALING
JOUrnal   ==> No        No | 1 - 99
JNLRead   ==> None      None | Updateonly | Readonly | All
JNLSYNCRRead ==> No     No | Yes
JNLUpdate ==> No        No | Yes
JNLAdd    ==> None      None | Before | After | All
JNLSYNCRWrite ==> Yes   Yes | No

RECOVERY PARAMETERS
RECOVery  ==> None      None | Backoutonly | All
Fwdrecovlog ==> No      No | 1-99
BACKuptype ==> STAtic   STAtic | DYNamic

SECURITY
RESsecnum : 00          0-24 | Public

```

Figure 26. The DEFINE panel for FILE

ADD(NO|YES)

Specifies whether records can be added to the file.

BACKUPTYPE(STATIC|DYNAMIC)

CICS VSAM files can be defined as eligible for backup while open for update. Possible values are:

STATIC

File is not eligible for backup while open for update.

DYNAMIC

Specify this along with the RECOVERY attribute of ALL to make the file eligible for backup while open for update.

BROWSE(NO|YES)

Specifies whether records can be retrieved sequentially from the file.

DATABUFFERS(2|value)

A value, in the range 2 through 32767, to define the number of buffers to be used for data. The minimum value you may specify is one more than the number of strings defined in the STRINGS attribute.

DELETE(NO|YES)

Specifies whether records can be deleted from the file.

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

DISPOSITION(SHARE|OLD)

The disposition of this file.

SHARE

Equivalent to the DISP=SHR parameter in JCL.

OLD

Equivalent to the DISP=OLD parameter in JCL.

DSNAME(name)

The data set name (as known to the operating system) to be used for this file. DSNAME can be 1 through 44 characters. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >.

At file open time, if no JCL statement exists for this file, the open is preceded by a dynamic allocation of the file using this DSNAME.

If a JCL statement exists for this file in the CICS start-up job, the DSNAME specified takes precedence over this DSNAME.

DSNSHARING(ALLREQS|MODIFYREQS)

Indicates whether VSAM data set name sharing will be used for the VSAM file. The possible values are:

ALLREQS

Data set name sharing will be set in the ACB when the file is opened and will therefore be used for all file requests.

MODIFYREQS

Data set name sharing will be set in the ACB when the file is opened only if an operation of DELETE, ADD, or UPDATE is set for the file.

FILE(name)

The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

FWDRECOVLOG(NO|value)

The value, in the range 1 through 99, specifies which journal the after images for forward recovery are written to. This value is only used if RECOVERY=ALL is specified.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see "How the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase

characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

INDEXBUFFERS(1|value)

A value, in the range 1 through 32767, to define the number of buffers to be used for the index. The minimum value you may specify is the number of strings defined in the STRINGS attribute.

JNLADD(NONE|BEFORE|AFTER|ALL)

Specify the add operations you want recorded on the journal nominated by the JOURNAL parameter. Possible values are:

NONE

Do not journal add operations.

ALL

Journal the file control write operation both before and after the VSAM I/O operation has completed.

BEFORE

Journal the file control write operation before the VSAM I/O operation.

AFTER

Journal the file control write operation after the VSAM I/O operation.

JNLREAD(NONE|UPDATEONLY|READONLY|ALL)

Specify the read operations you want recorded on the journal nominated by the JOURNAL parameter. Possible values are:

NONE

Do not journal read operations.

UPDATEONLY

Journal only READ UPDATE operations (not READ ONLY operations).

READONLY

Journal only READ ONLY operations (not READ UPDATE operations).

ALL

Journal all read operations.

JNLSYNCREAD(NO|YES)

Indicates whether you want the automatic journaling records, written for READ operations to the journal specified by JOURNAL, to be written synchronously or asynchronously.

JNLSYNCWRITE(YES|NO)

Indicates whether you want the automatic journaling records, written for WRITE operations to the journal specified by JOURNAL, to be written synchronously or asynchronously.

JNLUPDATE(NO|YES)

Specify whether you want REWRITE and DELETE operations recorded on the journal nominated by the JOURNAL parameter.

FILE

JOURNAL(NO|value)

Indicates whether you want automatic journaling for this file. For a data table, journaling is performed only for requests that result in VSAM I/O requests. The journaled data is in the format of the VSAM record and is used for user controlled journaling. The data to be journaled is identified by the JNLADD, JNLREAD, JNLSYNCREAD, JNLSYNCWRITE and JNLUPDATE parameters.

Note: Automatic journaling is independent of logging done to the system and forward recovery logs, as specified by the RECOVERY and FWDRECOVLOG parameters.

Possible values are:

NO

No journal activity for this file.

1 Journal data is recorded on the system log.

value

The journal identification in the range 2 through 99.

KEYLENGTH(value)

The length in bytes, in the range 1 through 255, of the logical key for remote files. If not defined here, the length option must be specified in the application program that refers to this file.

LSRPOOLID(1-8|None)

The identity of the local shared resource pool. For a data table the default value for LSRPOOLID is 1, unless a value has been coded for the NSRGROUP parameter, in which case the default value

+ **APAR PN77938**
+ emphasised the default rules

+ for LSRPOOLID is NONE.

1-8

The value, in the range 1 through 8, identifies the number of the VSAM shared resource pool that will be used by the VSAM data set associated with this file. The data set is defined as using VSAM Local Shared Resources (LSR).

None

Specifies that the data set associated with this file will use VSAM non-shared resources (NSR).

MAXNUMRECS(value)

If you have specified 'CICS' or 'USER' for the TABLE parameter, specify here the maximum number of entries to be accommodated in the data table, in the range 16 through 16777215.

NSRGROUP(value)

The NSRGROUP parameter only takes effect for files referencing data sets that use VSAM non-shared resources.

APAR PN77938

added the next sentence.

+
+
+
+

The NSRGROUP parameter must not be coded for a data table.

Code this with a symbolic name (up to eight characters) to group together file definitions that refer to the same VSAM base data set. The value is purely symbolic and need not refer to any particular file definition. It is merely necessary that all file definitions that need to be grouped together code the same name. You do not have to code this parameter to ensure correct processing, but if you do not provide it, performance of your system may be degraded.

The NSRGROUP parameter is associated with the VSAM concept of data set name sharing which causes VSAM to create a single control block structure for the strings and buffers required by all the files that relate to the same base data set.

When the first member of such a group of files is opened, the total number of strings to be allocated for all file entries in the group must be specified to VSAM (by means of the BSTRNO value in the Access Control Block). The VSAM control block structure is built this time regardless of whether the first file to be opened is associated with a path or base data set. The value of BSTRNO is calculated at this time by adding together the STRINGS values in all the file definitions with the same NSRGROUP parameter. After the first file in the group is opened, any new files added to the group will not affect the VSAM control block structure already built. This would only change if all the files open against the base were closed and then re-opened.

Data set name sharing is forced by CICS as the default for all VSAM files. Data set name sharing is not in effect if a file is opened for read-only processing with DSNSHARING=MODIFYREQS. A file with DSNSHARING=MODIFYREQS will still, however, contribute to the BSTRNO calculation.

If a file is using VSAM non-shared resources, and you do not provide an NSRGROUP parameter, the VSAM control block structure may be built with insufficient strings for later processing. When this happens, VSAM will invoke the dynamic string addition feature to provide the extra control blocks for the strings as they are required. This mechanism is, however, inefficient and the extra storage is not released until the end of the CICS run.

For files specifying that VSAM local shared resources are to be used (LSRPOOLID=n, where n is in the range 1 to 8), NSRGROUP has no effect.

Figure 27 on page 145 shows an example of how to specify the required file control definition for a VSAM base data set and an alternate index path.

```

CEDA DEFINE FILE(VSAM10B) GROUP(xxxxxx)
  DSNNAME(DTGCAT.VSAM10B)
  DISPOSITION(SHARE) ADD(YES)
  BROWSE(YES) DELETE(YES) READ(YES)
  UPDATE(NO) RECORDFORMAT(F)
  STRINGS(8) LSRPOOLID(NONE)
  RECOVERY(NONE) NSRGROUP(GROUP1)
  INDEXBUFFERS(8) DATABUFFERS(9)

CEDA DEFINE FILE(VSAM10P) GROUP(xxxxxx)
  DSNNAME(DTGCAT.VSAM10P)
  LSRPOOLID(NONE) DISPOSITION(SHARE)
  STRINGS(5) NSRGROUP(GROUP1)
  BROWSE(YES) DELETE(NO) READ(YES)
  ADD(NO) UPDATE(NO) RECORDFORMAT(F)
  RECOVERY(NONE) INDEXBUFFERS(5)
  DATABUFFERS(6)

```

Figure 27. VSAM base data set and alternate index path definition.

OPENTIME(FIRSTREF|STARTUP)

Indicates when the file will be opened. Possible values are:

FIRSTREF

The file will remain closed until a request is made to open it by:

- A master terminal command
- An EXEC CICS SET FILE OPEN command in an application program
- An implicit open.

STARTUP

The file will be opened immediately after CICS initialization by an automatically initiated CICS transaction (CSFU), unless the status of the file is UNENABLED when the file will be left closed.

PASSWORD(name)

The 1- to 8-character password that will be used to verify user access to the file.

CICS will mask the password you supply to avoid unauthorized access. You should therefore find a safe way of recording the password.

READ(YES|NO)

Indicates whether records on this file can be read.

RECORDFORMAT(V|F)

The format of the records on the file.

V The records are variable length.

F The records are fixed length. For VSAM files, code this only if the VSAM access method services definition specifies fixed size records (that is, the average size is equal to the maximum size), and all the records in the file are of that size. F is invalid

for user-maintained data tables. All user-maintained data tables are variable length.

RECORDSIZE(number)

- # specifies the maximum length in bytes of records in a
- # remote file. (The information is not required for local
- # files, it is obtained from the VSAM catalog.) The size
- # specified can be in the range 1 through 32767.

RECOVERY(NONE|BACKOUTONLY|ALL)

Indicates the recovery to be done for this file. (See the *CICS/ESA Recovery and Restart Guide* for further information.)

NONE

No recovery logging for this file.

BACKOUTONLY

Log before images to the system log. For CICS-maintained data tables, BACKOUTONLY specifies that the data table and its source data set are recoverable. They will both be updated in step and, if required, recovered in step. For user-maintained tables, this specifies only dynamic backup. No log records are written and, therefore, there is no recovery at emergency restart.

- # **Note:** If the JCT parameter in the SIT has a value
- # of NO, no recovery of any sort will be
- # performed.

- + **Note:** When BACKOUTONLY is specified for
- + VSAM ESDS files, CICS is unable to perform
- + backup of ADDs. To cope with this situation, you
- + should code user exit XDBFERR to avoid the file
- + being closed because of the error.

ALL

Log before images to the system log, and after images to the journal specified in the FWDRECOVLOG parameter.

Records written to the FWDRECOVLOG are independent of any automatic journaling options that may be set.

RECOVERY=ALL together with FWDRECOVLOG provide a means of separating the needs of a forward recovery utility from those of automatic journaling. Additional information, not available via automatic journaling, is recorded on the FWDRECOVLOG. RECOVERY=ALL plus FWDRECOVLOG is the recommended way to provide forward recovery support.

Existing forward recovery utilities that used the JREQ=(WU,WN) and JID=FCT macro settings can still be used with these settings. The RDO equivalents of these automatic journaling settings are JNLADD=BEFORE, JNLUPDATE=YES, and the JOURNAL parameter.

For CICS-maintained data tables, the data table and its source data set are logged, journaled, and recovered together. For user-maintained tables,

FILE

specifying ALL has the same effect as specifying BACKOUTONLY: only dynamic backout is provided. There is no forward recovery support for user-maintained tables.

- + **Note:** When ALL is specified for VSAM ESDS files,
- + CICS is unable to perform backout of ADDs. To
- + cope with this situation, you should code user exit
- + XDBFERR to avoid the file being closed because of
- + the error.

REMOTENAME(name)

The name by which this file is known in the system or region in which it is resident. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. If REMOTENAME is not specified, the name given in the FILE attribute will be used.

REMOTESYSTEM(name)

If you are operating in an ISC or MRO environment, and the file is held by a remote system, this specifies the name of the system or region in which the file is resident. The name can be up to four characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. If you specify REMOTESYSTEM, you may also supply a REMOTENAME, to specify the name of the file in the remote system.

RESSECNUM({0|value|PUBLIC})

The RESSECNUM keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

STATUS({ENABLED|DISABLED|UNENABLED})

The initial status of the file following a CICS initialization with START=COLD. You can change the status of a closed file with the master terminal transaction CEMT. The status of a file (enabled, disabled, or unenabled) following a CICS restart is recovered to its status at the previous shutdown.

ENABLED

Normal processing is allowed against this file.

DISABLED

Any request against this file from a command-level application program will cause the DISABLED condition to be passed to the program.

UNENABLED

This prevents the file being opened by an implicit open from an application program. Any such attempt to access the file will raise the NOTOPEN condition. By contrast, an explicit request to open the file (for example, a CEMT or EXEC CICS SET FILE OPEN command), will change the status to ENABLED before attempting to open the file.

STRINGS(1|value)

The number, in the range 1 through 255, of concurrent requests that can be processed against the file. When the number of requests reaches this value, CICS will queue any additional requests until one of the active requests terminates. This applies both to files using shared resources, and to those not using shared resources. *When coding STRINGS, be aware that a proportion (20%) of the specified number of strings are reserved by CICS for use in read-only requests.*

For VSAM files using shared resources, this number is not used by VSAM. It is used by CICS, not only as described above, but also to calculate the default value in the buffer pool definition.

TABLE(NO|CICS|USER)

Indicates the type of data table that you require.

NO

Data table not required.

CICS

CICS-maintained data tables. These automatically reflect all modifications to their source data set. If you specify CICS, you must also specify LSRPOOLID with a value of 1 through 8, and MAXNUMRECS with the value you require.

USER

User-maintained tables. These remain independent of their source data sets, and changes to the user-maintained tables are not reflected in corresponding source data sets. If you specify USER, you must also specify LSRPOOLID with a value of 1 through 8, RECORDFORMAT as V (or let this default to the value of V), and MAXNUMRECS with the value you require.

UPDATE(NO|YES)

Records on this file can (YES), or cannot (NO) be updated.

Chapter 14. LSRPOOL

The local shared resources (LSR) pool is a reserve of data buffers, strings, and Hiperspace⁴ buffers that VSAM uses when processing access requests for certain files. You use the LSRPOOL definition to define the size and characteristics of the pool. Up to eight LSR pools can be defined concurrently in the system, each identified by their LSRPOOLID. This LSRPOOLID is used to associate a FILE with an LSR pool if that file is to use shared resources.

When the LSRPOOL definition is installed in the active system, its information is stored and used when the pool with the specified ID is next built. A pool is built when the first file using a particular LSR pool is opened, and is dynamically deallocated only when no files are currently open against that pool. This means that when an LSRPOOL definition is installed into the system it may not take effect immediately.

CICS sets default attributes if an LSRPOOL is not defined, but you are advised to define the LSRPOOL anyway, for reasons of performance. In a production system, for example, delay may be incurred while pool requirements are being calculated by CICS. Another possible problem is that if files in the FCT are not allocated at the time the pool is built, the data set names will not be known to CICS. In this case, the pool is built based on the information available, but the subsequent performance of the system may suffer or files may fail to open.

You can associate the CSD with a particular LSRPOOL by specifying the CSDLSRNO system initialization parameter. The default is pool 1; you should ensure that sufficient buffers of an appropriate size are provided to permit the CSD to be used by CICS. See the *CICS/ESA System Definition Guide* for further information about CSDLSRNO and about calculating the buffer requirements for the CSD.

⁴ A Hiperspace buffer is a high-performance storage area in the MVS image. This area is used for reading and writing 4KB pages. The type of Hiperspace used by VSAM resides entirely in expanded storage, which is additional processor storage used only for paging to and from real storage.

Defining an LSRPOOL

```

Lsrpool    ==> ....
Group     ==> ....
Description ==>

Lsrpoolid   ==> 1           1 - 8
Maxkeylength ==>           0 - 255
Sharelimit  ==>           1 - 100
SStrings    ==>           1 - 255

DATA BUFFERS
DATA512     ==>           3 - 32767
DATA1K      ==>           3 - 32767
DATA2K      ==>           3 - 32767
DATA4K      ==>           3 - 32767
DATA8K      ==>           3 - 32767
DATA12k     ==>           3 - 32767
DATA16k     ==>           3 - 32767
DATA20k     ==>           3 - 32767
DATA24k     ==>           3 - 32767
DATA28k     ==>           3 - 32767
DATA32k     ==>           3 - 32767

INDEX BUFFERS
INDEX512    ==>           3 - 32767
INDEX1K     ==>           3 - 32767
INDEX2K     ==>           3 - 32767
INDEX4k     ==>           3 - 32767
INDEX8k     ==>           3 - 32767
INDEX12k    ==>           3 - 32767
INDEX16k    ==>           3 - 32767
INDEX20k    ==>           3 - 32767
INDEX24k    ==>           3 - 32767
INDEX28k    ==>           3 - 32767
INDEX32k    ==>           3 - 32767

HIPERSPACE DATA BUFFERS
HSDATA4K    ==>           0 - 16777215
HSDATA8K    ==>           0 - 16777215
HSDATA12K   ==>           0 - 16777215
HSDATA16K   ==>           0 - 16777215
HSDATA20K   ==>           0 - 16777215
HSDATA24K   ==>           0 - 16777215
HSDATA28K   ==>           0 - 16777215
HSDATA32K   ==>           0 - 16777215

HIPERSPACE INDEX BUFFERS
HSINDEX4K   ==>           0 - 16777215
HSINDEX8K   ==>           0 - 16777215
HSINDEX12K  ==>           0 - 16777215
HSINDEX16K  ==>           0 - 16777215
HSINDEX20K  ==>           0 - 16777215
HSINDEX24K  ==>           0 - 16777215
HSINDEX28K  ==>           0 - 16777215
HSINDEX32K  ==>           0 - 16777215

```

Figure 28. The DEFINE panel for LSRPOOL

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see “How the CSD is organized — groups and lists” on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with

DFH, because these characters are reserved for use by CICS.

LSRPOOL(name)

The name of the local shared resource pool being defined. The name may be up to eight characters in length.

If only DATA BUFFERS is specified, one set of buffers is built for the pool to be used for both the index and the data components of a VSAM KSDS data set.

If no data buffers are specified, CICS will calculate the buffers required for both data and index components, both components sharing the same set of buffers.

If INDEX BUFFERS is specified, two parts of the pool will be built, one for data and the other for index buffers. If you specify INDEX BUFFERS, you must also specify DATA BUFFERS.

LSRPOOLID({1|value})

The identifier of the local shared resource pool being defined. The value must be in the range 1 through 8.

MAXKEYLENGTH(value)

This value overrides part of the CICS resource calculation. It specifies the maximum key length of any of the files that are to share resources. If you do not code it, CICS will determine the maximum key length. The value must be in the range 0 through 255.

SHARELIMIT(value)

Code this if CICS is to calculate the maximum amount of resources required by the VSAM files that are to share resources. Because these resources are to be shared, some percentage of this maximum amount of resources must be allocated. The SHARELIMIT operand specifies, as an integer, the percentage of the maximum amount of VSAM resources to be allocated. If this parameter is omitted, 50 percent of the maximum amount of resources will be allocated. If both the STRINGS and SIZE parameters are coded, SHARELIMIT will have no effect.

The number can be any value from 1 through 100.

STRINGS(value)

The limit, in the range 1 through 255, of all the strings of the files in the pool.

DATA512(value)

The number, in the range 3 through 32767, of 512-byte data buffers you require.

DATA1K(value)

The number, in the range 3 through 32767, of 1KB data buffers you require.

DATA2K(value)

The number, in the range 3 through 32767, of 2KB data buffers you require.

DATA4K(value)

The number, in the range 3 through 32767, of 4KB data buffers you require.

DATA8K(value)

The number, in the range 3 through 32767, of 8KB data buffers you require.

DATA12K(value)

The number, in the range 3 through 32767, of 12KB data buffers you require.

DATA16K(value)

The number, in the range 3 through 32767, of 16KB data buffers you require.

DATA20K(value)

The number, in the range 3 through 32767, of 20KB data buffers you require.

DATA24K(value)

The number, in the range 3 through 32767, of 24KB data buffers you require.

DATA28K(value)

The number, in the range 3 through 32767, of 28KB data buffers you require.

DATA32K(value)

The number, in the range 3 through 32767, of 32KB data buffers you require.

INDEX512(value)

The number, in the range 3 through 32767, of 512-byte index buffers you require.

INDEX1K(value)

The number, in the range 3 through 32767, of 1KB index buffers you require.

INDEX2K(value)

The number, in the range 3 through 32767, of 2KB index buffers you require.

INDEX4K(value)

The number, in the range 3 through 32767, of 4KB index buffers you require.

INDEX8K(value)

The number, in the range 3 through 32767, of 8KB index buffers you require.

INDEX12K(value)

The number, in the range 3 through 32767, of 12KB index buffers you require.

INDEX16K(value)

The number, in the range 3 through 32767, of 16KB index buffers you require.

INDEX20K(value)

The number, in the range 3 through 32767, of 20KB index buffers you require.

INDEX24K(value)

The number, in the range 3 through 32767, of 24KB index buffers you require.

LSRPOOL

INDEX28K(value)

The number, in the range 3 through 32767, of 28KB index buffers you require.

INDEX32K(value)

The number, in the range 3 through 32767, of 32KB index buffers you require.

HSDATA4K(value)

The number, in the range 0 through 16777215, of 4KB Hiperspace data buffers you require.

HSDATA8K(value)

The number, in the range 0 through 16777215, of 8KB Hiperspace data buffers you require.

HSDATA12K(value)

The number, in the range 0 through 16777215, of 12KB Hiperspace data buffers you require.

HSDATA16K(value)

The number, in the range 0 through 16777215, of 16KB Hiperspace data buffers you require.

HSDATA20K(value)

The number, in the range 0 through 16777215, of 20KB Hiperspace data buffers you require.

HSDATA24K(value)

The number, in the range 0 through 16777215, of 24KB Hiperspace data buffers you require.

HSDATA28K(value)

The number, in the range 0 through 16777215, of 28KB Hiperspace data buffers you require.

HSDATA32K(value)

The number, in the range 0 through 16777215, of 32KB Hiperspace data buffers you require.

HSINDEX4K(value)

The number, in the range 0 through 16777215, of 4KB Hiperspace index buffers you require.

HSINDEX8K(value)

The number, in the range 0 through 16777215, of 8KB Hiperspace index buffers you require.

HSINDEX12K(value)

The number, in the range 0 through 16777215, of 12KB Hiperspace index buffers you require.

HSINDEX16K(value)

The number, in the range 0 through 16777215, of 16KB Hiperspace index buffers you require.

HSINDEX20K(value)

The number, in the range 0 through 16777215, of 20KB Hiperspace index buffers you require.

HSINDEX24K(value)

The number, in the range 0 through 16777215, of 24KB Hiperspace index buffers you require.

HSINDEX28K(value)

The number, in the range 0 through 16777215, of 28KB Hiperspace index buffers you require.

HSINDEX32K(value)

The number, in the range 0 through 16777215, of 32KB Hiperspace index buffers you require.

Chapter 15. MAPSET

Each interactive application using a display device can use specific screen layouts, or maps. These are not specified in the program itself. Instead, you use basic mapping support (BMS). This gives greater flexibility and allows the maps to be used by multiple invocations of the same program, or by several different programs. You specify maps, and the fields on them, using the DFHMSD, DFHMDI, and DFHMDF macros. For further guidance on this, see the *CICS/ESA Application Programming Guide*.

Instead of using the BMS map definition macros, you can define maps interactively with the Screen Definition Facility II (SDF II) program product, program numbers 5665-366 (for MVS) and 5664-307 (for VM). SDF II allows you to paint a screen interactively. You can then generate the screen to get the equivalent of a CICS/BMS map set. The test facilities of SDF II also enable you to see your map in its run-time appearance. For background information, see the *Screen Definition Facility II Primer for CICS/BMS Programs*, SH19-6118, and the *Screen Definition Facility II General Information* manual, GH19-6114.

An application can use a series of related maps at different times during the interaction with the user. It can also use several related maps at the same time and on the same display, to build up a complete screen.

These related maps belong to a map set, which you specify in a MAPSET definition. Even if your program has only one map, this must still belong to a map set. You can define your MAPSETs either by using CEDA or DFHCSDUP, or by setting the appropriate SIT options to enable them to be autoinstalled. See Chapter 11, "Autoinstall for programs, mapsets, and partitionsets" on page 123 for further information about autoinstall.

There is no link through resource definitions between a program and its map sets. Instead, you code the MAPSET name in the BMS SEND MAP and RECEIVE MAP commands in your program.

+
|
|
|

Defining a MAPSET

Mapset	==>	
Group	==>	
Description	==>	
RESident	==> No	No Yes
USAge	==> Normal	Normal Transient
USElpacopy	==> No	No Yes
Status	==> Enabled	Enabled Disabled
RSL	: 00	0-24 Public

Figure 29. The DEFINE panel for MAPSET

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see “How the CSD is organized — groups and lists” on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

MAPSET(name)

The name of this MAPSET definition. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. Do not use map set names beginning with DFH, because these characters are reserved for use by CICS.

For a BMS device-dependent map set, the map set name must be derived by adding the map set suffix to the original (1- to 7-character) map set name. The suffix depends on the parameter coded in the TERM operand of the DFHMSD macroinstruction that defined the map set.

To use device-dependent suffixes, you need to code BMS=(,DDS) in the DFHSIT macro. For programming information on map set suffixes, see the *CICS/ESA Application Programming Reference* manual.

RESIDENT({NO|YES})

The residence status of the map set.

NO

The map set is not to be permanently resident.

YES

The map set is to be loaded on first reference and is then to be permanently resident in virtual storage, but is to be pageable by the operating system.

RSL

The RSL keyword is not valid in CICS/ESA 4.1. See Appendix A, “Obsolete attributes retained for compatibility” on page 341 for information.

STATUS({ENABLED|DISABLED})

The map set status.

ENABLED

The map set may be used.

DISABLED

The map set may not be used.

USAGE({NORMAL|TRANSIENT})

Defines when the storage for this map set will be released.

NORMAL

When the use count of the map set reaches zero, it will become eligible for removal from storage as part of the normal dynamic storage compression process.

TRANSIENT

When the use count for this map set becomes zero, the storage for this map set is released. This value should be coded for map sets that are referenced infrequently.

USELPACOPY({NO|YES})

Defines whether the map set is to be used from the link pack area (LPA).

NO

The map set is not to be used from the LPA. It will be loaded into the CICS partition.

YES

The map set can be used from the LPA if LPA=YES is coded in the DFHSIT macro or as a start-up override. The use of the map set from the LPA requires that it has been installed there and that the map set is not named by the PRVMOD start-up option. For further guidance on this, see the *CICS/ESA Installation Guide*.

Chapter 16. PARTITIONSET

The screen areas of some display devices (the 8775 Display Terminal and the IBM 3290 Information Panel, for example), can be divided into **partitions**, which can be treated as several different small displays. Different programs or program steps in a transaction can write to or receive input from different partitions.

You specify the partition set with DFHPSD and DFHPDI macros, described for programming purposes in the *CICS/ESA Application Programming Reference* manual.

+ You specify each different partition configuration as a PARTITIONSET. PARTITIONSET
+ definitions are created using CEDA or DFHCSDUP, or they can be autoinstalled if the
| appropriate SIT options have been set. See Chapter 11, "Autoinstall for programs, mapsets,
| and partitionsets" on page 123 for information.

You can name the PARTITIONSET that you want the transaction to use in the TRANSACTION definition. When the transaction starts, the information is loaded into the internal buffer of the display device.

Alternatively, if you do not specify a PARTITIONSET, CICS sets the display device to its base state before the transaction is initiated.

The transaction may require CICS to load a PARTITIONSET, or change to a new one, by issuing the BMS SEND PARTNSET command. This loads the partition set dynamically, if its definition has been installed in the active CICS system.

| Instead of using the BMS partition definition macros, you can define partitions interactively
| with the Screen Definition Facility II (SDF II) program product, program numbers 5665-366
(for MVS) and 5664-307 (for VM). SDF II allows you to paint a screen interactively. You
| can then generate the screen to get the equivalent of a CICS/BMS partitionset. The test
| facilities of SDF II also enable you to see your partition in its run-time appearance. For
| general information, see the *Screen Definition Facility II Primer for CICS/BMS Programs*,
| SH19-6118, and the *Screen Definition Facility II General Information* manual, GH19-6114.

Defining a PARTITIONSET

PARTitionset	==>	
Group	==>	
Description	==>	
REsident	==>	No Yes
USAge	==>	Normal Transient
USElpacopy	==>	No Yes
Status	==>	Enabled Disabled
Rsl	:	00 0-24 Public

Figure 30. The DEFINE panel for PARTITIONSET

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see "How the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

PARTITIONSET(name)

The name of this PARTITIONSET definition. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. Do not use partition set names beginning with DFH, because these characters are reserved for use by CICS.

A partition set is a table that describes to CICS how to partition a display screen. Partition sets are created by coding and assembling a series of commands.

For a device-dependent partition set, the partition set name must be derived by adding the partition set suffix to the original (1- to 6-character) partition set name. The suffix depends on the parameter coded in the SUFFIX operand of the DFHPSD macro instruction that defined the partition set.

To use device-dependent suffixes, you need to code BMS=(, ,DDS) in the DFHSIT macro.

For programming information on partition set suffixes, see the *CICS/ESA Application Programming Reference* manual.

RESIDENT({NO|YES})

The residence status of the partition set.

NO

The partition set is not to be permanently resident.

YES

The partition set is to be loaded on first reference and is then to be permanently resident in virtual storage, but is to be pageable by the operating system.

RSL

The RSL keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

STATUS({ENABLED|DISABLED})

Enter the partition set status.

ENABLED

The partition set may be used.

DISABLED

The partition set may not be used.

USAGE({NORMAL|TRANSIENT})

Defines when the storage for this partition set will be released.

NORMAL

When the use count for this partition set reaches zero, it will become eligible for removal from storage as part of the normal dynamic program compression process.

TRANSIENT

When the use count for this partition set becomes zero, the storage for this partition set is released. This value should be coded for partition sets that are referenced infrequently.

USELPACOPY({NO|YES})

Defines whether the partition set is to be used from the link pack area (LPA).

NO

The partition set is not to be used from the LPA. It will be loaded into the CICS partition.

YES

The partition set can be used from the LPA if LPA=YES is coded in the DFHSIT macro or as a start-up override. The use of the partition set from the LPA requires that it has been installed there and that the partition set is not named by the PRVMOD start-up option. For more details on this, see the *CICS/ESA Installation Guide*.

PARTITIONSET

Chapter 17. PARTNER

You use the PARTNER definition to enable CICS application programs to communicate via APPC protocols to a partner application program running on a remote logical unit. This interaction between a CICS application program and a partner application program is called a **conversation**.

The PARTNER definition also facilitates the use of the call to the interface with the communications element of the System Application Architecture (SAA). For more information on the SAA communications interface, see the *Common Programming Interface Communications Reference*, SC26-4399-04.

To allow the SAA communications interface to be used, you must specify the following resources:

- A PROFILE definition (see “Defining a MAPSET” on page 152),
- A CONNECTION definition (see Chapter 12, “CONNECTION” on page 129),
- A SESSIONS definition (see Chapter 20, “SESSIONS” on page 175).

You can define your CICS partner information in one of two ways:

- Create a PARTNER definition.
- In an application program, by setting SYMDESTNAME to a null value, and issuing the appropriate CPI SET calls. See the *CPI-C Communications Reference* manual for further details.

PARTNER uses the following keywords:

PROFILE

This is the name of the communication profile that specifies the interaction between transactions and the logical units involved.

NETNAME

This is the name of the logical unit on which the partner application program will be running.

What happens to PARTNER definitions

When you install a PARTNER definition, CICS attempts to resolve references to CONNECTION and PROFILE definitions. CICS then creates an entry for the PARTNER definition in the **partner resource table (PRT)**. See the *CICS/ESA Intercommunication Guide* for more information.

Because it is possible for programs to use the SAA communications interface SET calls to change the PARTNER name and the TPNAME name, CICS does not check that the CONNECTION definition is present when the PARTNER resource is being installed.

If you leave out the PROFILE attribute, CICS will use the default profile DFHCICSA. Since it is not possible for an SAA communications interface program to set a different profile, the PROFILE must be installed before the PARTNER resource. However, the PARTNER install will not fail if the PROFILE is missing; instead, a run-time error will occur when the partner conversation is attempted.

The MODENAME specified in the PROFILE need not be specified in a corresponding SESSIONS definition, as it is possible for the SAA communications interface program to set a different value for MODENAME.

CICS programs, and SAA communications interface programs that do not use the SET calls, require that all the relevant definitions be installed. You can use CEDA CHECK to help find out if all required definitions are in a group.

Defining a PARTNER

```

PARTNer      ==>
Group       ==>
DEscription   ==>

REMOTE LU NAME
NETName     ==>
NETWork      ==>

SESSION PROPERTIES
Profile      ==> DFHCICSA

REMOTE TP NAME
Tpname      ==>
            ==>
Xtpname     ==>
            ==>
            ==>

```

Figure 31. The DEFINE panel for PARTNER

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see "How the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

NETNAME(name)

The NETNAME attribute specified in the CONNECTION definition. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. For further information on the CONNECTION definition and its NETNAME attribute, see Chapter 12, "CONNECTION" on page 129.

NETWORK(name)

You can use this optional attribute to specify the name of the network on which the partner LU is located. The name can be up to eight characters in length. The

acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

PARTNER(name)

The name of this PARTNER definition. The name must be exactly eight characters in length. The acceptable characters are A-Z and 0-9. Do not use partner names beginning with DFH, because these characters are reserved for use by CICS.

A partner definition specifies the SAA communications interface information required to establish a conversation with a partner program. For further guidance on this, see the *Common Programming Interface Communications Reference*, SC26-4399-04.

PROFILE(name)

Use this attribute to specify which communication profile is to be used for the session and conversation. The default PROFILE is DFHCICSA. The name can be up to 8 characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >. For information on how PROFILE resources are defined, see Chapter 18, "PROFILE" on page 163.

TPNAME(name)

Enter the name of the remote transaction program that will be running on the partner LU. The definition of a remote TP name is mandatory; you must specify either TPNAME or its alternative, XTPNAME.

This name can be up to 64 characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >. If this range of characters is not sufficient for a name that you wish to specify, you may use the XTPNAME attribute instead of TPNAME.

XTPNAME(value)

This attribute may be used as an alternative to TPNAME; you **must** specify one of the two, as the definition of a remote TP name is mandatory.

Enter a hexadecimal string up to 128 characters in length, representing the name of the remote transaction

program that will be running on the partner LU. All hexadecimal combinations are acceptable **except X'40'**. In order to specify an XTPNAME over 72 characters long to DFHCSDUP, put an asterisk in column 72. This causes the following line to be concatenated.

PARTNER

Chapter 18. PROFILE

You specify options that control the interactions between transactions and terminals or logical units as a PROFILE. The PROFILE is a means of standardizing the use of such options as screen size and printer compatibility, and the use of such functions as message journaling and the node error program.

MODENAME

A profile is associated with the communication between a transaction and an LUTYPE6.1 or APPC session to another system. For APPC sessions, you refer on the PROFILE definition to the MODENAME that is also named on the SESSIONS definitions. This MODENAME is the name of the mode set to which the sessions belong. See "APPC links and parallel sessions" on page 130 and "APPC (LUTYPE6.2) single session terminal" on page 131.

When installed in CICS, the information from the PROFILE definition creates an entry in the profile table. This entry is later used by each transaction that references that PROFILE.

There are CICS-supplied PROFILE definitions suitable for most purposes. Each TRANSACTION definition names the PROFILE to be used. If you do not specify a PROFILE, the transaction uses the PROFILE supplied for using a terminal in a standard way.

With CICS intercommunication facilities (for example, function shipping), a PROFILE is needed for the communication between the transaction and the session. The attributes of the CICS-supplied profiles are shown in "PROFILE definitions in group DFHISC" on page 371. The *CICS/ESA Intercommunication Guide* gives further information about the CICS-supplied PROFILES, and tells you about defining your own profiles.

Defining a PROFILE

PROFile	==>	
Group	==>	
DEscription	==>	
Scrnsize	==> Default	Default Alternate
Uctran	==> No	No Yes
MOdename	==>	
PRIntercomp	==> No	No Yes
JOURNALLING		
Journal	==> No	No 1-99
MSGJrn1	==> No	No INPut Output INOut
PROTECTION		
MSGInteg	==> No	No Yes
Onewte	==> No	No Yes
PROtect	==> No	No Yes
Chaincontrol	==> No	No Yes
PROTOCOLS		
DVsuprt	==> All	All Nonvtam Vtam
Inbfmh	==> No	No All Dip Eods
RAq	==> No	No Yes
Logrec	==> No	No Yes
RECOVERY		
Nepclass	==> 000	0-255
RTImout	==> No	No 1-7000

Figure 32. The DEFINE panel for PROFILE

CHAINCONTROL(NO|YES)

Specify YES if the application program can control the outbound chaining of request units. You must not code PROTECT(YES) as well. If you specify CHAINCONTROL(YES), then ONEWTE(YES) means one chain, and not one terminal control output request.

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

DVSUPRT(ALL|NONVTAM|VTAM)

Indicates the devices (terminals or logical units) that are to be supported. The access method used by a particular terminal or logical unit is specified in its associated TCTTE.

ALL

The profile can be used with any terminal or logical unit.

NONVTAM

The profile can be used only with non-VTAM terminals.

VTAM

The profile can be used only with logical units.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see "How the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

INBFMH(NO|ALL|DIP|EODS) - SNA LUs only

For profiles used with logical units, using this attribute you can specify whether a Function Management Header (FMH) received from a logical unit is to be passed to the application program.

NO

The FMHs are discarded.

ALL

All FMHs (except APPC FMHs and LU6.1 ATTACH and SYNCPOINT FMHs that are processed by CICS) are passed to the application program. This value is required for function shipping transactions such as CSMI, transactions which use distributed transaction processing, and for distributed program link requests.

DIP

The batch data interchange program (DFHDIP) is to process inbound FMHs. BMS will issue a batch data interchange receive request if a BMS receive request has been issued, and a batch data interchange receive request is issued instead of a terminal control receive request.

EODS

An FMH is passed to the application program only if it indicates end of data set (EODS).

JOURNAL({NO|journal-id})

You can specify with this attribute that you want automatic journaling of messages to take place, by specifying the identifier of the journal. If you want automatic journaling, you must also specify JCT=YES or JCT=suffix in the DFHSIT macro.

NO

No automatic journaling of messages is to take place.

journal-id

The journal ID to be used for automatic journaling. This may be any number from 1 through 99. Journal number 1 is the system log.

LOGREC({NO|YES})

Enter LOGREC(YES) if the design of the application requires that each EXEC CICS RECEIVE request be satisfied by a logical record. This option allows existing 2770- and 2780-based application programs to be attached to a batch logical unit (for example, 3790 or 8100) without modification to the program.

MODENAME(name)

The name that identifies a group of sessions for use on an APPC connection. The name can be up to 8 characters in length, and must be the name of a VTAM LOGMODE entry defined to VTAM. It must not be the reserved name SNASVCMG. If you omit the modename it defaults to blanks. See the *CICS/ESA Intercommunication Guide* for more information about VTAM modenames.

If a transaction that specifies this profile has been started using an EXEC CICS START command, the MODENAME will be used for allocation of the principal facility. If a transaction performs an EXEC CICS ALLOCATE command specifying this profile, the MODENAME will be used for allocation of the alternate facility.

If you do not specify a MODENAME, CICS will select a session from any one of the mode sets that have been defined.

The CICS-supplied profile DFHCICSA is used, if PROFILE is not specified on an EXEC CICS ALLOCATE command. For function shipping, the profile DFHCICSF is always used. MODENAME is not specified on the

definition for either of these profiles, but you can add a MODENAME if you make your own copy. You must then ensure that the mode sets using your MODENAME have been defined in the TERMINAL or SESSIONS definition for all the systems with which communication will take place using APPC.

If a MODENAME is specified and you wish to remove it, delete completely the value previously specified by pressing the ERASE EOF key.

MSGINTEG({NO|YES}) - SNA LUs only

YES means that a definite response is to be requested with an output request to a logical unit. You cannot specify YES for a pipeline transaction.

MSGJRNL({NO|INPUT|OUTPUT|INOUT})

Indicates which messages are to be automatically journaled. If you specify a value other than NO, you must also supply a value for the JOURNAL attribute.

NO

No message journaling is required.

INPUT

Journaling is required for input messages.

OUTPUT

Journaling is to be performed for output messages.

INOUT

Journaling is to be performed for input and output messages.

NEPCLASS({0|value}) - VTAM only

The node error program transaction class. This value overrides the value specified on the TYPETERM and SESSION definitions.

- + **0** This results in a link to the default node error program module for VTAM devices, or is the default value for non-VTAM devices.
- +
- +

value

The transaction-class for the (non-default) node error program module. The value can be in the range 1 through 255. For programming information on the node error program, see the *CICS/ESA Customization Guide*.

ONEWTE({NO|YES})

Specify YES if the transaction is permitted only one write operation or EXEC CICS SEND during its execution. YES has the effect of forcing the LAST option on the first write of the transaction. Any additional write requests are treated as errors, and the task is made ready for abnormal termination.

You must specify YES for a PIPELINE transaction.

PRINTERCOMP({NO|YES})

This attribute defines the level of compatibility required for the generation of data streams to support the printer compatibility option for the BMS SEND TEXT command.

PROFILE

NO

Each line of output starts with a blank character, so that the format is equivalent to that on a 3270 display where an attribute byte precedes each line.

YES

No blank character is inserted, so that forms-feed characters included as the first character of your data are honored and the full width of the printer is available for your data.

If you use the BMS forms feed option, you should specify YES.

PROFILE(name)

The name of this PROFILE definition. The name can be up to eight characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >. Do not use profile names beginning with DFH, because these characters are reserved for use by CICS.

```
# Note: If you use a comma (,) in a name, you will be
#         unable to use those commands such as
#         CEMT INQUIRE PROFILE(value1,value2)
#         where the comma serves as a list delimiter. See
#         the CICS/ESA CICS-Supplied Transactions
#         manual for information about using lists of
#         resource identifiers.
```

A profile specifies the options that will control the interaction between CICS and a terminal or logical unit. A profile name is specified on the transaction definition to indicate the set of options that will control the communication between the transaction and its principal terminal. You can also specify a profile name on an EXEC CICS ALLOCATE command to indicate the options that will control communication between the transaction and the allocated session.

CICS supplies a number of profile definitions that are suitable for most purposes. For guidance on the names of the definitions, see the *CICS/ESA Operations and Utilities Guide*. Further guidance is also given in the *CICS/ESA Intercommunication Guide*.

PROTECT({NO|YES}) - SNA LUs only

Enter PROTECT(YES) to provide recovery for output message. This option provides message integrity (see the MSGINTEG option on page 165), and also causes message logging to take place. CICS also records the contents of deferred write requests that are pending at a syncpoint, and records the receipt of the definite response (associated with the deferred write) on the system log for message recovery and resynchronization purposes. Journaling support is required during generation of the CICS system.

If you specify PROTECT(YES):

- You must specify MSGINTEG(YES). This ensures the integrity response is received.

- Definitions for the transaction CSLG and program DFHZRLG must be available.

RAQ({NO|YES}) - SNA terminals only

Indicates whether the 'read ahead queuing' option is required.

NO

The transaction will obey SNA protocols and only SEND and RECEIVE when in the correct mode. If it does not follow the protocol, then it may be abended with code ATCV.

YES

The transaction may not obey SNA protocols, and CICS queues incoming data on temporary storage until the data is specifically requested by the transaction. RAQ(YES) is provided only for compatibility with transactions that support both bisynchronous devices and logical units, and its use is not recommended. If you enter RAQ(YES), the temporary storage program must be generated.

RTIMOUT({NO|value})

The time-out value for the read time-out feature. The task that is timed out will receive an AKCT or AZCT abend. (Note that if a value is specified and you wish to let it default to NO, you must completely delete the value previously specified.)

```
# RTIMOUT has no effect for MRO or basic (unmapped)
# APPC connections.
```

NO

The read time-out feature is not required.

value

This is an interval (MMSS for minutes and seconds) after which the task will be terminated if no input has been received from the terminal. The maximum value that can be specified is 70 minutes. The value specified in this option is rounded up to units of 16.78 seconds. Thus, the minimum value (after rounding-up) is 16.78 seconds.

SCRNSIZE({DEFAULT|ALTERNATE})

This attribute specifies whether the DEFAULT or ALTERNATE buffer size for a 3270 display or printer is to be used. For further information on the choice of screen sizes and buffer sizes, refer to the ALTSCREEN and DEFSCREEN attributes on the TYPETERM definition.

The SCRNSIZE value will be ignored if the TYPETERM definition has ALTSCREEN(0,0) and DEFSCREEN(0,0). That is, the screen size will be assumed from the related TERMMODEL attribute in the TYPETERM definition; the page size will be taken from PAGESIZE; the ALTPAGE value will be ignored. The 3270 EW command will be inserted for output requests with the ERASE option.

DEFAULT

If the TYPETERM definition has non-zero ALTSCREEN or non-zero DEFSCREEN, the default screen size mode will be applied, using the erase write (EW) command. That is, whenever the terminal issues a terminal output request with the ERASE option, the 3270 EW command will be inserted in the data stream. The screen size specified in the DEFSCREEN attribute will be assumed, and BMS will use the value specified in the PAGESIZE attribute as the page size.

ALTERNATE

If the TYPETERM definition has non-zero ALTSCREEN, the alternate screen size mode will be applied, using the erase write alternate (EWA) command. That is, whenever a terminal output request with the ERASE option is issued, the 3270 EWA command will be inserted in the data stream. The ALTSCREEN value will be assumed as the screen size, and BMS will use the value in ALTPAGE as the page size.

SCRNSIZE(ALTERNATE) may be used for all CICS service transactions (for example, CSMT).

Note: Both DEFAULT and ALTERNATE can be overridden by the DEFAULT and ALTERNATE options on the SEND MAP, SEND TEXT, and SEND CONTROL commands. See the *CICS/ESA Application Programming Reference* for programming information about these commands.

UCTRAN({NO}|YES) – VTAM only

This attribute specifies whether terminal input is to be translated to uppercase before passing to programs for the transaction using this profile.

You can also request translation to uppercase at the terminal level on the associated TYPETERM definition (see page 240) but you should be aware of the following points:

- A TYPETERM UCTRAN(YES) definition overrides a PROFILE UCTRAN(NO) definition. So, if you

specify TYPETERM UCTRAN(YES), a PROFILE UCTRAN(NO) has no effect.

- A PROFILE UCTRAN(YES) definition overrides a TYPETERM UCTRAN(NO) definition.
- Specifying TYPETERM UCTRAN(TRANID) causes the tranid to be translated to uppercase so that CICS can locate the transaction definition. All other input received by the application is translated according to what is specified for PROFILE UCTRAN.
- UCTRAN(YES) on a profile definition does not cause translation of the input data until an EXEC CICS RECEIVE or CONVERSE is executed. This means that if the transaction is routed through a dynamic routing program, for example DFHDYP, the copy of the input data passed to the routing program is unaffected by the UCTRAN option of the PROFILE definition.

Note: In a transaction routing environment where your VTAM terminals have a remote definition on the AOR, and the AOR has a different UCTRAN value from the TOR, the TOR value of UCTRANST (as specified in an EXEC CICS SET TERMINAL command) overrides that on the AOR.

Table 11 shows which portion of the terminal input is translated (transaction id and/or data) according to the setting of the UCTRAN on the PROFILE and TYPETERM resource definitions.

Table 11. The effect of UCTRAN parameters on tranid and data translation

Profile (PROFILE)	Terminal (TYPETERM)		
	UCTRAN (YES)	UCTRAN (NO)	UCTRAN (TRANID)
UCTRAN (YES)	Tranid: Yes Data: yes	Tranid: No Data: Yes	Tranid: Yes Data: Yes
UCTRAN (NO)	Tranid: Yes Data: Yes	Tranid: No Data: No	Tranid: Yes Data: No

PROFILE

Chapter 19. PROGRAM

You use the PROGRAM definition to describe the control information for a program that is stored in the program library and used to process a transaction, or part of a transaction. For example, this is where you would tell CICS whether the program can handle data located above the 16MB line. You can create PROGRAM definitions either by using CEDA or DFHCSDUP, or by setting the appropriate SIT options and allowing programs to be autoinstalled. See Chapter 11, "Autoinstall for programs, mapsets, and partitionsets" on page 123 for information on autoinstall for programs.

Installing PROGRAM definitions

Use the INSTALL command to install a new PROGRAM definition in your CICS system. If the program is a new version of an existing program, you must install it again using the INSTALL command, then use CEMT to make the new version available:

```
CEMT SET PROGRAM(pgmid) NEWCOPY
```

See the *CICS/ESA CICS-Supplied Transactions* for further information about CEMT.

Defining a PROGRAM

```

PROGram      ==>
Group        ==>
DEscription  ==>
Language     ==>
RELoad       ==> No           CObol | Assembler | Le370 | C | Pli | Rpg
RESident     ==> No           No | Yes
USAge        ==> Normal      Normal | Transient
USElpacopy   ==> No           No | Yes
Status       ==> Enabled     Enabled | Disabled
RS1          : 00             0-24 | Public
Cedf         ==> Yes         Yes | No
DATAlocation ==> Below       Below | Any
EXECKey      ==> User       User | CICS
REMOTE ATTRIBUTES
REMOTESystem ==>
REMOTENAME   ==>
Transid      ==>
EXECutionset ==> Fullapi     Fullapi | Dp1subset

```

Figure 33. The DEFINE panel for PROGRAM

CEDF(Yes|No)

This defines the action of the execution diagnostic facility (EDF) when the program is running under EDF control.

Yes

The EDF diagnostic screens are displayed. If the program is translated with the NOEDF option, only the program initiation and termination EDF screens are displayed. See Table 12.

No The EDF diagnostic screens are not displayed.

Table 12. The effect on programs of CEDF(NO) and NOEDF

CEDF option on PROGRAM	EDF option on translator	
	EDF	NOEDF
YES	ALL EDF screens	Program initiation and termination screens only
NO	NO EDF screens	NO EDF screens

Note: The table shows how the CEDF option on the program resource definition interacts with the EDF option specified for the translator.

DATALOCATION({BELOW|ANY})

Commands using the SET option can return a data address to an application program; this operand defines the location of the data. For example, in the command EXEC CICS RECEIVE SET(ptr-ref), ptr-ref will be less than 16MB if DATALOCATION(BELOW) is specified or defaulted to, but may be greater than 16MB if DATALOCATION(ANY) is specified. Note that DATALOCATION does not affect the operation of the GETMAIN command. See the *CICS/ESA Application Programming Reference* for programming information about where CICS obtains storage in response to a GETMAIN command.

BELOW

The program can handle only 24-bit addresses and must therefore only be given data located below the 16MB line. If necessary, data will be copied below the 16MB line before passing its address to the application program.

ANY

The program can handle 31-bit addresses. The address of the data can be above or below the 16MB line. The DATALOCATION options are independent from the addressing mode of the link-edited program. Programs link-edited with addressing mode AMODE=24 cannot access data above 16MB; you should therefore ensure that the DATALOCATION option you specify is compatible with the addressing mode of the link-edited application program. For example:

- You are recommended to specify ANY for all 31-bit programs, unless they pass CICS data addresses on to other 24-bit programs.
- Specify DATALOCATION(BELOW) for an AMODE=24 program, unless storage addresses are being passed to a program that can access storage above 16MB, or the program explicitly switches addressing mode.

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

EXECKEY(USER|CICS)

This specifies the key in which CICS gives control to the program, and determines whether the program can modify CICS-key storage. For all except reentrant programs (that is, programs link-edited with the RENT attribute), EXECKEY also defines, in conjunction with the residency mode, into which of the DSAs CICS loads the program (see note 2).

USER

This specifies that CICS is to give control to the program in user key when it is invoked. CICS loads the program into one of the user-key shared DSAs—either the SDSA or the ESDSA, depending on the residency mode specified for the program (see note 2 below).

In a CICS region with storage protection only active, a user-key program has read and write access to all user-key storage, but read-only access to CICS-key storage.

In a storage protection **and** transaction isolation environment, a user-key program has read and write access to the user-key task-lifetime storage of its own task only, and to any shared DSA storage, if the transaction is defined with ISOLATE(YES).

However, if a transaction is defined with ISOLATE(NO) in a transaction isolation environment, its user-key programs also have read and write access to the user-key task-lifetime storage of other transactions that are defined with ISOLATE(NO).

User-key programs always have read-only access to CICS-key storage.

CICS

This specifies that CICS is to give control to the program in CICS key when it is invoked. CICS loads the program into one of the CICS-key DSAs—either the CDSA or the ECDSA, depending on the residency mode specified for the program (see note 2).

In a CICS region with storage protection active, a CICS-key program has read and write access to CICS-key and user-key storage of its own task and all other tasks, whether or not transaction isolation is active.

Notes:

1. First-level global user exit programs, task-related user exit programs, user-replaceable programs, and PLT programs always execute in CICS key, regardless of the EXECKEY definition.
2. If the program is link-edited with the RENT attribute, CICS loads the program into one of the read-only DSAs—either the RDSA or the ERDSA, depending on the residency mode specified for the program, regardless of the EXECKEY option. The read-only

DSAs are allocated from read-only storage only if RENTPGM=PROTECT is specified as a system initialization parameter.

EXECUTIONSET(FULLAPI|DPLSUBSET)

This indicates whether you want CICS to link to and run a program as if it were running in a remote CICS region.

FULLAPI

Specify FULLAPI if you want CICS to link to the program and run it without the API restrictions of a DPL program. The program can use the full CICS API.

DPLSUBSET

Specify DPLSUBSET if you want CICS to link to the program and run it with the API restrictions of a remote DPL program. See the *CICS/ESA Application Programming Guide* for details of the API restrictions for a DPL program.

The EXECUTIONSET parameter applies only when the REMOTESYSTEM name is the same name as the local CICS region. Its purpose is to test programs in a local CICS environment as if they were running as DPL programs.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see “How the CSD is organized — groups and lists” on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

LANGUAGE(COBOL|ASSEMBLER|LE370|C|PLI|RPG)

indicates the program language. There is no default language, and if you omit the language, or define it incorrectly, the CICS program manager deduces the correct language and ignores your incorrect definition.

If you intend to share the CSD with a level of CICS prior to CICS/ESA 4.1, you should not leave this field blank, as it will default to COBOL in the earlier release, which may not be correct.

COBOL

This is an OS/VS COBOL or a VS COBOL II program.

ASSEMBLER

This is an assembler language program.

LE370

LANGUAGE(LE370) must be specified if the program exploits multi-language support, or if the program has been compiled by the Language Environment/370-enabled compiler.

PROGRAM

C This is a C/370 program not compiled by a Language Environment/370 enabled compiler.

PLI
This is a PL/I program.

RPG
Not applicable to CICS/ESA.

PROGRAM(name)

The name of this PROGRAM definition. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase. Do not use program names beginning with DFH, because these characters are reserved for use by CICS.

To use the program in an active CICS system, it must have been link-edited into one of the libraries specified as part of the DFHRPL DD statement or, if it is reentrant, it may have been placed in the link pack area (LPA). For more information about installing application programs, see the *CICS/ESA System Definition Guide*.

RELOAD({NO|YES})

Indicates whether a program control link, load, or XCTL request is to bring in a fresh copy of a program.

NO
Any valid copy of the program currently in storage will be reused for the request.

YES
A fresh copy of the program will be brought into storage for every request. For programming information about the RELOAD(YES) option, see the *CICS/ESA Application Programming Reference* manual. Furthermore, each of these program copies must be removed from storage explicitly, using a storage control FREEMAIN request, when it is no longer required and before the transaction terminates. If the relevant FREEMAIN(s) are not issued, areas of the DSA/EDSA will become tied up with inaccessible program copies, potentially causing storage shortage or fragmentation.

RELOAD(YES) can be used to load tables or control blocks that are modified by execution of the associated program(s). It should not be specified for the first program loaded for a task. This is because the task would have no way of issuing a FREEMAIN for the program.

You must specify RELOAD(YES) for non-reentrant programs.

REMOTENAME(name)

The name by which the program is known in the remote CICS region. If you specify REMOTESYSTEM and omit REMOTENAME, the REMOTENAME parameter defaults to the same name as the local name (that is, the PROGRAM name on this resource definition).

REMOTESYSTEM(name)

Specify the name of a remote CICS region if you want CICS to ship a distributed program link (DPL) request to another CICS region. The name you specify must be the name of the CONNECTION resource definition for the link to the remote CICS.

Note that an application program can also specify a remote system explicitly on the program LINK command, using the SYSID parameter of the EXEC CICS LINK PROGRAM(name) command. The rules of precedence are as follows:

1. If an application program issues a DPL request, and the SYSID option on the EXEC CICS LINK command specifies a remote CICS region, CICS ships the request automatically without reference to the program resource definition.
2. If an application program issues a DPL request, but the SYSID is the same name as the local CICS region or the SYSID option is not specified, CICS checks the program resource definition. If the program definition specifies a remote system name, CICS ships the request to the remote system, otherwise CICS runs the program locally.

The rules for specifying the remote system name are the same as for the CONNECTION parameter of the CONNECTION resource definition.

Note: You must not specify remote attributes for any user-written CICS programs, such as the dynamic transaction routing or autoinstall user programs.

RESIDENT({NO|YES})

The residence status of the program.

NO
The program is not to be permanently resident. This value must be specified if RELOAD(YES) is specified.

YES
The program is to be loaded on first reference and is then to be permanently resident in virtual storage, but is to be pageable by the operating system. When you specify RESIDENT(YES), CICS assumes a specification of USAGE(NORMAL).

RSL

The RSL keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

STATUS({ENABLED|DISABLED})

The program status.

ENABLED
The program may be used.

DISABLED
The program may not be used.

TRANSID(name)

Specifies the name of the transaction you want the remote CICS to attach, and under which it is to run the remote program. If you do not specify a transaction name on the TRANSID parameter, the remote region executes the DPL program under one of the following CICS-supplied default mirror transactions:

- CPMI** This is the CICS mirror transaction for LU6.2 connections that specify data conversion.
- CSMI** This is the CICS ISC mirror transaction for MRO and LU6.2 connections with sync level 2.
- CVMI** this is the CICS/VM mirror transaction for LU6.2 connections with sync level 1.

USAGE({NORMAL|TRANSIENT})

Defines when the storage for this program will be released.

NORMAL

When the use count for this program reaches zero, it will become eligible for removal from storage as part of the normal dynamic program compression process.

This value must be specified if RELOAD(YES) is specified.

TRANSIENT

When the use count for this program becomes zero, the storage for this program is released. This value should be coded for programs that are referenced infrequently.

USELPACOPY({NO|YES})

Defines whether the program is to be used from the link pack area (LPA).

NO

The program is not to be used from the LPA. It will be loaded into the CICS address space.

YES

The program can be used from the LPA if LPA=YES is specified as a system initialisation parameter. The use of the program from the LPA requires that it has been installed there and that the program is not named by the PRVMOD start-up option. For guidance on this, see the *CICS/ESA Installation Guide*.

PROGRAM

Chapter 20. SESSIONS

Before two systems can communicate using intersystem communication (ISC) or multiregion operation (MRO), they must be logically linked through one or more sessions. The nature of the logical link determines how they can communicate. You specify the logical link in the SESSIONS definition. CICS does **not** use the SESSIONS name when the definition has been installed in the active system. This name is used only to identify the definition in the CSD.

You use the CONNECTION attribute of the SESSIONS resource definition to name the CONNECTION with which these SESSIONS are associated when they are installed in the active systems.

If you use the INSTALL command to install a new SESSIONS definition for MRO, when there is already a definition installed, you must then close down all interregion communication (IRC) and open it again, before you will be able to use the definition. You should adopt the following procedure:

1. Close IRC down:

```
CEMT SET IRC CLOSED
```

2. Install the resource definitions:

```
CEDA INSTALL GROUP(groupname)
```

3. When you have successfully installed the group containing the definitions:

```
CEMT SET IRC OPEN
```

MRO links and sessions

When you install a SESSIONS definition for MRO, you are telling CICS about a set of parallel sessions between this CICS and another CICS. The number of sessions is determined by the SENDCOUNT and RECEIVECOUNT attributes. The SEND sessions are identified by names created from the SENDPFX and SENDCOUNT. The RECEIVE sessions are identified by names created from the RECEIVEPFX and RECEIVECOUNT.

APPC (LUTYPE6.2) links and parallel sessions

When you install the SESSIONS definition, the sessions are grouped (for the benefit of VTAM) into a modeset, which is identified by the MODENAME. The individual sessions are named by a counter; the first session created will be named -999, the second -998, and so on. The value of this counter is retained over a warm or emergency start. The number of sessions created is controlled by the MAXIMUM attribute on the SESSIONS definition.

LUTYPE6.1 CICS-CICS ISC links and sessions

The way in which the sessions are identified by CICS depends on the way you defined them, using SENDPFX, SENDCOUNT, RECEIVEPFX, and RECEIVECOUNT like MRO sessions, or using SESSNAME as for CICS-IMS sessions.

Note: You should use APPC for all new CICS-CICS ISC links.

LUTYPE6.1 CICS-IMS links and sessions

When you install the SESSIONS definitions in the active CICS system, CICS identifies each session by the SESSNAME.

INDIRECT connections

Because the association between an INDIRECT link and the intermediate systems used for communicating with it is made at installation time, it is advisable for you to install the definition for the intermediate system before the definition for the INDIRECT link. If you install the INDIRECT link first, it will remain dormant until the intermediate definition is installed, and until any other already-installed connections which make reference to it are resolved. For example, System A is indirectly connected with system C through system B.

SESSIONS

In system A, you should install the following definitions, in this order:

1. The intermediate system:

```
CONNECTION(B) NETNAME(B) ACCESSMETHOD(IRC) ...
```

2. The INDIRECT link

```
CONNECTION(C) NETNAME(C) ACCESSMETHOD(INDIRECT) INDSYS(B) ...
```

Defining a SESSION

```
Sessions      :
Group         :
Description   ==>
SESSION IDENTIFIERS
Connection    ==>
SESSName     ==>
NETnameq     ==>
M0dename     ==>
SESSION PROPERTIES
Protocol      ==> Appc           Appc | Lu61 | Exci
Maximum      ==> 001 , 000      0-999
RECEIVEPfx   ==>
RECEIVECount ==>                1-999
SENDPfx      ==>
SENDCount    ==>                0-999
SENDSize     ==> 04096          1-30720
RECEIVESize  ==> 04096          1-30720
SESSPriority ==> 000            0-255
Transaction  :
OPERATOR DEFAULTS
OPERId       :
OPERPriority : 000              0-255
OPERRsl      : 0
OPERSecurity : 1
PRESET SECURITY
USERId       ==>
OPERATIONAL PROPERTIES
Autoconnect  ==> No             No | Yes | All
INservice    :
Buildchain   ==> Yes           Yes | No
USERArealen ==> 000            0-255
IOarealen    ==> 00000 , 00000 0-32767
RELreq       ==> No           No | Yes
DIScreq      ==> No           No | Yes
NEPclass     ==> 000          0-255
RECOVERY
RECOVOption  ==> Sysdefault    Sysdefault | Clearconv | Releasesess
| Uncondrel | None
RECOVNotify  : None           None | Message | Transaction
```

Figure 34. The DEFINE panel for SESSIONS

AUTOCONNECT({NO|YES|ALL})

Indicates how connections are to be established. What you have to specify for LU6.1 and APPC sessions is discussed below:

APPC

For a VTAM-connected system that has AUTOCONNECT(YES) or (ALL) on the CONNECTION definition:

NO

CICS will not attempt to bind any sessions when the connection is established. However, one or more user sessions may be allocated as part of any ACQUIRE CONNECTION processing which takes place.

YES or ALL

Contention winner session is established (that is, BIND is performed) during CICS initialization, or when communication with VTAM is started using the CEMT SET VTAM OPEN command. If the connection cannot be made at this time because the remote system is unavailable, the link must be subsequently acquired using the CEMT SET CONNECTION(sysid) INSERVICE ACQUIRED command, unless the remote system becomes available in the meantime and itself initiates communications.

+ You should not specify ALL for connections to
+ other CICS systems. This can lead to a bind
+ race condition, and is discussed in the
+ *CICS/ESA Intercommunication Guide*.

For a VTAM-connected system that has AUTOCONNECT(NO) on the CONNECTION definition:

NO

CICS will not attempt to bind any sessions when the connection is established. However, one or more user sessions may be allocated as part of any ACQUIRE CONNECTION processing which takes place.

YES

Contention winner sessions will be established when the connection is acquired by issuing CEMT SET CONNECTION(sysid) ACQUIRED, or when the remote system itself initiates communication.

ALL

All sessions, not just contention winners, will be established when the connection is acquired by issuing CEMT SET CONNECTION(name) ACQUIRED, or when the remote system itself initiates communication.

LU6.1

Specify AUTOCONNECT(YES) on the SESSIONS if you want the connection to be established at initialization or CEDA install.

Specify AUTOCONNECT(NO) on the SESSIONS if you don't want the connection to be established at initialization or CEDA install.

BUILDCHAIN({YES|NO})

Indicates whether CICS is to perform chain assembly prior to passing the input data to the application program.

YES

Any terminal input/output area (TIOA) received by an application program from this logical unit will contain a complete chain.

NO

Any TIOA received by an application program from this logical unit will contain one request unit (RU).

CONNECTION(name)

The name of the CONNECTION definition that you want to use with this SESSIONS definition. The name can be up to four characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

Note that the CONNECTION definition must be in the same GROUP as the SESSIONS definition.

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

DISCREQ({NO|YES})

Indicates whether disconnect requests are to be honored. DISCREQ applies to LUTYPE6.1 ISC sessions (see the Release Guide for ISSUE DISCONNECT (LUTYPE6.1)), but not to MRO sessions where CICS is not dealing with VTAM devices.

DISCREQ does not apply to APPC (LUTYPE6.2) sessions. When using APPC, individual sessions are acquired as transactions need them, then are subsequently freed. As it is possible to have multiple sessions between APPC logical units, there should never be a problem of one request holding up another. It is not possible to disconnect an individual APPC session; instead, you can issue a CEMT SET CONNECTION RELEASED command.

NO

CICS is not to honor a disconnect request for a VTAM device.

SESSIONS

YES

CICS is to honor a disconnect request for a VTAM device, and issue a VTAM CLSDST macro instruction to terminate the VTAM session with that logical unit.

CESF LOGOFF or GOODNIGHT commands issued from the terminal also cause disconnection if you code DISCREQ(YES).

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see "How the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

INSERVICE({YES|NO})

The INSERVICE keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

IOAREALEN({0|value1},{0|value2})

The length, in bytes, of the terminal input/output area to be used for processing messages transmitted on the MRO link.

(value1, value2)

Value1 specifies the minimum size of a terminal input/output area to be passed to an application program when a RECEIVE command is issued.

If value2 is not specified, or is less than value1, it defaults to the value of value1.

You can specify value2 as greater than or equal to value1. In this case, when the size of an input message exceeds value1, CICS uses a terminal input/output area value2 bytes long. If the input message size also exceeds value2, the node abnormal condition program sends an exception response to the terminal.

You can waste both real and virtual storage by specifying an IOAREALEN value that is too large for most messages transmitted on your MRO link. On the other hand, if you specify an IOAREALEN value that is either zero or smaller than most of your messages, excessive FREEMAIN and GETMAIN activity may occur. This will result in additional processor requirements.

MAXIMUM({1|value1},{0|value2}) (APPC only)

Indicates the maximum number of sessions that are to be supported for the modeset. Value1 must be greater than or equal to value2.

1|value1

The maximum number of sessions in the group. This value can be in the range 1 through 999. The default is 1.

0|value2

The maximum number of sessions that are to be supported as contention winners. This value can be in the range 0 to 999. The default is 0. Note that this operand has no meaning for a single session connection. (For further information on the effects of the MAXIMUM option, refer to the *CICS/ESA Intercommunication Guide*.)

SNA allows some resources (for example, switched lines) to be defined in the network as **limited resources**. At bind-time, VTAM indicates to CICS whether the bind is over a limited resource. When a CICS task frees a session across a limited resource, CICS unbinds the session if no other task wants to use it. If the sessions are to use limited resources, specify MAXIMUM(value1,0). This causes any unbound session to be reset so that either side can then bind it as a winner when it is next required. For more information on limited resources, see the *CICS/ESA Intercommunication Guide*.

MODENAME(name) (APPC only)

The name that identifies a group of sessions for use on an APPC connection. The name can be up to eight characters in length, and must be the name of a VTAM LOGMODE entry defined to VTAM. It must not be the reserved name SNASVCMG. If you omit the modename it defaults to blanks. See the *CICS/ESA Intercommunication Guide* for more information about VTAM modenames.

The MODENAME must be unique for each group of sessions defined for any one intersystem link. That is, the MODENAME must be unique among the SESSIONS definitions related to one CONNECTION definition. It will be passed to VTAM as the LOGMODE name.

NEPCLASS({0|value})

The node error program transaction class. This value acts as the default.

0 This will result in a link to the default node error program module.

value The transaction-class for the (non-default) node error program module. The value can be in the range 1 through 255. For programming information about the node error program, see the *CICS/ESA Customization Guide*.

NETNAMEQ(name)

The name by which the remote IMS system knows this particular session. This is used for CICS-IMS sessions. The name can be up to eight characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ? ! : | " = ~ , ; < and >.

OPERID(code)

The OPERID keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

OPERPRIORITY({0|number})

The OPERPRIORITY keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

OPERRSL({0|number[,...]})

The OPERRSL keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

OPERSECURITY({1|number[,...]})

The OPERSECURITY keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

PROTOCOL({APPC|LU61|EXCI})

Indicates the type of protocol that is to be used for an intercommunication link (ISC or MRO).

APPC (LUTYPE6.2)

Advanced program-to-program communication (APPC) protocol. Specify this for CICS-CICS ISC.

LU61

LUTYPE6.1 protocol. Specify this for CICS-CICS ISC, for CICS-IMS, or for MRO.

EXCI

The external CICS interface. Specify this to indicate that the sessions are for use by a non-CICS client program using the external CICS interface. If you specify EXCI, you must leave SENDCOUNT blank.

RECEIVECOUNT({blank|number})

The number of MRO, LUTYPE6.1, or EXCI sessions that usually receive before sending.

For MRO, receive sessions can only receive before sending.

blank

These sessions can send only; there are no receive sessions.

number

Specifies the number of receive sessions on connections that specify blank, LU61, or EXCI on the protocol parameter of the CONNECTION definition. CICS uses the number to generate the last two or three characters of the session names (see RECEIVEPFX for details).

If you are using the default receive prefix (<), or your own 1-character prefix, specify a number in the range 1 through 999.

If you specify a 2-character prefix, the number is restricted to the range 1 through 99.

Except for external CICS interface (EXCI) connections, the RECEIVECOUNT in this system should equal SENDCOUNT in the other system.

Note: CICS restricts the number of sessions for an EXCI address space to 25. When this limit is reached, IRP rejects further requests for a session with SYSTEM_ERROR reason code 608.

RECEIVEPFX(<|prefix)

Specifies a 1- or 2-character prefix that CICS is to use as the first 1 or 2 characters of the receive session names (the names of the terminal control table terminal entries (TCTTEs) for the sessions).

Prefixes must not cause a conflict with an existing connection or terminal name.

< (MRO and EXCI sessions)

For MRO sessions, if you do not specify your own receive prefix, CICS enforces the default prefix—the less-than symbol (<), which is used in conjunction with the receive count to generate receive session names.

CICS creates the last three characters of the session names from the alphanumeric characters A through Z, and 1 through 9. These 3-character identifiers begin with the letters AAA, and continue in ascending sequence until the number of session entries reaches the limit set by the RECEIVECOUNT value. Note that receive session names are generated **after** the send sessions, and they follow in the same sequence.

For example, if the last session name generated for the send sessions is <AAJ, using the default prefix (<) CICS generates the receive session names as <AAK, <AAL, <AAM, and so on. (This method of generation of session identifiers is the same as for APPC sessions, except for the initial prefix symbol.)

Note: If you specify your own prefix, CICS generates the session names as in earlier releases, which is the same as for LUTYPE6.1 sessions.

prefix (LUTYPE6.1 sessions)

If the sessions are on LUTYPE6.1 ISC connections, you must specify a 1- or 2-character prefix. Do not use the default < symbol for LUTYPE6.1 sessions.

For LUTYPE6.1 sessions (and MRO if you specify your own 1- or 2-character prefix) CICS generates session names by appending a number to the prefix, either in the range 1 through 99, or 1 through 999. The number begins with 1 and is incremented by 1 until the specified RECEIVECOUNT is reached.

RECEIVE SIZE({4096|value})

Indicates the maximum VTAM request unit (RU) size that these sessions are capable of receiving. The value

SESSIONS

must be between 1 and 30720 for LU61 sessions, or 256 and 30720 for APPC sessions. The default is 4096.

The value specified will be transmitted to the connected logical unit. This value may be rounded down by CICS, depending on what value you specified, because the value must be transmitted in an architected form. The value may be negotiated down still further at BIND time.

If CICS is the secondary LU session, then this indicates the maximum VTAM request unit (RU) size that these sessions are capable of sending.

RECOVNOTIFY({NONE|MESSAGE|TRANSACTION})

The RECOVNOTIFY keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

RECOVOPTION({SYSDEFAULT|CLEARCONV|RELEASESESS|UNCONDREL|NONE})

This option applies to the recovery of sessions in a CICS region running with VTAM persistent sessions, or with XRF.

VTAM persistent sessions: In a CICS region running with persistent session support, this option specifies how you want CICS to recover the session, and return the terminal to service on system restart within the persistent session delay interval.

XRF: In a CICS region running with XRF support, this option specifies how you want CICS to recover the session, and return the terminal to service after an XRF takeover.

For all recovery options other than NONE, if the action taken is a VTAM UNBIND, the UNBIND is followed by a VTAM SIMLOGON.

SYSDEFAULT

VTAM persistent sessions: In a CICS region running with persistent sessions support, this specifies that CICS is to select the optimum procedure to recover a session on system restart within the persistent session delay interval, depending on the session activity and on the characteristics of the terminal.

Although sessions are recovered, any transactions in-flight at the time of the failure are abended and not recovered. Transactions are also abended if the recovered session is being used by another CICS region over an APPC connection.

CICS recovers the session with the least possible impact, in one of the following ways:

- If the session was not busy at the time that CICS/ESA failed, no action is required.
- If the session was busy at the time that CICS/ESA failed, CICS issues a DEALLOCATE(ABEND) (equivalent to an EXEC

CICS ISSUE ABEND) for the APPC conversation in progress at the time of the failure.

- If neither of the above applies, the session is unbound.

XRF: If AUTOCONNECT(YES) is specified, the session is restarted. If AUTOCONNECT(NO) is specified, the session is unbound.

CLEARCONV

VTAM persistent sessions: CLEARCONV is not supported for APPC sessions. It defaults to SYSDEFAULT.

XRF: If AUTOCONNECT(YES) is specified, the session is restarted. If AUTOCONNECT(NO) is specified, the session is unbound.

RELEASESESS

VTAM persistent sessions: CLEARCONV is not supported for APPC sessions. It defaults to SYSDEFAULT.

XRF: If AUTOCONNECT(YES) is specified, the session is restarted. If AUTOCONNECT(NO) is specified, the session is unbound.

UNCONDREL

Requires CICS to send an UNBIND request to release the active session. The UNBIND is sent whether or not the session was busy at the time of system restart (in the case of persistent sessions support) or the takeover (in the case of XRF).

NONE

VTAM persistent sessions: In a CICS region running with persistent sessions support, this specifies that the session is not to be recovered at system restart within the persistent session delay interval: in effect, the sessions on the modegroup have no persistent sessions support. LU6.2 sessions are unbound and the modegroup CNOS value is reset to zero. After system restart, the session is reconnected automatically if you specify AUTOCONNECT(YES).

XRF: In a CICS region running with XRF support, this specifies that the logon state is not tracked by the alternate system, and the terminal session is not automatically recovered after a takeover; in effect, the terminal has no XRF support. After takeover, the terminal is reconnected automatically by the alternate system, if you specify AUTOCONNECT(YES).

RELREQ=({NO|YES})

Indicates whether CICS is to release the logical unit.

NO

CICS is not to release the logical unit upon request by another VTAM application program.

YES

CICS is to release the logical unit, if the logical unit is not currently part of a transaction.

SENDCOUNT(blank|number)

The number of MRO or LUTYPE6.1 sessions that usually send before receiving.

For MRO, send sessions must send before they can receive.

blank

These sessions can receive only; there are no send sessions.

You must leave this field blank when the sessions are on an external CICS interface (EXCI) connection.

number

Specifies the number of send sessions on connections that specify blank or LU61 on the protocol parameter of the CONNECTION definition. CICS uses the number to generate the last two or three characters of the session names (see SENDPFX for details).

If you are using the default send prefix (>), or your own 1-character prefix, specify a number in the range 1 through 999.

If you specify a 2-character prefix, the number is restricted to the range 1 through 99.

Except for external CICS interface (EXCI) connections the SENDCOUNT in this system should equal RECEIVECOUNT in the other system.

SENDPFX(>|prefix)

Specifies a 1- or 2-character prefix that CICS is to use as the first 1 or 2 characters of the send session names (the names of the terminal control table terminal entries (TCTTEs) for the sessions).

Prefixes must not cause a conflict with an existing connection or terminal name.

> (MRO sessions)

For MRO sessions, if you do not specify your own send prefix, CICS enforces the default prefix—the greater-than symbol (>), which is used in conjunction with the send count to generate send session names.

CICS creates the last three characters of the session names from the alphanumeric characters A through Z, and 1 through 9. These 3-character identifiers begin with the letters AAA, and continue in ascending sequence until the number of session entries reaches the limit set by the SENDCOUNT value.

For example, using the default prefix (>) CICS generates session names as >AAA, >AAB, >AAC, and so on. (This method of generation of session

identifiers is the same as for APPC sessions, except for the initial symbol.)

Note: If you specify your own prefix, CICS generates the session names as in earlier releases, which is the same as for LUTYPE6.1 sessions.

prefix (for LUTYPE6.1 sessions)

If the sessions are on LUTYPE6.1 ISC connections, you must specify a 1- or 2-character prefix. Do not use the default > symbol for LUTYPE6.1 sessions.

For LUTYPE6.1 sessions (and MRO if you specify your own 1- or 2-character prefix) CICS generates session names by appending a number to the prefix, either in the range 1 through 99, or 1 through 999. The number begins with 1 and are incremented by 1 until the specified SENDCOUNT is reached.

SENDSIZE({4096|value})

Indicates the maximum VTAM request unit (RU) size that these sessions are capable of sending. The value must be between 1 and 30720 for LU61 sessions, or between 256 and 30720 for APPC sessions. The default is 4096. The value may be negotiated down at bind time. Increasing the value of SENDSIZE will cause more storage to be allocated for the session but may decrease the number of physical messages sent between the two nodes.

If CICS is the secondary LU session, then this parameter indicates the maximum VTAM request unit (RU) size that these sessions are capable of receiving. The value must be between 256 and 30720.

SESSIONS(name)

The name of this SESSIONS definition. The name can be up to eight characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >.

This name is used to identify the SESSIONS definition on the CSD. It is not used within the active CICS system.

SESSNAME(name)

The symbolic identification to be used as the local half of a session qualifier pair in a CICS intercommunication parallel session.

The name can be up to four characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . & ! : ; " , and ~. If you type a name of less than four characters, it will be padded with trailing blanks to four characters.

SESSPRIORITY({0|number})

Establishes the terminal priority. This decimal value (0 through 255) is used in establishing the overall transaction processing priority. (Transaction processing priority is equal to the sum of the terminal priority, transaction priority, and operator priority; this must not exceed 255.)

SESSIONS

TRANSACTION(name)

This keyword is now obsolete, but is supported to provide CSD compatibility for earlier releases of CICS where it is still valid. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

+ **USERAREALEN({0|value})**

- + Specify this with the length, in bytes (0 to 255), of the user area for this terminal. It should be made as small as possible. The TCT user area is initialized to zeros when the session is installed.
- + The TCT user area may be located above or below the 16Mb line in virtual storage. Where it is located depends on the value of the TCTUALLOC operand of the DFHSIT macro. You should ensure that this is specified

- + correctly to allow successful operation of any programs that are not capable of handling 31-bit addressing.

USERID(name)

A user identifier used for signon (SEC=YES or MIGRATE) and referred to in security error messages, security violation messages and the audit trail. It must be a valid userid defined to the security manager. Operators are unable to sign on. All access to protected resources depends on USERID.

This USERID overrides a SECURITYNAME specified on the CONNECTION definition.

The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

Chapter 21. TERMINAL

CICS needs a definition for each terminal device with which it communicates. Terminal devices include visual display units, printers, operating system consoles, and more specialized devices such as facsimile machines.

The unique and possibly dynamic properties of terminals are defined in the TERMINAL definition in the CSD.

However, many of your terminals have identical properties, and it is unnecessary to define each of them separately and fully to CICS. There are two ways you can reduce the time and effort needed to define each terminal. They are:

- **TYPETERM definitions**, with or without the QUERY function. Each TERMINAL definition must refer to a TYPETERM definition, which defines the common, often more complex, and usually static properties. Together, information from the TERMINAL and TYPETERM definitions makes up a terminal entry in the TCT (a TCTTE).

One TYPETERM can represent a lot of the properties of many terminals. Some of these properties can be left undefined at the time of creating the TYPETERM definition. These properties can be determined, for each terminal, from the QUERY structured field at logon time.

- **Autoinstall**: using one TERMINAL definition to represent many terminals. In addition, autoinstall reduces the virtual storage required for the terminal control table (TCT) if some of your terminals are not logged on when CICS is active.

There are, however, still more properties that many terminals have in common, to the extent that their TERMINAL definitions would all be identical. So a facility is provided that avoids the need for each terminal to have its own resource definition installed in the TCT the whole time CICS is active.

Instead, you let CICS create and install the resource definition dynamically when the terminal is needed, at logon time. To do this, CICS uses a **model TERMINAL definition** from the CSD. This process is known as automatic installation, or autoinstall.

If you are involved in planning for and managing CICS communications resources such as terminals, you should read Chapter 9, "Autoinstall for VTAM terminals" on page 105 for further information.

Installing TERMINAL definitions

For groups of TERMINAL definitions, adopt the following procedure:

1. Make sure that nobody is using the terminals which you want to install. Remember that CEMT INQUIRE TERMINAL puts a lock on the TCT entry and this also prevents installation of a group containing that terminal.
2. Make sure that there are no ATIs outstanding for the terminals.
3. Use CEMT, with a generic name if appropriate:
CEMT SET TERMINAL(trmid) RELEASED OUTSERVICE NOATI
4. Install the resource definitions:
CEDA INSTALL GROUP(groupname)
5. Use CEMT to make the terminals available again:
CEMT SET TERMINAL(trmid) ACQUIRED INSERVICE ATI

TERMINAL and TYPETERM definitions are resolved at installation to become a TCT entry, sometimes known as a TCT terminal entry or TCTTE. Each definition contains only some of the information necessary to build a TCT entry.

Each TYPETERM definition must be installed before, or at the same time as the TERMINAL definitions that reference it. When you are using CEDA to install groups, if the TYPETERMs

TERMINAL

are in a separate group from the TERMINALs, you must install the TYPETERMs group before the TERMINALs. You may include TYPETERMs and TERMINALs in the same group for testing purposes, although we do not recommend it for long term use. (See “What should be in a group?” on page 18.)

Because the TERMINAL definition is not necessarily in the same group as its associated TYPETERM definition, the global catalog is used to store the TYPETERMs.

Changing a TYPETERM definition and then reinstalling it has no effect on an already installed terminal entry, even though the TERMINAL definition used to create it refers to the TYPETERM. To change the terminal entry, you must reinstall both the TYPETERM and the TERMINAL.

Installing autoinstall model TERMINAL definitions

If you define a model TERMINAL for autoinstall, you install it just as you would an ordinary resource definition, that is, by installing the group containing it. However, TERMINAL definitions specified as AUTINSTMODEL(ONLY) are only stored in the AMT at this time; they do not result in a TCTTE on the active CICS system. These model TERMINAL definitions are stored in the AMT until needed by CICS to create a definition for an actual terminal logging on through VTAM using autoinstall. The resulting TERMINAL definition is “automatically installed” at this time.

This is what happens when you install a TERMINAL definition, either at system initialization or using INSTALL:

AUTINSTMODEL(NO)

A TCT entry is created for the terminal.

AUTINSTMODEL(YES)

A TCT entry is created for the terminal, and the model definition is stored for later use by the autoinstall process.

AUTINSTMODEL(ONLY)

The model definition is stored for later use by the autoinstall process.

If you install two model TERMINAL definitions with the same AUTINSTNAME, the second one will replace the first.

For further information about autoinstall and model TERMINAL definitions, see Chapter 9, “Autoinstall for VTAM terminals” on page 105.

Types of terminals

Different types of terminal have different characteristics, and the way you use them may vary, too. Here is some guidance on defining terminals that are used in particular ways:

- Terminals for printing
- Pipeline terminals for VTAM pooled sessions
- Devices with LDC lists
- APPC single session terminals.

Terminals for printing

A TERMINAL definition for a display device can name TERMINAL definitions for printers, using the PRINTER and ALTPRINTER attributes. Such a reference is not resolved when the TERMINAL definitions are installed. Instead, the reference is resolved when the printer is needed by the display device.

There are several ways in which printed output can be created and sent to a printer.

- BMS page building
- Screen copying, using one of the following:
 - A hardware copy key
 - A local copy key
 - The ISSUE PRINT command.

For programming information about creating output by these methods, see the *CICS/ESA Application Programming Reference* manual.

The TYPETERM and TERMINAL definitions are used for both printers and display devices. There are a number of attributes that apply only to printers, or have special meanings for printers. There are also some attributes that you need to specify for a display device that is to be used for screen-copying.

Printers

You need a TERMINAL definition for each printer. Specify NO for AUTINSTMODEL, unless you are using autoinstall for printers. (For more information about this, see “Autoinstall and output-only devices” on page 109.)

TERMINAL

The name of the printer is the TERMINAL name on the resource definition for that printer.

(The PRINTER attribute is used on a display device definition, to refer to a printer device to be used for output from the display. You do not specify PRINTER on the printer definition. See “Associating printers with display devices” on page 186.)

DEVICE

The TERMINAL definition for each printer must refer to a TYPETERM with an appropriate DEVICE type. The DEVICE attribute, and in one case, the SESSIONTYPE attribute, determine whether a TYPETERM defines printers or display devices. The values that you can specify for printers are:

DEVICE	SESSIONTYPE	Printers
3270P	-	All printers that support the 3270 data stream (not SNA-connected).
LUTYPE3	-	All printers that support the 3270 data stream (SNA-connected).
SCSPRINT	-	All printers that support the SNA character set (SNA-connected).
3790	SCSPRINT	IBM 3793 keyboard-printers that support the SNA character set (SNA-connected).

AUTOPAGE

AUTOPAGE must be YES for printers, but you do not need to worry about it, because RDO fills it in for all printer DEVICE types. Autopaging means that BMS multiple page messages are printed continuously, without operator intervention. This is what is normally required for a printer. (Contrast the requirement for multiple page messages, displayed on a 3270-type display, when the operator wants to finish reading a page, before requesting the next page to be delivered.)

Only BMS SEND commands with the PAGING option use autopaging. BMS SEND with TERMINAL or SET, does not use autopaging.

You need at least one TYPETERM definition for each type of printer you use. You may need more, if you want to allow printers to be used only for some functions and not for others.

PAGESIZE

ALTPAGE

For BMS, the PAGESIZE attribute determines the default page size, and also the size of the print buffer. You specify the number of lines in the page (the length) and the number of characters in each line (the width).

Another attribute, ALTPAGE, indicates the page size to be used when the alternate screen size (ALTSCREEN) is selected. The width you specify in ALTPAGE must be the same as the width specified in the ALTSCREEN attribute. However, the length of ALTPAGE and ALTSCREEN can be different. This could be useful if you are using the same BMS map to display and to print. For instance, you could make the screen one line longer than the page, to reserve the bottom line of the screen for error messages.

The ALTPAGE, DEFSCREEN, and ALTSCREEN attributes do not normally apply to printers.

FORMFEED

You need to define FORMFEED as YES for BMS page building.

Associating printers with display devices

If you want to copy the contents of a screen direct to a printer you need to associate a specific printer with the display device, in the TERMINAL definition. You do this using the PRINTER and ALTPRINTER attributes.

PRINTER

ALTPRINTER

The alternative printer is used if the primary printer is unavailable. Both these attributes can be set dynamically by the autoinstall control program, if the display device definition is autoinstalled.

PRINTERCOPY

ALTPRINTCOPY

You specify YES for PRINTERCOPY, if you want to use the hardware COPY feature on the primary printer.

You specify YES for ALTPRINTCOPY, if you want to use the hardware COPY feature on the alternative printer.

For programming information about screen copying, see the *CICS/ESA Application Programming Reference* manual and for further guidance see the *CICS/OS/VS 3270 Data Stream Device Guide*.

Pipeline TERMINAL definitions

When you define a 3600 pipeline logical unit, you generate a TCTTE which is associated with a pool of TCTTEs. As messages enter CICS from the 3600 pipeline logical unit, a task is attached to process this message, using as anchor block one of the TCTTEs from the pool. In this way, consecutive messages sent via the pipeline logical unit can be processed concurrently, the number of concurrent transactions being limited by the number of TCTTEs in the pool. The number of TCTTEs in the pool should represent the high water mark of inquiry activity. In this way, the pipeline facility allows less TCTTEs to be defined to CICS than the total number of pipeline inquiry terminals.

All the TERMINAL definitions within a named POOL must be in the same group on the CSD. TASKLIMIT must have been specified on at least one of the definitions in the group. If it is specified on more than one definition, the maximum value of TASKLIMIT over the definitions in the group is used.

TERMINAL and TYPETERM definitions are resolved just as they are for ordinary terminals.

You should install PIPELINE terminals by GROUP. If you do not, the results can be unpredictable.

Defining pipeline terminals

You define pipeline terminals with the following attributes:

POOL

All the TERMINAL definitions having the same POOL name belong to the same pipeline pool. The presence of a value for the POOL attribute distinguishes these from ordinary TERMINAL definitions.

NETNAME

Names the VTAM session that is used.

TASKLIMIT

You must specify the maximum number of concurrent tasks that can be active for the pool of terminals on at least one of the TERMINAL definitions.

One TYPETERM would normally suit all the definitions. The TYPETERM may be in another group.

SESSIONTYPE

You use this attribute on the TYPETERM definition to identify the TYPETERM as representing pipeline terminals. You specify PIPELINE as the value.

Pipeline transactions are associated with a PROFILE definition that has the ONEWTE attribute. The programs associated with these transactions are only permitted one write or EXEC CICS SEND operation, or else they are terminated with an ATCC abend code. This means that CICS tasks rapidly appear and disappear across the pool of sessions.

There is an example of definitions for a pool of pipeline terminals on page 200.

Devices with LDC lists

For 3600, 3770 batch, 3770 and 3790 batch data interchange, and LUTYPE 4 logical units, you can specify the name of an LDC (logical device code) list. The list specifies which LDCs are valid for this logical unit and, optionally, which device characteristics are valid for each LDC. The first LDC in this list is the default when CICS must choose a default LDC for a logical unit.

All the TERMINAL definitions for devices that reference a particular LDC list must name the same TYPETERM, because the LDCLIST attribute is on the TYPETERM definition.

The LDC list itself must be defined using a DFHTCT TYPE=LDCLIST macro. It may be a local LDC list or an extended local LDC list. There is an example of the coding for each on page 231, and further guidance in "Logical device codes" on page 302. You use the label coded on the macro instruction to identify the LDC list on the TYPETERM definition, specifying it in the LDCLIST attribute.

APPC (LUTYPE6.2) single session terminal

An APPC (LUTYPE6.2) single session terminal can be defined as a TERMINAL, with a reference to a TYPETERM with DEVICE(APPC). When these definitions are installed, the resources they define are known to CICS as a connection and a modeset, just as they are when defined in RDO as CONNECTION and SESSIONS. The name of the connection is the TERMINAL name and the name of the modeset is the MODENAME on the TERMINAL definition.

TERMINAL

An APPC terminal may be a PS/2, an Application System/400 (AS/400), a System/38, or similar. You can define an APPC terminal either by a TERMINAL-TYPETERM definition or by a CONNECTION-SESSIONS definition. Both kinds of definition can be autoinstalled (see Chapter 9, “Autoinstall for VTAM terminals” on page 105 and Chapter 10, “Autoinstall for APPC connections” on page 119 for information about autoinstall). If you decide to use the TERMINAL-TYPETERM method, the following attributes are important:

DEVICE

The TERMINAL definition references a TYPETERM with APPC (advanced program-to-program communications) specified as the DEVICE. One such TYPETERM definition suffices for many terminals.

MODENAME

This is the name that CICS uses to identify the session when the definition is installed in the active system.

Terminals for transaction routing

Transaction routing enables terminals in one CICS system to invoke transactions in another CICS system.⁵ You can use transaction routing between systems connected by MRO or by an APPC link. (See the *CICS/ESA Intercommunication Guide* for guidance.) If you use transaction routing, there are three possible methods of defining the terminals.

When a terminal belonging to (local to and fully defined in) one system invokes a transaction belonging to another system, it is known to the application-owning region as a **remote terminal**. The application-owning system needs to have access to at least a partial definition of the remote terminal. This partial definition is often known as a **remote definition**.

The terminal-owning system has a complete definition of the terminal: this is known as a **local definition**.

Making a partial definition of a terminal available to the application-owning CICS system(s) can be done in one of the following ways:

- Having a separate terminal definition for each system involved, each on a CSD that is accessible to that system. One is created as a local definition and one or more are created as remote definitions. We refer to this as **duplicating terminal definitions**, because there is more than one resource definition for the same terminal (the definitions are not necessarily exact duplicates of each other).
- Having one dual-purpose definition in one, shared CSD, available to all the systems involved, both terminal-owning and application-owning. This method is known as **sharing terminal definitions**.
- Having, in the CSD available to the terminal-owning system, one **local** terminal definition that can be shipped to other systems when required. This method is known as **shipping terminal definitions**.

These three different methods of making a partial definition available, involve three different ways of using the TYPETERM and TERMINAL definitions. Local, remote, and dual-purpose terminal definitions are described more fully in:

- “Local definition” on page 189
- “Remote definition” on page 189
- “Dual-purpose definition” on page 190.

⁵ In this section, the word “system” means a CICS system communicating with other systems using MRO or ISC.

APPC devices for transaction routing

APPC transaction routing allows an APPC device belonging to one CICS system to invoke transactions in another CICS system. The three methods of defining a terminal for transaction routing, described in “Terminals for transaction routing” on page 188, are also applicable to APPC devices. These methods can involve the use of either TERMINAL-TYPETERM or CONNECTION-SESSION definitions. For APPC single session terminals, the TERMINAL-TYPETERM definition method is recommended (see “APPC (LUTYPE6.2) single session terminal” on page 187.)

Local definition

This is a full definition of the terminal, installed in the terminal-owning system. It is used in the **duplicating** and in the **shipping** methods.

- No REMOTESYSTEM, REMOTESYSNET, or REMOTENAME is needed on the TERMINAL or CONNECTION definition.

If the REMOTESYSTEM named on the TERMINAL definition is actually the name of the local system (as specified by the SYSIDNT system initialization operand), the definition is treated as a local terminal when it is installed, rather than a remote one.

- SHIPPABLE on the TYPETERM is NO if duplicating definitions.
- SHIPPABLE on the TYPETERM is YES if shipping definitions.
- SHIPPABLE on the TYPETERM is obligatory for APPC devices.
- All other necessary attributes on the TERMINAL and TYPETERM definitions must be specified.

Remote definition

This is a partial definition of the terminal, installed in the application-owning region and intermediate regions. It contains the minimum information necessary for the terminal to access a transaction in that system. You create remote definitions only if you are using the **duplicating** method.

- The REMOTESYSTEM name must be the name of the CONNECTION definition for the next region in the transaction routing path to the terminal-owning region.
- REMOTESYSNET must be the netname (generic applid) of the terminal-owning region. If REMOTESYSTEM names a direct CONNECTION to the terminal-owning region, REMOTESYSNET is not required, *unless the terminal-owning region is a member of a VTAM generic resource group, and the direct connection is an APPC link.*
- The REMOTENAME is the name that the terminal or APPC device is known by in the terminal-owning region. It may be the same as or different from the TERMINAL or CONNECTION name, which must be the name the terminal is known by in the application-owning system. REMOTENAME defaults to the TERMINAL name if not specified.
- SHIPPABLE on the TYPETERM is NO for non-APPC devices.
- SHIPPABLE on the TYPETERM is obligatory for APPC devices.
- Some of the attributes on the TERMINAL and TYPETERM definitions may be omitted. For further guidance about these definitions, see the *CICS/ESA Intercommunication Guide*.

The following terminals and logical units cannot use transaction routing and therefore cannot be defined as remote:

- Pooled 3600 or 3650 pipeline logical units
- IBM 2260 terminals
- The MVS console
- Pooled TCAM terminals

Dual-purpose definition

This is a full definition of the terminal, installed in the terminal-owning system and in the application-owning and intermediate regions. It is used in the **sharing** method only.

- The REMOTESYSTEM name must be the SYSIDNT of the terminal-owning region. This must also be the name of the CONNECTION that you define from the application-owning region to the intermediate region (if any), and from the intermediate region to the terminal-owning region. All the CONNECTION definitions on the path from the application-owning region to the terminal-owning region must be given this same name.
- REMOTESYSNET must be the netname (generic applid) of the terminal-owning region.
- The REMOTENAME is the same as the TERMINAL or CONNECTION name.
- SHIPPABLE on the TYPETERM is NO for non-APPC devices.
- SHIPPABLE on the TYPETERM is obligatory for APPC devices.
- All other necessary attributes on the TERMINAL and TYPETERM definitions must be specified.

Duplicating your definitions: maintaining local and remote definitions separately

1. You create a local definition for the terminal, in the CSD of the terminal-owning system, or on a shared CSD.
2. You create a remote definition for the terminal, in the CSD of the application-owning system, or in a shared CSD. If you have more than one application-owning system, you may need more than one remote definition, but if the systems are sharing a CSD, you may be able to use the same remote definition for them all.
3. If your systems share a CSD, you make sure the definitions are in different groups, because:
 - You want to install them in different systems
 - The TERMINAL names are probably the same, so the definitions cannot be in the same group.
4. You install the local definition in the terminal-owning system. This definition can be autoinstalled.
5. You install the remote definition in the application-owning system(s).

Advantages

- You can use this method whether or not your systems share a CSD, so you can use it for transaction routing between different MVS images.
- This is the **only** method you can use, if your terminals are known by different names in different systems.
- You can use automatic transaction initiation (ATI) in the application-owning system for a remote terminal without having to set up XALTENF and XICTENF exits. (See the *CICS/ESA Customization Guide* for programming information on these exits.) This is especially useful for printers, because they must be acquired **before** any ATI can be successful.

Disadvantages

- The CSD is using at least twice the necessary amount of disk storage for your definitions.
- The TCT is using more than the necessary amount of virtual storage for your definitions.
- There will be more effort than necessary involved in maintaining the definitions.
- You can autoinstall the terminal in the terminal-owning system, but you cannot autoinstall it in the application-owning systems.

Duplicating definitions

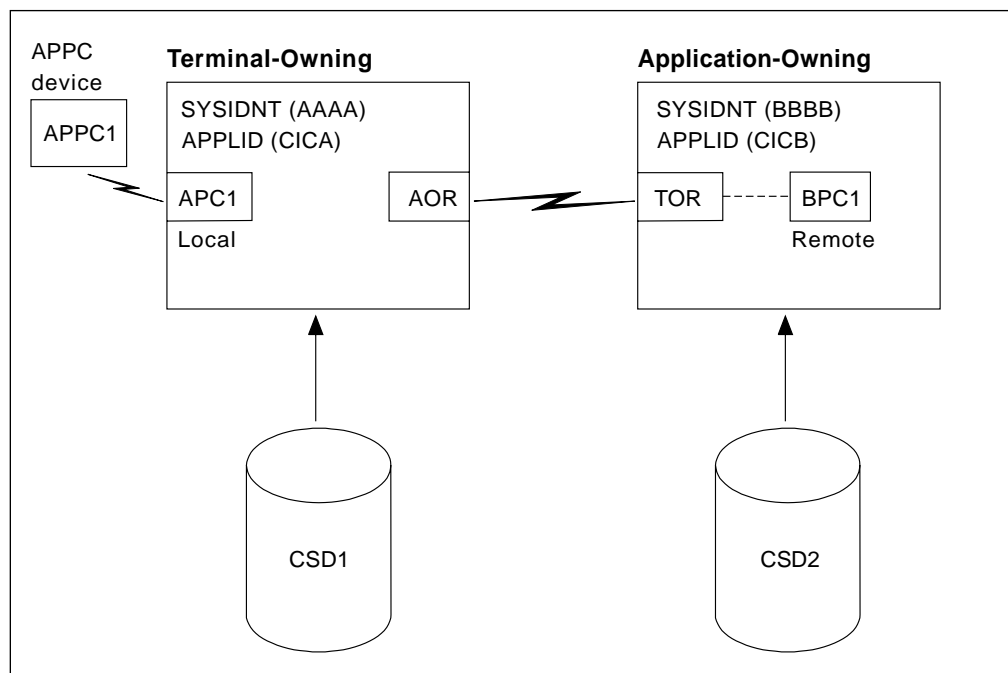


Figure 35. Maintaining local and remote definitions separately. The definitions may be on the same CSD.

Connections	Definitions in the AOR	Definitions in the TOR
Connection between the systems	<code>CONNECTION(TOR) NETNAME(CICA)</code>	<code>CONNECTION(AOR) NETNAME(CICB)</code>
Connection to APPC device, using TERMINAL and TYPETERM	<code>TERMINAL(BPC1)</code> <code>TYPETERM(APPC0001)</code> <code>NETNAME(APPC1)</code> <code>REMOTESYSTEM(TOR)</code> <code>REMOTENAME(APC1)</code> and <code>TYPETERM(APPC0001) DEVICE(APPC)</code>	<code>TERMINAL(APC1)</code> <code>TYPETERM(APPC0001)</code> <code>NETNAME(APPC1)</code> and <code>TYPETERM(APPC0001) DEVICE(APPC)</code>
Connection to APPC device, using CONNECTION and SESSIONS	<code>CONNECTION(BPC1) NETNAME(APPC1)</code> <code>REMOTESYSTEM(TOR)</code> <code>REMOTENAME(APC1)</code>	<code>CONNECTION(APC1) NETNAME(APPC1)</code> and <code>SESSIONS(APPC0001)</code> <code>CONNECTION(APC1)</code>

Note: The `REMOTESYSNET` option is not required because there is a direct connection between the AOR and the TOR.

Sharing dual-purpose definitions

1. You create a dual-purpose definition for the terminal, in the shared CSD.
2. You install the definition in all the systems: it becomes a remote definition in a system whose `SYSDNT` is different from the `REMOTESYSTEM` name; it becomes a local definition in a system whose `SYSDNT` is the same as the `REMOTESYSTEM` name.

Advantages

- You can use automatic transaction initiation (ATI) in the application-owning system for a remote terminal without having to set up XALTENF and XICTENF exits. (See the *CICS/ESA Customization Guide* for programming information on these exits.) This is especially useful for printers, because they must be acquired **before** any ATI will be successful.
- Disk storage use is reduced because you need only one CSD record.
- Maintenance is reduced because you need only one CSD record.

Disadvantages

- The TCT is using more than the necessary amount of virtual storage for your definitions.
- You cannot use this method if your systems do not share a CSD, so you can use it only for transaction routing within the same MVS image.
- You cannot use this method if your terminals are known by different names in different systems.

Sharing definitions

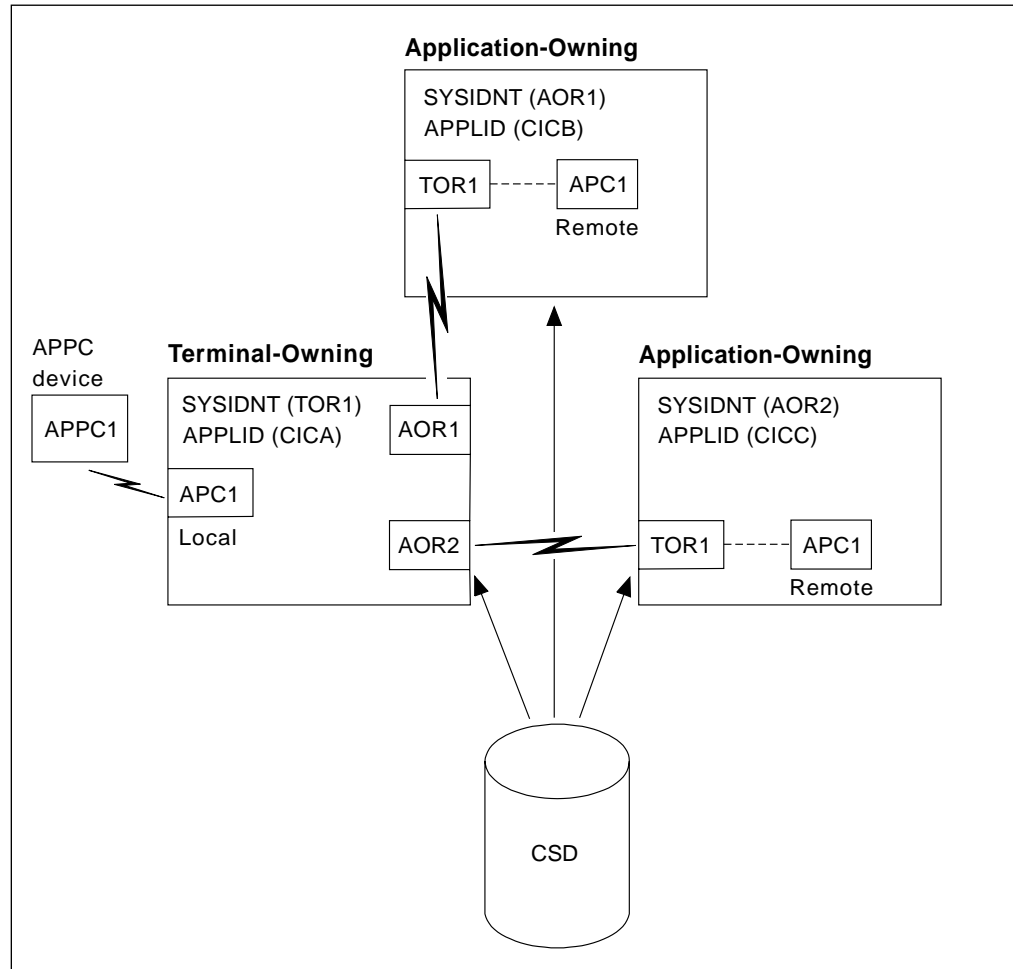


Figure 36. Sharing dual-purpose definitions

Connections	Definitions in the AORs	Definitions in the TOR
Connection between the systems	CONNECTION(TOR1) NETNAME(CICA)	CONNECTION(AOR1) NETNAME(CICB)
Connection to APPC device, using TERMINAL and TYPETERM	-	TERMINAL(APC1) TYPETERM(APPC0001) NETNAME(APPC1) REMOTESYSTEM(TOR1) REMOTENAME(APC1) and TYPETERM(APPC0001) DEVICE(APPC)
Connection to APPC device, using CONNECTION and SESSIONS	-	CONNECTION(APC1) NETNAME(APPC1) REMOTESYSTEM(TOR1) REMOTENAME(APC1) and SESSIONS(APPC0000) CONNECTION(APC1)
Note: The REMOTESYSNET option is not required because there are direct connections between the AOR and the TOR.		

Shipping terminal definitions to application-owning systems

1. You create a local definition for the terminal, in the CSD of the terminal-owning system. (This can be a shared CSD.)
2. You install the local definition in the terminal-owning system. (This definition can be installed at system initialization, or by using CEDA INSTALL, or by autoinstall.)
3. When the terminal invokes a transaction belonging to another system, the information necessary to create a remote definition is shipped to that system, and a temporary definition is installed there, automatically.

If the local definition was autoinstalled, the shipped definition lasts until the terminal is logged off. Otherwise, the shipped definition lasts until the local definition is installed again, or until the link between the systems is broken.

Advantages

- You can use this method whether or not your systems share a CSD, so you can use it for transaction routing between different MVS images.
- Disk storage use is reduced because you need only one CSD record.
- Maintenance is reduced because you need only one CSD record.
- Virtual storage use is reduced because you install the definition in only one system.
- You can autoinstall the terminal in the terminal-owning system, and **in effect** autoinstall it in the application-owning systems. (Shipping terminal definitions has the same effect as autoinstall, but does not itself involve the autoinstall process.)

Disadvantages

- You cannot use this method if your terminals are known by different names in different systems.
- You may need to use the global exits XALTENF and XICTENF if you use ATI in the transaction owning system. See the *CICS/ESA Customization Guide* for programming information on these exits.

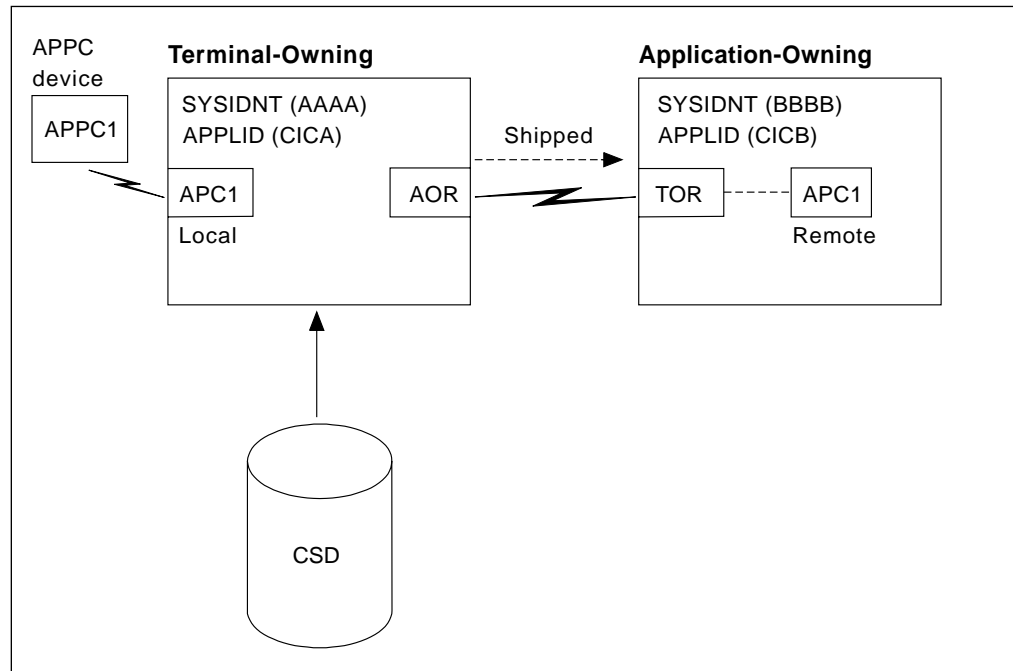


Figure 37. Shipping local definitions to an application-owning system

The definitions in the TOR to connect to the APPC device would be either a pair of TERMINAL and TYPETERM definitions or a CONNECTION and SESSIONS pair of definitions. Both are given here:

```

TERMINAL(APC1) TYPETERM(APC0001) NETNAME(APPC1)
and
TYPETERM(APPC0001) DEVICE(APPC) SHIPPABLE(YES)
      (SHIPPABLE is mandatory for device type APPC)

CONNECTION(APC1) NETNAME(APPC1)
and
SESSIONS(APPC0000) CONNECTION(APC1)
    
```

Transaction routing — summary

You should consider carefully these methods of defining terminals for transaction routing, and decide which is most suitable for your network before you start to use RDO.

We recommend the **shipping method**, unless you use terminals that are known by different names in different systems. Note that for ATI to work with the shipping method in a transaction owning system, you may need to use the XALTENF and XICTENF global exits. (See the *CICS/ESA Customization Guide* for programming information on these exits). If you use the same names in different systems and you don't want to use global exits to ensure that ATI works, we recommend the **sharing method** for systems with a shared CSD.

You should not need to use the **duplicating method** unless you use terminals that are known by different names in different systems, or if you use ATI to acquire terminals but do not have a shared CSD, and you don't want to use the XALTENF and XICTENF global user exits.

You could use a mixture of methods: perhaps shipping for display terminals, and duplicating for printers that need ATI to acquire them but without the use of the XALTENF and XICTENF global user exits.

More information about defining intercommunication resources

Before you start creating definitions for these resources, see the *CICS/ESA Intercommunication Guide* for further guidance. There you will find useful examples of the attributes you must specify for different types of links and sessions.

Extended recovery (XRF) for terminals

There are three TYPETERM attributes that relate specifically to the use of XRF for terminals.

A note on terminology: XRF-capable, or class 1, terminals are SNA terminals connected to both active and alternate CICS systems through a boundary network node communications controller, using NCP and VTAM with XRF capability.

RECOVNOTIFY

This applies only to class 1 terminals. You use it to specify how the terminal user is notified when an XRF takeover has occurred. You can specify either a simple message to be displayed, or a transaction that can do more than that. In either case, all the terminals for which you specify MESSAGE get the same message, and all the terminals for which you specify TRANSACTION get the same transaction.

There are two messages, DFHXRC1 and DFHXRC2, that you may edit in mapset DFHXMSG. Alternatively you can change the name of the map passed to the node error program. You specify the transaction using the RMTRAN system initialization operand, which defaults to the 'good morning' transaction specified in GMTRAN system initialization operand.

MESSAGE is more efficient than TRANSACTION, and minimizes the takeover time.

RECOVOPTION

You use this to influence how CICS recovers the terminal session and returns the terminal to service, after an XRF takeover. The **NONE** option leaves the terminal with no XRF support: the logon status of the terminal is not tracked by the alternate system and the terminal session is not automatically recovered after a takeover. The system default allows CICS to optimize the recovery of the terminal. However, there are circumstances in which you must specify one of the other options.

For XRF-capable, or class 1, terminals there are five options: NONE and four others. If, at takeover, a terminal is executing a transaction in the active CICS, the transaction is backed out, but the session remains busy. **SYSDEFAULT** allows CICS to clean up this busy session in the most efficient way. CICS uses one of three methods, in this order of preference:

1. Send an end-bracket (EB) indicator to close the in-bracket state.
2. Send a CLEAR request to reset all session states.
3. Send an UNBIND request to release the session.

If the terminal cannot handle an unexpected EB, specify **CLEARCONV**: CICS sends a CLEAR or an UNBIND request. This might apply if the terminal is programmable (intelligent).

If the terminal cannot handle an unexpected EB indicator or a CLEAR request, specify **RELEASESESS**: CICS sends an UNBIND request to release the session.

If the terminal must know at all times to which system it is connected, specify **UNCONDREL**: CICS sends an UNBIND request to release the session, **whether it is busy or not**. This might apply if the terminal is programmable (intelligent).

If, for any reason, an UNBIND request is sent, a new session is initiated by CICS, if the terminal is capable of performing a SIMLOGON.

TERMINAL

For other terminals (class 2 and class 3), there is effectively a choice between NONE (no XRF support) and any one of the other options. Any option other than NONE causes the alternate system to track the logon status of the terminal and, if the terminal was logged on, to initiate a logon, and to re-establish a session, if possible.

XRFSIGNOFF

You use this option to set the signon characteristics of a group of terminals. You can either allow the group to be signed-on after takeover, or you can force the group to be signed off. You might choose to force signoff to prevent unauthorized use of the terminal if it became active again after a takeover while the authorized user was absent.

There is a hierarchy of options for signon/signoff.

- DFHSIT XRFSOFF=FORCE|NOFORCE
- TYPETERM XRSIGNOFF(NOFORCE|FORCE)
- CICS RACF segment

For further guidance on extended recovery for terminals, see the *CICS/ESA 3.3 XRF Guide*.

Defining a TERMINAL

```

TErmina1 ==>
Group ==>
Description ==>
AUTINSTModel ==> No No | Yes | Only
AUTINSTName ==>

TERMINAL IDENTIFIERS
TYpeterm ==>
NEtname ==>
CONSOLE ==> No No | 0-99
CONSName ==>
REMOTESYSem ==>
REMOTEName ==>
REMOTESYSNet ==>
Modename ==>

ASSOCIATED PRINTERS
PRINTER ==>
PRINTERCopy ==> No No | Yes
ALTPRINTER ==>
ALTPRINTCopy ==> No No | Yes

PIPELINE PROPERTIES
P0ol ==>
TAsklimit ==> No No | 1-32767

OPERATOR DEFAULTS
OPERId :
OPERPriority : 000 0-255
OPERRs1 : 0 0-24,...
OPERSecurity : 1 1-64,...

PRESET SECURITY
# USERid ==>
NAtlang ==>

TERMINAL USAGES
TTransaction ==>
TEmpriority ==> 000 0-255
Inservice ==> Yes Yes | No

SESSION SECURITY
Securityname ==>
ATTachsec ==> Local Local | Identify | Verify | Persistent
| Mixidpe
BINDPassword ==> PASSWORD NOT SPECIFIED
# BINDSecurity ==> No No | Yes
USEDfltuser ==> No No | Yes
    
```

Figure 38. The DEFINE panel for TERMINAL

ALTPRINTCOPY({NO|YES})

YES means that the hardware COPY feature is to be used to satisfy a PRINT request on the printer named in ALTPRINTER. For further details, see the PRINTERCOPY attribute.

ALTPRINTER(name)

The name of a 3270 printer to be used as an alternative to the PRINTER named in this TERMINAL definition, if

the PRINTER is unavailable. The name may be up to four characters in length. For further details, see the PRINTER attribute. You should not specify an ALTPRINTER without specifying a PRINTER. If you do, ALTPRINTER will be ignored.

The printer you name must be owned by the same CICS system that owns this TERMINAL definition.

TERMINAL

If you want to specify the hardware COPY feature for the alternative printer, specify YES for ALTPRINTCOPY on this TERMINAL definition.

ATTACHSEC({LOCAL|IDENTIFY|VERIFY|PERSISTENT|MIXIDPE}) (APPC only)

Indicates the level of attach time user security required for the connection.

LOCAL

The authority of the user is taken to be that of the link itself, and you will be relying on link security alone to protect your resource.

IDENTIFY

Incoming attach requests must specify a user identifier. You should specify IDENTIFY when the connecting terminal has a security manager.

VERIFY

Incoming attach requests must specify a user identifier and a user password. You should specify VERIFY when the connecting terminal has no security manager and hence cannot be trusted.

PERSISTENT

This involves a user signon to a remote system that persists over multiple conversations until the user signs off from the remote system. In this way, the user's ID and password are passed only on the first (signon) attach. Subsequent attach requests require only the user's ID.

MIXIDPE

This represents a connection able to support attaches using either or both IDENTIFY and PERSISTENT security types. The security type used depends on the incoming attach. As in previous releases, IDENTIFY implies that there is a degree of trust between the two systems that allows this system to accept the signon logic of the other system. In effect this is a distributed security manager, with one system doing the signon and the other doing the security check. These two options, PERSISTENT and MIXIDPE, are valid only with VTAM as the access method, and when APPC is being used.

AUTINSTMODEL({NO|YES|ONLY})

Indicates whether this TERMINAL definition can be used as a model terminal definition for autoinstall. For more information on autoinstall and model TERMINAL definitions, see Chapter 9, "Autoinstall for VTAM terminals" on page 105.

NO

This definition will not to be used as a model for autoinstall. It will be used only as a definition for a specific device that will not be autoinstalled.

YES

This definition will be used as a definition for a specific device that will not be autoinstalled. The

definition will also be used as a model for automatic installation.

ONLY

This definition will be used only as a model for autoinstall. It will not be used as a definition for a specific device.

AUTINSTNAME(name)

The name that this model definition will be known by in the autoinstall control program. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

You need only specify this if AUTINSTMODEL is YES or ONLY. You can let it default to the TERMINAL name followed by four blanks, as long as this is acceptable to the autoinstall control program. For more information about autoinstall models, autoinstall names, and the autoinstall control program, see Part 3, "Autoinstall" on page 103.

BINDPASSWORD(password) (APPC only)

A password of up to 16 hexadecimal characters (0-9, A-F). A password of less than 16 characters will be padded on the right with hexadecimal zeros. A password of more than 16 characters will be ignored.

CICS will mask the password you supply to avoid unauthorized access. You should therefore find a safe way of recording the password.

If you supply a password, an identical password must be supplied in the CONNECTION definition for the remote system to ensure bind time security.

BINDSECURITY({NO|YES}) (APPC only)

This indicates whether an ESM is being used for bind-time security.

No No external bind-time security is required. However, if you define a password on the BINDPASSWORD parameter, CICS defaults to using its own internal security mechanism for bind security checking.

Yes

If security is active and the XAPPC system initialization parameter is set to YES, then CICS will attempt to extract the session key from RACF in order to carry out bind-time security. If no RACF profile is available, the bind fails.

CONSOLE({NO|number})

- + If CICS is running under a release of MVS earlier than MVS/ESA SP 4.1, use this attribute to define a system console to CICS. The console being defined should either be an existing console with the specified number, or be predefined in the CONSOLnn member of SYS1.PARMLIB.
- + |
- + |
- + |
- + |
- + If CICS is running under MVS/ESA SP 4.1 or later, you can install a CICS console definition without having an existing console, and without the console being

+ previously defined in the CONSOLnn member of
 + SYS1.PARMLIB.

+ Specify a number in the range 01 through 250, but not
 + 128. However, before you can use the console, it must
 + be either defined to MVS in the CONSOLnn member of
 + SYS1.PARMLIB, or dynamically allocated by a product
 + such as NETVIEW. The primary MVS console should
 + therefore be specified with a CONSOLE number of 1, if
 it is intended to use that console as a CICS terminal.
 The first MVS secondary console should be specified
 with a number of 2, the second MVS secondary console
 with a number of 3, and so on.

If you want to initiate a CICS transaction and issue a
 command to it in a JCL statement, you need to create a
 TERMINAL definition with a CONSOLE number of 00.
 For guidance about using JCL to issue CICS commands,
 see the *CICS/ESA Operations and Utilities Guide*.

CONSNAME(name)

If CICS is running under MVS/ESA SP 4.1 or later, you
 should use the CONSNAME parameter rather than the
 CONSOLE parameter to define a console device to
 CICS. For migration purposes, however, it is still
 possible to code CONSOLE(number).

| Using CONSNAME, you can install a CICS console
 | definition without having an existing console, and without
 | the console being previously defined in the CONSOLnn
 | member of the MVS SYS1.PARMLIB. However, before
 | you can use the console, you must define the name to
 | MVS, either in the CONSOLnn member of
 | SYS1.PARMLIB or by dynamic allocation.

The length of CONSNAME must be from two to eight
 characters and must begin with an alphabetic character
 or one of #, @, or \$. It uniquely identifies the console
 device within a sysplex, regardless of which MVS image
 it is connected to; it corresponds to the name defined for
 the console in the MVS SYS1.PARMLIB member,
 CONSOLnn.

To define a TSO user as a console device, code
 CONSNAME(name), where name is the TSO userid.
 This enables a TSO user authorized to use the TSO
 CONSOLE command to initiate CICS transactions. The
 TSO userid does not have to be defined in the
 CONSOLnn member of SYS1.PARMLIB member.

The equivalent of CONSOLE(00) is
 CONSNAME(INTERNAL); specify this if you want to
 initiate a CICS transaction and issue a command to it in
 a JCL statement. For further guidance about using JCL
 to issue CICS commands, see the *CICS/ESA Operations
 and Utilities Guide*.

- # **Notes:**
- # 1. Define either CONSOLE or CONSNAME; you
 # cannot define both.
 - # 2. If the name DFHCONxx (where xx can be any valid
 # characters) is specified, the terminal will be used by
 # CICS as a 'pool' console for use when processing a
 # modify command from a console with a name which
 # is not defined to CICS.

DESCRIPTION(text)

You can provide a description of the resource you are
 defining in this field.

The DESCRIPTION text can be up to 58 characters in
 length. There are no restrictions on the characters that
 you may use. However, if you use parentheses, you
 should ensure that for each left parenthesis there is a
 matching right one.

GROUP(groupname)

Every resource definition must have a GROUP name.
 The resource definition becomes a member of the group
 and is installed in the CICS system when the group is
 installed. For more information about groups, see "How
 the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in
 length. The characters allowed are A-Z 0-9 @ # and
 \$. Lowercase characters are treated as uppercase
 characters. Do not use group names beginning with
 DFH, because these characters are reserved for use by
 CICS.

INSERVICE({YES|NO})

The status of the terminal that is being defined.

- YES** Transactions may be initiated and messages may
 automatically be sent to the terminal.
- NO** Indicates that the terminal can neither receive
 messages nor transmit input.

**MODENAME(name) (APPC single session terminals
 only)**

The name will be passed to VTAM as the LOGMODE
 name. The name may be up to eight characters in
 length, but may not have the reserved name
 SNASVCMG. The name follows assembler language
 rules. It must start with an alphabetic character. The
 acceptable characters are: A-Z 0-9 \$ @ and #.
 Lowercase characters are converted to uppercase.

For further guidance on the LOGMODE name, see the
CICS/ESA Installation Guide.

NATLANG({blank|E|K})

The language in which all NLS-enabled messages will
 be displayed for this terminal.

Use only one character, which can be A-Z 0-9 @ #
 and \$.

TERMINAL

blank If you leave this blank and do not supply a value, CICS will use the system default as specified in the system initialization table (SIT).

E English
K Kanji.

NETNAME(name)

The network name that identifies the terminal to ACF/VTAM. The name may be up to eight characters in length. The name follows assembler language rules. It must start with an alphabetic character. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

If you do not specify a name, the NETNAME defaults to the TERMINAL name.

The NETNAME must be unique; that is, you cannot install two terminals or a terminal and a connection with the same NETNAME.

OPERID(code)

The OPERID keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

OPERPRIORITY({0|number})

The OPERID keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

OPERRSL({0|number[,...]}))

The OPERRSL keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

OPERSECURITY({1|number[,...]}))

The OPERRSL keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for information.

POOL(name)

The pool name for a 3600 or 3650 pipeline terminal pooled with other pipeline terminals.

When you define a 3600 pipeline logical unit, you generate a TCTTE which is associated with a pool of TCTTEs. A pool of pipeline TCTTEs can be used by one pipeline logical unit, or it can be shared by a number of pipeline logical units.

The pool name is used only as a method of identifying the related TERMINAL definitions on the CSD. It is not used within the active CICS system. The name can be up to 8 characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

For a pipeline terminal, you must specify a TYPETERM with SESSIONTYPE(PIPELINE) specified. You must specify a TASKLIMIT on at least one of the pool of pipeline terminals. You must name the same group for each of the pipeline terminals in a pool.

An example of the definitions for a pool of pipeline terminals follows.

```
DEFINE TERMINAL(ttt1) GROUP(g) POOL(poolid)
      TYPETERM(xxxxxxxx) NETNAME(nnnnnn1)...
```

```
DEFINE TERMINAL(ttt2) GROUP(g) POOL(poolid)
      TYPETERM(xxxxxxxx) NETNAME(nnnnnn2)...
```

```
DEFINE TERMINAL(ttt3) GROUP(g) POOL(poolid)
      TYPETERM(xxxxxxxx) NETNAME(nnnnnn3)...
```

```
DEFINE TERMINAL(ttt4) GROUP(g) POOL(poolid)
      TASKLIMIT(nn) TYPETERM(xxxxxxxx)
      NETNAME(nnnnnn4)...
```

```
DEFINE TYPETERM(xxxxxxxx) GROUP(g)
      DEVICE(3600|3650) SESSIONTYPE(PIPELINE)
```

PRINTER(name)

The name of the primary 3270 printer to be used to respond to an ISSUE PRINT command, or a PRINT request from an operator pressing a program access (PA) key. The name may be up to four characters in length. The name is the TERMINAL name on the definition for the printer. If you name a PRINTER here, the TYPETERM referenced by this TERMINAL definition must have PRINTADAPTER(NO).

The printer you name must be owned by the same CICS system that owns this TERMINAL definition.

You may also name an ALTPRINTER which will be used as an alternative, if this PRINTER is unavailable.

You can name a PRINTER if this TERMINAL definition is for one of the following:

- A 3270 display without the printer-adapter feature
- A 3270 display attached to a 3274 control unit
- A 3276 control unit display station
- A 3790 in 3270 compatibility mode.

If you want to specify the hardware COPY feature, specify PRINTERCOPY(YES) on this TERMINAL definition.

Note that SNA character string (SCS) printers accept only 3790 data streams; they do not accept 3270 data streams. They therefore cannot be used to print the contents of a display unit's buffer.

If you use a program attention key (for example, PA1) to print the contents of the screen on an associated VTAM printer, the screen size of the printer will be chosen according to the the SCRNSIZE operand as defined in the CICS-supplied default profile DFHCICST. This profile is defined with SCRNSIZE(DEFAULT), and if you want to use the alternate screen size of the printer, you will have to change the profile entry for transaction CSPP.

PRINTERCOPY({NO|YES})

YES specifies that the hardware COPY feature is to be used to satisfy a print request on the PRINTER named in this TERMINAL definition.

CICS will use the hardware COPY feature of the 3270 to perform the print, unless a task is currently attached to the display.

You need not specify COPY(YES) on the TYPETERM definition, because this is implied by PRINTERCOPY(YES) on the TERMINAL definition.

If you have named an ALTPRINTER as well as a PRINTER, you may specify ALTPRINTCOPY(YES).

To use the COPY feature, both the printer and the display terminal must be on the same 3270 control unit. Otherwise, either the COPY may fail, raising an error condition, or, if the display device address is valid for the printer's control unit, copying might be performed from a different display.

Do not specify PRINTERCOPY(YES) if, in a networking environment, the 3270 control unit is connected to a TCAM system in one domain, and a CICS system in another domain has access to the control unit via VTAM. This is because the hardware COPY address is not available to CICS and cannot therefore be used by terminals attached to such a control unit.

The COPY command is invalid for a 3270 compatibility mode display.

REMOTENAME(name)

The name by which the terminal is known in the system or region that owns the terminal. The name can be up to four characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >.

REMOTESYSNET(name)

The network name (APPLID) of the region that owns the terminal. The name can be up to eight characters in length. It follows assembler language rules, and must start with an alphabetic character. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

REMOTESYSNET is used where there is no direct link between the region in which this definition is installed and the terminal-owning region. You do not need to specify REMOTESYSNET if either of the following is true:

- You are defining a local terminal (that is, REMOTESYSTEM is not specified, or specifies the name of the local system).
- REMOTESYSTEM names a direct link to the terminal-owning region. However, there is one special case: if the terminal-owning region is a member of a VTAM generic resources group and the direct link is an APPC connection, you **do** need to specify REMOTESYSNET.

REMOTESYSTEM(name)

The name that identifies the intercommunication link to the system that owns the terminal. The name can be up to four characters in length. The acceptable characters

are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

This is the CONNECTION name on the CONNECTION definition for the intercommunication link. If it is not specified, or if it is specified as the name of the local system, then this terminal will be local to this system. If the name is that of another system, the terminal will be remote. You can therefore use the same definition for the terminal in both the local system and a remote system.

If there are intermediate systems between this CICS and the terminal-owning region, REMOTESYSTEM should specify the first link in the path to the TOR. If there is more than one possible path, it should specify the first link in the preferred path.

REMOTESYSTEM is ignored if you specify AUTINSTMODEL(YES) or (ONLY).

SECURITYNAME(name)

The security name of the remote system.

In a CICS system with security initialized (SEC=YES or MIGRATE), the security name is used to establish the authority of the remote system.

Note: If USERID is specified in the session definition (DEFINE SESSIONS command) associated with the connection definition, it will override the userid specified in the SECURITYNAME parameter, and will be used for link security.

The security name (or USERID on the sessions definition) must be a valid RACF userid on your system. Access to protected resources on your system is based on the RACF user profile and its group membership.

For further information on defining connections for MRO, LUTYPE6.1, and APPC, see the *CICS/ESA CICS-RACF Security Guide*.

TASKLIMIT(NO|number)

The number of concurrent tasks allowed to run in a pipeline session or in a pool of pipeline sessions. The number can be in the range 1 through 32767.

When you define a 3600 pipeline logical unit, you generate a TCTTE which is associated with a pool of TCTTEs. As messages enter CICS from the 3600 pipeline logical unit, a task is attached to process this message, using as anchor block one of the TCTTEs from the pool. In this way, consecutive messages sent via the pipeline logical unit can be processed concurrently, the number of concurrent transactions being limited by the number of TCTTEs in the pool. The number of TCTTEs in the pool should represent the high water mark of inquiry activity. In this way, the pipeline facility allows less TCTTEs to be defined to CICS than the total number of pipeline inquiry terminals.

TERMINAL(name)

This is the terminal identifier. The name can be up to four characters in length. For a terminal, the acceptable characters are: A-Z a-z 0-9 \$ @ # . / _ % & ¢ ?

TERMINAL

? ! : | " = ~ , ; < and >. For an APPC LU6.2 single session terminal, the acceptable characters are A-Z 0-9 \$ @ #. You should not have a TERMINAL definition and a CONNECTION definition with the same name. The value CERR is reserved for the identification generated for the error console.

If the name supplied is less than four characters, it will be left-justified and padded with blanks up to four characters.

Note: If you use a comma (,) in a name, you will be unable to use those commands such as
CEMT INQUIRE TERMINAL(*value1,value2*)
CEMT SET TERMINAL(*value1,value2*)
where the comma serves as a list delimiter. See the *CICS/ESA CICS-Supplied Transactions* manual for information about using lists of resource identifiers.

The TERMINAL name must be unique, and not match any other TERMINAL or CONNECTION name. It becomes part of the name of the TCT entry (the TCTTE), when this TERMINAL definition is installed.

Both a CONNECTION and a TERMINAL have a TCTTE. The CONNECTION's TCTTE is called a TCSE, but it is treated in exactly the same way as a TCTTE for a TERMINAL, that is, is it looked after by the Table Manager. The key used by the Table Manager for TCTTEs is the termid/sysid. It is not possible to have a CONNECTION and a TERMINAL with the same name.

If you specify AUTINSTMODEL(ONLY), you need not worry about making the TERMINAL name unique, as it will **not** be used as the name of a TCT entry. If you specify AUTINSTMODEL(YES), the TERMINAL name will be used as the name of the TCT entry that is installed in the TCT when the TERMINAL definition is installed; the names of the TCT entries for the autoinstalled terminals will be determined by the autoinstall user program.

If you are coding this operand for a 3270 display, the only CICS functions the operator is able to invoke, other than this transaction, are paging commands and print requests.

If the terminal is to be associated with a transient data destination, then the terminal name and the destination identification in the DCT must be the same.

TERMPRIORITY({0|number})

Establishes the terminal priority. This decimal value (0 through 255) is used in establishing the overall transaction processing priority. (Transaction processing priority is equal to the sum of the terminal priority, transaction priority, and operator priority, not to exceed 255.)

TRANSACTION(name)

Code this with a 1- to 4-character transaction name. The acceptable characters are: A-Z 0-9 \$ @ and #.

For VTAM non-3270 devices, a TRANSACTION name of less than four characters requires a delimiter.

It specifies the name of the transaction that is to be initiated each time input is received from the terminal when there is no active task. For guidance on what happens when a transaction is initiated, see the *CICS/ESA Application Programming Guide*.

If you code this operand for a 3270 display, the only CICS functions the operator will be able to invoke, other than this transaction, are paging commands and print requests.

You are unlikely to code the TRANSACTION parameter for a 3270 display or SCS printer. It is optional for 3601 logical units, but is mandatory for 3614 logical units.

If this operand is coded for a 3790 Communication System, and multiple sessions are used to connect the same 3791, the same TRANSACTION name should be specified for all sessions.

TYPETERM(name)

The name of the TYPETERM definition to be associated with this TERMINAL definition. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

The TYPETERM definition specifies many attributes for a number of similar terminals. For more information, see Chapter 24, "TYPETERM" on page 217 and "Installing resource definitions" on page 19. This attribute is mandatory for all TERMINAL definitions.

The TYPETERM definition should already be installed when you install this TERMINAL definition.

All TERMINAL definitions used to define pooled consoles (those with a Consname of DFHCONxx) should specify the same TYPETERM.

USEDFTUSER({NO|YES}) (APPC only)

specifies the kind of security checking that will take place for each inbound attach FMH.

No Indicates that each inbound attach FMH will be checked for the presence of those fields required by the ATTACHSEC option and if the required fields are not present a protocol violation message will be issued and the attach will fail. NO is the default.

Yes Indicates that some checks on the validity of the attach FMH are bypassed. This provides the same level of security as in previous releases of CICS. See 'Attach Time Security and the USEDFTUSER option' in the *CICS/ESA CICS-RACF Security Guide*.

USERID(name)

A user identifier used for signon and referred to in security error messages, security violation messages and the audit trail. It must be a valid userid defined to the security manager.

TERMINAL

Chapter 22. TRANCLASS

The TRANCLASS definition allows you to define a transaction class. Transactions that are defined as belonging to a transaction class are subject to scheduling constraints before they are allowed to execute. If transactions belonging to an active transaction class are already running, any new transactions are queued. Use the MAXACTIVE attribute to specify the maximum number of transactions that you want to run. To limit the size of the queue, you can use the PURGETHRESH attribute.

By putting your transactions into transaction classes you can control how CICS dispatches tasks. For example, you can separate transactions into those which are heavy resource users and those which are of lesser importance, such as the “Good morning” broadcast messages. You can then use the attributes on the TRANCLASS definition to control the number of active tasks allowed from each transaction class.

The DEFINE panel for TRANCLASS is shown in Figure 39.

Defining a TRANCLASS

```

CEDA DEFINE TRANClass( XXXX ) GROUP(MYGROUP)
TRANClass      : XXXX
Group          : MYGROUP
Description    ==>
CLASS LIMITS
Maxactive      ==>          0-999
Purgethresh    ==> No      No | 1-1000000

```

Figure 39. CEDA DEFINE TRANCLASS panel

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see “How the CSD is organized — groups and lists” on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

MAXACTIVE(value)

This is the maximum number of transactions in this TRANCLASS that are allowed to be active. You must specify a MAXACTIVE value when you define a TRANCLASS, in the range 0 through 999.

New transactions attached when the number of active transactions has reached the MAXACTIVE limit are considered for queueing subject to the PURGETHRESH limit.

Defining a TRANCLASS with a zero MAXACTIVE value signifies that **all** tasks are to be queued.

PURGETHRESH({NO|number})

This is an optional purge threshold for the transaction class; it defines a threshold number at which transactions queuing for membership of the transaction class are purged. Specify it if you want to limit the number of transactions queueing in this transaction class. It can have the following values:

NO

The size of the queue is unlimited (other than by the storage available to attach tasks).

number

The purge threshold number in the range 1—1 000 000.

If you specify this as 1, no transactions are allowed to queue. If you specify it as any other number (n), the size of the queue is restricted to $n-1$. All new transactions attached after the limit of $n-1$ is reached are purged.

TRANCLASS

Example of PURGETHRESH: In the case of a transaction class where the maximum number of active tasks (MAXACTIVE) is set to 50, and the purge threshold (PURGETHRESH) is set to 10 to limit queuing transactions, CICS begins to abend new transactions for the class when:

- The number of active transactions reaches 50, and
- The number of transactions queuing for membership of the transaction class has reached 9.

CICS accepts new transactions for this transaction class queue only when the number queued falls below the maximum size of the queue (9 in our example).

TRANCLASS(name)

is the name of the transaction class. Transactions belonging to a transaction class are subject to scheduling constraints before they are allowed to execute. The reserved TRANCLASS name DFHTCL00 is used to indicate that the transaction does not belong to any transaction class.

For compatibility with releases that support a TCLASS attribute, CICS provides the following TRANCLASS equivalents:

TCLASS	TRANCLASS
NO	DFHTCL00
1	DFHTCL01
2	DFHTCL02
3	DFHTCL03
4	DFHTCL04
5	DFHTCL05
6	DFHTCL06
7	DFHTCL07
8	DFHTCL08
9	DFHTCL09
10	DFHTCL10

Sample definitions for these transaction classes are in group DFHTCL, supplied as part of DFHLIST.

Note: If a transaction is run and its associated TRANCLASS definition is not installed, the transaction will run without any of the scheduling constraints specified in the TRANCLASS. Message DFHXM0212 is issued as a warning.

TRANCLASS can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters.

Chapter 23. TRANSACTION

A CICS application consists of one or more programs written to perform a specific function. A particular invocation of such an application is known as a **transaction**, and the CICS transaction manager identifies it by its transaction identifier (TRANSID). You tell CICS how you want your transaction to run, primarily in a TRANSACTION definition. The name of this definition, the TRANSACTION name, is the same as the TRANSID.

In the TRANSACTION definition you specify options related to functions provided by CICS itself. The transaction priority, security key, and the length of the transaction work area (TWA) are examples of these options.

You also link the transaction with other resources by coding the names of their definitions in the TRANSACTION definition. These other resources are PROGRAM, PROFILE, PARTITIONSET, REMOTESYSTEM, and TRANCLASS.

PROGRAM

You specify options related to the software implementation of your application in the PROGRAM definition. This defines the program to which control is to be given to process the transaction. The TRANSACTION definition references the PROGRAM definition.

PROFILE

You do **not** have to specify, for each transaction, the options that will control the interaction with a terminal or logical unit. Instead, the TRANSACTION definition references a PROFILE definition, which specifies them for a number of transactions.

REMOTESYSTEM

For transaction routing, instead of specifying a PROGRAM name in the TRANSACTION definition, you specify the name of a REMOTESYSTEM. This can be the name of another CICS system, which itself is defined to this CICS system in a CONNECTION definition of the same name.

If you name a REMOTESYSTEM, you may also supply a REMOTENAME, which is the name of the **transaction** to be run in the remote system. The remote system decides which program it will give control to.

If you specify a REMOTESYSTEM name that corresponds to the system in which the definition is installed, CICS installs a local transaction resource definition. Otherwise CICS installs a remote transaction resource definition.

TRANCLASS

This specifies the name of the transaction class to which the transaction belongs. Transactions belonging to a transaction class are subject to scheduling constraints before they are allowed to execute. The constraints are specified in the associated TRANCLASS definition. A full description of the TRANCLASS attribute can be found on page 215.

The TRANSACTION definition is thus the central resource definition for an application. The related resource definitions are discussed below. Figure 1 on page 9 shows how all the resource definitions relate to each other.

initialization parameter specifies CONFDATA=SHOW, CONFDATA on the transaction definition is ignored.

If the system initialization parameter specifies CONFDATA=HIDETC, the following options are effective:

NO

CICS does not suppress any user data. VTAM and MRO initial user data is traced in trace point AP FC92. FEPI user data is traced in the normal CICS FEPI trace points.

YES

CICS suppresses user data from the CICS trace points.

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

DTIMOUT({NO|value})

Specifies whether deadlock time-out will be applied to the task. If the execution of the task gets suspended (for example, through lack of storage), a purge of the task is initiated if the task stays suspended for the longer than the DTIMOUT value. If the purge leads to a transaction abend, the abend code used depends on which part of CICS suspended the task. When using CEDF, the user task should, if possible, specify DTIMOUT(NO), or a large value.

Note: When using CEDF, if any DTIMOUT value has been specified for the user task, the DTIMOUT value is ignored while the user task is suspended and a CEDF task is active. Therefore the suspended user task cannot terminate with a deadlock timeout (abend AKCS) while a CEDF task is waiting for a user response.

For DTIMOUT to be effective, SPURGE must be set to YES.

CICS inhibits deadlock time-out at certain points. DTIMOUT is not triggered for terminal I/O waits. Because the relay transaction does not access resources after obtaining a session, it has little need for DTIMOUT except to trap suspended allocate requests. However, for I/O waits on a session, the RTIMOUT option can be specified on PROFILE definitions for transaction routing on mapped APPC connections.

It is important that you define some transactions with a DTIMOUT value, because deadlock time-out is the mechanism that CICS uses to deal with SOS (short on storage) situations.

NO

The deadlock time-out feature is not required.

value

The length of time (MMSS for minutes and seconds) after which the deadlock time-out facility will terminate a suspended task. The maximum value that you can specify is 68 minutes; this is accurate to intervals of one second.

DUMP({YES|NO})

Indicates whether a call is made to the dump domain to produce a transaction dump if the transaction terminates abnormally.

YES

A call is made to the dump domain to produce a transaction dump. Note that the final production or suppression of the transaction dump is controlled by the transaction dump table. For more information about the dump table, see the *CICS/ESA Problem Determination Guide*.

If no transaction dump table entry exists for the given dump code when a transaction abends, a temporary entry is created for which the default is to produce a transaction dump.

You control dump table entries for transaction dumps using the CEMT transaction (for more information see the *CICS/ESA CICS-Supplied Transactions* manual) or the EXEC CICS SET TRANDUMPCODE command (for programming information, see the *CICS/ESA System Programming Reference* manual).

NO

No call is made to the dump domain, suppressing any potential transaction dump.

Note: This operand has no effect on the following:

- An EXEC CICS DUMP command, which always produces a dump.
- The system dumps for dump codes AP0001 and SR0001 that CICS produces in connection with ASRA, ARSB, or ASRD abends. If you specify NO on the transaction DUMP option, CICS suppresses the transaction dump, but not the system dump.

DYNAMIC({NO|YES})

Indicates whether or not the transaction can be dynamically routed to a remote region, using the CICS dynamic transaction routing facility.

NO

Creates a local or remote definition according to the REMOTESYSTEM attribute.

YES

Allows the dynamic transaction routing program to determine the local or remote status dynamically at invocation time. For programming information about

TRANSACTION

the dynamic transaction routing program, see the *CICS/ESA Customization Guide*.

EXTSEC({NO|YES})

The EXTSEC keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see "How the CSD is organized — groups and lists" on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

INDOUBT({BACKOUT|COMMIT|WAIT})

Indicates the action required if the transaction is using intercommunication, and abends at a critical time during syncpoint or abend processing. For guidance on using the INDOUBT option, see the *CICS/ESA Intercommunication Guide*.

BACKOUT

The effects of the transaction will be backed-out. This must be specified for recoverable files.

COMMIT

The effects of the transaction will be committed. You should use INDOUBT(COMMIT) if you do not want dynamic transaction backout.

WAIT

Changes to recoverable temporary storage are locked until the session is recovered. The resources are then committed or backed out in step with the remote system.

ISOLATE({YES|NO})

indicates whether CICS is to isolate the transaction's user-key task-lifetime storage to provide transaction-to-transaction protection. (See the TASKDATAKEY option for a description of user-key storage.) Isolate means that the user-key task-lifetime storage is protected from both reading and writing by the user-key programs of other transactions—that is, from programs defined with EXECKEY(USER).

Note: The ISOLATE option does not provide any protection against application programs that execute in CICS key—that is, from programs defined with EXECKEY(CICS).

YES

If you specify ISOLATE(YES), the transaction's user-key task-lifetime storage is isolated from the user-key programs of all other transactions—that is, from programs defined with EXECKEY(USER), but not from programs defined with EXECKEY(CICS).

Also, the user-key task-lifetime storage of **all** other transactions is protected **from** the user-key programs of transactions defined with ISOLATE(YES).

NO

If you specify ISOLATE(NO), the transaction's task-lifetime storage is isolated from the user-key programs of those transactions defined with ISOLATE(YES). The transaction's storage is not, however, isolated from user-key programs of other transactions that also specify ISOLATE(NO), because with this option, the transactions are all allocated to the common subspace.

Note also that the user-key task-lifetime storage of all transactions defined with ISOLATE(YES) is protected **from** the user-key programs of transactions defined with ISOLATE(NO).

You should specify ISOLATE(NO) for those transactions that require to share any part of their user-key task-lifetime storage.

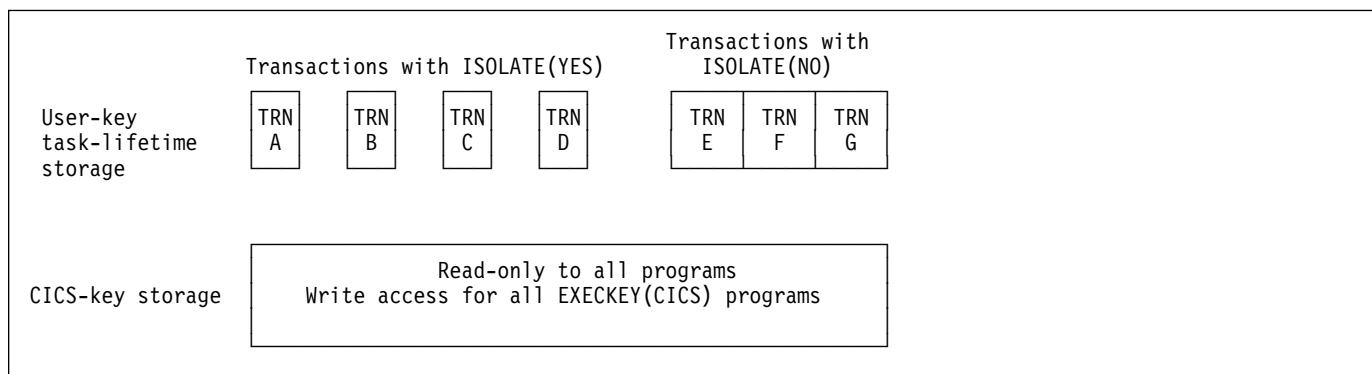


Figure 41. Conceptual view of transactions with and without transaction isolation. In this diagram, the elements of user-key task-lifetime storage of each of the transactions A, B, C, and D are isolated from each other, and from transactions E, F, and G. For the transactions A, B, C and D, therefore, their storage is addressable by their own user-key programs only.

The user-key task-lifetime storage of transactions E, F, and G, however, is accessible by all the user-key application programs of transactions E, F and G, but is isolated from the user-key programs of transactions A, B, C, and D. All of the illustrated transactions have read-only access to CICS-key storage.

LOCALQ({NO|YES})

Indicates whether queuing on the local system is to be performed.

NO

No local queuing is to be performed.

YES

Local queuing can be attempted for an EXEC START NOCHECK request when the system is not available and the system name is valid. A system is defined as not available when:

- The system is OUT OF SERVICE when the request is initiated.
- The attempt to initiate any session to the remote system fails and the corrective action taken by the abnormal condition program (DFHZNAC) or the node error program (DFHZNEP) is to place the system OUT OF SERVICE.
- No sessions to the remote system are immediately available, and your XISCONA global user exit program specifies that the request is not to be queued in the issuing region.

Local queuing should be used only for those EXEC START commands that represent time independent requests. The delay implied by local queuing will affect the time at which the request is actually started. It is your responsibility to ensure that this condition is met.

You can use the intersystem communication program global user exit XISLCLQ to override the setting of the LOCALQ attribute. For programming information on the user exits in the intersystem communication program, see the *CICS/ESA Customization Guide*.

PARTITIONSET(name|KEEP|OWN)

The name of the partition set that is to be the default application partition set. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

Note that you should delete a previously specified name by pressing the ERASE EOF key, if you are not specifying a new name.

If you do not specify a partition set name, or either of the reserved names, CICS will destroy existing partitions prior to the first BMS output to the terminal from the transaction.

name

If you specify a name, CICS will destroy existing partitions and will load the named partition set before the first BMS output to the terminal from the transaction. (Existing partitions are not destroyed if the terminal partition set matches the application partition set.)

This name must not be the same as that specified in PROGRAM(name).

KEEP

If you specify the reserved name KEEP, the transaction will use the application partition set for this terminal, whatever it may be. This option is normally used for successor transactions in a chain of pseudo-conversational transactions.

OWN

If you specify the reserved name OWN, the transaction will perform its own partition management.

PRIMEDSIZE({0|value})

The PRIMEDSIZE attribute is obsolete in CICS/ESA 4.1, but is supported to provide CSD compatibility for earlier releases of CICS where it is still valid. See Appendix A,

TRANSACTION

“Obsolete attributes retained for compatibility” on page 341 for more information.

PRIORITY({1|value})

The transaction priority. This one- to three-digit decimal value from 0 to 255 is used in establishing the overall transaction processing priority. (Transaction processing priority is equal to the sum of the terminal priority, transaction priority, and operator priority, not to exceed 255.)

PROFILE({DFHCICST|name})

The name of the PROFILE definition that specifies the processing options used in conjunction with the terminal that initiated the transaction. The name can be up to eight characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >.

The processing options provided by the default DFHCICST are shown in “PROFILE definitions in group DFHISC” on page 371. Note that DFHCICST is not suitable for use with a distributed program link. You should specify instead DFHCICSA, which has INBFMH=ALL.

PROGRAM(name)

The name of the program to which CICS gives control to process this transaction. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

You should ensure that this name is not the same as that specified in PARTITIONSET(name).

- + **Note:** If a name is specified for REMOTESYSTEM, and
- + it differs from that of the current system, no
- + name need be specified for PROGRAM. If, in
- + these circumstances, a name is specified for
- + PROGRAM, it may be ignored.

If this transaction definition is for use on a remote program link request, the program name you specify on this parameter must be the name of the CICS mirror program, DFHMIRS. See the TRANSID option on the PROGRAM resource definition on page 173.

REMOTENAME({local-name|remote-name})

The name of this transaction as it is known in a remote system, if it is to be executed in a remote system or region using intersystem communication. The remote system can be another CICS region or an IMS system. REMOTENAME can be one through four characters in length if the REMOTESYSTEM parameter specifies another CICS region, or one through eight characters in length if REMOTESYSTEM specifies an IMS system. IMS uses 8-character names, and if REMOTENAME is less than 8, IMS will translate it into a useable format. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >.

If you specify REMOTESYSTEM and omit REMOTENAME, the value of REMOTENAME defaults to

the local name; that is, the TRANSACTION name on this definition. Note that the transaction need not necessarily reside on the remote system or region.

REMOTESYSTEM(name)

The name of the CONNECTION definition of the intercommunication link on which the transaction attach request will be sent. The name can be up to 4 characters in length. The acceptable characters are: A-Z 0-9 \$ @ and #. Lowercase characters are converted to uppercase.

This attribute is used for CICS function request shipping (asynchronous processing and transaction routing). For more details of these methods of intercommunication, see the *CICS/ESA Intercommunication Guide*.

RESSEC({NO|YES})

Code this to indicate whether resource security checking is to be used for resources accessed by this transaction.

NO

All resources will be available to any user who has the authority to use this transaction.

YES

An external security manager will be used. For more details about external security checking, see the *CICS/ESA CICS-RACF Security Guide*.

RESTART({NO|YES})

Indicates whether the transaction restart facility is to be used to restart those tasks that terminate abnormally and are subsequently backed out by the dynamic transaction backout facility.

If RESTART(YES) is specified, the task that failed is restarted from the beginning of the initial program. If dynamic transaction backout fails, or if restart is suppressed dynamically, DFHPEP will be invoked in the normal way. The transaction restart facility is especially useful in such situations as a program isolation deadlock, where the task can be restarted automatically rather than resubmitted manually. For more details of automatic transaction restart, see the *CICS/ESA Recovery and Restart Guide*.

NO

The restart facility is not required.

YES

The restart facility is to be used.

RSL({0|value|PUBLIC})

The RSL keyword is not valid in CICS/ESA 4.1. See Appendix A, “Obsolete attributes retained for compatibility” on page 341 for information.

RUNAWAY(SYSTEM|number)

The amount of time, in milliseconds, for which any task running under this transaction definition can have control of the processor before it is assumed to be in a runaway condition (logical loop). When this interval expires, CICS can abnormally terminate the task.

SYSTEM

Specifies that CICS is to use the ICVR system initialization parameter value as the runaway time limit for this transaction.

number

Specifies the runaway time limit in the range 0—2700000.

If you specify 0 (zero) it means there is no limit and that no runaway task detection is required for the transaction.

SHUTDOWN(DISABLED|ENABLED)

applies to transactions associated with a terminal, and indicates whether the transaction can be run during CICS shutdown. This supplements the XLT option on EXEC CICS PERFORM SHUTDOWN, so in order for a transaction to be attached during shutdown it must either be defined as SHUTDOWN(ENABLED) or named in the XLT specified in the EXEC CICS SHUTDOWN command.

DISABLED

The transaction is disabled from running during CICS shutdown.

ENABLED

The transaction is enabled to run during CICS shutdown.

SPURGE({NO|YES})

Indicates whether the transaction is initially “system purgeable” or not.

SPURGE=NO prevents a transaction being purged by the deadlock time-out (DTIMOUT) facility, an EXEC CICS ... PURGE command, TWAOC (Cancel Task) being set in the node error program (NEP), or a CEMT SET ... PURGE command.

SPURGE=YES allows such purges to go ahead as far as the user is concerned. CICS may, however, prevent the purge if it is not safe to allow a purge at the point the transaction has reached.

Note that SPURGE=NO does not prevent a transaction being purged by the read time-out (RTIMOUT) facility, an EXEC CICS SET ... FORCEPURGE command, or a CEMT SET TRANSACTION(tranid) FORCEPURGE command. SPURGE only determines the initial value, which can be changed by the transaction while it is running.

NO

The transaction is not initially system purgeable.

YES

The transaction is initially system purgeable.

STATUS({ENABLED|DISABLED})

The transaction status.

ENABLED

Allows the transaction to be executed normally.

DISABLED

Prevents the transaction being executed.

STORAGECLEAR(NO|YES)

indicates whether task-lifetime storage for this transaction is to be cleared or not upon release. This can be used to prevent other tasks accidentally viewing any confidential or sensitive data that was being stored by this transaction in task lifetime storage.

TASKDATAKEY(USER|CICS)

This specifies the storage key of the storage CICS allocates at task initialization for the duration of the task (task-lifetime storage), and which is accessible by the application. These storage areas are the EXEC interface block (EIB) and the transaction work area (TWA).

TASKDATAKEY also specifies the key of the storage that CICS obtains on behalf of all programs that run under the transaction. The program-related storage that CICS allocates in the specified key includes:

- The copies of working storage that CICS obtains for each execution of an application program
- The storage CICS obtains for the program in response to implicit and explicit GETMAIN requests. For example, the program can request storage by a GETMAIN command, or as a result of the SET option on other CICS commands.

You must specify TASKDATAKEY(USER) if any of the programs in the transaction are defined with EXECKEY(USER). If you specify TASKDATAKEY(CICS) for a transaction, an attempt to run any program in user key under this transaction leads to a task abend, with abend code AEZD.

USER

CICS obtains user-key storage for this transaction. Application programs executing in any key can both read and modify these storage areas.

Note: User-key programs of transactions defined with ISOLATE(YES) have access only to the user-key task-lifetime storage of their own tasks.

User-key programs of transactions defined with ISOLATE(NO) also have access to the user-key task-lifetime storage of other tasks defined with ISOLATE(NO).

See the description of the EXECKEY option on the PROGRAM definition for more information about task storage protection.

CICS

CICS obtains CICS-key storage for this transaction. Application programs executing in CICS key can both read and modify these storage areas. Application programs executing in user key can only read these storage areas.

TRANSACTION

TASKDATALOC(BELOW|ANY)

This specifies whether task life-time storage acquired by CICS for the duration of the transaction can be located above the 16MB line in virtual storage. These areas, which relate to specific CICS tasks, include the EXEC Interface Block (EIB) and the Transaction Work Area (TWA).

You must specify TASKDATALOC(BELOW) if any of the programs that make up the transaction does the following:

- Runs in 24-bit addressing mode (this also applies to task-related user exits running on behalf of the transaction)
- Issues local DL/I requests using the DLI CALL interface.

For transactions that do not satisfy any of these conditions you can specify ANY to obtain the associated virtual storage constraint relief.

CICS polices the use of TASKDATALOC(ANY). In particular:

- An attempt to invoke an amode 24 program running under a transaction defined with TASKDATALOC(ANY) results in an AEZC abend.
- An attempt to issue an EXEC CICS command or call a task related user exit whilst running amode(24) with TASKDATALOC(ANY) specified, results in an AEZA abend.
- An amode 31 program running as a transaction with TASKDATALOC(ANY) that attempts to invoke a task related user exit that is forced to run amode 24, results in an AEZB abend.
- If a task-related user exit that is forced to run in amode 24 is enabled for task start, CICS forces TASKDATALOC(BELOW) for all transactions for the remainder of the CICS run.

BELOW

Storage areas that CICS acquires for the transaction must be located below the 16MB line.

ANY

Storage areas that CICS acquires for the transaction can be located above the 16MB line in virtual storage.

TASKREQ(value)

This allows a transaction to be initiated by pressing a PF key, by using a light pen, or by using a card. Possible values are:

- PA1, PA2, or PA3** - for PA keys.
- PF1 through PF24** - for PF keys.
- OPID** - for the operator identification card reader.
- LPA** - for a light-pen-detectable field on a 3270 device.

MSRE - for the 10/63 character magnetic slot reader.

Here are some notes on the use of PF and PA keys:

- If a PA or PF key is specified in the PRINT operand of DFHSIT or a SIT override, you cannot use the same PF key as the TASKREQ to initiate a transaction.
- PA or PF keys specified in the SKRxxxx operand of DFHSIT or a SIT override as page retrieval keys will be interpreted as such during a page retrieval session. You can use the same keys to initiate transactions at other times. The keys should be defined with the following values:

```
TASKREQ=KEY-ID  
PROGRAM=DFHTPR  
TWSIZE=1024  
TPURGE=NO  
SPURGE=NO
```

- If you define a transaction with PROGRAM(DFHTPR), and a TASKREQ key, the key will initiate the transaction and open the page retrieval session at the same time.

Note: For transactions that had a TASKREQ but no TRANSID before migration to RDO, there may be an incompatibility in the use of EIBTRNID by application programs. (See the *CICS/ESA Application Programming Guide* for information.)

TCLASS(NO|value)

As a result of the introduction of TRANCLASS, the TCLASS attribute is obsolete in CICS/ESA 4.1. If you already use TCLASS, you can still access it by using compatibility mode (see "Compatibility mode (CSD sharing)" on page 19 for information). See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for a description of TCLASS.

TPNAME(name)

The name of the transaction that may be used by an APPC partner if the 4 character length limitation of the TRANSACTION parameter is too restrictive. This name can be up to 64 characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ? ! : | " = ~ , ; < and >. If this range of characters is not sufficient for a name that you wish to specify, you may use the XTPNAME attribute instead of TPNAME.

TPURGE(NO|YES)

Indicates (for non-VTAM terminals only) that the transaction can be purged because of a terminal error.

NO

The task cannot be purged when a terminal error occurs. Manual intervention by the master terminal operator will be required when this happens.

YES

The task can be purged when a terminal error occurs.

TRACE({YES|NO})

Indicates whether the activity of this transaction is to be traced.

YES

Trace the activity for this transaction.

NO

Do not trace the activity for this transaction.

Note: The CICS-provided transaction definitions

(previously PCT entries) for CEDF and CSGM have TRACE=NO coded.

TRANCLASS(name)

is the name of the transaction class to which the transaction belongs. Transactions belonging to a transaction class are subject to scheduling constraints before they are allowed to execute. The reserved TRANCLASS name DFHTCL00 is used to indicate that the transaction does not belong to any transaction class.

In previous releases of CICS, you could specify a transaction class by using the now obsolete TCLASS attribute. TCLASS specified the value of the associated class. Compatibility with these classes is preserved if you specify the following TRANCLASS equivalents:

TRANSACTION

TCLASS	TRANCLASS
--------	-----------

NO	DFHTCL00
1	DFHTCL01
2	DFHTCL02
3	DFHTCL03
4	DFHTCL04
5	DFHTCL05
6	DFHTCL06
7	DFHTCL07
8	DFHTCL08
9	DFHTCL09
10	DFHTCL10

Note: If a transaction is run and its associated TRANCLASS definition is not installed, the transaction will run without any of the scheduling constraints specified in the TRANCLASS. Message DFHXM0212 is issued as a warning.

TRANCLASS can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters.

TRANSACTION(name)

The name of the transaction, or transaction identifier (TRANSID). The name can be up to 4 characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ? ! : | " = ~ , ; < and >. Do not use transaction names beginning with C because this character is reserved for use by CICS. Also, the '%' and '&' characters can cause problems to RACF if transaction security is active. You should check that the transaction names are acceptable to the external security manager that you are using.

Note: If you use a comma (,) in a name, you will be unable to use those commands such as

```
CEMT INQUIRE TRANSACTION(value1,value2)
CEMT SET TRANSACTION(value1,value2)
```

where the comma serves as a list delimiter. See the *CICS/ESA CICS-Supplied Transactions* manual for information about using lists of resource identifiers.

If you wish to use other special characters in a transaction identifier, use the XTRANID attribute to specify another name which can be used to initiate the transaction. You must also specify a TRANSACTION name, because this is the name by which the TRANSACTION definition is known on the CSD.

When defining a transaction you must also name either a PROGRAM or a REMOTESYSTEM.

TRANSEC

The TRANSEC keyword is not valid in CICS/ESA 4.1. See Appendix A, "Obsolete attributes retained for compatibility" on page 341 for more information.

TRPROF=(DFHCICSS|name)

The name of the PROFILE for the session that will carry intersystem flows during ISC transaction routing. The name can be up to eight characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ? ! : | " = ~ , ; < and >.

You can specify this only for remote transactions.

TWASIZE({0|value})

The size (in bytes) of the transaction work area to be acquired for this transaction. Specify a one- to five-digit decimal value in the range 0 through 32767.

Notes:

1. Your storage may be corrupted if your TWASIZE is too small.
2. Do not change the TWASIZE of the CICS-supplied transactions.

XTPNAME(value)

This attribute may be used as an alternative to TPNAME. Enter a hexadecimal string up to 128 characters in length, representing the name of the transaction that may be used by an APPC partner. All hexadecimal combinations are acceptable **except X'40'**. In order to specify an XTPNAME over 72 characters long to DFHCSDUP, put an asterisk in column 72. This causes the following line to be concatenated.

XTRANID(value)

You can use this optional attribute to specify another name to be used instead of the TRANSACTION name for initiating transactions. The name may be up to eight hexadecimal digits in length. Because XTRANID is specified in hexadecimal code, you can use a name which contains unprintable characters, or characters not allowed for TRANSACTION names.

(See also TASKREQ, another transaction alias that can be specified.)

value

A 4-byte transaction identifier in hexadecimal notation (the identifier therefore uses up to eight hexadecimal digits). If you specify fewer than eight hexadecimal digits, the identifier will be padded on the right with blanks. XTRANID should not begin with X'C3', or anything less than or equal to X'40' and should not end with X'FFFFFF', as all these values are reserved for use by CICS. A value of X'00000000' is also not allowed.

Chapter 24. TYPETERM

A TYPETERM definition is an extension of a TERMINAL definition; it specifies a set of attributes common to a group of terminals. The TYPETERM attributes are listed on page 225. If you have a number of terminals with the same properties, you would define one TYPETERM with the required values, then name that TYPETERM in each TERMINAL definition (or in the autoinstall model definition if you are using autoinstall).

Every TERMINAL definition must name a TYPETERM definition. This single attribute represents many other characteristics, and thus can save considerable effort, and reduce the chance of making mistakes. TYPETERMs make it easier to define your terminals if you have many terminals of the same kind.

Two TYPETERM attributes are worthy of note here, because they further simplify the terminal definition process:

- DEVICE
- QUERY.

DEVICE

DEVICE specifies the **device type** that the TYPETERM represents. This is a key attribute, because the default values for a number of other attributes depend on the value you supply for it. You must supply a value for the DEVICE attribute, because there is no default.

Some attributes are always the same for every device of the same type. You do not need to define all these attributes yourself, because RDO knows what they are. All you need to tell RDO is the device type of your terminals, when you define the TYPETERM for them. Values for the fixed attributes are supplied automatically.

For a list of terminals supported by RDO, see "Devices supported" on page 221. There is also a list of valid values for the DEVICE attribute of TYPETERM in Table 13 on page 219. This shows you the other attribute values supplied for different device types. In some cases, these values depend also on your values for SESSIONTYPE and TERMMODEL, but these too have defaults that depend on the DEVICE specified.

Apart from ordinary display devices, printers, and other more specialized input and output devices, you can create a TYPETERM definition for your CICS consoles.

QUERY

The QUERY attribute allows you to leave some features of your terminals undefined until they are connected. Information about these attributes can then be obtained by CICS itself using the QUERY structured field.

The attributes for which you can use QUERY are all TYPETERM attributes. They are:

ALTPAGE
ALTSCREEN
APLTEXT
BACKTRANS
CGCSGID
COLOR
EXTENDEDDES
HILIGHT
MSRCONTROL
OBFORMAT
OUTLINE
PARTITIONS
PROGSYMBOLS
SOSI
VALIDATION

TYPETERM

The use of QUERY overrides any values that are explicitly defined for all the TYPETERM attributes listed above, **except ALTSCREEN**. QUERY-supplied ALTSCREEN values are used only if no ALTSCREEN values are explicitly defined in the TYPETERM.

You can use QUERY for 3270 devices with the extended 3270 data stream. The DEVICE types for which you can use QUERY are:

3270
3270P
LUTYPE2
LUTYPE3
SCSPRINT

You can specify that QUERY be used in one of two ways:

- QUERY(COLD) specifies that the QUERY is only to be issued when the terminal is first connected after a cold start.
- QUERY(ALL) specifies that the QUERY is to be issued each time the terminal is connected.

The QUERY function is particularly useful with configurable devices, such as the IBM Personal System/2 (PS/2) and the IBM 3290. It enables you to reconfigure the device between logging off and logging on to CICS, without having to change any resource definitions.

The QUERY function is also particularly useful when used in conjunction with autoinstall.

Note that the QUERY facility obtains only the information required by CICS. If an application program needs to determine other device characteristics, it still needs to send a QUERY structured field and analyze the reply.

To summarize, you may need only one TYPETERM definition for each device type. If the attributes that can be determined by QUERY differ among the terminals, you still need only one TYPETERM for each device type. If other attributes of your terminals vary, you may need more than one TYPETERM definition for a device type.

There are some CICS-supplied TYPETERM definitions suitable for the more frequently used terminals. These are described in "TYPETERM definitions in group DFHTYPE" on page 361.

Terminal definition — summary

Now you have seen that, if all your terminals are basically the same, you could get away with having only one TYPETERM definition, and one TERMINAL definition with AUTINSTMODEL(YES). You might, perhaps, use QUERY to deal with different features used by your terminals.

Dependent default values

See “Dependent default values” on page 26 for an explanation of dependent default values.

<i>Table 13 (Page 1 of 2). Default values for different devices</i>								
Values specified by you:			Default values supplied by RDO:					
DEVICE	SESSION TYPE	TERM MODEL	DEF. SCR.N.	PAGE SIZE	AUTO PAGE	BRAC -KET	BUILD CHAIN	ROU TD MSGS
3270 (3277, L3277) (a)		1	12,40	12,40	N	Y	N	All
3270		2 (dft)	24,80	24,80	N	Y	N	All
3275		1	12,40	12,40	N	Y	N	All
3275		2 (dft)	24,80	24,80	N	Y	N	All
3270P (3284, L3284, 3286, L3286) (b)		1	12,40	12,40	Y	Y	N	All
3270P		2 (dft)	24,80	24,80	Y	Y	N	All
APPC			0,0	1,40	Y	Y*	Y*	None*
CONSOLE			0,0*	1,124*	N	N*	N*	None
LUTYPE2		1	12,40	12,40	N	Y*	Y*	All
LUTYPE2		2 (dft)	24,80	24,80	N	Y*	Y*	All
LUTYPE3		1	12,40	12,40	Y	Y*	N	All
LUTYPE3		2 (dft)	24,80	24,80	Y	Y*	N	All
LUTYPE4			0,0	50,80	Y	Y*	N	All
BCHLU	(dft)		0,0	12,80	Y	Y*	N	All
BCHLU	BATCHDI		0,0	12,80	Y	Y*	N	All
BCHLU	USERPROG		0,0	12,80	Y	Y*	N	All
INTLU			0,0	12,80	Y	Y	N	All
SCSPRINT			0,0*	24,80	Y	Y*	N	All
TLX or TWX	CONTLU (dft)		0,0	1,40	Y	Y	N	All
TLX or TWX	INTLU		0,0	1,40	Y	Y	N	All
3600	(dft)		0,0	1,40	Y	Y	N	All
3600	PIPELINE		0,0	6,30	Y	Y	N	All
3614			0,0	1,40	Y	Y	N	All
3650	USERPROG (dft)		0,0	3,80	Y	Y	N	All
3650	3270		12,40	23,80	Y	Y	N	All
3650	3653		0,0	6,30	Y	Y	N	All
3650	PIPELINE		0,0	6,30	Y	Y	N	All
3767			0,0	12,80	Y	Y*	N	All
3767C			0,0	12,80	Y	Y*	N	All
3767I			0,0	12,80	Y	Y*	N	All
3770	(dft)		0,0	12,80	Y	Y*	N	All
3770	USERPROG		0,0	12,80	Y	Y*	N	All
3770	BATCHDI		0,0	12,80	Y	Y*	N	All
3770B	(dft)		0,0	12,80	Y	Y*	N	All
3770B	BATCHDI		0,0	12,80	Y	Y*	N	All
3770B	USERPROG		0,0	12,80	Y	Y*	N	All
3770C			0,0	12,80	Y	Y*	N	All

TYPETERM

Table 13 (Page 2 of 2). Default values for different devices

Values specified by you:			Default values supplied by RDO:					
3770I			0,0	12,80	Y	Y*	N	All
3790	(dft)		0,0	1,40	Y	Y*	N	None*
3790	SCSPRINT		0,0	24,80	Y	Y*	N	All
3790	USERPROG		0,0	12,80	Y	Y*	N	All
3790	BATCHDI		0,0	12,80	Y	Y*	N	All
3790	3277CM (c)	1	12,40	12,40	N	Y*	Y*	All
3790	3277CM (d)	2 (dft)	24,80	24,80	N	Y*	Y*	All
3790	3284CM (e)	1	12,40	12,40	Y	Y*	N	All
3790	3284CM (f)	2 (dft)	24,80	24,80	Y	Y*	N	All
3790	3286CM (e)	1	12,40	12,40	Y	Y*	N	All
3790	3286CM (f)	2 (dft)	24,80	24,80	Y	Y*	N	All

Notes:
 * denotes that this value is forced; it is not just a default but it is mandatory for the DEVICE type.
 (a) 3277 and L3277 are converted by RDO to 3270.
 (b) 3284, 3286, L3284, and L3286 are converted to 3270P.
 (dft) indicates the default SESSIONTYPE or TERMMODEL value assigned by RDO if you specify the DEVICE only.
 (c) is converted to DEVICE(LUTYPE2) TERMMODEL(1).
 (d) is converted to DEVICE(LUTYPE2) TERMMODEL(2).
 (e) is converted to DEVICE(LUTYPE3) TERMMODEL(1).
 (f) is converted to DEVICE(LUTYPE3) TERMMODEL(2).

Devices supported

This chapter lists the device names that you can use on the TYPETERM.

Table 14 is a list of terminal types supported by RDO. If you have devices other than those shown, you may be able to install them on an earlier version of CICS (before CICS/ESA 3.1) and use transaction routing.

To use Table 14, find the family number of your device in the leftmost column (headed **Terminal or System Type**). Then look across at the second column (headed **Units**) to see if your type of units are specifically mentioned. Next look at the third column (headed **Attachment**) to see what device type you should use on your TYPETERM definition.

There is further explanation of the information in Table 14 in the notes that follow. The following abbreviations have been used:

local	channel or adapter attached
s/s	start/stop transmission
SDLC	synchronous data link control
sw	switched
BSC	binary synchronous
nonsw	non-switched communications.

Table 14 (Page 1 of 2). VTAM terminals and subsystems supported by RDO

Terminal or System Type	Units	Attachment	Notes
3101		Supported as TWX 33/35	1,10
3230		Supported as 2741 (BSC/BTAM) or INTLU (VTAM)	1
3270	3178, 3179, 3180, 3262, 3271, 3272, 3274, 3276, 3290	local, SDLC, BSC nonsw	1,2, 3
	3275 3277, 3278, 3279, 3284, 3286, 3287, 3288, 3289	BSC sw or nonsw	3
3270PC	3270PC, 3270PC/G, 3270PC/GX	Supported as 3270	
3287	models 11, 12	SDLC supported as SCSVRT	13
3600	3601, 3602, 3690, 3604, 3610, 3612, 3618, 3614, 3624	SDLC, BSC nonsw	1,4,5,14
3630	3631, 3632, 3643, 3604	attached as 3600	4,11
3640	3641, 3644, 3646, 3647	SDLC attached as INTLU	13
	3642	SDLC attached as SCSVRT	13
	3643	SDLC supported as LUTYPE2	13
	3645	SDLC supported as SCSVRT	13
3650	3651, 3653, 3275, 3284	SDLC	4
3680	3684	Supported as 3790/3650	4
3730	3791	Supported as 3790	4
3767		SDLC s/s supported as 2740/2741	1
3770	3771, 3773, 3774	SDLC	1,4,6
	3775, 3776, 3777	BSC supported as 2770	
3790	3791	SDLC or local	4,7

TYPETERM

Table 14 (Page 2 of 2). VTAM terminals and subsystems supported by RDO			
Terminal or System Type	Units	Attachment	Notes
4300	4331, 4341, 4361, 4381	BSC or SDLC	4,8
4700	4701-1	Supported as 3600	1,4,5
5280		Supported as 3741 (BSC/BTAM) or 3270 (BTAM or VTAM)	1
5520		SDLC supported as 3790 full-function LU BSC supported as 2770 SDLC attached as APPC	1,4
5550		Supported as 3270	
5937		SDLC/BSC attached as 3270	1,3
6670		SDLC BSC supported as 2770	1
8100	8130/8140 processors with DPCX	Supported as 3790	4
	DPPX/BASE using Host Presentation Services or Host Transaction Facility	Attached as 3790	4
	DPPX/DSC or DPCX/DSC (including 8775 attach)	Supported as 3270	4,12
8775		SDLC supported as LUTYPE2	
8815		Supported as APPC	
Displaywriter		Supported as APPC SNA for EDDS Supported as 3270; attached as 2741 (s/s) or 3780 (BSC) SDLC attached as APPC	1
Personal Computer		Supported as 3270 and as APPC	1,14
PS/2		Supported as 3270 and as APPC	1,14
Scanmaster		Supported as APPC	
Series/1		Attached as System/3; supported as 3650 Pipeline (VTAM) or 3790 (full function LU)	1,4
System/32	5320	SDLC supported as 3770	1,4,9
		BSC supported as 2770	4
System/34	5340	SDLC supported as 3770 BSC attached as System/3	1,4,9 4
System/36		Supported as System/34 SDLC attached as APPC	1
System/38	5381	SDLC attached as 3770 SDLC attached as APPC BSC attached as System/3	1,4,9 4
AS/400	5381	SDLC attached as 3770 SDLC attached as APPC BSC attached as System/3	1,4,9 4
System/370		BSC or SDLC and APPC	1,4,8
System/390		BSC or SDLC and APPC	1,4,8
TWX 33/35		VTAM (via NTO) ss sw BTAM ss sw	1,10
WTTY		VTAM (via NTO) ss nonsw BTAM ss sw	1,10

Notes:

1. These devices are also supported by BTAM and you may connect them to CICS by using transaction routing from CICS releases before CICS/ESA 3.1.
2. CICS supports the 3290 through both the terminal control and the BMS interfaces. The 3290 can be in one of three states: default, alternate, or partitioned. Up to 16 partitions can be defined. The 3290 also has the programmed symbols and extended highlighting features, as well as two kinds of data validation feature, mandatory fill and mandatory enter. A 3290 terminal can be configured as from one to five logical units. You define the size of each logical unit when setting up the 3290. You must ensure that the resource definitions of each logical unit match the set-up size, to prevent unpredictable results. Up to four interactive screens in any configuration can be active at the same time, but only one interactive screen can be defined as having programmed symbols at any time; that is, all programmed symbol sets must be assigned to the interactive screen.

To display long lines of data, such as the 132-character lines of CEMT output, you should specify a default screen width of 132 characters.

If you intend to use the large buffer, you might have to specify a much larger value for IOAREALEN (see page 231). Whether you need to do this depends on whether operators are likely to modify, or enter, large quantities of data. If a terminal is used for inquiry only, or for limited data entry, IOAREALEN need not be large.

3. SDLC 3270s are supported only through VTAM, and the switched BSC 3275 (feature #3440) is supported only through BTAM. Printers attached to local or SDLC 3274s and SDLC 3276s are supported through VTAM either as LU Type 3 using the 3270 printer data stream or as LU Type 1 using the SCS data stream (which is a subset of that used for SDLC 3767, 3770, and 3790 printers). The 3288 is supported as a 3286 Model 2. CICS supports the 3270 copy feature (#1550).
4. Devices and features supported by a system or programmable controller are generally transparent to CICS. In some cases, CICS provides specific device support, in which case the units are listed.
5. SDLC is supported through VTAM. BSC attachment (RPQ 8K0598) is supported through BTAM. The 3614 is supported both for loop attachment to the 3601 and SDLC attachment to the host via a 3704/3705 Communications Controller.
The 3614 is supported by CICS as BSC only when loop-attached to the 3601/3602 Controllers. The 3624 is supported as a 3614.
The 3690 is supported as a 3602.
6. CICS supports the Data Transfer Function of the SDLC Programmable Models of the 3770 Data Communication System. In using this function, you are responsible for allocating data sets and managing the program library.
7. CICS does not support the 3790/Data Entry Configuration using 3760s. The #9165 or #9169 configuration is required to support the CICS enhancement first made available in Version 1 Release 3.0.

Printers on 3790 systems are supported with one of the following:

- a function program provided by you, or
- 3270 data stream compatibility with a 3270 printer data stream (LU3), or
- an SCS data stream supporting a subset of that for SDLC 3767 (LU1).

When operating in 3270 mode, the 3288 Model 2 is supported as a 3286 Model 2.

8. System/370 and 4300 attachment by BSC requires a suitable telecommunications program (for example, the VSE/3270 Bisync Pass Through Program) in the system connected to CICS. CICS is **not** a suitable program for the remote CPU. Attachment by SDLC is supported by CICS intersystem communication.
9. The System/32 with its SNA/SDLC workstation system utility program, and the System/34 and System/38, are supported as compatible versions of an appropriately featured 3770 Communication System operating as a batch logical unit. The System/34

or System/38 user-written program is responsible for supporting the correct SNA sequences of the attached subsystem.

10. TWX and WTTY are supported through VTAM via the Network Terminal Option program product (5735-XX7), with attachments as defined by NTO.

TWX (Line Control Type) is attached through BTAM on eight-level code at 110 bps on common-carrier switched 150 bps networks.

WTTY is attached at 50 bps on common-carrier switched networks where the terminals supported are those interfacing through IBM World Trade Corporation Telegraph Terminal Control with Telegraph Line Adapter.

The transmission code used is International Telegraph Alphabet No. 2 (CCITT No. 2). CICS/ESA does not support autocal or automatic host disconnect via WRITE break.

11. The 3643 is supported as a 3604.
12. 8775 support includes validation of mandatory fill and mandatory enter field attributes.
13. Attachment is via the Loop Adapter #4830, #4831, and Data Link Adapter #4840 of the 4331 processor.
14. 3270 support requires that the 3278/3279 Emulation Adaptor be installed in the Personal Computer or PS/2.
15. The 3600 pipeline logical unit is designed to provide high throughput for particular types of transaction, such as credit card authorization or general inquiry applications. To achieve a high throughput of inquiry messages and their replies, the CICS pipeline session uses a restricted set of the communication protocols that are used with the 3601 logical unit. These restrictions result in a full duplex message flow whereby many inquiry messages are outstanding at any one time, and the replies may flow back in a different order from that of the original inquiries. The 4700/3600 application program controlling the inquiry terminals is responsible for maintaining the protocols as well as correlating replies with inquiries and controlling message flow to the group of terminals associated with the pipeline logical unit.

Defining a TYPETERM

+

```

TYPeterm ==>
Group ==>
DEscription ==>
RESOURCE TYPE
DEvice ==> (See Table 13 on page 219)
TERmodel ==> (See Table 13 on page 219)
SESSiontype ==> (See Table 13 on page 219)
LDclist ==>
SHippable ==> No No | Yes
MAPPING PROPERTIES
PAGesize ==> 024 , 080 0-999
ALTPage ==> 000 , 000 0-999
ALTSuffix ==>
FMhparm ==> No No | Yes
OBOperid ==> No No | Yes
PAGING PROPERTIES
AUTOPage ==> No | Yes
DEVICE PROPERTIES
DEFscreen ==> 000 , 000 0-999
ALTScreen ==> 0-999
APLKybd ==> No No | Yes
APLText ==> No No | Yes
AUDiblealarm ==> No No | Yes
COLor ==> No No | Yes
COPy ==> No No | Yes
DUAlcasekybd ==> No No | Yes
EXTendedds ==> No No | Yes
HILight ==> No No | Yes
Katakana ==> No No | Yes
LIGhtpen ==> No No | Yes
MSrcontrol ==> No No | Yes
OBFormat ==> No No | Yes
PARTitions ==> No No | Yes
PRIntadapter ==> No No | Yes
PROGsymbols ==> No No | Yes
VALidation ==> No No | Yes
FORmfeed ==> No No | Yes
HORizform ==> No No | Yes
VERticalform ==> No No | Yes
TEXTKybd ==> No No | Yes
TEXTPrint ==> No No | Yes
Query ==> No No Cold | All
OUTline ==> No No | Yes
SOSi ==> No No | Yes
BACKtrans ==> No No | Yes
CGcsgid ==> 00000 , 00000 0-65535
SESSION PROPERTIES
AScii ==> No No | 7 | 8
SENdsiZe ==> 00000 0-30720
RECEivesize ==> 0-30720
BRacket ==> Yes Yes | No
LOGMODE ==>
LOGMODECom ==> No No | Yes
LOGMode ==>
DIAGNOSTIC DISPLAY
ERRLastline ==> No No | Yes
ERRIntensify ==> No No | Yes
ERRColor ==> NO NO Blue | Red | Pink | Green
| Turquoise | Yellow | NEutral
No | Blink | Reverse | Underline
ERRHilight ==> No
continued...

```


#

Figure 42 (Part 1 of 2). The DEFINE panel for TYPETERM

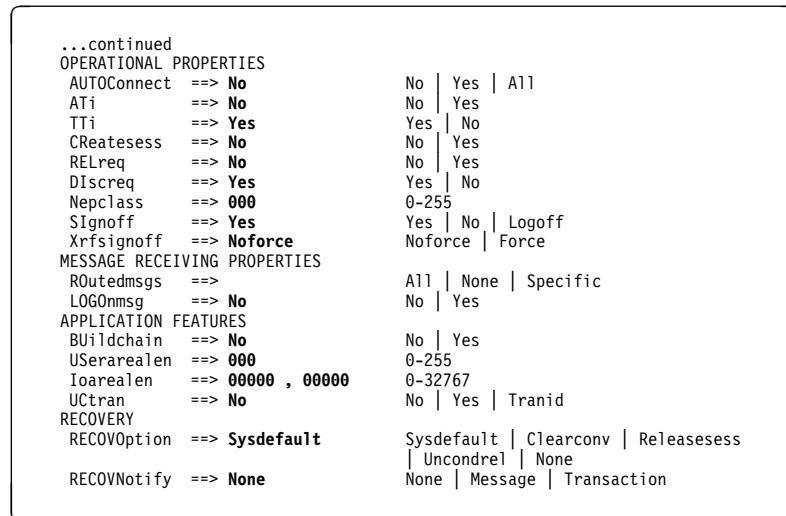


Figure 42 (Part 2 of 2). The DEFINE panel for TYPETERM

ALTPAGE({0|rows}, {0|columns})

Indicates the page size to be used by BMS for this terminal entry when ALTSCREEN has been selected as the screen size. The default is the same as PAGESIZE. The values for both rows and columns must be in the range 0 through 999.

Unexpected results will occur if the columns value of ALTPAGE is different from that of ALTSCREEN. The lines value of ALTPAGE can usefully be less than that of ALTSCREEN, perhaps to reserve the bottom line of the screen for error messages.

If you use the QUERY structured field (see page 217), the alternate page size used will be the size set up as the alternate screen size. For querable terminals it is possible to have ALTPAGE set to zero and the ALTSCREEN value defined explicitly by the CINIT BIND. If ALTPAGE is not zero, it is possible to have different values for the ALTPAGE and the ALTSCREEN.

lines x columns must not exceed 32767.

ALTSCREEN(rows,columns)

Defines the 3270 screen size to be used for a transaction that has SCRNSIZE(ALTERNATE) specified in its profile. The values that can be specified are:

Device	Alternate Screen Size
3276-1, 3278-1	(12,80)
3276-2, 3278-2	(24,80)
3276-3, 3278-3	(32,80)
3276-4, 3278-4	(43,80)
3278-5	(27,132)
3279-2A, 3279-2B	(24,80)
3279-3A, 3279-3B	(32,80)

Note that there is no validity checking performed on the screen size selected, and that incorrect sizes may lead to unpredictable results.

For BSC devices, both the alternate and default screen sizes are determined by the device hardware. The alternate screen size is the maximum screen size. For the 3290 display, both the default and alternate screen sizes are determined by the customer setup procedure. For further guidance, see the *IBM 3290 Information Panel Description and Reference* manual, GA23-0021.

For SNA devices (LUTYPE2 and LUTYPE3), both alternate and default screen sizes can be any value chosen by yourself, up to the maximum physical screen size. In particular, both the alternate and default screen sizes can be the maximum screen size, or the default screen size can be the maximum screen size with no alternate screen size specified. The SNA bind is generated by CICS from this information. There is no need for you to provide logmode table entries, or to customize the device.

For non-SNA 3270 and LUTYPE2 devices, you can use the QUERY structured field to determine the alternate screen size that has been set up for the display. To use QUERY, you leave the DEFSCREEN to default to (24,80) and you leave ALTSCREEN unspecified. The alternate screen size will be the size set up by the terminal user. Otherwise, QUERY(COLD) or QUERY(ALL) will have no effect on the alternate screen size. Leaving ALTSCREEN not specified without using QUERY under the conditions above will result in an alternate screen size of (00,00). For more information about QUERY, see page 217.

If dual screen sizes are used, you can make CICS transactions use the alternate screen size by coding SCRNSIZE(ALTERNATE) in their associated profiles. If an application consists of several pseudo-conversationally linked transactions, you should specify SCRNSIZE(ALTERNATE) in the profiles for each

of these transactions if the application uses the alternate screen size.

For 3287 and 3289 printers, the value specified must equal the buffer size of the particular device. For non-SNA 3287 and 3289 printers, the sizes depend on the feature ordered, not on the model number. For SNA printers, there are no features, and any two sizes can be specified from the list of valid sizes. When printing to a printer whose associated TERMINAL definition has PRINTERCOPY(YES) specified, the ALTSCREEN value should match the screen size of the terminal whose screen is to be printed. If the values differ, unpredictable results may occur.

ALTSUFFIX({blank|number})

A 1-character numeric suffix (specified in the SUFFIX operand of the application programmer's DFHMSD TYPE={DSECT|MAP} macro).

blank

Leave this attribute blank if you do not want a suffixed map set.

number

This suffix will be appended by BMS to map set names if the screen size being used is the same value as the alternate screen size, that is, if the transaction has SCRNSIZE(ALTERNATE) specified in the PROFILE definition, or if the default and alternate screen size are the same. In this case, BMS map selection routines will attempt to load the map set with the suffix specified in the ALTSUFFIX operand. If there is no such map set, BMS will try to load a map set suffixed with M or L and, if this load fails, BMS will try to load an unsuffixed map set version. If the transaction uses default screen size, BMS will first try to load a map set suffixed with M or L and, if this load fails, BMS will try to load an unsuffixed map set version.

To use a suffixed map set, you must code the system initialization parameter BMS=(,,DDS).

APLKYBD({NO|YES})

Indicates whether the 3270 device has the APL keyboard feature.

APLTEXT({NO|YES})

Indicates whether the 3270 device has the APL text feature. Do not specify YES for a 3288 printer, with or without TEXTPRINT(YES). The APLTEXT feature is used in conjunction with the TEXTKYBD and APLKYBD operands.

You can use the QUERY structured field to determine whether the device is set up to use the APL text feature. (See page 217.)

ASCII({NO|7|8})

Indicates whether the terminal has an ASCII feature.

NO

This terminal does not have an ASCII feature. This is the default.

- 7** Specify this to communicate with ASCII-7 terminals. Devices configured with the ASCII-7 feature must be LUTYPE2 or LUTYPE3 without extended 3270 features. Only the following devices are supported:

3274 Model 1C and 51C
3276 Model 12
3278
3287.

Any terminal configured with the ASCII-7 option will have all FM data outbound from CICS converted to ASCII-7, and all FM data inbound to CICS converted to EBCDIC. Only FM request data will be translated. All other data in the RU such as LU status or sense data will be assumed to be in EBCDIC on output. ASCII-7 does *not* support data streams that contain extended attributes, such as structured fields and function management headers.

The ASCII-7 support is available on 3274-1C as an option on the configuration of the standard microcode. The use of the ASCII-7 option is determined at session initiation by BIND parameters set by CICS as a result of the TCT definition described above.

- 8** Specify this to communicate with ASCII-8 terminals. Devices configured with the ASCII-8 feature can be LUTYPE1, LUTYPE2, or LUTYPE3 with or without extended 3270 and SCS data stream features.

Any terminal configured with the ASCII-8 option will have all FM data outbound from CICS converted to ASCII-8, and all FM data inbound to CICS converted to EBCDIC. All FM request data will be translated. This will include the AID, cursor address, FM headers and structured fields. Any other form of the RU such as LU status or sense data will be assumed to be in EBCDIC on input and will be transmitted in EBCDIC on output.

Note that this ASCII-8 support is intended only for devices which will operate in EBCDIC but will translate or retranslate the data stream to or from ASCII-8 as is done by this CICS support. This is because the data stream is treated as a character string and any binary number fields will be translated byte by byte as though they were graphic characters, thus they may not represent their true value while in ASCII form.

The ASCII-8 support is available as a microcode RPQ on the 3274 and is mutually exclusive with the ASCII-7 option. The use of the ASCII-8 option is determined at session initiation by BIND parameters set by CICS as a result of the TCT definitions described above.

TYPETERM

ATI({NO|YES})

Indicates whether transactions are allowed to be started at the terminal by automatic transaction initiation.

ATI(YES) allows transactions to be started at the terminal by transient data control or by an EXEC CICS START command issued by another transaction. If there is already a transaction at the terminal, the ATI transaction is held until it ends.

If you specify ATI(YES), you must specify an IOAREALEN of at least one byte, except for DEVICE(APPC) when ATI and IOAREALEN have forced default values of YES and 0.

If ATI is specified as YES and CREATESESS is specified as YES then if a transaction is initiated when the terminal is not ACQUIRED, it will be automatically acquired.

See also the TTI attribute.

AUDIBLEALARM({NO|YES})

Specify this for the audible alarm feature for a 3270 display or for a 3270 printer attached to a 3651 controller.

AUTOCONNECT({NO|YES|ALL})

AUTOCONNECT(YES) or (ALL) specifies that the session with the terminal is to be established (that is, BIND is to be performed) during CICS initialization, or when communication with VTAM is started using the CEMT SET VTAM OPEN command. If the connection cannot be made at this time because the terminal is unavailable, the link must be subsequently acquired using the CEMT SET TERMINAL(termid) INSERVICE ACQUIRED command, unless the terminal becomes available in the meantime and itself initiates communications.

Note: If you use the VTAM LOGAPPL function, do not code AUTOCONNECT(YES), as this can lead to race conditions, causing errors or hung logical units.

NO

CICS will not attempt to bind sessions when the connection is established.

YES

CICS will attempt to bind as a contention winner session, when the connection is established.

ALL

Not applicable.

For background information about AUTOCONNECT, see the *CICS/ESA Intercommunication Guide*.

AUTOPAGE({NO|YES})

Indicates whether BMS autopaging is to be used. See page 185 for details. You should specify YES for printer TYPETERMs and NO for display device TYPETERMs. The default will depend on the value you specify for the DEVICE keyword: the default values are indicated in Table 13 on page 219.

BACKTRANS({NO|YES})

Indicates whether the device has the background transparency feature.

You can use the QUERY structured field to determine whether the device is set up to use the background transparency feature (see page 217).

BRACKET({YES|NO})

Specify this if bracket protocol is not to be enforced for this logical unit. The default will depend on the value you specify for the DEVICE keyword (see Table 13 on page 219).

YES

Bracket protocol is to be used. This option is required for the 3790 inquiry and full function logical units. BRACKET(YES) is forced for many DEVICE types. (See Table 13 on page 219.)

NO

Bracket protocol is not to be used. You must specify BRACKET(NO) for a 3614 logical unit and the 3650 Host Command Processor (HCP) session.

BUILDCHAIN({NO|YES})

Specify BUILDCHAIN(YES) if CICS is to perform chain assembly prior to passing the input data to the application program.

The default will depend on the value you specify for the DEVICE keyword (see Table 13 on page 219).

NO

Any TIOA received by an application program from this logical unit will contain one request unit (RU).

YES

Any terminal input/output area (TIOA) received by an application program from this logical unit will contain a complete chain.

CGCSGID({0,0|value1,value2})

The coded graphic character set global identifier (CGCSGID) enables application programs to determine the character set supported at the device.

This information may be obtained from a QUERY structured field for some devices (see page 217). For others, you must supply this information here, so that application programs can retrieve it using the EXEC CICS ASSIGN command.

0,0 No CGCSGID is specified.

value1,value2

The CGCSGID consists of 2 five-digit decimal numbers which can take values in the range 1 through 65535. value1 is the graphic character set global identifier (GCSGID) and value2 is a specification of the code points for the set, the code page global identifier (CPGID).

COLOR({NO|YES})

Indicates that the 3270 device or the SCS printer has the extended color feature, which allows colors to be selected for each field or character.

You can use the QUERY structured field to determine whether the device is set up to use the color feature. (See page 217.)

COPY({NO|YES})

COPY(YES) specifies that the copy feature for a 3270 display or printer is included in the 3270 control unit. Leave it to default to COPY(NO) for 3270 compatibility mode logical units, because COPY(YES) will be ignored.

See also the PRINTERCOPY and ALTPRINTCOPY attributes of the TERMINAL definition.

For further details about screen copying, see the *CICS/OS/VS 3270 Data Stream Device Guide*.

CREATESESS({NO|YES})

Indicates whether sessions are to be created.

NO

Specify this for a status that prevents internally generated session requests from actually creating a session. During CICS execution, this status can only be generated by a CEMT command.

CREATESESS(NO) prevents EXEC START requests and automatic transaction initiation (ATI) requests for this terminal causing a session to be created. This means that the requests are either queued or rejected when no session is currently established.

YES

Specify this for a status that allows internally generated session requests to create a session. During CICS execution, this status can only be generated by a CEMT command.

CREATESESS(YES) allows EXEC START requests and automatic transaction initiation (ATI) requests for this terminal to cause a session to be created automatically.

DEFSCREEN({rows|columns})

Defines the 3270 screen size or 3270 printer page size to be used on this device when attached to a transaction or used by BMS for which SCRNSIZE(DEFAULT) has been specified in the PROFILE definition. The default will depend on the value you specify for the DEVICE keyword (see Table 13 on page 219). The values that can be specified for a BSC 3270 are:

Device	Screen Size
3278-1	(12,40)
3278-2	(24,80)
3276-3, 3278-3	(24,80)
3276-4, 3278-4	(24,80)
3278-5	(24,80)

Device	Screen Size
3279-2A, 3279-2B	(24,80)
3279-3A, 3279-3B	(24,80)

For BSC devices, both default and alternate screen sizes are determined by the terminal hardware. The default screen size is 24,80, except for the 3278-1 where it is 12,40.

For SNA devices (LUTYPE2 and LUTYPE3), both default and alternate screen sizes can be any value you choose, up to the maximum physical screen size (see ALTSCREEN). In particular, both default and alternate screen sizes can be the maximum screen size, or the default screen size can be the maximum screen size with no alternate screen size specified. The SNA bind is generated by CICS from this TCT information. You do not need to provide logmode table entries, or to customize the device.

DESCRIPTION(text)

You can provide a description of the resource you are defining in this field.

The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, you should ensure that for each left parenthesis there is a matching right one.

DEVICE(name)

Specify the device type which this TYPETERM will define. This attribute is mandatory for all TYPETERM definitions.

If you type DEVICE(xxxx), where xxxx is a valid device type, on the command line, together with SESSIONTYPE and TERMMODEL if appropriate, other attributes will be given appropriate default values. For further guidance, see "Dependent default values" on page 26. Entering or overtyping the DEVICE, SESSIONTYPE, or TERMMODEL values on the overtyping-to-modify panel will **not** provide these defaults. Using the ALTER command to change the DEVICE will leave you with the old defaults for the dependent attributes.

The valid keywords and the defaults for each device type are listed in Table 13 on page 219. The recommended keywords for non-SNA VTAM 3270 devices are 3270 and 3270P for displays and printers, respectively. The following keywords can also be specified and are retained for compatibility with previous releases:

- Displays: 3277 and L3277
- Printers: 3284 and L3284, 3286 and L3286.

For SNA VTAM 3270 devices use the LUTYPE2 or LUTYPE3 keyword as appropriate. LUTYPE2 logical units are those defined by SNA, which accept a 3270-display data stream. LUTYPE3 logical units are

TYPETERM

those defined by SNA, which accept a data stream similar to that for a 3270 printer.

For a list of device types supported by CICS/ESA, see "CICS terminals list" on page 324. See also Table 13 on page 219 for a list of valid device names and the default attributes that they generate.

DISCREQ=({YES|NO})

Indicates whether disconnect requests are to be honored.

YES

CICS is to honor a disconnect request for a VTAM device, and issue a VTAM CLSDST macroinstruction to terminate the VTAM session with that logical unit.

In addition, CESF LOGOFF or GOODNIGHT from the terminal will cause disconnection if you specify YES.

YES is essential if the TYPETERM definition is referenced by AUTINSTMODEL TERMINAL definitions, so that autoinstalled terminal entries can be deleted automatically.

NO

CICS is not to honor a disconnect request for a VTAM device.

DUALCASEKYBD({NO|YES})

Indicates whether a 3270 display has a typewriter keyboard or an operator console keyboard. Both uppercase and lowercase data can be transmitted with either of these keyboards.

ERRCOLOR({NO|color})

Indicates whether the error message will be displayed in color. Coding ERRCOLOR(color) implies ERRLASTLINE(YES).

The colors you can specify are:

BLUE
RED
PINK
GREEN
TURQUOISE
YELLOW
NEUTRAL.

ERRHIGHLIGHT({NO|BLINK|REVERSE|UNDERLINE})

Indicates whether the error message will be displayed with highlighting.

ERRINTENSIFY({NO|YES})

Indicates whether the error message will be displayed in an intensified field. Coding ERRINTENSIFY(YES) implies ERRLASTLINE(YES).

ERRLASTLINE({NO|YES})

Indicates where error messages will be displayed.

NO

An error message will be displayed at the current cursor position and without any additional attributes.

YES

An error message will be displayed starting at the beginning of the line nearest the bottom of the screen such that the message will fit on the screen.

Because all error messages occupy the same line, if the messages are received in quick succession they will overlay one another and the earlier messages may disappear before being read.

EXTENDEDDEDS({NO|YES})

Indicates whether the 3270 device or the SCS printer supports extensions to the 3270 data stream. EXTENDEDDEDS(YES) is implied if you specify YES for any one of the COLOR, HIGHLIGHT, PROGSYMBOLS, QUERY, or VALIDATION (3270 only) attributes.

- + If extended data stream (EXTENDEDDEDS) is set to YES,
- + the device will support the write structured field
- + COMMAND and Outbound Query structured field.

You can use the QUERY structured field to determine whether the device is set up to use the extended data stream (see page 217).

Use of the QUERY structured field sets EXTENDEDDEDS to YES if query is valid.

FMHPARM({NO|YES})

BMS is to accept user-supplied parameters for inclusion in the function management header built by BMS. You should specify YES only if the DEVICE type is 3650.

FORMFEED({NO|YES})

If the devices for which you are defining this TYPETERM have the forms feed feature, you can specify FORMFEED(YES). If you specify FORMFEED(YES), BMS will use the forms feed character when formatting output documents.

If DEVICE(SCSPRINT) is specified, BMS inserts a form feed character at the beginning of the data stream. This causes the device to skip to the top margin of a new page before starting to print. The top margin is defined by a set vertical format (SVF) data stream, and may be a line number equal to or greater than one. If a SVF data stream has not been sent to the printer, the top margin is line one. The line counter in the device is set to 1 when the operator sets up the paper. Note that the device may also perform an automatic form feed if an attempt is made to print beyond a bottom margin. The bottom margin is also determined by the SVF data stream and will default to the maximum presentation line (MPL). The MPL is the last line on the page and its value represents the page or forms length in terms of a number of lines (that is, physical page size times the line density). Both the MPL and the line density can be determined by the SVF data stream. Otherwise the MPL (the number of lines) can be set up on the device by the operator.

If DEVICE(3270), DEVICE(3270P), DEVICE(LUTYPE2), or DEVICE(LUTYPE3) is specified, FORMFEED(YES) must be used in conjunction with the FORMFEED option in the BMS SEND commands. Use of form feed on display devices provides for a skip to a new page when the screen data is copied to a printer. The parameters discussed above for SCSPRINT operation do not apply when the devices are operating as 3270P or LUTYPE3 devices. In this case there is only the concept of a forms length, and this can only be set on the device by the operator. Refer to the *CICS/ESA Application Programming Reference* manual for programming information on the use of the FORMFEED option.

GROUP(groupname)

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see “How the CSD is organized — groups and lists” on page 15.

The GROUP name can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and \$. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

HIGHLIGHT({NO|YES})

Indicates whether the 3270 device or SCS printer has the extended highlight facility, which enables fields or characters to be displayed in reverse-video, underline mode, or blink (3270 only).

You can use the QUERY structured field to determine whether the device is set up to use the extended highlight facility. (See page 217.)

HORIZFORM({NO|YES})

If the devices for which you are defining this TYPETERM have the horizontal form feature, you can specify HORIZFORM(YES). The devices that can use this feature are batch, batch data interchange, interactive, SCSPT or LUTYPE4 logical units.

NO

The HTAB(tab,...) parameter in the BMS map definition will be ignored.

YES

BMS will use horizontal tabbing when formatting output documents.

IOAREALEN({0|value1},{0|value2})

Indicates the terminal input/output area, length in bytes, to be passed to a transaction.

If you specify ATI(YES), you must specify an IOAREALEN of at least one byte.

(value1, value2)

Value1 specifies the minimum size of a terminal input/output area to be passed to an application program when a RECEIVE command is issued.

If value2 is not specified, or is less than value1, it defaults to the value of value1.

You can specify value2 as greater than or equal to value1. In this case, when the size of an input message exceeds value1, CICS uses a terminal input/output area value2 bytes long. If the input message size also exceeds value2, the node abnormal condition program sends an exception response to the terminal.

The maximum value that may be specified for IOAREALEN is 32767 bytes.

KATAKANA({NO|YES})

Indicates whether Katakana support is required. Katakana terminals cannot display mixed case output; uppercase characters appear as uppercase English characters, but lowercase characters appear as Katakana characters. If you have any Katakana terminals connected to your CICS system, you should therefore specify MSGCASE=UPPER in the system initialization table. For further information about the MSGCASE parameter, see the *CICS/ESA System Definition Guide*.

NO

Katakana support is not required.

YES

Katakana support is required. All lowercase characters sent to the terminal from the following transactions will be translated to uppercase:

CBRC CDBC CDBI CEBR
CECI CEDA CEDF CEMT
CEOT CESN CEST CMSG
CRTE CSPG CWTO

+ LDCLIST(listname)

+ The name of a logical device code (LDC) list. The name
+ may be up to eight characters in length. The name
+ follows assembler language rules. It must start with an
+ alphabetic character. The acceptable characters are:
+ A-Z 0-9 \$ @ and #. Lowercase characters are
+ converted to uppercase. The LDCLIST and its contents
+ must be defined by macroinstruction(s).

+ A local LDC list is defined by:

```
+ listname DFHTCT TYPE=LDCLIST,  
+ LDC(aa=nnn,bb=nnn,...)
```

+ An extended local LDC list is defined by:

```
+ listname DFHTCT TYPE=LDC,LOCAL=INITIAL  
+ DFHTCT TYPE=LDC=(aa=nnn)....  
+ DFHTCT TYPE=LDC=(bb=nnn)....  
+ DFHTCT TYPE=LDC,LOCAL=FINAL
```

+ You specify this **listname** as the value for the LDCLIST
+ attribute on the TYPETERM definition.

+ This attribute applies only to 3600, 3770 batch, 3770
+ and 3790 batch data interchange, and LUTYPE4 logical

TYPETERM

+ units. The list specifies which LDCs are valid for this
 + logical unit and, optionally, which device characteristics
 + are valid for each LDC. The first LDC generated in this
 + list is the default when CICS must choose a default LDC
 + for a logical unit. For further guidance, see "Logical
 + device codes" on page 302.

LIGHTPEN({NO|YES})

Indicates whether a 3270 display has the selector pen feature.

LOGMODE(blank|0|name)

This determines how CICS builds the BIND to be sent to the logical unit.

blank

+ A defined terminal definition uses the BIND image
 + generated by the CICS definitions for this device by
 + means of this TYPETERM definition and its
 + associated TERMINAL definitions. An autoinstalled
 + terminal uses the fields specified in the incoming
 + CINIT.

name

+ This is the LOGMODE name from a VTAM logon
 + mode table that has been set up for use by this
 + logical unit. The name may be up to eight
 + characters in length and must follow assembler
 + language rules. The name must start with an
 + alphabetic character. The acceptable characters
 + are: A-Z 0-9 \$ @ and #. Lowercase characters are
 + converted to uppercase. This allows you to override
 + the BIND image provided by CICS for the logical
 + unit. For further information, see the appropriate
 + CICS subsystem guide. You cannot specify a
 + LOGMODE name when the terminal is a
 + cross-domain resource.

+ The TCTTE is updated to reflect the logmode bind
 + image fields. These include SEND and RECEIVE
 + sizes and default and alternate screen sizes. If the
 + logmode indicates that the terminal may not be
 + queried, the extended data stream fields are all set
 + to zero.

0 (zero)

+ This causes CICS to use some of the information
 + from the BIND image contained in the CINIT coming
 + from the logical unit. The BIND image in the CINIT
 + was created by VTAM based on the LOGMODE
 + entry defined for the logical unit requesting to log on
 + to CICS. The NIB is built with LOGMODE=0 and
 + BNDAREA=0. When the TYPETERM's SENDSIZE
 + and RECEIVESIZE have been specified as zero,
 + CICS replaces them with the values from the
 + LOGMODE's RUSIZES.

+ The TCTTE is updated to reflect the incoming CINIT
 + fields. These include SEND and RECEIVE sizes
 + and default and alternate screen sizes. If the
 + logmode indicates that the terminal may not be
 + queried, the extended data stream fields are all set
 + to 0.

Note: You should use LOGMODE(0) only in EXCEPTIONAL circumstances. Although the LU is bound with the VTAM definition, CICS keeps the main session characteristics from the CICS definition. For example, if a printer is defined to VTAM as LUTYPE1 but to CICS as an LUTYPE3 with LOGMODE(0), CICS accepts the bind but sends LUTYPE3 control characters to the printer, giving rise to incorrect results. This restriction does not apply to pipeline terminals.

APAR PQ15147

added the LOGMODECOM parameter

LOGMODECOM({NO|YES})

indicates LOGMODE compatibility. It shows whether
 # CICS is to make LOGMODE work the way it does in
 # releases earlier than CICS/ESA 4.1.

Background

In releases prior to CICS/ESA 4.1 the
 # TYPETERM LOGMODE(0|name) parameter
 # behaves differently for terminals that are not
 # XRF-capable.⁶ In the earlier releases,
 # LOGMODE(0|name) for a non-XRF-capable
 # terminal causes the TCTTE to be set up
 # from TYPETERM fields such as SENDSIZE,
 # RECEIVESIZE, BRACKETS etc. rather
 # than from the specified LOGMODE.
 # XRF-capable terminals behave differently -
 # their TCTTE fields are set up from the bind
 # specified in the LOGMODE.

In CICS/ESA 4.1 and later releases the
 # TCTTEs for both kinds of terminal are set
 # up from the bind specified in the
 # LOGMODE.

There are a few isolated cases where the
 # device does not obey the protocols and the
 # user is unable to change the TYPETERM or
 # VTAM LOGMODE to match each other in
 # such a way that the device accepts them.
 # For this reason the LOGMODECOM
 # parameter is introduced to allow CICS to
 # work as in earlier releases.

Use this parameter only in exceptional circumstances - it should only be used for devices that do not obey the protocols and where you are unable to change the TYPETERM or the LOGMODE to make the device work with CICS. IT IS NOT INTENDED TO MAKE THIS PARAMETER AVAILABLE IN RELEASES LATER THAN CICS/ESA 4.1,

NO

causes LOGMODE(0|name) to work as described in this book.

YES

causes LOGMODE(0|name) to work as it did in releases before CICS/ESA 4.1 for terminals that are

not XRF-capable.⁶ That is, the TCTTE is set up from the TYPETERM and not from the LOGMODE.

LOGMODECOM(YES) is ignored for XRF-capable terminals.

The effect of LOGMODECOM(YES) with LOGMODE(0|name) is as follows:

- # • For terminals that are not XRF-capable, the TCTTE is NOT updated to reflect the LOGMODE BIND image fields.
- # • For defined terminals this means that the TYPETERM definitions are used in the TCTTE.
- # • For autoinstalled terminals this means that initial incoming CINIT BIND fields override the equivalent fields in the TCTTE.
- # • Use of LOGMODECOM(YES) forces RECOVPTION(NONE).

Note: If LOGMODE(0), LOGMODECOM(YES), SENDSIZE(0) and RECEIVESIZE(0) are all used, the TYPETERM SEND/RECEIVE sizes are used for a defined terminal - whereas, in earlier releases the RUSIZES from the CINIT were used. If this causes a problem modify the TYPETERM to specify the correct SEND and RECEIVE sizes.

When LOGMODE(blank) is specified, or LOGMODE is not supplied, LOGMODECOM(YES) is not valid.

LOGONMSG({NO|YES})

Indicates whether the 'good morning' transaction, specified in the system initialization parameter GMTRAN, will be:

- # • automatically initiated when the logical unit is first logged on to CICS through VTAM.
- # • initiated after the terminal user's TIMEOUT period has expired under certain conditions.

Note that if you have specified ERRLASTLINE(YES), the message written by the transaction will not overwrite the error message line.

NO

The 'good morning' transaction will not be initiated.

? **Note:** If you are using a non-SNA terminal such as a Telnet 3270, LOGONMSG(NO) does not automatically release the keyboard lock. You need to press the Reset key to release the keyboard lock.

YES

The good morning transaction will be initiated as follows:

• The transaction runs when the OPNDST exit is successfully completed and a session is established. Note that the transaction is initiated by means of automatic task initiation (ATI) and competes with other ATI transactions for use of the terminal. You must specify ATI(YES) for this TYPETERM.

• After the terminal user's TIMEOUT period has expired (specified in the External Security Manager (ESM)) and either of the following has been specified in the TYPETERM definition :-

- # – SIGNOFF(YES) or
- # – SIGNOFF(LOGOFF) and DISCREQ(NO).

GMTRAN is set up to be the next transaction to be initiated by the terminal operator.

? **Note:** If you are using a non-SNA terminal such as a Telnet 3270, LOGONMSG(YES) also automatically releases the keyboard lock.

MSRCONTROL({NO|YES})

Indicates whether the terminal, an 8775 or 3643, has a magnetic slot reader. This option is not valid for SCS printers.

You can use the QUERY structured field to determine whether the device is set up to use a magnetic slot reader. (See page 217.)

NEPCLASS({0|value})

The node error program transaction class. This value acts as the default.

0 This will result in a link to the default node error program module.

value

The transaction-class for the (non-default) node error program module. The value can be in the range 1 through 255. For programming information about the node error program, see the *CICS/ESA Customization Guide*.

OBFORMAT({NO|YES})

OBFORMAT(YES) means that outboard formatting will be used. If the devices for which you are defining this TYPETERM use BMS outboard formatting, you can specify OBFORMAT(YES).

You can specify OBFORMAT(YES) for two DEVICE types only:

- 3650, SESSIONTYPE(3270)
- LUTYPE2, for an 8100 Information System using the DPPX operating system with DPPX/DPS Version 2 for presentation services.

For further guidance, see the:

#⁶ An XRF-capable terminal is one with BIT X'01' defined in offset 4 of the incoming CINIT indicating that the terminal is capable of supporting a backup session. It is an SNA terminal whose boundary network node is a 327x/3745 or equivalent.

TYPETERM

- *CICS/OS/VS IBM 3650/3680 Guide*
- *CICS/OS/VS IBM 3790/3730/8100 Guide*
- *DPPX/Distributed Presentation Services Version 2: System Programming Guide*

You can use the QUERY structured field to determine whether the device is set up to use outboard formatting. (See page 217.)

OBOPERID({NO|YES})

The outboard operator identifiers will be used by CICS in order to support the BMS routing facilities required for this terminal. This option only applies to the 3790 and 3770 batch data interchange logical units.

OUTLINE({NO|YES})

Indicates whether the device supports field outlining.

You can use the QUERY structured field to determine whether the device is set up to use field outlining. (See page 217.)

PAGESIZE({rows|columns})

Indicates the default page size for this printer. The default page size is used by BMS when the default screen size has been selected in the DEFSCREEN parameter. (See page 229.)

rows

Indicates the number of lines in the page. The PAGESIZE rows value can usefully be less than the DEFSCREEN rows value, perhaps to reserve the bottom line of a screen for error messages (see the ERLASTLINE parameter), if the same BMS map is being used for both printing and display.

columns

Indicates the number of characters in each line. Unexpected results will occur if the columns value specified in PAGESIZE differs from the columns value specified in DEFSCREEN.

lines × columns must not exceed 32767.

The default value will depend on the value you specify for the DEVICE keyword. See Table 13 on page 219 for details.

BMS uses the page size values when preparing output data streams. The specified number of characters in each line of the page should not exceed the physical line width of the terminal. In the case of printers that automatically perform a new-line function on reaching the end of the carriage (for example, 3270 printers) the line width specified here should be less than the physical line width. This will ensure that the formatting of the output data is governed entirely by the new-line (NL) characters supplied by BMS or by you, not by new-line functions performed by the device itself, which would produce additional lines of output, resulting in a physical page depth greater than that specified here.

For 3270-type printers, the hardware limits the amount of data that BMS may transmit. If the map or application

program request specifies L40, L64, or L80, or does not specify NLEOM on the SEND MAP command, the product of lines and columns specified in PAGESIZE must not be greater than the buffer size. If the BMS request specifies NLEOM, the page length may be any number of lines, but the product of lines and columns specified in the DEFSCREEN or the ALTSCREEN attributes must not exceed the buffer size of the device. In other words, the number of characters that BMS transmits must not exceed the physical buffer size of the printer.

Note: BMS divides a large page into smaller segments for transmission. PAGESIZE should therefore correspond to the required *logical* page size (linewidth × number of lines), and the DEFSCREEN value should correspond to the actual buffer size.

For a VTAM 3600, the PAGESIZE specified is used if a BMS page build operation is attempted without specifying a logical device code (LDC). A default device type of 3604 is assumed.

For 3770, LUTYPE4, or 3790 batch data interchange logical units, the PAGESIZE specified is used if a BMS page build operation is requested without specifying a logical device code (LDC). The default device type is the console printer. Take care when routing a message to a list of terminals. If the PAGESIZE you have defined (or allowed to default) is too small to accommodate the message, the transaction will abend.

+

APAR PN87384

+

added the next paragraph

+

For cumulative text processing, the maximum allowed buffer size is 32767. If this is exceeded, BMS will internally force a reduced page length to ensure that the PAGESIZE stays within the limit.

+

+

PARTITIONS({NO|YES})

Indicates whether a device is to use partitions. This option is not valid for SCS printers.

You can use the QUERY structured field to determine whether the device is set up to use partitions. (See page 217.)

PRINTADAPTER({NO|YES})

For the 3275: specifies the printer adapter feature and corresponding 3284 Printer Model 3 on the 3275 Display Station. This feature makes the 3284 eligible for print requests through the PA key from the host 3275.

For LUTYPE2 logical units: specifies that for print requests initiated by the PRINT key or by an ISSUE PRINT command, printer allocation will be handled by the 3790, or by the 3274 or 3276, according to the printer authorization matrix for both VTAM and non-VTAM attachments. Further, 3270 printers attached to the same 3790 are available for print requests sent to the 3270-display logical unit by a terminal control print request or initiated by the operator. If PRINTADAPTER

is NO, printer allocation is determined by the PRINTER and ALTPRINTER attributes of the TERMINAL definition.

If output is created on the screen by BMS requests with the PRINT option, by BMS requests with the NLEOM option, or by the CMSG command, the contents of the screen are automatically copied to a 3270 printer, whether or not the CICS-defined PRINT key (usually a PA key) was pressed.

PROGSYMBOLS({NO|YES})

Indicates whether the programmed symbol (PS) facility can be used on this 3270 device or SCS printer. The facility enables up to six 191-character sets, with customer-defined and program-loaded fonts and codes, to be stored and accessed.

You can use the QUERY structured field to determine whether the device is set up to use programmed symbols. (See page 217.)

QUERY({NO|COLD|ALL})

Indicates whether the QUERY structured field is to be used to determine the characteristics of the device. For more information about this function, and a list of the attributes which can be determined by using it, see page 217.

NO

The QUERY function is not to be used.

COLD

The QUERY function will be used to determine the characteristics of the device only when the device is first connected after a cold start of CICS. The device characteristics will be stored in the CICS global catalog for use on subsequent warm and emergency starts.

ALL

The QUERY function will be used to determine the characteristics of the device each time the device is connected.

RECEIVESIZE(256]value)

Defined terminal (non-autoinstalled) For a non-autoinstalled terminal, you should specify this with the maximum size of a request unit that can satisfy a VTAM RECEIVE request. The RECEIVESIZE value is transmitted to the connected logical unit, and must be in

the range 0 through 30720. It may be rounded down by CICS, because it must be transmitted in an architected form.

The effect of RECEIVESIZE depends on whether a RECEIVE RUSIZE is present in the VTAM LOGMODE table. Table 15 shows what RECEIVE RUSIZE is used to bind a session for each possible combination of TYPETERM and LOGMODE values.

Table 15. RECEIVE RUSIZE for defined (non-autoinstalled) terminals

RECEIVE RUSIZE (VTAM)	TYPETERM RECEIVESIZE	RUSIZE USED IN BIND
0	0	0
0	specified	TYPETERM RECEIVESIZE size
specified	0	0
specified	specified	TYPETERM RECEIVESIZE size

Note: The exception to this table is LOGMODE(0). If you specify this in your TYPETERM definition, VTAM values are used, irrespective of what else is specified.

APAR PQ15147

If LOGMODE(0), LOGMODECOM(YES) and RECEIVESIZE(0) are all used CICS uses a RECEIVESIZE of 0 where in earlier releases the CINIT RUSIZE was used. If this causes a problem the TYPETERM RECEIVESIZE should be made the same as the LOGMODE RUSIZE.

APPC terminal For an APPC (LUTYPE6.2) single session terminal, 256 would be a suitable value.

Autoinstalled terminal For an autoinstalled terminal, a non-zero value for RECEIVESIZE specifies either the maximum or actual RECEIVE RUSIZE value used in binding a session for a logical unit defined with this TYPETERM.

The effect of RECEIVESIZE depends on whether a RECEIVE RUSIZE is present in the VTAM LOGMODE table. Table 16 on page 236 shows what RECEIVE RUSIZE is used to bind a session for each possible combination of TYPETERM and LOGMODE values.

TYPETERM

Table 16. RECEIVE RUSIZE for autoinstalled terminals

RECEIVE RUSIZE (VTAM)	TYPETERM RECEIVESIZE	RUSIZE USED IN BIND
0	0 and BUILDCHAIN(YES)	256
0	0 and BUILDCHAIN(NO)	0
0	specified	TYPETERM RECEIVESIZE size
specified	0	VTAM RECEIVE RUSIZE size
specified less than or equal to TYPETERM RECEIVESIZE	specified	VTAM RECEIVE RUSIZE size
specified greater than TYPETERM RECEIVESIZE	specified	this combination is invalid and results in message DFHZC5963

RECOVNOTIFY({NONE|MESSAGE|TRANSACTION})

This option applies to the recovery of sessions for terminals in a CICS region running with either VTAM persistent sessions or with XRF. It is for use in situations where a terminal user may have to take action, such as sign on again, after a CICS restart. Use RECOVNOTIFY to specify how such a user should be notified.

VTAM persistent sessions: In a CICS region running with persistent session support, this specifies how a terminal end user is notified that their terminal session has been recovered.

XRF: In a CICS region running with XRF support, this specifies how the terminal user is notified that an XRF takeover has occurred.

This option is not applicable to APPC sessions.

NONE

There is no notification that a takeover has occurred.

MESSAGE

A message is displayed on the screen to say that the system has recovered. The message is specified in two BMS maps; DFHXRC1 and DFHXRC2 for XRF; and DFHXRC3 and DFHXRC4 for VTAM persistent sessions. These maps are in map set DFHXMSG. If reduced takeover time is important, use MESSAGE rather than TRANSACTION.

The terminal must be defined with the ATI(YES) option, and must be capable of displaying a BMS map.

TRANSACTION

A transaction is initiated at the terminal. The name of the transaction is specified by the RMTRAN system initialization parameter. (The default transaction for RMTRAN is the one specified in the GMTRAN system initialization parameter: the good-morning transaction.)

For the TRANSACTION operand, the terminal must be defined with the ATI(YES) option. If reduced takeover time is important, use MESSAGE rather than TRANSACTION.

RECOVOPTION({SYSDEFAULT|CLEARCONV|RELEASESESS|UNCONDREL|NONE})

This option applies to the recovery of sessions in a CICS region running with VTAM persistent sessions, or with XRF.

VTAM persistent sessions: In a CICS region running with persistent session support, this option specifies how you want CICS to recover the session, and return the terminal to service on system restart within the persistent session delay interval.

XRF: In a CICS region running with XRF support, this option specifies how you want CICS to recover the session, and return the terminal to service after an XRF takeover.

For all recovery options other than NONE, if the action taken is a VTAM UNBIND, the UNBIND is followed by a VTAM SIMLOGON.

SYSDEFAULT

VTAM persistent sessions: In a CICS region running with persistent sessions support, this specifies that CICS is to select the optimum procedure to recover a session on system restart within the persistent session delay interval, depending on the session activity and on the characteristics of the terminal.

Although sessions are recovered, any transactions in-flight at the time of the failure are abended and not recovered. Transactions are also abended if the recovered session is being used by another CICS region over an APPC connection.

CICS recovers the session with the least possible impact, in one of the following ways:

- If the terminal was not executing a transaction at the time of the CICS failure, no recovery action is required, and CICS takes the appropriate recovery notification action as defined by RECOVNOTIFY.
- If the terminal was busy (that is, executing a transaction) when CICS failed, CICS first tries to recover the session by sending a VTAM end-bracket indicator. If the end-bracket does not recover the session (for example, CICS may be in RECEIVE mode), CICS issues a CLEAR command. If the terminal does not

support the CLEAR command, the recovery action taken is a VTAM UNBIND followed by a SIMLOGON.

See the *CICS/ESA Recovery and Restart Guide* for more information about persistent sessions.

XRF: In a CICS region running with XRF support, this specifies that CICS is to select the optimum procedure to recover a busy session at takeover, depending on the session activity and on the characteristics of the terminal.

CLEARCONV

Prevents CICS from sending an end-bracket indicator to close an in-bracket session. Instead CICS sends a CLEAR request, to reset the conversation states. If the session does not support the CLEAR request, CICS sends an UNBIND request. The CLEAR or UNBIND is sent only if the session was busy at the time of system restart (in the case of persistent sessions) or the takeover (in the case of XRF).

RELEASESESS

Requires CICS to send an UNBIND request to release the active session. The UNBIND is sent only if the session was busy at the time of system restart (in the case of persistent sessions), or the takeover (in the case of XRF). Following the UNBIND, the session is queued for SIMLOGON. If the session is not busy, the requested recovery notification is carried out.

UNCONDREL

Requires CICS to send an UNBIND request to release the active session. The UNBIND is sent whether or not the session was busy at the time of system restart (in the case of persistent sessions support) or the takeover (in the case of XRF). Following the UNBIND, the session is queued for SIMLOGON.

NONE

VTAM persistent sessions: In a CICS region running with persistent sessions support, this specifies that the terminal session is not to be recovered at system restart within the persistent session delay interval: in effect, the terminal has no persistent sessions support. LU6.2 sessions are unbound but the latest negotiated CNOS value is returned to the CICS system after the restart. After system restart, the terminal is reconnected automatically if you specify AUTOCONNECT(YES), subject to the operation of the AIRDELAY system initialization parameter (AIRDELAY=0 overrides AUTOCONNECT(YES), and the terminal is not reconnected).

APAR PN88279
added the next paragraph.

Specify RECOVOPTION(NONE) if this terminal or autoinstall model is to be used with persistent sessions but the terminal may be the subject of an EXEC CICS ISSUE PASS LUNAME() LOGONLOGMODE.

XRF: In a CICS region running with XRF support, this specifies that the logon state is not tracked by the alternate system, and the terminal session is not automatically recovered after a takeover; in effect, the terminal has no XRF support. After takeover, the terminal is reconnected automatically by the alternate system, if you specify AUTOCONNECT(YES).

APAR PQ15147

LOGMODECOM(YES): forces a value of RECOVOPTION(NONE).

RELREQ=({NO|YES})

Specify this to indicate whether CICS is to release the logical unit.

NO

CICS is not to release the logical unit upon request by another VTAM application program.

YES

CICS is to release the logical unit, if the logical unit is not currently part of a transaction.

ROUTEDMSGS({ALL|NONE|SPECIFIC})

This specifies which messages are to be routed to this terminal by an EXEC CICS ROUTE command. The default will depend on the value you specify for the DEVICE keyword. See Table 13 on page 219 for details.

ALL

BMS will route to this terminal messages that are destined for *all* terminals as well as those specifically destined for *this* terminal.

NONE

BMS will not route any messages to this terminal, whether they are destined for all terminals or for this terminal specifically.

SPECIFIC

BMS will route messages to this terminal when they are destined specifically for this terminal, but not when they are destined for *all* terminals.

SENDSIZE(0|value)

Defined terminal (non-autoinstalled) For a non-autoinstalled terminal, you should specify this with the maximum size of a request unit that can satisfy a VTAM RECEIVE request. The SENDSIZE value is transmitted to the connected logical unit, and must be in the range 0 through 30720. It may be rounded down by CICS, because it must be transmitted in an architected form.

TYPETERM

The effect of SENDSIZE depends on whether a RECEIVE RUSIZE is present in the VTAM LOGMODE table. Table 17 on page 238 shows what RECEIVE RUSIZE is used to bind a session for each possible combination of TYPETERM and LOGMODE values.

Table 17. SEND RUSIZE for defined (non-autoinstalled) terminals

SEND RUSIZE (VTAM)	TYPETERM SENDSIZE	RUSIZE USED IN BIND
0	0	0
0	specified	TYPETERM SENDSIZE size
specified	0	This combination is invalid and results in a bind failure with message DFHZC2403
specified	specified	TYPETERM SENDSIZE size

Note: The exception to this table is LOGMODE(0). If you specify this in your TYPETERM definition, VTAM values are used, irrespective of what else is specified.

APAR PQ15147

If LOGMODE(0), LOGMODECOM(YES) and SENDSIZE(0) are all used CICS uses a SENDSIZE of 0 where in earlier releases the CINIT RUSIZE was used. If this causes a problem the TYPETERM SENDSIZE should be made the same as the LOGMODE RUSIZE.

APPC terminal For an APPC (LUTYPE6.2) single session terminal, 256 would be a suitable value.

Autoinstalled terminal For an autoinstalled terminal, a non-zero value for SENDSIZE specifies either the maximum or actual SEND RUSIZE value used in binding a session for a logical unit defined with this TYPETERM.

The effect of SENDSIZE depends on whether a SEND RUSIZE is present in the VTAM LOGMODE table. Table 18 shows what SEND RUSIZE is used to bind a session for each possible combination of TYPETERM and LOGMODE values.

Table 18. SEND RUSIZE for autoinstalled terminals

SEND RUSIZE (VTAM)	TYPETERM SENDSIZE	RUSIZE USED IN BIND
0	0	0
0	specified	TYPETERM SENDSIZE size
specified	0	VTAM SEND RUSIZE size
specified less than or equal to TYPETERM SENDSIZE	specified	VTAM SEND RUSIZE size
specified greater than TYPETERM SENDSIZE	specified	this combination is invalid and results in message DFHZC5963

SESSIONTYPE(type)

Indicates the type of session that can be used for a VTAM SNA logical unit. For details, see Table 13 on page 219.

SHIPPABLE({NO|YES})

Indicates whether the definition is allowed to be sent to a remote system if this device tries to initiate a remote transaction. This function may be used for any terminal, whether autoinstalled, or with its own TERMINAL definition. The shipping will not work unless the terminal has a definition installed, by one of these methods, in the local system.

Using SHIPPABLE(YES) means that you do not need to ensure that a definition of the terminal exists on the remote system for a locally defined terminal to initiate a transaction in that system. This can be useful when the remote system cannot share the CSD file with the local system.

A definition for the terminal must already be installed in (or already shipped to) the remote system.

For guidance on deciding whether to use SHIPPABLE(YES), see "Terminals for transaction routing" on page 188.

NO

This definition cannot be shipped to a remote system.

YES

This definition can be shipped to a remote system.

SIGNOFF({YES|NO|LOGOFF})

+ Specify this to control the actions taken when GNTRAN (CESF or user-defined transaction) is attached and attempts to sign off the terminal. If you are using RACF 1.9 or later, you must specify the TIMEOUT limit in the RACF segment.

YES

When the specified time has elapsed since the last
input from the operator, the terminal is automatically
signed off from CICS, and the next transaction to be
initiated by the operator will be the good morning
(GMTRAN) transaction.

NO

The terminal will not be timed out.

LOGOFF

When the specified time has elapsed since the last
input from the operator, the terminal is automatically
signed off from CICS. If DISCREQ(YES) has been
specified it is then logged off from VTAM, otherwise
it performs like SIGNOFF(YES).

LOGOFF is useful for an autoinstall model, because
it has the effect that virtual storage is not wasted on
entries for terminals that have been timed out.

+ If GNTRAN fails to attach because of unprocessed
+ data in the terminal buffer (resulting in a BID
+ failure), the terminal will be signed off and logged
+ off. GNTRAN will not run and will have no effect.

APAR PN92166

added this note

+ **Note:** You cannot change the value of this attribute
+ when DEVICE(APPC) is specified. The default value
+ in that case is SIGNOFF(NO).

SOSI({NO|YES})

Indicates whether the device supports mixed EBCDIC
and double-byte character set (DBCS) fields.

You can use the QUERY structured field to determine
whether the device is set up to use mixed EBCDIC and
DBCS fields. (See page 217.)

TERMMODEL({1|2})

Specify this with the model number of the terminal. If
the device is a component of the 3270 Information
Display System, this operand must be specified as
follows:

- 1 Specify 1 for the 3270 Model 1 displays and printers
(for example, 3277 Model 1) with a default screen or
buffer size of 12x40 (480 bytes/characters).
TERMMODEL(1) is the default for 3270 Model 1
printers and displays.

Specify 1 for the 3275 Display Station Model 11.
The CICS support obtained is identical to that
obtained by coding TERMMODEL(1) for 3275
Display Station Model 1.

- 2 Specify 2 for the 3270 displays and printers (for
example, 3278 Model 4) with a default screen or
buffer size of 24x80 (1920 bytes/characters).
TERMMODEL(2) is the default for the 3286 printer
in 3270 compatibility mode.

Specify 2 for the 3275 Display Station Model 12.
The CICS support obtained is identical to that
obtained by coding TERMMODEL(2) for 3275
Display Station Model 2.

TEXTKYBD({NO|YES})

Indicates whether the 3270 device has the text-keyboard
feature.

TEXTPRINT({NO|YES})

Indicates whether the 3288 printer has the text-print
feature.

TTI({YES|NO})

Indicates whether transactions can be initiated at the
terminal by a user.

YES

Transactions can be initiated at the terminal by a
user. If you also specify ATI(YES), transactions can
also be initiated automatically. In this case, the
automatic transaction initiation, either by transient
data control or interval control, sets a condition in an
appropriate terminal control table terminal entry. If
both ATI and TTI are specified as YES, and if there
is no transaction at the terminal, terminal control
initiates the user-defined task. This task is expected
to send messages to the terminal.

For a terminal used in the processing of
transactions such as inquiries or order entries, you
would specify TTI(YES) and ATI(NO). This also
applies to a display station or hard-copy terminal to
which no messages are sent without a terminal
request and through which transactions are entered.
Note that this is the only specification allowed for
3790 inquiry logical units.

NO

Transactions cannot be initiated at the terminal by a
user. If you specify NO, you should specify
ATI(YES) to allow transactions to be initiated
automatically. An example of this type of terminal is
one that is located in a remote location, such as a
warehouse, and is unattended but may receive
messages.

TYPETERM(name)

The name of this extension of terminal definitions. The
name can be up to 8 characters in length. The
acceptable characters are: A-Z 0-9 \$ @ and #.
Lowercase characters are converted to uppercase.

This name is referred to in all the TERMINAL definitions
using this TYPETERM. Note that this TYPETERM
definition must be installed before, or at the same time
as the TERMINAL definitions that reference it.

UCTRAN({NO|YES|TRANID})

Indicates whether the input data stream from a terminal
is to be translated to uppercase. The input data stream
may include a transaction identifier as well as program
data. CICS supports transaction identifier definition in

TYPETERM

mixed case, and the UCTRAN parameter can be used to ensure that the correct transaction is located.

NO

No uppercase translation is performed.

YES

All the data input from the terminal, both the transaction identifier if present and the program data, is translated to uppercase before any processing.

TRANID

When the input data stream includes a transaction identifier CICS will translate it to uppercase before attempting to locate its definition. However all the input data, both the transaction identifier and the program data, will be passed to the program without any translation.

Therefore both the YES and the TRANID parameters allow transaction identifiers to be defined in uppercase and to be entered from the terminal in either uppercase or lowercase, but the TRANID parameter causes the transaction identifier and program data to be passed to the program without any translation.

You can also request translation to uppercase at the transaction level on PROFILE definitions (see “Defining a PROFILE” on page 164), but you should be aware that a TYPETERM UCTRAN(YES) definition overrides a PROFILE UCTRAN(NO) definition. So, if you specify TYPETERM UCTRAN(YES), a PROFILE UCTRAN(NO) has no effect. Translation can be overridden by the application program for all RECEIVE requests except the first, by using the ASIS option.

Table 19 shows which portion of the terminal input is translated (transaction id and/or data) according to the setting of the UCTRAN on the PROFILE and TYPETERM resource definitions.

Profile (PROFILE)	Terminal (TYPETERM)		
	UCTRAN (YES)	UCTRAN (NO)	UCTRAN (TRANID)
UCTRAN (YES)	Tranid: Yes Data: yes	Tranid: No Data: Yes	Tranid: Yes Data: Yes
UCTRAN (NO)	Tranid: Yes Data: Yes	Tranid: No Data: No	Tranid: Yes Data: No

USERAREALEN({0|number})

Specify this with the length, in bytes (0 to 255), of the user area for this terminal. It should be made as small as possible. The TCT user area is initialized to zeros at system initialization.

The TCT user area may be located above or below the 16Mb line in virtual storage. Where it is located depends on the value of the TCTUALLOC operand of the DFHSIT macro. You should ensure that this is specified correctly to allow successful operation of any programs you may have that are not capable of handling 31-bit addressing.

VALIDATION({NO|YES})

Indicates whether the 8775 device has the extended validation feature, which allows fields to be defined as TRIGGER, MANDATORY FILL, or MANDATORY ENTER. Or the 3290 device has the validation feature, which allows fields to be defined as MANDATORY FILL or MANDATORY ENTER. This option is not valid for SCS printers. If VALIDATION(YES) is specified for an SCS printer an error message will be raised, but the option will not be generated.

You can use the QUERY structured field to determine whether the device is set up to use the validation feature. (See page 217.)

VERTICALFORM({NO|YES})

If the devices for which you are defining this TYPETERM have the vertical form feature, you can specify VERTICALFORM(YES). The devices that can use this feature are batch, batch data interchange, interactive, SCSPRT or LUTYPE4 logical units.

NO The VTAB(tab,...) parameter in the BMS map definition will be ignored.

YES BMS will use vertical tabbing when formatting output documents.

XRFSIGNOFF({NOFORCE|FORCE})

Sets the sign-on characteristics of a group of terminals.

If you have a collection of terminals in a security-sensitive area, for example, you might choose to force sign-off of those terminals after a takeover, to prevent the use of the terminal in the absence of the authorized user. (This could happen if the authorized user left the terminal during the takeover, and the terminal became active again while it was unattended.)

This option works in conjunction with the system initialization parameter XRFSOFF and the XRFSOFF entry in the CICS RACF segment (if you are running RACF 1.9). The relationship between them is explained on page 196.

TYPETERM

Part 5. Macro reference information

This part gives reference information for the macros needed for each table. The information is divided into a number of chapters, one for each table, in alphabetic order as listed below.

- Chapter 25, "Introduction to resource definition with macros" on page 245
- Chapter 26, "CLT — command list table" on page 253
- Chapter 27, "DCT — destination control table" on page 257
- Chapter 28, "DL/I directories — DDIR and PDIR" on page 267
- Chapter 29, "FCT — file control table" on page 269
- Chapter 30, "JCT — journal control table" on page 279
- Chapter 31, "MCT — monitoring control table" on page 285
- Chapter 32, "PLT — program list table" on page 291
- Chapter 33, "RST — recoverable service table" on page 295
- Chapter 34, "SRT — system recovery table" on page 297
- Chapter 35, "TCT — terminal control table" on page 299
- Chapter 36, "TLT — terminal list table" on page 329
- Chapter 37, "TST — temporary storage table" on page 331
- Chapter 38, "XLT — transaction list table" on page 337.

Each chapter describes the function of the table and outlines the macros used to create it. The operands for each macro are given in a syntax box, which is followed by an explanation of each operand, in alphabetic order.

"Chapter 35, TCT — terminal control table" is further subdivided according to access method and type of device.

Macro reference information

Chapter 25. Introduction to resource definition with macros

This part tells you which CICS system table defines each resource, and how to prepare the tables. You use macros to define:

- Non-VTAM networks
- Non-VTAM terminals
- Non-VSAM files
- Databases
- Journals
- Queues
- Monitoring resources
- System recovery resources

CICS uses an external security manager for all its security management.

BTAM is not supported in this version of CICS. To gain access to CICS/ESA 4.1 from BTAM terminals, define the BTAM terminals in a version of CICS earlier than 3.2.1, and use transaction routing to gain access to the current release. BTAM terminals must be defined as remote resources in this release, and as local resources in the earlier release. This book contains information about the definition of remote BTAM terminals: information about the definition of local BTAM terminals can be found in the documentation for your earlier release of CICS.

You must use **resource definition online (RDO)** for VSAM files, and to define programs, map sets, partition sets, transactions, and profiles. You must also use RDO to define VTAM terminals, and links and sessions with MRO (multiregion operation) and ISC (intersystem communication) systems. RDO is described in Chapter 1, "Resource definition—an introduction" on page 3.

Introduction to CICS control tables and macros

CICS is configured under your control during system initialization. You select a system initialization table (SIT) and, through it, CICS selects other control tables. Each control table is created separately and may be recreated at any time before system initialization. You have to prepare the required control tables by coding the appropriate macros. For each table, the macros automatically generate the necessary linkage editor control statements.

This chapter gives you an outline description of the control tables.

You may need to read the *CICS/ESA System Definition Guide* for further information about the following areas related to control tables:

- A **system initialization table (SIT)** is required for the system to be operational. Other tables are needed only if you are using the corresponding CICS facilities. For details of the SIT, you should read the *CICS/ESA System Definition Guide*.
- To determine the **job control language (JCL)** needed for the control tables, and to find out how to link-edit and assemble the macro statements that you have to code.
- For further information about whether CICS loads a table above or below the 16MB boundary. The CICS table macros contain linkage editor control statements that determines this; Table 20 on page 246 shows whether specific tables are loaded above or below the boundary.

The control tables that can be defined by macros are also shown in Table 20 on page 246.

Introduction to resource definition with macros

Table 20. Control tables definable by macros. The last column shows whether the table is loaded above or below the 16MB line.

Control Table	What the Table Defines	Above the line?
Command list table	Sets of commands and messages for an XRF takeover	Yes
Destination control table	Extrapartition, intrapartition, and indirect destinations	Yes
DL/I directories	Databases and program specification blocks	No
File control table	Files and VSAM LSRPOOLS	Yes
Journal control table	The system log and user journals	No
Monitoring control table	Monitoring activity	Yes
Program list table	Sets of related programs	Yes
Recoverable service table	Sets of recoverable service elements	Yes
System recovery table	Abend codes for which recovery will be attempted	Yes
Terminal control table	Non-VTAM terminal networks	No
Terminal list table	Sets of related terminals	No
Temporary storage table	Special processing for temporary storage	Yes
Transaction list table	Sets of related transactions	Yes
Resource control table	A CICS - DB2 connection. For details of the RCT, see the <i>DB2 Version 2 Administration Guide</i> .	
System initialization table	System startup parameters. For details of the SIT, see the <i>CICS/ESA System Definition Guide</i> .	

A brief description of each of these CICS control tables follows.

Command list table

The command list table (CLT) is used for XRF (extended recovery facility). If you are using XRF, you must have a CLT; it is used only by the alternate CICS system. The CLT contains a list of commands that are passed to JES or MVS for execution. It also provides the authorization for canceling the active CICS system. See Chapter 26, "CLT — command list table" on page 253 for details of the CLT.

Destination control table

The destination control table (DCT) contains an entry for each extrapartition, intrapartition, and indirect destination. Extrapartition entries address data sets external to the CICS region. Indirect destination entries redirect data to a destination controlled by another DCT entry. Intrapartition destination entries contain the information required to locate the queue in the intrapartition data set. See Chapter 27, "DCT — destination control table" on page 257 for details of the DCT.

File control table

The file control table (FCT) describes files and data tables that are processed by file management. The files defined in the FCT can be VSAM or BDAM. VSAM files can be defined with the resource definition online (RDO) facility described in Chapter 13, "FILE" on page 141. The FCT for BDAM files is assembled using the DFHFCT macros as described in Chapter 29, "FCT — file control table" on page 269.

Journal control table

The journal control table (JCT) describes the system log and user journals and their characteristics for access through journal management. See Chapter 30, "JCT — journal control table" on page 279 for details of the JCT.

Monitoring control table

The monitoring control table (MCT) describes the monitoring actions (data collection) to be taken at each user event monitoring point (EMP). Different actions can be specified for each monitoring class at each EMP. See Chapter 31, “MCT — monitoring control table” on page 285 for details of the MCT.

Program list table

The program list table (PLT) contains a list of related programs. You may want to generate several PLTs to specify a list of programs that are to be executed in the initialization programs phase of CICS startup, executed during the first or second quiesce stages of controlled shutdown, or both, or enabled or disabled as a group by a master terminal ENABLE or DISABLE command. See Chapter 32, “PLT — program list table” on page 291 for details of the PLT.

Recoverable service table

The recoverable service table (RST) is used for CICS IMS/ESA DBCTL (database control) support. If you are using XRF and DBCTL, you must have an RST: it is used by the active CICS system. The RST contains a list of recoverable service elements which define the DBCTL configuration. It defines which DBCTL CICS will connect to. See Chapter 33, “RST — recoverable service table” on page 295 for details of the RST.

Terminal control table

The terminal control table (TCT) is retained to define non-VTAM terminal networks. See Chapter 35, “TCT — terminal control table” on page 299 for details of the TCT.

Terminal list table

The terminal list table (TLT) allows terminal or operator identifications, or both, to be grouped logically. A TLT is required by the supervisory terminal operation (CEST), to define and limit the effective range of the operation. It can also be used by a supervisory or master terminal operation (CEMT) to apply a function to a predetermined group of terminals. A TLT can be used, singly or in combination with other TLTs, to provide predefined destinations for message switching. See Chapter 36, “TLT — terminal list table” on page 329 for details of the TLT.

Temporary storage table

Application programs can store data in temporary storage for later retrieval. For the data to be recoverable by CICS if the system terminates abnormally, data identifiers have to be specified in the temporary storage table (TST). A generic data identifier can be coded so that any unique temporary storage identifier (generated dynamically in a program) that begins with the same characters as the generic identifier (in the TST) can automatically acquire the same properties as the TST entries. Resource security level checking can be done on the temporary storage queues. See Chapter 37, “TST — temporary storage table” on page 331 for details of the TST.

Transaction list table

The transaction list table (XLT) is a list of logically-related transaction identifications. The XLT defines a list of transaction identifications that can be initiated from terminals during the first quiesce stage of system termination, or a group of transaction identifications that can be disabled or enabled through the master terminal. See Chapter 38, “XLT — transaction list table” on page 337 for details of the XLT.

A note on the system initialization table, the resource control table, and data conversion tables

The system initialization table (SIT) contains parameters used by the system initialization process. In particular, the SIT identifies (by suffix characters) the versions of CICS system control programs and CICS tables that you have specified are to be loaded. This book does not tell you how to code the SIT. For detailed information about the SIT, you should read the *CICS/ESA System Definition Guide*.

The resource control table (RCT) is required only by IBM DATABASE 2 users. It describes the relationship between CICS transactions and DB2 resources. The RCT is not covered in this book. For details of the RCT, you should refer to the *DB2 Version 2 Administration Guide*.

If the CICS/ESA system is using ISC to communicate with a member of the CICS family that runs on a hardware platform that does not use EBCDIC (such as CICS OS/2 or CICS/6000 which use ASCII), then a data conversion table may be needed. The conversion table defines how data is to be changed from ASCII format at the workstation to EBCDIC format at the CICS/ESA host. The DFHCNV macros which you use to create the table are described in the *CICS Family: Communicating from CICS on System/390* manual.

DL/I directories

If you use CICS-IMS/ESA DBCTL (database control) exclusively to manage your CICS system's use of DL/I, you need not define the DL/I directories (DDIR and PDIR) using CICS.

However, if you use DL/I in the CICS address space (local DL/I), you need two directories:

- The DL/I database directory (DDIR)
- The DL/I program specification block directory (PDIR)

The DDIR is a directory of all physical databases to be used by the CICS system.

The PDIR is a directory of all the program specification blocks (PSBs) that are accessed by the CICS system.

If you function-ship requests to a remote database manager (remote DL/I), you need only one directory, the PDIR.

The macros for creating these directories are described in Chapter 28, "DL/I directories — DDIR and PDIR" on page 267.

TYPE=INITIAL

Most of the tables must start with a TYPE=INITIAL macro. For some tables you can provide information that applies to the whole table, on the TYPE=INITIAL macro.

The TYPE=INITIAL macro establishes the control section (CSECT) for the CICS system table, and produces the necessary linkage editor control statements. CICS automatically generates the address of the entry point of each table through the DFHVM macro that is generated from each TYPE=INITIAL macro. The entry point label of the table is DFHxxxBA. Only the END statement need be specified.

Naming and suffixing the tables

The tables are named as follows:

Table	Name
Command list table	DFHCLTxx
Destination control table	DFHDCTxx
File control table	DFHFCTxx
Journal control table	DFHJCTxx
Monitoring control table	DFHMCTxx
Program list table	DFHPLTxx
Recoverable service table	DFHRSTxx
System recovery table	DFHSRTxx
Terminal control table	DFHTCTxx
Terminal list table	DFHTLTxx
Temporary storage table	DFHTSTxx
Transaction list table	DFHXLTxx

The first six characters of the name of each table are fixed. You can specify the last two characters of the name, using the SUFFIX operand. The SUFFIX operand is specified on the TYPE=INITIAL macro for each table.

Suffixes allow you to have more than one version of a table. A suffix may consist of one or two characters. Valid characters are @, and the ranges A through Z and 0 through 9. (Note, however, that you should not use **NO** or **DY**.) You select the version of the table to be loaded into the system during system initialization, by specifying the suffix in the appropriate operand in the SIT.

For example:

```
DFHSIT...,FCT=MY,...
```

Note: The TYPE=INITIAL macros have a STARTER operand that is not listed in the descriptions of the individual macros in the main body of this book. Coding STARTER=YES enables you to use the \$ character in your table suffixes. The default is STARTER=NO. This operand should be used only with starter system modules.

TYPE=FINAL

Most of the tables, again with the single exception of the SIT, must end with a TYPE=FINAL macro. The TYPE=FINAL macro creates a dummy entry to signal the end of the table. It must be the last statement before the assembler END statement. The format is always like this:

	DFHxxT	TYPE=FINAL
--	--------	------------

Format of macros

The CICS macros are written in assembler language and, like all assembler language instructions, are written in the following format:

Name	Operation	Operand	Comments
blank or symbol	DFHxxxxx	One or more operands separated by commas	

The operand field is used to specify the services and options to be generated. Operands are always in a keyword format and any parameters are specified according to the following general rules:

- If the parameter associated with the operand is shown entirely in capital letters (for example, TYPE=INITIAL), the operand and parameter must be specified exactly as shown.
- If the parameter associated with the operand is shown in lowercase letters (for example, FILE=name), the operand must be specified exactly as shown and a value, address, or name must be substituted for the lowercase letters.
- Commas and parentheses must be specified exactly as shown, except that a comma following the last operand specified must be omitted.
- The parentheses may be omitted when only one parameter of a particular operand is used.
- Because a blank character indicates the end of the operand field, the operand field must not contain blanks except within quotes, after a comma on a continued line, or after the last operand of the macro. The first operand on a continuation line must begin in column 16.
- When a CICS macro is written on more than one line, each line containing part of the macro (except the last line) must contain a character (for example, an asterisk) in column 72, indicating that the macro is continued on the next line.

Syntax notation

The symbols [], { }, | and ,... are used in this book to show the operands for the macros as clearly as possible. **Do not use these symbols in your specifications.** They act only to indicate how a command or macro can be written; their definitions are given below:

- [] indicates optional operands. You may or may not need to specify the operand enclosed in the brackets (for example, [FB]), depending on whether the associated option is desired. If more than one item is enclosed within brackets (for example, [BLOCKED|UNBLOCKED]), you can specify either one or none. Any default value available is indicated like this: default. The default will be taken if you do not specify an option from the group.
- { } indicates that a choice must be made. One of the operands from the list within braces separated by a | symbol (for example, {YES|NO}) must be specified, depending on which of the associated services is desired. Any default value is indicated like this: default.
- | indicates that a choice must be made between the operands that are separated by this symbol.
- ,... indicates that more than one set of operands can be designated.

To simplify the syntax notation in the case where one or more operands may be specified, the notation:

```
PARAM=( [A] [ ,B] [ ,C] [ ,D] )
```

indicates that any number or none of A, B, C, or D may be specified. Do not code a leading comma. If you specify only one operand, you need not code the enclosing parentheses.

For example:

```
PARAM=A
PARAM=(A,B)
PARAM=(B,D)
PARAM=(C)
```

are all valid interpretations of the above notation.

The default value of an operand is shown like this: default. For example:

```
PARAM={A|B|C}
```

means that you can code A, B, or C as the value for this operand. If you do not code the operand at all, the value A is assumed.

Chapter 26. CLT — command list table

The command list table (CLT) is used by the **extended recovery facility** (XRF). The CLT contains a list of MVS system commands and messages to the operator, to be issued during takeover. Typically, the function of these commands is to tell alternate systems to take over from their actives in the same MRO-connected configuration. It also contains the name of the alternate system, with the jobname of the active system that it is allowed to cancel. (See DFHCLT TYPE=LISTSTART FORALT operand.) This provides a security check against the wrong job being canceled, when the alternate system takes over. In addition, the DFHCLT TYPE=INITIAL macro gives JES routing information, needed to send cancel commands to the appropriate MVS system. If you are using XRF, you **must** have a CLT: it is used only by the alternate CICS system.

For security reasons, you should link-edit the CLT into a library authorized using APF. For virtual storage constraint relief considerations, you should link-edit using a MODE control statement specifying AMODE(31),RMODE(ANY). The table should be link-edited as reentrant. The CLT is not loaded into the CICS nucleus.

For guidance on the use of CLTs in various configurations, see the *CICS/ESA 3.3 XRF Guide*.

Your CLT can contain the following statements:

- DFHCLT TYPE=INITIAL
- DFHCLT TYPE=LISTSTART
- DFHCLT TYPE=COMMAND
- DFHCLT TYPE=WTO
- DFHCLT TYPE=LISTEND
- DFHCLT TYPE=FINAL (see page 249).

Note: Although the CLT may be shared by a number of alternate systems, take care that MVS is not given too many redundant commands during takeover. For example, in multiregion operation, using one CLT with commands for several regions, region 1 would send valid commands to other regions, but they would in turn send redundant commands to region 1 and to each other.

Control section — DFHCLT TYPE=INITIAL

The DFHCLT TYPE=INITIAL macro establishes the entry point and the beginning address of the CLT being defined.

Table 22. DFHCLT TYPE=INITIAL		
label	DFHCLT	TYPE=INITIAL [.,JES={JES2 JES3}] [.,JESCHAR=value] [.,JESID= (mvsname1,jesname1,spoolno1) [.,(mvsname2,jesname2,spoolno2) ...])] [.,SUFFIX=xxx]

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

JES={JES2|JES3}

Specifies the version of JES being used. If you have active and alternate CICS in different CECs, you must use the same version of JES on both CECs.

If you are using JES3, then you need release 2.2.0 for full support of all CLT functions in a two-CEC environment. JES2 accepts commands issued by programs. For JES3, this feature was only introduced for release 2.2.0. With an earlier JES3 release, it is possible that a takeover in a two-CEC environment can only proceed after the operator has manually canceled the failing active CICS. This should occur only when the active does not realize that it is failing and continues to run.

JESCHAR=value

Specifies the one-character prefix to be used for commands to be passed to JES. If you omit this keyword:

- JESCHAR=\$ is the default for JES=JES2.
- JESCHAR=* is the default for JES=JES3.

JESID=((mvsname1,jesname1,spoolno1) [.,(mvsname2,jesname2,spoolno2),...])

Specifies the JES routing code that corresponds to the MVS name and JES name of an active CICS system. You must use this option if active and alternate CICS are in different CECs.

mvsname1

This is the SMFID, as specified in SYS1.PARMLIB member SMFPRMxx, for the CEC on which the active CICS system executes.

jesname1

This is the JES2 or JES3 subsystem name for the JES under whose control the active system executes. It is defined in the MVS/ESA SCHEDULR sysgen macro and also in SYS1.PARMLIB member IEFSSNxx.

spoolno1

For JES2 this is the multiaccess spool member number of the JES2 for the active CICS system. It is defined in the JES2 initialization parameter MASDEF SSID(n). For JES3 this is the processor name of the JES3 for the active CICS system. It is defined in the JES3 initialization parameter MAINPROC NAME=name. See the *MVS/ESA JES2 Initialization and Tuning* or *MVS/ESA JES3 Initialization and Tuning* manual.

You may extend this, using more groups of mvname, jesname, and spoolno, so that the CLT can be used to refer to more CEC/JES combinations.

Specifying alternate systems — DFHCLT TYPE=LISTSTART

This macro defines the start of the set of commands and messages that the alternate CICS issues when it takes over from the active CICS. (There may be no commands or messages, but you still need a CLT, so that authorization checks can be made.)

label	DFHCLT	TYPE=LISTSTART ,FORALT=((applid1,jobname1) [, (applid2,jobname2),...])

FORALT=((applid1,jobname1)[,(applid2,jobname2),...])

Specifies pairs of alternate and active CICS systems.

applid1

The name of the alternate CICS that will issue the set of commands and messages when it takes over. This name must be the **specific APPLID**, defined in the SIT APPLID operand. It is used as an authorization check.

jobname1

The name of the active CICS from which the alternate is taking over. This name must be the **MVS JOBNAME** for the active CICS. It is used as a security check, to ensure that the alternate system does not attempt to cancel any job other than one of that name.

You may extend this, using more pairs of applid and jobname, so that you can use one CLT for several alternate CICS systems.

Specifying takeover commands — DFHCLT TYPE=COMMAND

This macro allows you to specify the commands to be used by the alternate CICS system during takeover.

label	DFHCLT	TYPE=COMMAND ,COMMAND=command-string

COMMAND=command-string

Defines a command that is passed to MVS for execution. CICS does not interpret this command.

The command that is issued in this way most frequently is CEBT PERFORM TAKEOVER.

In multiregion operation (MRO), where there is a simple hierarchy of **master** and **dependent** regions, a failing master region can issue this command to each of its dependent regions, if it is necessary that they also move to another CEC.

In a more complex multiregion operation, a failing master region can issue this to its **coordinator** region, and the coordinator can issue the same command to other masters and dependents in the same hierarchy of regions. Hence, many MRO-connected regions can move together to another CEC, without operator intervention.

Here are some examples:

- A master region without a coordinator sends a command to a dependent region:

```
DFHCLT TYPE=COMMAND,
      COMMAND='MODIFY CICSDEP,CEBT PERFORM
      TAKEOVER'
```

- A master region sends a command to its coordinator region:

```
DFHCLT TYPE=COMMAND,
      COMMAND='MODIFY CICS CRD,CEBT PERFORM
      TAKEOVER'
```

- A coordinator region sends commands to master and dependent regions:

```
DFHCLT TYPE=COMMAND,
      COMMAND='MODIFY CICS MAS,CEBT PERFORM
      TAKEOVER'
```

```
DFHCLT TYPE=COMMAND,
      COMMAND='MODIFY CICS DEP,CEBT PERFORM
      TAKEOVER'
```

- You can also issue other commands to any other job running under MVS:

```
DFHCLT TYPE=COMMAND,
      COMMAND='MODIFY jobname,command
      string'
```

Messages to the operator — DFHCLT TYPE=WTO

These two instructions define a message that is written to the system operator.

label	DFHCLT	TYPE=WTO ,WTOL=addr
addr	WTO	'message to operator' [,ROUTCDE=(number)] [,DESC=(number)] ,MF=L

WTOL=addr

Specifies the address of a list format WTO macro that defines the message and any associated route codes and descriptor codes.

The MF (macro format), ROUTCDE (routing code), and DESC (descriptor) operands of the WTO macro are described in the *MVS/ESA Authorized Assembler Programming Reference* manual.

An example is to send a request to the operator:

```

          DFHCLT TYPE=WTO,
          WTOL=wtoad
wtoad    WTO    'switch local terminals, please',
          MF=L

```

Closing the command list — DFHCLT TYPE=LISTEND

This instruction defines the end of the set of commands and messages issued by an alternate system when it takes over from an active system.

Table 26. DFHCLT TYPE=LISTEND

label	DFHCLT	TYPE=LISTEND

DFHCLT TYPE=LISTEND

Chapter 27. DCT — destination control table

The destination control table (DCT) contains an entry for each **transient data destination**. A destination can be intrapartition, extrapartition, indirect, or remote. You code different DFHDCT macros for each type. The macros specify the symbolic name for each destination, and also other information that CICS needs.

CICS uses several destinations for its own purposes. These entries must be included in the generation of the DCT, if the associated functions are being used. See “Required entries in the destination control table” on page 264 for details of the required entries.

DFHDCT macro types

The following macros define transient data destinations:

- DFHDCT TYPE=INITIAL establishes the control section and necessary linkage editor control statements for the DCT.
- DFHDCT TYPE=SDSCI defines the data control block (DCB), for an extrapartition destination.
- DFHDCT TYPE=EXTRA defines an extrapartition destination: a destination that is outside the CICS region.
- DFHDCT TYPE=INDIRECT defines an indirect destination: a logical destination that points to another destination. (This allows several logical destinations to be merged into one physical destination.)
- DFHDCT TYPE=INTRA defines an intrapartition destination: a destination that is within the CICS region.
- DFHDCT TYPE=REMOTE defines a destination that is owned by another CICS system or region.
- DFHDCT TYPE=FINAL concludes the DCT (see page 249).

Control section — DFHDCT TYPE=INITIAL

The DFHDCT TYPE=INITIAL macro establishes the entry point and beginning address for the DCT being defined.

Table 27. DFHDCT TYPE=INITIAL

DFHDCT	TYPE=INITIAL [,SUFFIX=xx] [,USERID=name]

For general information about TYPE=INITIAL macros, see “TYPE=INITIAL” on page 248.

SUFFIX=xx

Two characters that will be concatenated with DFHDCT to create the name of the destination control table.

USERID=name

Code this with the userid that you want CICS to use for security checking for any trigger-level TYPE=INTRA entry that does not specify its own userid.

If you omit the userid from a trigger-level entry, and also from this TYPE=INITIAL macro, CICS uses the CICS default userid, specified on the DFLTUSER system initialization parameter.

See the USERID description for the TYPE=INTRA macro for information about surrogate user checking.

Data set control information — DFHDCT TYPE=SDSCI

This macro defines the data control block (DCB) for an extrapartition transient data destination.

You must also code a DFHDCT TYPE=EXTRA macro, to define the destination.

Extrapartition data sets can be blocked or unblocked, of fixed or variable length.

Table 28. DFHDCT TYPE=SDSCI

DFHDCT	TYPE=SDSCI ,DSCNAME=name [,BLKSIZE=length] [,BUFNO={1 number}] [,ERROPT={IGNORE SKIP}] [,RECFORM={FIXUNB FIXUNBA FIXUNBM FIXBLK FIXBLKA FIXBLKM VARBLK VARBLKA VARBLKM VARUNB VARUNBA VARUNBM}] [,RECSIZE=length] [,REWIND={LEAVE REREAD}] [,TYPEFLE={INPUT OUTPUT RDBACK}]

TYPE=SDSCI

Indicates that this DCT entry contains data set control information.

BLKSIZE=length

Code this with the length of the block, in bytes.

For V format data sets, each block consists of a block descriptor word followed by one or more logical records. The value coded for BLKSIZE must include four bytes for the block descriptor word, and also make allowance for the largest possible logical record (which, itself, includes 4 bytes of record descriptor word).

If the data set already exists, BLKSIZE can be omitted. However, if BLKSIZE is coded for an input data set it should match the data set BLKSIZE.

DFHDCT TYPE=EXTRA

BUFNO={1|number}

Code this with the number of buffers to be provided. Any number up to 255 may be coded.

DSCNAME=name

Code this with the 1- to 8-character data set control name. This name must be the same as that coded in the DSCNAME operand of any associated DFHDCT TYPE=EXTRA macro.

The name used for DSCNAME must be used as the ddname on the DD statement, and will also be used as the name for the DCB that is created.

The name must not start with the letters "DFH", which are reserved for use by CICS, unless it is describing one of the standard destinations listed under "Required entries in the destination control table" on page 264. Use of the prefix "DFH" may cause assembly errors and future compatibility problems, because the DSCNAME parameter becomes an externally-specified name.

ERROPT={IGNORE|SKIP}

Code this with the error option to be performed if an I/O error occurs.

IGNORE

The block that caused the error is accepted.

SKIP

The block that caused the error is skipped.

RECFORM={FIXUNB|FIXUNBA|FIXUNBM|FIXBLK|FIXBLKA|FIXBLKM|VARBLK|VARBLKA|VARBLKM|VARUNB|VARUNBA|VARUNBM}

Code this with the record format of the data set.

FIXUNB

Fixed unblocked records

FIXBLK

Fixed blocked records

VARBLK

Variable blocked records

VARUNB

Variable unblocked records

FIXUNBA, FIXBLKA, VARBLKA, VARUNBA

Fixed unblocked, fixed blocked, variable blocked, or variable unblocked records respectively, **together with** ASA control characters

FIXUNBM, FIXBLKM, VARBLKM, VARUNBM

Fixed unblocked, fixed blocked, variable blocked, or variable unblocked records respectively, **together with** machine control characters

RECSIZE=length

Code this with the length of the record, in bytes.

For V format data sets, each logical record consists of a record descriptor word followed by a data record. The value coded for RECSIZE must include 4 bytes for the

record descriptor word (or LLBB), and also make allowance for the largest possible data record.

RECSIZE=length need be coded only for RECFM=FIXBLK.

REWIND={LEAVE|REREAD}

Code this with the disposition of a tape data set.

LEAVE

The current tape is positioned to the logical end of the data set.

REREAD

The current tape is positioned to reprocess the data set.

TYPEFLE={INPUT|OUTPUT|RDBACK}

Code this with the type of data set.

INPUT

An input data set.

OUTPUT

An output data set.

RDBACK

An input data set that is to be read backward.

An extrapartition SDSCI can be either input or output, not both.

For more information on the above operands, see the *MVS/ESA Data Administration: Macro Instruction Reference* manual.

Extrapartition destinations — DFHDCT TYPE=EXTRA

Destinations outside the CICS region (but which are allocated to CICS) are specified in the DFHDCT TYPE=EXTRA macro. This macro must be generated once for every extrapartition destination.

Extrapartition destinations are used for:

- Sending data outside the CICS region; for example, data created by a transaction, for processing by a batch program
- Retrieving data from outside the region; for example, data received from terminals, as input to a transaction.

Extrapartition data is sequential and is managed by QSAM.

Table 29. DFHDCT TYPE=EXTRA

	DFHDCT	TYPE=EXTRA
#		,DESTID=name
#		,DSCNAME=name
#		[,LENGTH=length]
		[,OPEN={INITIAL DEFERRED}]
		[,RMTNAME=name]
		[,SYSIDNT=name]

TYPE=EXTRA

Indicates an extrapartition destination.

DESTID=name

Code this with the symbolic name of the extrapartition destination. The symbolic name is used in the transient data operations to specify the destination.

Any DESTID of more than four characters is truncated on the right.

The DESTID should not start with the letter C, which is reserved for defining the destinations required by some CICS facilities. This applies to DFHDCT TYPE=EXTRA, TYPE=INDIRECT, and TYPE=INTRA. See "Required entries in the destination control table" on page 264 for a list of these destinations.

You must not use special characters, lower case, or mixed case characters in a DESTID name.

DSCNAME=name

Code this with the same file name you used in DFHDCT TYPE=SDSCI.

If two or more extrapartition destinations refer to the same SDSCI, only one destination can be open at the same time.

If OPEN=INITIAL is specified for each of these destinations, the choice of destination to be opened will be arbitrary. To avoid this, define one of the destinations as extrapartition and the others as indirect on the first.

LENGTH=length

Specifies, as a decimal value, the record length in bytes of fixed-length records in the queue. The length you specify must correspond to the RECSIZE length on the associated SDSCI entry in the DCT.

A CICS region that references a remote DCT entry requires the length of the record. If you do not specify it on the DCT entry, the application program must specify it on the WRITEQ and READQ requests

If you omit the SYSIDNT parameter, LENGTH is ignored.

OPEN={INITIAL|DEFERRED}

Code this with the initial status of the data set.

INITIAL

The data set is to be opened by system initialization.

DEFERRED

The data set remains closed until you indicate that you want to open it by using the CEMT INQUIRE|SET TDQUEUE command.

RMTNAME=name

code this with the 1- to 4-character name by which the destination is known in the CICS region in which the destination resides (the remote region).

If you omit this parameter, CICS uses the name specified on the DESTID parameter (that is, the local and remote names are the same).

This parameter is meaningful only when you specify the SYSIDNT parameter.

SYSIDNT=name

Identifies the CICS region in which the remote transient data queue resides. The 4-character alphanumeric name specified must match the SYSIDNT system initialization parameter specified on the region that "owns" the queue (the region in which the queue is a local resource).

If you omit SYSIDNT, the queue is treated as a local queue.

Indirect destinations — DFHDCT TYPE=INDIRECT

An indirect destination is specified by the DFHDCT TYPE=INDIRECT macro. The indirect destination does not point to an actual data set, but to another destination. This may be extrapartition, intrapartition, or remote. It may even be another indirect destination.

For example, you can give a different symbolic name (DESTID) to each of several different message types. You then have the flexibility to send all these message types to the same physical destination (INDDEST), or to send them to different physical destinations.

The DFH\$TDWT sample program demonstrates how you can use indirect destinations to send different categories of message to the same terminal. For programming information about DFH\$TDWT, see the *CICS/ESA Customization Guide*.

If you use EXEC CICS INQUIRE TDQUEUE, information will always be returned about an indirect queue. (This does not, however, guarantee that the inquiry transaction will be able to use EXEC CICS INQUIRE TDQUEUE for the ultimate target queue.)

If the QUEUE operand of an EXEC CICS WRITEQ TD (or READQ or DELETEQ) command specifies an indirect queue, access is determined by the security setting of the ultimate target queue.

Table 30. DFHDCT TYPE=INDIRECT

	DFHDCT	TYPE=INDIRECT ,DESTID=name ,INDDEST=name

TYPE=INDIRECT

Indicates an indirect destination.

DESTID=name

Code this with the 1- through 4-character symbolic name of the indirect destination. The symbolic name is used when writing to the destination.

DFHDCT TYPE=INTRA

You must not use special characters, lower case, or mixed case characters in a DESTID name.

INDEST=name

Code this with the name (DESTID) of a transient data destination. The destination can be intrapartition, extrapartition, remote, or indirect. If there is no DCT entry for the destination with this name, an assembly error results.

Intrapartition destinations — DFHDCT TYPE=INTRA

This macro specifies a destination for data that is to be stored temporarily.

An intrapartition destination may be a terminal, a file, or another system. A single data set, managed by VSAM, is used to hold the data for all intrapartition destinations. This macro must be coded once for every intrapartition destination.

You can specify a transaction to process the records and a **trigger level** for each intrapartition destination. The trigger level represents a number of records. When this number of records has been accumulated, the specified transaction is initiated.

The intrapartition destination may be defined as logically recoverable, physically recoverable, or not recoverable.

Logically recoverable destinations are restored (after individual transaction failures and after total system failures) to the status they had at the end of the last completed LUW. (A **logical unit of work (LUW)** begins at start of task or at a **synchronization (sync) point**, and ends at end of task or at a syncpoint.)

Physically recoverable destinations are restored (after a total system failure) to the status they had when the system failure occurred.

Recovery does not occur if DCT=(,COLD) is coded in the DFHSIT macro or in the system initialization overrides.

DFHDCT	TYPE=INTRA
	,DESTID=name [,DESTFAC={(<u>TERMINAL</u> [,trmidnt]) FILE[,USERID=name]}(SYSTEM,sysidnt)} [,DESTRCV={NO PH LG}] [,RMTNAME=name] [,SYSIDNT=name] [,TRANSID=name] [,TRIGLEV={1 number}]

TYPE=INTRA

Indicates an intrapartition destination.

DESTID=name

Code this with the symbolic name of the intrapartition destination. The symbolic name is used to identify the intrapartition queue for I/O operations. It must not be more than four characters in length.

You must not use special characters, lower case, or mixed case characters in a DESTID name.

DESTFAC={(TERMINAL[,trmidnt]) |

FILE[,USERID=name]}(SYSTEM,sysidnt)}

Code this with the type of destination that the queue represents.

(TERMINAL[,trmidnt])

The transient data destination is to be associated with the terminal identified by trmidnt. The terminal must be defined to CICS using either an RDO TERMINAL definition, or a DFHTCT TYPE=TERMINAL macro.

If you do not specify trmidnt, it defaults to the value of DESTID. If ATI is used, as specified in the TRANSID and TRIGLEV operands, the transaction that is initiated is associated with the specified terminal, which must be available before the transaction can be initiated.

FILE

The transient data destination is to be used as a file of data records that are not associated with a particular terminal or system. ATI does not require a terminal to be available.

USERID={name}

Code this with the userid that you want CICS to use for security checking for the trigger-level transaction specified on the TRANSID operand. USERID is valid only when the destination is defined as DESTFAC=FILE.

The trigger-level transaction runs under the authority of the specified userid, which must be authorized to all the resources used by the transaction.

If you omit the userid from a qualifying trigger-level entry, CICS uses the userid specified on the TYPE=INITIAL macro. If you omit the userid from the TYPE=INITIAL macro also, CICS uses the CICS default userid, specified on the DFLTUSER system initialization parameter. You must ensure that the CICS region userid of any CICS region in which this DCT is installed, is defined as a surrogate for all the userids specified in the DCT. This is because, during initialization, CICS performs a surrogate user security check against the CICS region userid. If the surrogate security check fails, CICS deactivates automatic transaction initiation by trigger-level for the intrapartition queue for which the surrogate check failed.

For further information about surrogate user security, refer to the *CICS/ESA CICS-RACF Security Guide*.

(SYSTEM,sysidnt)

The transient data destination is to be associated with a system identified by sysidnt. The system must be defined to the local CICS system using an RDO CONNECTION definition.

The primary purpose of coding DESTFAC=SYSTEM,sysidnt is to initiate a distributed transaction processing (DTP) session. For details of DTP considerations in application programming, please refer to the *CICS/ESA Application Programming Guide*.

DESTRCV={NO|PH|LG}

Code this to indicate the recoverability attributes of the destination in the event of an abnormal termination of either CICS or the transaction processing the destination.

NO

This destination is not recoverable, and automatic logging is not to be performed to keep track of accesses to this destination (the default).

Queue records are held on one or more control intervals (CIs); each CI is released as soon as the last record on it has been read.

PH This destination is physically recoverable, and automatic logging is to be performed to keep track of accesses by application programs. If emergency restart occurs, this destination is to be recovered to its status at the time CICS terminated.

Queue records are held on one or more control intervals (CIs); each CI is released as soon as the last record on it has been read.

LG This destination is logically recoverable, and automatic logging is to be performed to keep track of accesses by application programs. If a transaction that accessed this destination was in-flight at the time of abnormal termination, in the subsequent emergency restart or dynamic transaction backout this destination is restored to the status it had before the in-flight unit of work modified it.

When this destination is accessed, the task that issued the WRITEQ TD or READQ TD command is enqueued upon the input or output end of the transient data queue, respectively. This enqueue will be maintained until the task terminates or issues a syncpoint request to signal the end of a logical unit of work. This is necessary to ensure the integrity of the data being accessed. Because the enqueues are thus maintained for a longer period of time, an enqueue lockout is possible if an application program that accesses this destination performs what is effectively more than one logical unit of work against it without defining each separate logical unit of work to CICS by issuing the syncpoint request. Furthermore, when a DELETEQ request is issued for a logically recoverable

destination, **both** the input and output ends of the queue are enqueued upon. This increases the probability of an enqueue lockout.

Queue records are held on one or more control intervals (CIs); each CI is marked for release as soon as the last record on it has been read. However, the release does not occur until the end of task or until after the next user syncpoint.

RMTNAME=name

code this with the 1- to 4-character name by which the destination is known in the CICS region in which the destination resides (the remote region).

If you omit this parameter, CICS uses the name specified on the DESTID parameter (that is, the local and remote names are the same).

This parameter is meaningful only when you specify the SYSIDNT parameter.

SYSIDNT=name

Identifies the CICS region in which the remote transient data queue resides. The 4-character alphanumeric name specified must match the SYSIDNT system initialization parameter specified on the region that "owns" the queue (the region in which the queue is a local resource).

If you omit SYSIDNT, the queue is treated as a local queue.

TRANSID=name

Code this to identify the transaction that is to be automatically initiated when the trigger level is reached. The purpose of transactions that are initiated in such a way is to read records from the destination. If this operand is omitted, or if TRIGLEV=0 is coded, some other means must be used to schedule transactions to read records from the destinations.

This transaction must not reside in a remote CICS system,

+

APAR PN87931

+

added to this sentence

+

or be defined as dynamic. If it does, the transaction initiation fails and a warning message is issued to the console.

TRIGLEV={1|number}

Code this with the number of records to be accumulated before a task is automatically initiated to process them. (This number is known as the **trigger level**.) If you code the TRANSID operand, TRIGLEV defaults to 1. The maximum TRIGLEV is 32767.

If you have coded DESTFAC=TERMINAL, the task is not initiated until the terminal is available.

If you have coded DESTFAC=FILE, no terminal is necessary for the task to be initiated. For a non-terminal destination, if a **maximum task, short-on-storage** or **no-space** condition exists, the task is not initiated. This

DFHDCT TYPE=REMOTE

is also true during stages 1 and 2 of initialization, and during the final stage of shutdown. It is initiated when the stress condition no longer exists and a subsequent TD WRITE occurs.

If a VTAM terminal is defined, with TRIGLEV=1, as the destination for CSTL on two ISC CICS systems, a performance problem may arise when both systems repeatedly acquire and release the terminal in order to write out the session started and session ended messages.

During CICS operation, the trigger level can be changed using the CEMT transaction. If the trigger level is reduced to a number that is equal to or less than the number of records accumulated so far, the task is initiated when the next record is put to the destination.

Remote destinations — DFHDCT TYPE=REMOTE

The DFHDCT TYPE=REMOTE macro defines a transient data destination that is owned by another CICS system or region. The destination must also have a complete definition in the system or region in which it resides.

Note that if a transient data request includes the SYSID operand, CICS does not refer to the DCT but identifies the specified destination as remote.

DFHDCT	TYPE=REMOTE
	,DESTID=name ,SYSIDNT=name [,LENGTH=length] [,RMTNAME=name]

TYPE=REMOTE

Indicates that this DCT entry identifies a remote transient data destination.

DESTID=name

Code this with a 4-character name by which the destination is known to application programs in the local system or region. For further information, see the RMTNAME operand below.

You must not use special characters, lower case, or mixed case characters in a DESTID name.

+ **APAR PN76660**
+ added this sentence

+ The DESTID name should not start with a 'C' unless the name is known to CICS.

SYSIDNT=name

Code this with the 4-character alphanumeric name of the system or region in which the remote transient data destination resides. The name specified must be the same as that given in the CONNECTION name of the RDO definition. (For more guidance information about the CONNECTION option, see Chapter 12, "CONNECTION" on page 129.)

LENGTH=length

Code this with the length in bytes of fixed records for a remote destination. The value specified must correspond to that specified for the DCT in the system or region in which the destination resides. If a value is not specified for the LENGTH operand, the LENGTH parameter must be given in READQ or WRITEQ requests in the application program.

RMTNAME=name

Code this with the 4-character name by which the destination is known in the system or region in which that destination resides. If this operand is omitted (the normal case), the name specified in the DESTID operand is used.

If more than one system or region has a destination with the same name, the DESTID operand allows the definition of an alias that routes a transient data request to a specific system or region. RMTNAME defines the common name, SYSIDNT defines the system or region, and DESTID defines the unique alias. A transient data request using the alias identifies the remote name **and** the system or region to which the request is shipped.

Examples:

+ **APAR PN76660**
+ changed the examples to remove references to system 'C' and replace them with system 'E'.
+

1. Destination A001 is owned by system A.

SYSTEM	DESTID	RMTNAME
A	A001	A001
B	B001	A001
D	D001	A001
E	E001	A001

+ In system A, both RMTNAME and DESTID are A001. In systems B, D, and E, RMTNAME is A001, but DESTID must be different.

+ 2. Four systems A, B, D, and E each own a different destination, but each destination has the same name X001.

Each system has a definition for its local destination and each of the three remote destinations.

<i>SYSTEM</i>	<i>DESTID</i>	<i>SYSIDNT</i>	<i>RMTNAME</i>
A	A002	A	X001
A	B002	B	X001
+ A	D002	D	X001
+ A	E002	E	X001

Each remote definition in system A:

- Has the RMTNAME *X001*
- Defines the remote system with the SYSIDNT parameter *B*, *D*, or *E*
- Uses the DESTID parameter to define a unique alias for use by application programs in the local region.

DFHDCT examples

Figure 43 shows an example of the coding for a DCT. This DCT includes an extrapartition destination and three intrapartition destinations.

```
DFHDCT TYPE=INITIAL
DFHDCT TYPE=SDSCI,DSCNAME=AAAXTRA, *
      RECFORM=FIXUNB *
DFHDCT TYPE=EXTRA,DSCNAME=AAAXTRA, *
      DESTID=BETA
DFHDCT TYPE=INTRA,DESTID=GAMA
DFHDCT TYPE=INTRA,DESTID=SAMA
DFHDCT TYPE=INTRA,DESTID=DAMA, *
      TRIGLEV=5,DESTFAC=TERMINAL, *
      TRANSID=AUTO
DFHDCT TYPE=FINAL
END
```

Figure 43. Extrapartition and intrapartition destinations

Required entries in the destination control table

The following destinations are used by some of the CICS-supplied transactions. The copybook DFH\$DCTR in the CICS sample library (CICS410.SDFHSAMP) contains the + macros for these destinations. This contains the + TYPE=EXTRA entries. The TYPE=SDSCI entries are + contained in DFH\$DCTD, which should also be used.

- These entries must be included in the generation of the DCT, if the associated functions are being used.
- These destinations must always be enabled. If you define any of them as indirect destinations, their final target destination and any associated transactions (if applicable) must also be enabled.
- These entries may **not** be coded as remote destinations, either directly or by using the DFHDCT TYPE=INDIRECT macro to make the final destination remote.
- You must **not** code these destinations, either directly or indirectly, as logically recoverable intrapartition destinations.
- You **may** code these destinations as physically recoverable. The advantage of this is that the contents of the transient data queue are not lost across an emergency restart and may provide useful diagnostic information (for example, the CSMT destination). The disadvantage is that emergency restart may take longer because more data has to be read from the log (in extreme cases, CICS may not be able to make an emergency restart).
- Except where noted otherwise, data records written to these destinations will have a variable length of up to 120 bytes.

If you define these destinations as extrapartition then the associated SDSCI or DD statement must specify V format records with a minimum BLKSIZE of 128 bytes.

DESTID=CADL (needed to log VTAM resource definitions)

For VTAM resources, this destination keeps a log of each RDO definition installed in the active CICS system. The log records both the installation of entries in the TCT, and the deletion of autoinstalled entries from the TCT. It records definitions installed by each of the following methods:

- Autoinstall
- CEDA INSTALL
- At system initialization.

For details about how to define CADL and CSDL, see the *CICS/ESA System Definition Guide*.

DESTID=CAIL (needed to log autoinstall terminal model definitions)

The autoinstall terminal model manager (AITM) uses this

destination to log all autoinstall terminal model entries installed in, and deleted from, the TCT.

DESTID=CCPI (needed for CPI Communications messages)

The common programming interface for communications (CPI Communications) writes messages to this destination.

DESTID=CCSE (needed for C language support)

Under CICS, C/370 directs the C standard streams to transient data queues. (You should note that the queue names are fixed in CICS/ESA 4.1: thus, the C standard streams cannot be redirected to other queues.) C programs write to the CCSE queue by writing to stderr. You may code this destination as extrapartition, intrapartition, or indirect. If you do not provide a DCT entry for the CCSE queue, writing to stderr in C programs will fail.

DESTID=CCSI (optional, for C language support)

The CCSI queue is reserved for stdin, the C standard stream for input data. You should note that, although the CCSI queue name is reserved for stdin, any attempt to read from stdin in CICS/ESA 4.1 will result in EOF being returned. It is for this reason that this destination has been described as an "optional" entry in the DCT. You may code this destination as extrapartition, intrapartition, or indirect.

DESTID=CCSO (needed for C language support)

CCSO is associated with stdout, the C standard stream for output data. C programs write to the CCSO queue by writing to stdout. You may code this destination as extrapartition, intrapartition, or indirect. If you do not provide a DCT entry for the CCSO queue, writing to stdout in C programs will fail.

The copybook DFH\$DCTR has an entry provided for both output queues directed to //COUT. If you use the CICS-supplied sample definitions as the basis for your own DCT, you should ensure that you add a DD statement for the COUT data set to your CICS start-up job streams.

DESTID=CDBC (needed for DBCTL DFHDB81xx messages)

CDBC is defined as an indirect queue, which points to the CSML extrapartition queue. Only DBCTL DFHDB81xx messages use this data log; the remaining messages use either the terminal or the console.

DESTID=CDUL (needed for transaction dump messages)

CDUL is the destination for transaction dump messages. If a transaction dump is requested, for example after a transaction abend, a message will be written to this destination to show that a dump has been taken or to give a reason if the dump was suppressed.

DESTID=CESE (needed for run-time output from AD/CYCLE Language Environment/370)

For Language Environment/370, when available, all run-time output is written to this transient data queue. For further information, see the *Language Environment/370 Programming Guide*.

DESTID=CMIG (needed for migration log)

CMIG is a **migration log**, that receives messages reporting the use of functions that are no longer supported in CICS/ESA 4.1 (for example, the EXEC CICS ADDRESS CSA command). You can define CMIG as an intrapartition, extrapartition, or indirect destination.

DESTID=CPLI, DESTID=CPLD (both needed for CICS PL/I support)

CPLI is the destination for SYSPRINT output, and CPLD is the destination for PLIDUMP output. The minimum logical record size is 137 for CPLI and 125 for CPLD. If these destinations are extrapartition (direct or indirect), they must be V format. See the installation manual for your PL/I compiler for more details.

DESTID=CRDI (needed to log program resource definitions)

This destination provides a log of installed resource definitions for programs, transactions, maps, and mapsets.

DESTID=CSCS (needed for the signon transaction)

CSCS receives a message giving details of each signon and sign-off. It also receives a message about each rejected attempt at sign on and each resource authorization failure. This destination can be of any type.

DESTID=CSDL (needed to log RDO commands)

The resource definition online (RDO) transactions write to this destination all commands that result in changes to the CICS system definition (CSD) file or active CICS system.

CSDL is required only if you use RDO and want to keep a log of commands.

The maximum length of data records written to CSDL is 128 bytes. If you define CSDL as extrapartition, then the associated SDSCI or DD statement should specify V format records with a minimum BLKSIZE of 136 bytes.

DESTID=CSFL (needed to log file resource definitions)

CSFL is a log of all file resource definitions installed in the active CICS system. Deletions of file resource entries are also logged here.

DESTID=CSKL (needed to log transaction and profile resource definitions)

CSKL is a log of all transaction and profile resource

definitions installed in the active CICS system. Deletions are also logged here.

DESTID=CSML (needed for the sign-off transaction)

CICS sign-off writes data to this destination.

DESTID=CSMT (needed for terminal error and abend messages)

The terminal abnormal condition program (DFHTACP) and abnormal condition program (DFHACP) write terminal error and ABEND messages, respectively, to this destination. You may code this destination as extrapartition, intrapartition, or indirect.

DESTID=CSNE (needed for node error messages)

The node abnormal condition program (DFHZNAC) and the node error program (DFHZNEP) write terminal error messages and data to this destination. You can code this destination as extrapartition, intrapartition, or indirect.

DESTID=CSPL (needed to log program resource definitions)

CSPL is a log of all program resource definitions installed in the active CICS system. Deletions are also logged here.

DESTID=CSRL (needed to log partner resource definitions)

CSRL is a log of all partner resources installed in the active CICS system. Deletions are also recorded here. For more information about partner resources, see Chapter 17, "PARTNER" on page 159.

DESTID=CSSL (needed for recovery utility statistics)

The recovery utility program (DFHRUP) writes statistics to this destination. This destination needs a minimum logical record length of 132 bytes and a minimum blocksize of 136 bytes.

DESTID=CSTL (needed for terminal I/O error messages)

The terminal abnormal condition program (DFHTACP) writes terminal I/O error messages to this destination. You may code this destination as extrapartition, intrapartition, or indirect.

DESTID=CSZL (needed for the Front End Programming Interface)

If you have installed the CICS FEPI feature, CSZL is used as the destination for FEPI messages. For information on FEPI transient data destinations, see the *CICS/ESA Front End Programming Interface User's Guide*.

DESTID=CSZX (needed for the Front End Programming Interface)

If you have installed the CICS FEPI feature, CSZX is intended for use with a triggered transaction. For information on FEPI transient data destinations, see the *CICS/ESA Front End Programming Interface User's Guide*.

DFHDCT required entries

Chapter 28. DL/I directories — DDIR and PDIR

If you use local DL/I you must define two directories to CICS:

- The DL/I database directory (DDIR)
- The DL/I program specification block directory (PDIR)

The DDIR is a directory of all physical databases to be used by the CICS system.

The PDIR is a directory of all the program specification blocks (PSBs) that are accessed by the CICS system.

To use remote DL/I, you must define the PDIR to CICS. For DBCTL, you must define the DL/I directories to DBCTL using IMS-supplied macros.

DL/I database directory — DDIR

The DL/I database directory (DDIR) contains an entry for each physical database description (DBD). To create a DDIR, you have to code DFHDLDBD TYPE=INITIAL, TYPE=ENTRY, and TYPE=FINAL macros. The TYPE=FINAL (see page 249) must be followed by:

```
END DFSIDMD0
```

You code a DFHDLDBD macro for each physical DL/I database that CICS uses via local DL/I.

In a non-data-sharing environment, the database may or may not be shared, depending on the JCL disposition and the VSAM SHAREOPTION with which it is defined.

In a data-sharing environment, the database has a JCL disposition of SHR and is defined with a VSAM cross-system share option of 3. Note that the database can be freely accessed by any address space that is *not* signed on to DBRC. Your operating procedures must ensure that, while a database is being shared under DBRC control, it cannot be *updated* by a region that is not signed on to DBRC.

Control section — DFHDLDBD TYPE=INITIAL

The DFHDLDBD TYPE=INITIAL macro has the following format and operands:

	DFHDLDBD	TYPE=INITIAL [,{DLI DL1}=y.y.y] [.SUFFIX=xx]

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

{DLI|DL1}=y.y.y

Indicates the IMS/ESA level in the form:

Version.Release.Modification level. The levels of IMS/ESA supported are 3.1.0 and 4.1.0. If DLI/DL1 is not specified or specified incorrectly, IMS/ESA level 3.1.0 is assumed.

Physical databases — DFHDLDBD TYPE=ENTRY

The DFHDLDBD TYPE=ENTRY macro has the following format and operands:

Table 34. DFHDLDBD TYPE=ENTRY

	DFHDLDBD	TYPE=ENTRY ,DBD=dbdname [.ACCESS={RO RD UP EX}]

TYPE=ENTRY

Specifies that an entry is to be generated in the list. The maximum number of entries that can be included in the list is 5000.

DBD=dbdname

Specifies the name of the database description block (DBD). In a CICS intercommunication environment, only those DBDs that reside in the given CICS system or region need appear in the DDIR. Thus, if an application program in the local system or region makes a request for a database on a remote system, the corresponding DBDs need not appear in the DDIR for the local system or region. Any databases that are to be accessed by sharing regions should be included in the DDIR. Logical DBDs should not be included in the DDIR.

ACCESS={RO|RD|UP|EX}

Specifies the initial access intent of CICS against the database. The access intent can be changed subsequently by means of the master terminal transaction CEMT. Access intent is applicable both to databases that are involved in IMS/ESA data sharing and to databases that are being shared only via VSAM share options. If sharing via VSAM share options, ACCESS=RO or RD will result in the database being opened for input, and ACCESS=UP or EX will result in the database being opened for update. Normal VSAM sharing rules will then apply.

RO Code this to specify **read-only** access. This means that the status of the data that is read is not guaranteed (changes on the database may not have been committed). There is also the possibility of a reading failure (for example, GET NEXT processing may fail because pointers are currently being changed). Read-only is the only

access available in database level sharing when another subsystem is to be authorized for update.

- RD** Code this to specify **read** access. This means that the data will be read with full integrity. No uncommitted data will be read.
- UP** Code this to specify **update** access. This means that this CICS subsystem is to be allowed to update (insert, delete, replace) records in the database.
- EX** Code this to specify **exclusive** access. This means that this CICS subsystem requires exclusive access to the database.

DL/I PSB directory — PDIR

To create a program specification block directory (PDIR), you have to code DFHDLPSB TYPE=INITIAL, TYPE=ENTRY, and TYPE=FINAL macros. The TYPE=FINAL (see page 249) must be followed by:

```
END DFSIDIR0
```

Control section — DFHDLPSB TYPE=INITIAL

The DFHDLPSB TYPE=INITIAL macro has the following format and operands:

	DFHDLPSB	TYPE=INITIAL [, {DLI DL1}=y.y.y] [, SUFFIX=xx]

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

{DLI|DL1}=y.y.y

Indicates the IMS/ESA level in the form: Version.Release.Modification level. The levels of IMS/ESA supported are 3.1.0 and 4.1.0. If DLI/DL1 is not specified or specified incorrectly, IMS/ESA level 3.1.0 is assumed.

Program specification blocks — DFHDLPSB TYPE=ENTRY

The DFHDLPSB TYPE=ENTRY macro has the following format and operands:

	DFHDLPSB	TYPE=ENTRY ,PSB=psbname [,MXSSASZ=value] [,RMTNAME=name] [,SYSIDNT=name]

TYPE=ENTRY

Indicates that an entry is to be generated in this list. The maximum number of entries that can be included in the list is 32760.

PSB=psbname

Specifies the name of the program specification block (PSB) accessed through the local or remote DL/I. The PSBs required by IMS/ESA batch application programs that participate in a shared database session must be represented in this macro.

If an application program is to access a PSB through another CICS system using remote DL/I, there must be an entry for the PSB in the CICS PDIR. The entry must specify the SYSIDNT and MXSSASZ (and, optionally, RMTNAME) operands.

MXSSASZ=value

Specifies the maximum size of a segment search argument to be used for this PSB. This operand is only required if the SYSIDNT operand is specified.

Note: An excessively large value for MXSSASZ will cause considerable increase in performance cost, and may lead to a data stream being shipped which is too large for the connected CICS system.

RMTNAME=name

Indicates the name by which the PSB is known in the remote system or region and need only be specified when the SYSIDNT operand is used. The default is the psbname specified in the PSB operand. If the original application program that makes the request against this PSB is a batch program using shared database support, the PSB must be local to this system.

SYSIDNT=name

Indicates the four-character alphanumeric name of the remote system or region for which the PSB is applicable when CICS intercommunication facilities are used. The name specified must be the same as that in the SYSIDNT operand in the SIT of the remote system. The local system or region is assumed if this operand is omitted.

Chapter 29. FCT — file control table

The file control table (FCT) describes to CICS the user files that are processed by file management. CICS user files correspond to physical data sets that must have been defined to MVS and allocated to the CICS system before they are used. Because CICS file management processes only VSAM and BDAM data sets, any sequential data sets must be defined as extrapartition destinations by using the DFHDCT macro.

| To define VSAM files, you must use CEDA or DFHCSDUP.
| See Chapter 13, "FILE" on page 141 for information on
| defining VSAM files.

DFHFCT macro types

The following macros specify file characteristics, and some of the characteristics of data sets referenced by the files:

- DFHFCT TYPE=INITIAL establishes the beginning of the FCT.
- DFHFCT TYPE=FILE defines the characteristics of a file, such as record characteristics and types of service allowed. (Note that TYPE=DATASET is retained for compatibility with previous releases, and means exactly the same as TYPE=FILE.)
- DFHFCT TYPE=GROUP is used to migrate file definitions to the CSD.
- DFHFCT TYPE=REMOTE defines a file that is owned by another system or region.
- DFHFCT TYPE=FINAL concludes the FCT. (See page 249.)

Control section — DFHFCT TYPE=INITIAL

The DFHFCT TYPE=INITIAL macro establishes the control sections into which the FCT is assembled, and must be coded as the first statement in the source used to assemble the FCT.

	DFHFCT	TYPE=INITIAL [,SUFFIX=xxx] [,MIGRATE={ YES COMPLETE}]
--	--------	---

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

MIGRATE={**YES**|COMPLETE}

This operand controls the building of FCT entries for VSAM files that are **eligible** for resource definition online (RDO). The only way RDO-eligible resources may be moved to the CSD from the macro source is to use DFHCSDUP, as described under "YES" below. For more detailed guidance, see "MIGRATE macro-defined resource definitions from tables to the CSD" on page 97.

YES

Indicates that you want to generate the necessary data to migrate your RDO-eligible resources. The records generated from the macro source are designed to be used as input to the DFHCSDUP utility program. An MNOTE warning message is issued for each RDO-eligible resource. The DFHCSDUP MIGRATE command will convert them into resource definitions on the CSD. These can then be managed using RDO.

COMPLETE

Use of COMPLETE means that FCT entries are not generated from the macro source for any RDO-eligible files. For each one, the assembly produces an MNOTE. This means that you can keep your FCT macro source code after you have migrated your definitions.

If you continue to assemble an FCT for resources that are not eligible for RDO, you should continue to use MIGRATE=COMPLETE.

Local files — DFHFCT TYPE=FILE

The DFHFCT TYPE=FILE macro describes to CICS file control the physical and operational characteristics of a BDAM file. This macro includes operands that provide information about the access method, record characteristics, and the types of service allowed for the file. This information is used to generate control information used by CICS as well as a DCB or ACB.

Table 38. DFHFCT TYPE=FILE

DFHFCT	TYPE=FILE
	,ACCMETH=BDAM ,FILE=name [,DISP={OLD SHR}] [,DSNAME=name] [,FILSTAT={{ENABLED DISABLED UNENABLED},{OPENED CLOSED}}] [,JID={NO SYSTEM nn}] [,JREQ={ALL}(request[,request,...])] [,LOG={NO YES}] [,RECFORM= ({UNDEFINED VARIABLE FIXED}) [,{BLOCKED UNBLOCKED}] [,DCB format])] [,SERVREQ=(request[,request,...])] [,BLKKEYL=length] [,BLKSIZE=(length[,length])] [,KEYLEN=length] [,LRECL=length] [,RELTYPE={BLK DEC HEX}] [,RKP=number] [,SRCHM=number] [,VERIFY=YES]

TYPE=FILE

Indicates that this macro will describe the characteristics of a file. (Note that TYPE=DATASET is retained for compatibility with previous releases, and means exactly the same as TYPE=FILE.)

ACCMETH=BDAM

Specifies the access method of the data set that is allocated to the file.

BDAM

Basic Direct Access Method.

Note: On a cold start of CICS, only BDAM files are installed from DFHFCT; VSAM file definitions are not installed. If you wish to use existing VSAM file definitions, you must migrate your DFHFCT definitions to the CSD before using CICS/ESA 4.1.

BLKKEYL=length (BDAM only)

Code this with a decimal value from 1 through 255, which represents the length of the physical key in the BDAM physical record. You must code this operand only for files that reference data sets with physical keys (that is, those with SERVREQ=KEY specified). If a data set contains blocked records, and deblocking is to be performed by using a logical key (that is, a key embedded within each logical record), the logical key length must be specified by using the KEYLEN operand.

If necessary, CICS can place a record under exclusive control by building an ENQ argument by concatenating the data set name, the block reference, and the physical key. An ENQ will then be issued using a maximum of 255 bytes of this argument. If the argument exceeds 255 bytes in length, the ENQ will place a range of keys under exclusive control.

BLKSIZE=(length[,length]) (DAM only)

Code this with the length of the block, in bytes. The way you calculate the BLKSIZE depends on the RECFORM. For UNDEFINED or VARIABLE blocks, the length must be the maximum block length. For FIXED length blocks, you calculate the BLKSIZE as follows:

BLKSIZE = LRECL for unblocked records.

BLKSIZE = (LRECL x blocking factor) for blocked records.

If you wish to have a BLKSIZE value generated in the DCB, you must specify that value in the second parameter of the operand; for example, BLKSIZE=(250,250), where the first 250 relates to the FCT and the second 250 relates to the DCB. If the second parameter is not coded, the DCB is generated without a BLKSIZE value.

Warning

- CICS assumes that the block size of the BDAM data set is the size you have specified on the first BLKSIZE parameter. If the value specified is smaller than the actual block size of the data set, storage violations or other unpredictable results are probable during use of the file.
- If you specify the second parameter (the DCB value) the value that you code must always be the actual block size. We recommend that you either omit the second parameter, or make it equal to the first parameter.

DISP={OLD|SHR}

Code this to set the disposition of the data set that will be allocated to this file. If no JCL statement exists for this file when it is opened, the open will be preceded by a dynamic allocation of the file using this disposition. If a JCL statement does exist, it will take precedence over this disposition.

OLD The disposition of the data set will be set to OLD if dynamic allocation is performed.

SHR The disposition of the data set will be set to SHR if dynamic allocation is performed.

Note: You must specify the disposition of the data set, either with the DISP operand, or:

- In a JCL statement
- With CEMT SET
- With EXEC CICS SET

If you specify the disposition in a JCL statement, you **must** specify the data set name in the JCL statement as well.

DSNAME=name

Code from 1 to 44 characters to specify the JCL data set name (DSNAME) to be used for this file. If no JCL statement exists for this file when it is opened, the open will be preceded by a dynamic allocation of the file using

this DSNNAME. If a JCL statement does exist, it will take precedence over this DSNNAME.

You must specify the data set name, either with the DSNNAME operand, or:

- In a JCL statement
- With CEMT SET
- With EXEC CICS SET.

If you specify the data set in a JCL statement, you *must* specify the disposition in the JCL statement as well.

Note: You define the CICS system definition file (CSD) by SIT parameters, not in the FCT.

FILE=name

Code this with a 1- to 8-character symbolic name by which this FCT entry is to be identified. This name is known as the file name and is used by CICS or by CICS application programs to refer to the data set with which this FCT entry has been associated. (Note that the DATASET operand is retained for compatibility with previous releases, and means exactly the same as the FILE operand.)

As well as identifying the FCT entry, this name is also used as the DDNAME when the associated data set is allocated to CICS. The allocation is achieved either by using JCL statements in the start-up job stream, or dynamically, by using the DSNNAME and DISP values in the FCT.

You must not use file names that start with the character string DFH for your own files, because CICS reserves the right to use any file name beginning with DFH. In addition, using the character string FCT for a file name prefix can cause assembly errors.

FILSTAT={(ENABLED|DISABLED|UNENABLED), {OPENED|CLOSED)}

Code this to set the initial status of the file.

The first operand determines the initial enablement state of the file. It is used only during a cold start. (On a warm or emergency start, the file state is determined by the state at the time of the previous shutdown.)

The second operand specifies whether an attempt will be made to open the file at the end of CICS initialization. It applies to cold, warm, and emergency starts.

ENABLED

Normal processing is to be allowed against this file.

DISABLED

Any request against this file from an application program will cause the DISABLED condition to be passed to the program.

UNENABLED

This option is valid only with the CLOSED option. It may be used to prevent the file being opened on first reference. An attempt to access the file in this state will raise the NOTOPEN condition.

OPENED

The file will be opened by an automatically initiated CICS transaction (CSFU) after CICS initialization. (On a warm or emergency start, a file remains UNENABLED, if that was its state at the time of the previous shutdown. CSFU ignores an OPENED option on an UNENABLED file, and leaves the file closed.)

CLOSED

The file is to remain closed until a request is made to open it by the master terminal function, by an EXEC CICS SET command, or by an implicit open.

For each combination of initial states, files will be opened as follows:

(ENABLED,CLOSED)

The file will be opened on first reference. This is the default.

(ENABLED,OPENED)

The file will be opened by the automatically-initiated transaction CSFU after CICS initialization, unless a user application or master terminal function has opened it first.

(DISABLED,OPENED)

The file will be opened by the automatically-initiated transaction CSFU after CICS initialization, unless a user application or master terminal function has *explicitly* opened it first.

(DISABLED,CLOSED)

The file will be opened only by an explicit OPEN request (for example, from the master terminal transaction).

(UNENABLED,CLOSED)

The file will be opened only by an explicit OPEN request. The file state after it has been opened is (ENABLED,OPENED).

Note: For performance reasons, the default CSD entry for transaction CSFU is defined with DTIMOUT=10 (seconds). This can cause a transaction timeout abend if there is a delay in opening a file during CICS startup. See page 209 for an explanation of the DTIMOUT value on TRANSACTION definitions.

JID={NO|SYSTEM|nn}

Code this if automatic journal activity is to take place for this FCT entry, and to identify the journal to be used to record the journaled data. The operations that will cause data records to be journaled are specified in the JREQ parameter.

NO

Journal activity will not occur for this file.

SYSTEM

The system log is to be used for journaling.

DFHFCT TYPE=FILE

nn The identifier of a user journal. You can code a value in the range 2 through 99.

Note: Automatic journaling can be specified if you wish to record file activity for subsequent processing by yourself (for example, user-written data set I/O recovery). It must not be confused with automatic logging (specified with LOG=YES), which is required if CICS is to perform data set backout to remove in-flight task activity during emergency restart or dynamic transaction backout.

JREQ={ALL|(request[,request,...])}

Code this with the file operations that are to be automatically journaled, and whether the journaling operation is to be **synchronous** or **asynchronous** with file activity.

When a synchronous journal operation is executed for a READ request, control is not returned to the program that issued the file control request until the data read is written in the journal data set. When a synchronous journal operation is executed for a WRITE request, the output operation to the data set is not initiated until the data is written in the journal data set.

When an asynchronous journal operation is executed for a READ request, control can be returned as soon as the data read is moved to the journal I/O buffer. When an asynchronous journal operation is executed for a WRITE request, the output operation to the data set can be initiated as soon as the data is moved to the journal I/O buffer.

Synchronization defaults provide asynchronous operation for READs and synchronous operation for WRITEs.

If you have requested automatic journaling, the contents of the journal may not accurately reflect the actual changes to a data set, because the request is journaled before the response from the I/O operation is tested.

If this operand is omitted and JID is coded, JREQ will default to JREQ=(WU,WN).

ALL Journal all file activity with READ asynchronous and WRITE synchronous.

ASY Asynchronous journal operation for WRITE operations.

RO Journal READ ONLY operations.

RU Journal READ UPDATE operations.

SYN Synchronous journal operation for READ operations.

WN Journal WRITE NEW operations.

WU Journal WRITE UPDATE operations.

KEYLEN=length (BDAM only)

Code this with the length of the logical key for the deblocking of the BDAM data set to which this file refers.

The logical key for BDAM data sets is embedded and located through the use of the RKP operand. The length

of the physical key is coded in the BLKKEYL operand, and can be different from the value specified for KEYLEN.

This operand must always be coded when logical keys are used in blocked BDAM data sets.

LOG={NO|YES}

This operand specifies the recovery attributes of the file. Specify LOG=YES if you want automatic logging. This enables backout (recovery) of incomplete changes to the data set referenced by this file, in the event of an emergency restart or transaction abend. Whenever a change – update, deletion, or addition – is made to the data set, the “before” image is automatically recorded in the CICS system log. (Automatic logging should not be confused with automatic journaling.)

NO

Automatic logging is not to be performed.

YES

Automatic logging is to be performed.

When a request is made to change the contents of the data set referenced by the file, the record being updated, added, or deleted is enqueued upon, using the record identification together with the address of the CICS control block representing the base data set. This enqueue is maintained until the task terminates or the application issues a syncpoint request to signal the end of a logical unit of work. This ensures the integrity of the altered data. Because the enqueues are thus maintained for a longer period of time, an enqueue lockout can occur if an application program that accesses this data set performs what is effectively more than one logical unit of work against it, without defining each separate logical unit of work to CICS by issuing a syncpoint request. Also, long-running tasks could tie up storage resources.

Always ensure consistency of recovery attributes between files referring to the same base data set cluster or its paths.

To look at the recovery attributes, use the CEMT INQUIRE DSNAME, or EXEC CICS INQUIRE DSNAME, commands on the base cluster to which the file refers. If you use INQUIRE FILE RECOVSTATUS, you will not necessarily find the recovery attribute that CICS has enforced on the base cluster (if there are inconsistencies between files referring to the same base cluster). We explain this in the next paragraph. If all files are consistent, the recovery attribute on the file will be the same as on the base cluster.

CICS takes the actions shown in the following list when opening a file for update processing. (That is, if you set SERVREQ=ADD, DELETE, or UPDATE. If you set only SERVREQ=READ and/or BROWSE, CICS does not make these consistency checks.)

- If an FCT entry refers to an alternate index (AIX) path and LOG=YES is specified, the AIX must be in the upgrade set for the base. This means that any changes made to the base data set are also reflected in the AIX. If the AIX is not in the upgrade set, the attempt to open the FCT entry for this AIX path fails.
- If a file is the first to be opened for update against a base cluster since the last cold start, the recovery attribute (LOG=YES or NO) is copied into the base cluster block.
- If a file is not the first to be opened for update against a base cluster since the last cold start, the LOG= parameter on the FCT is checked against the LOG= parameter held in the base cluster block. These are the possibilities:
 - The base cluster has LOG=NO:
 - If the FCT entry is defined with LOG=NO, the open proceeds normally.
 - If the FCT entry is defined with LOG=YES, and other files are currently open against the base cluster, the attempt to open the file fails. If there are no other files open against the base, the open is allowed, and the recovery attribute of the base cluster is updated to LOG=YES. This is the only occasion when the recovery attribute of the base can be safely changed from its original setting.
 - The base cluster has LOG=YES:
 - If the FCT entry is defined with LOG=NO, the open proceeds, but the base recovery attribute LOG=YES overrides the setting on the FCT entry.
 - If the FCT entry is defined with LOG=YES, the file open proceeds normally.

CICS also checks whether you have specified **forward** recovery for another file referencing the same data set.

Any change in setting of the recovery options, or a failure to open a data set for an FCT entry, results in a message to the operator. If necessary, the recovery options must be changed. To change the recovery attributes (held in the base cluster block) of a VSAM data set, you can use the CEMT SET DSNAME REMOVE or EXEC CICS SET DSNAME REMOVE commands. This deletes the base cluster block, so CICS has no record of prior recovery settings for this VSAM data set. The next file to open against this data set causes a new base cluster block to be built and, if the file is opened for update, the data set takes on the recovery attributes of this file.

The base cluster block, together with its recovery attributes, is preserved even when all the files relating to it are closed, and across warm and emergency restarts.

The file control program decides whether before- and/or after-images are required by inspecting the **data set's** recovery attributes (held in the base cluster block) and not the attributes of the files open against that VSAM data set.

In those cases, listed above, in which the file open proceeds and the data set attributes are used, the file entry is not changed, and the file control program just looks at the base cluster block to obtain the correct recovery settings. The file definitions are not changed because, if they were, the FCT source would not reflect the object code. This could be a problem if the file were later dynamically allocated to another VSAM data set.

LRECL=length (BDAM only)

Code this with the maximum length (in bytes) of the logical record. The value specified is also the length of records in a fixed length remote file. See the DFHFCT TYPE=REMOTE macro for further information on remote files.

RECFORM=([UNDEFINED|VARIABLE|FIXED]), [BLOCKED|UNBLOCKED],[DCB format])

Code this to describe the format of physical records in the data set. The default is UNDEFINED.

For BDAM data sets, **blocking** refers to CICS blocking, and has no meaning for BDAM. You must specify BLOCKED or UNBLOCKED for all data sets of FIXED or VARIABLE format.

BLOCKED

Specify this option when each physical record is to be viewed by CICS as a block consisting of more than one logical record.

DCB (BDAM only)

Code this with the record format to be inserted in the DCB; for example, RECFORM=(FIXED,BLOCKED,FBS).

The DCB format sub-parameter of the RECFORM operand is the only way you can put record format information into the DCB when the FCT is assembled. The first two sub-parameters of the RECFORM operand do not generate information in the DCB.

FIXED

Records are fixed length.

UNBLOCKED

Specify this option when no CICS block structure is to be used. That is, when there is one CICS logical record for each BDAM physical record.

UNDEFINED (BDAM only)

Records are of undefined length. (If you specify a data set as UNDEFINED, you must allow for an additional 8 bytes for the count field, when calculating the BLKSIZE.)

DFHFCT TYPE=FILE

VARIABLE

Records are variable length.

RELTYPE={BLK|DEC|HEX}

Code this if relative addressing is being used in the block reference portion of the record identification field of the BDAM data set referenced by this file. If the RELTYPE operand is omitted, absolute addressing is assumed (that is, MBBCCHHR).

BLK Relative block addressing is being used.

DEC The zoned decimal format is being used.

HEX The hexadecimal relative track and record format is being used.

RKP=number

Code this with the starting position of the key field in the record relative to the beginning of the record (position zero for BDAM data sets). With variable-length records, this operand must include the 4-byte LLbb field at the beginning of each logical record. This operand must always be coded for data sets that have keys within each logical record, or when browsing.

RKP=number (BDAM only)

Code this with the starting position of the key field in the record relative to the beginning of the record (position zero for BDAM data sets). With variable-length records, this operand must include the 4-byte LLbb field at the beginning of each logical record. This operand must always be coded for data sets that have keys within each logical record, or when browsing.

SERVREQ={request[,request],...}

Code this to define the types of service request that can be processed against the file. The parameters that can be included are as follows:

ADD

Records can be added to the file.

BROWSE

Records may be sequentially retrieved from the file.

KEY

Records are to be retrieved from or added to the file. This parameter is mandatory if the data set referenced by the file is a keyed BDAM data set. It must not be coded for other files.

NOEXCTL

Records are not to be placed under exclusive control when a read for update is requested.

If you do not specify NOEXCTL, BDAM exclusive control is provided by default. This provides integrity in the system. For BDAM, you may specify LOG=YES with SERVREQ=NOEXCTL. This will request only a CICS enqueue and suppress the

BDAM exclusive control, thus providing CICS integrity for the update only until a syncpoint.

Note: The CICS enqueue is at the record level within the CICS region, and lasts until a syncpoint, whereas the BDAM exclusive control operates on a physical block, is system-wide, and lasts only until the update is complete.

READ

Records in this file can be read. READ is assumed, if you specify BROWSE or UPDATE.

UPDATE

Records in this file can be changed.

SERVREQ={request[,request],...}

Code this to define the types of service request that can be processed against the file. The parameters that can be included are as follows:

ADD

Records can be added to the file.

BROWSE

Records may be sequentially retrieved from the file.

DELETE (VSAM only)

Records can be deleted from this file.

KEY (BDAM only)

Records are to be retrieved from or added to the file. This parameter is mandatory if the data set referenced by the file is a keyed BDAM data set. It must not be coded for other files.

NOEXCTL (BDAM only)

Records are not to be placed under exclusive control when a read for update is requested.

If you do not specify NOEXCTL, BDAM exclusive control is provided by default. This provides integrity in the system. For BDAM, you may specify LOG=YES with SERVREQ=NOEXCTL. This will request only a CICS enqueue and suppress the BDAM exclusive control, thus providing CICS integrity for the update only until a syncpoint.

Note: The CICS enqueue is at the record level within the CICS region, and lasts until a syncpoint, whereas the BDAM exclusive control operates on a physical block, is system-wide, and lasts only until the update is complete.

READ

Records in this file can be read. READ is assumed, if you specify BROWSE or UPDATE.

REUSE

There is no need to use this option because CICS now processes VSAM load mode files and work files automatically, without the need for a separate FCT entry.

For compatibility with previous CICS releases, SERVREQ=REUSE is still valid. It was used to specify that a file should be opened in VSAM load mode. Using SERVREQ=REUSE forces the VSAM restrictions for load mode files on the FCT entry, as follows:

- A SERVREQ=REUSE file cannot be recoverable. Any specification relating to recovery will be ignored.
- You must use mass sequential insertion to load the file.
- The file must then be closed and reopened before any further processing can take place. That is, after writing the first record or sequence of records, you must close the file and open it through another DFHFCT TYPE=FILE entry that does not specify REUSE.

These restrictions do not apply to the automatic handling of load mode and work files, that is, when you have not specified SERVREQ=REUSE.

UPDATE

Records in this file can be changed.

SRCHM=number

Code this if multiple track search for keyed records is to be provided. This operand is applicable only to BDAM keyed data sets.

number

The number of tracks or blocks to be searched.
The default is 0.

SRCHM=number (DAM only)

Code this if multiple track search for keyed records is to be provided. This operand is applicable only to DAM keyed data sets.

number

The number of tracks or blocks to be searched.
The default is 0.

VERIFY=YES

Code this if you want to check the parity of disk records after they are written. If this operand is omitted, records are not verified after a write request.

VERIFY=YES (DAM only)

Code this if you want to check the parity of disk records after they are written. If this operand is omitted, records are not verified after a write request.

Configurator: This section is intended to help you use the DFHFCT TYPE=FILE macro to define your files. Each TYPE=FILE instruction describes the characteristics of the file, and of the data set referenced by the file.

Table 39. DFHFCT TYPE=FILE instructions

	BDAM			
	Blocked		Unblocked	
	With key	Without key	With key	Without key
BLKKEYL	R		R	
SRCHM	O		O	
VERIFY	O	O	O	O
RELTYPE	R ¹	R ¹	R ¹	R ¹
LRECL		R	R	R
R				
BLKSIZE	R ³	R	R ²	R
KEYLEN	R ⁵			
RKP	R ⁴		R ⁴	
RECFORM	O	O	O	O
O				
FILSTAT	O	O	O	O
O				
SERVREQ	R ⁶	O	R ⁶	O

Notes:

R Required
O Optional

1. Required if relative type addressing is to be used.
2. If SERVREQ=BROWSE or SERVREQ=ADD, this value must be BLKSIZE + BLKKEYL for unblocked records.
3. If SERVREQ=BROWSE or SERVREQ=ADD, this value must be LRECL x blocking factor + BLKKEYL for blocked records.
4. Required if key exists within logical records.
5. Required if deblocking by key for BDAM
6. SERVREQ=KEY is required.

Migrating FCT definitions — DFHFCT TYPE=GROUP

Use this macro to name the groups into which FCT definitions will be put when you migrate to resource definition online. This macro can appear as many times as required and at any point in the macro source. Each time it appears, it defines the CSD group into which subsequent definitions will be put until the next DFHFCT TYPE=GROUP macro occurs.

Table 40. DFHFCT TYPE=GROUP

DFHFCT	TYPE=GROUP [,GROUP=name]

DFHFCT TYPE=REMOTE

GROUP=name

Code this with the name of the group to which subsequent definitions will be migrated. The name can be up to eight alphanumeric characters, but must not begin with DFH. The default name is FCTxx where xx is the value coded for SUFFIX in the DFHFCT TYPE=INITIAL macro. If an error is found, the existing group name continues.

If a group with the name you specify does not already exist, it will be created. If it does exist, subsequent definitions will be added to it.

GROUP=name

Code this with the name of the group to which subsequent definitions will be migrated. The name can be up to eight alphanumeric characters, but must not begin with DFH. The default name is FCTxx where xx is the value coded for SUFFIX in the DFHFCT TYPE=INITIAL macro. If an error is found, the existing group name continues.

If a group with the name you specify does not already exist, it will be created. If it does exist, subsequent definitions will be added to it.

For details on how to migrate your FCT to the CSD, see the *CICS/ESA Migration Guide*.

Remote files — DFHFCT TYPE=REMOTE

The DFHFCT TYPE=REMOTE macro defines a file that resides in a remote system or region. You can use function request shipping to access this file from the local system. The remote definition that this macro creates gives less information than is needed for the local definition created by the TYPE=FILE macro. Each file that you define using TYPE=REMOTE also has a TYPE=FILE entry in its own system.

You name the system that owns the file using the SYSIDNT operand. The file may be known by a different name in the system that owns it; in this case, you need to code the RMTNAME operand.

Table 41. DFHFCT TYPE=REMOTE

	DFHFCT	TYPE=REMOTE ,SYSIDNT=name [,KEYLEN=length] [,LRECL=length] [,RMTNAME=name]
--	--------	--

TYPE=REMOTE

Indicates that this FCT entry identifies a file that resides in a remote system or region.

FILE=name

Code this with a 1- to 8-character file name, which is the name used by the application programs in the same system as this FCT. (Note that the DATASET operand is retained for compatibility with previous releases, and means exactly the same as the FILE operand.)

KEYLEN=length

Code this with the default key length for a file control request that is sent to a remote system.

For DAM, the value must be equal to the total length of the RIDFLD option required to access the file. In the case of remote blocked files, this length must be the length of the RIDFLD option required to deblock by key. If you do not code this operand, the KEYLENGTH option must be specified in the application program that refers to this file.

KEYLEN=length

Code this with the default key length for a file control request that is sent to a remote system.

For DAM, the value must be equal to the total length of the RIDFLD option required to access the file. In the case of remote blocked files, this length must be the length of the RIDFLD option required to deblock by key. If you do not code this operand, the KEYLENGTH option must be specified in the application program that refers to this file.

LRECL=length

Code this with the default data length (in bytes) for a READ WRITE or REWRITE request that is sent to a remote system.

LRECL=length

Code this with the default data length (in bytes) for a READ WRITE or REWRITE request that is sent to a remote system.

RMTNAME=name

Code this with a 1- to 8-character name by which the file is known to the system or region in which it resides. If this operand is omitted (the normal case), the name coded in the FILE operand will be used.

This allows a single file to be known by different names in different systems. For example, A001 is owned by system A. On the definition in system A, both FILE and RMTNAME are A001. In systems B, C, and D the RMTNAME is always A001, but the FILE may be B001, C001, or any other name.

In addition, this allows each system to own a different file with the same name. For example, system A has a local definition for X002, which it owns. System A also has definitions for B002, C002, and D002. These three files, however, are each owned by another system, and the RMTNAME in each of the definitions in system A is X002. In the system that owns it, each of the four files is known as X002, but in each system, transactions can distinguish between the four, using a different FILE for each.

RMTNAME=name

Code this with a 1- to 8-character name by which the file is known to the system or region in which it resides. If this operand is omitted (the normal case), the name coded in the FILE operand will be used.

This allows a single file to be known by different names in different systems. For example, A001 is owned by system A. On the definition in system A, both FILE and RMTNAME are A001. In systems B, C, and D the RMTNAME is always A001, but the FILE may be B001, C001, or any other name.

In addition, this allows each system to own a different file with the same name. For example, system A has a local definition for X002, which it owns. System A also has definitions for B002, C002, and D002. These three files, however, are each owned by another system, and

the RMTNAME in each of the definitions in system A is X002. In the system that owns it, each of the four files is known as X002, but in each system, transactions can distinguish between the four, using a different FILE for each.

SYSIDNT=name

Code this with the 4-character alphanumeric name of the system or region in which the file is resident. The name given must be the name of the CONNECTION definition for the system (or the SYSIDNT operand in the DFHTCT TYPE=SYSTEM macro), or the SYSIDNT in an explicit remote request in an application program.

SYSIDNT=name

Code this with the 4-character alphanumeric name of the system or region in which the file is resident. The name given must be the name of the CONNECTION definition for the system (or the SYSIDNT operand in the DFHTCT TYPE=SYSTEM macro), or the SYSIDNT in an explicit remote request in an application program.

Example

Figure 44 illustrates the coding that is required to create an FCT entry for a BDAM file.

```
DFHFCT TYPE=FILE, *
      FILE=DAM83, *
      ACCMETH=BDAM, *
      SERVREQ=(READ,BROWSE,KEY), *
      BLKSIZE=172, *
      RECFORM=(FIXED,BLOCKED), *
      LRECL=86, *
      RELTYPE=HEX, *
      KEYLEN=6, *
      BLKKEYL=6, *
      RKP=0, *
      FILSTAT=(ENABLED,OPENED)
```

Figure 44. File control table example — BDAM file

Chapter 30. JCT — journal control table

The journal control table (JCT) describes the system log and user journals to CICS, for access through journal management. The JCT contains control information and operating system control blocks for each journal.

Elements of DFHJCT

You define the journal control table with the following macro instructions:

- Control Section — DFHJCT TYPE=INITIAL
- Journal Entries — DFHJCT TYPE=ENTRY
- End of journal control table — DFHJCT TYPE=FINAL (see page 249).

Journal devices

A journal may reside on tape or disk and may occupy either one or two tape drives or disk data sets. The SMF data set may be used for user journals (see “System management facility (SMF)” on page 280). This is specified through the JTYPE keyword of the DFHJCT TYPE=ENTRY macro. If you use a disk drive, you are recommended to dedicate it to the use of journaling. (However, if XRF is to be used, the CICS system journal must be on two disk data sets.)

Standard labeled tape volumes can be used for system log and user journal data sets.

Journal tapes are normally rewound and unloaded at end-of-volume or when the journal is closed. If two tape drives are assigned to a journal, the device is automatically switched at end-of-volume.

Journal data sets on disk are reused when filled. If two data sets are specified, the system switches back and forth between them. If JOUROPT=PAUSE is specified through the DFHJCT TYPE=ENTRY macro instruction, a data set will not start to be reused until the console operator allows it. This protects the data until the operator verifies that it is no longer needed; the operator may wish to dump the data to tape or process it in some other way before it is destroyed.

Journal archive support is obtained by coding JOUROPT=AUTOARCH. This invokes automatic journal archiving for the journal specified by JFILEID.

All disk data sets for journaling use must be preformatted using the DFHJCJFP utility prior to use in a CICS execution. Similarly, tapes must be preformatted using the DFHFTAP utility prior to their first use. After this has been done, both data sets and tapes can be reused for successive CICS executions without reformatting.

When using an X data set for the system log, if you have opted not to use the archiving facility (that is, you have **not** coded JOUROPT=AUTOARCH in the JCT) you can specify where logging output will continue after an emergency restart, by means of the JOUROPT operand.

Journal buffers

Journal records are blocked, variable-length records. CICS writes a block label record as the first record of each block and adds a system prefix to each journal record written.

Each journal defined to CICS employs two, equally-sized, buffers. The buffering system works like this:

1. Records are added to buffer 1 until a write-to-disk (or write-to-tape) operation is required. This happens when the buffer is full, or when a record requiring immediate output (STARTIO) is placed in the buffer.
2. Subsequent records are added to buffer 2.
3. After the first write operation has completed successfully, the next is instigated from buffer 2, when this becomes necessary.
4. Subsequent records are added to buffer 1, overwriting the original contents; and so on.

Buffer size: The size of each of the two buffers for each journal is specified by the BUFSIZE operand of the corresponding DFHJCT TYPE=ENTRY macro.

+ The minimum buffer size is 512 bytes or the sum of the + following:

- 46 bytes for the block length field and the block label record
- 30 bytes for the record length field and the common root of the system prefix
- The length of the variable portion of the system prefix
- Sufficient space to satisfy the largest journal output request made through the journal control request, including:
 - The length of the user prefix (plus two bytes) if specified by the PFXLGTH operand in the journal control output request
 - The length of the journal record as specified by the JCDLGTH operand in the journal control output request.

The maximum buffer size is 32760, or the track capacity, whichever is the smaller. Users of SMF 110 type records and SMF data sets should read “System management facility (SMF)” on page 280.

DFHJCT TYPE=INITIAL • DFHJCT TYPE=ENTRY

Other factors that need to be considered in selecting buffer size include:

- The amount of logging being done for file control files. File control makes both synchronous and asynchronous journal requests without specifying STARTIO. In a lightly loaded system, large journal buffers can delay tasks because the buffers are not being written out often enough.
 - If DL/I logging is being done through CICS journaling, the minimum buffer size that can be specified is 1100 bytes.
 - The volume of records to be written.
 - The lengths of the records.
 - The percentage of synchronous requests. (When a synchronous request is made, the record is moved to the output area and the block is written regardless of its length. Control is not returned to the program that issued the journal output request until the data is recorded on the journal device.)
- + • The minimum buffer size that can be specified is 512.
- If VTAM is in use, the system log must be defined with a minimum buffer size of 1600 bytes.

The following statistics are gathered for each journal to assist in tuning:

- The number of output requests made
- The number of blocks written
- The average length of blocks written
- The number of times the current buffer (that is, the one to which records were currently being added) was physically full and therefore ready to perform a write operation, but was delayed because the alternate buffer had not completed its own output.

Each journal task acquires space for a TCA, a JCA, and the specified buffer size at the time it is created during system initialization. The TCA has a TWA length of zero.

System management facility (SMF)

CICS user journals that are used for output only (except the system log) can optionally use the system management facility (SMF) 110 type record and the SMF dataset to record data instead of tapes and disks. To do this, you will need to create the necessary journal control table (JCT) entries by specifying FORMAT=SMF and JTYPE=SMF on the DFHJCT TYPE=ENTRY macro instruction.

When using SMF data sets the maximum buffer size is 32756.

Control section — DFHJCT TYPE=INITIAL

The control section into which the JCT is assembled is established by the DFHJCT TYPE=INITIAL macro instruction. This macro must be coded as the first statement in the source used to assemble the journal control table.

Table 42. DFHJCT TYPE=INITIAL

	DFHJCT	TYPE=INITIAL [,SUFFIX=xxx]
--	--------	-------------------------------

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

Journal entries — DFHJCT TYPE=ENTRY

Each journal referred to during CICS execution must have a JCT entry as generated by the DFHJCT TYPE=ENTRY macro.

Table 43. DFHJCT TYPE=ENTRY

	DFHJCT	TYPE=ENTRY ,JFILEID={SYSTEM nn} ,BUFSIZE=nnnnn [,ARCHJCL={DFH\$ARCH member}] [,FORMAT=SMF] [,JOUROPT={([CRUCIAL] [,RETRY {[,AUTOARCH]} [,LRU EXTA] [,PAUSE]])}] [,JTYPE={TAPE1 TAPE2 DISK1 DISK2 SMF}] [,LABEL={NO STANDARD}] [,LAYOUT={LINEAR CYCLIC}] [,OPEN={INITIAL DEFERRED}] [,SYSWAIT={STARTIO ASIS}] [,VOLCNT=number]
--	--------	---

TYPE=ENTRY

Indicates that an entry is to be generated in this table.

ARCHJCL={DFH\$ARCH|member}

Code this with the name of the partitioned data set member name to be used for this journal when submitting an archive job. It should not be coded if JOUROPT=AUTOARCH is not coded.

BUFSIZE=nnnnn

- + Code this with a decimal number indicating the size of each buffer (in bytes) to be used for journal I/O operations. (Note that the two buffers associated with each journal are identical in size. Coding BUFSIZE defines the size of both.) The minimum size is 512. The maximum is 32760 for tape. For CKD disk devices the maximum is the lesser of the maximum track capacity of the device and 32760. For DL/I logging, the CICS system log must have a minimum buffer size of 1100 bytes.

For a discussion of buffer size, see “Buffer size” on page 279.

FORMAT=SMF

Code this if journal records will be written in SMF format. If JTYPE=SMF, you must code FORMAT=SMF.

JFILEID={SYSTEM|nn}

Code this with the journal identification for this entry.

SYSTEM

Code this if the journal being defined is the CICS system log. The system log is required if CICS is to perform automatic logging of changes to CICS resources, to provide for emergency restart or an XRF takeover.

The CICS system log must have an associated BUFSIZE value of at least 1100 bytes if DL/I is used.

nn A decimal number between 2 and 99 that identifies the user journal to be used.

JOUROPT={([CRUCIAL],[RETRY]([AU TOARCH] [,([LRU|EXTA],[PAUSE]))])}

Code this with the journaling option or options that are to apply to the journal represented by this entry.

Note: LRU (the default) and EXTA cannot be used with user journals.

For details about these options, see the *CICS/ESA Recovery and Restart Guide*.

AUTOARCH

Code this to invoke automatic journal archiving for the journal specified by the JFILEID operand. If AUTOARCH is coded, JTYPE=DISK2 **must** be coded. Use of AUTOARCH ensures that data sets are archived promptly and are not overwritten until archived. The AUTOARCH option is particularly useful if you use the VSAM recovery and backout utilities, because automatic archiving offers greater security and reduces the possibility of a delay caused by archiving just before you use an offline utility.

If you code AUTOARCH, you cannot code LRU, EXTA, or PAUSE.

CRUCIAL

The CRUCIAL option determines what action CICS takes on encountering an I/O error for a journal.

The action that CICS takes depends on the kind of I/O error encountered.

- If an **unrecoverable output I/O error** occurs for a journal, CICS issues a message suggesting that you initiate a non-immediate shutdown of CICS. If the journal has been specified as CRUCIAL, CICS then issues another message asking you to acknowledge the shutdown request. Transactions attempting to use the journal will terminate abnormally.

The journal task for the specified journal terminates abnormally in this error situation, and if you have not specified the CRUCIAL option, or if you do not perform the shutdown at this time, CICS execution will continue, but the journal will remain unavailable for the remainder of the run.

- If the I/O error occurs during **end of volume processing**, and you have specified the CRUCIAL option for the journal, CICS execution is abnormally terminated with a dump. If you have not specified the CRUCIAL option, execution continues and the journal is unavailable for the duration of the run; the journal task of the journal is abnormally terminated.

LRU (system journals only)

After DFHJ01X has been used during an emergency restart, logging will continue on the less recently used data set. If you code LRU, you must not code EXTA. If you code AUTOARCH, you cannot code LRU. For an explanation of which data set is used by CICS for logging purposes, see the *CICS/ESA Recovery and Restart Guide*.

EXTA (system journals only)

This option forces CICS to use data set DFHJ01A after DFHJ01X has been used during an emergency restart. This is convenient if your DFHJ01B data set is small. If you code EXTA, you must not code LRU. If you code AUTOARCH, you cannot code EXTA.

PAUSE

You code this to ensure that a journal data set is not reused until the operator authorizes it. If you do not code this option, the data set is automatically reused, thus overwriting the previous journal records. If you code AUTOARCH, you cannot code PAUSE. In an IMS/ESA data-sharing environment you should use this option if you do not use the AUTOARCH option; otherwise, clashes with IMS/ESA utilities are likely.

Disk journals:

If PAUSE is specified and the journal is defined as a **single** data set, CICS closes the journal when it is full, and issues messages DFH4583 and DFH4584. Message DFH4583 indicates that the journal is closed and ready to be copied, and DFH4584 tells the operator that CICS is waiting for a reply. All tasks using the journal are held up until CICS receives a reply, which gives the operator an opportunity to copy the data set before allowing it to be overwritten. Any transactions that cause output to the journal will be held up until the copy operation is complete and the operator has replied to the message. So if you use single-data-set journals, make sure they are large enough to hold all the journal output for a given CICS run.

DFHJCT TYPE=ENTRY

If PAUSE is specified for a journal defined as a *pair of data sets*, message DFHJC4583 is issued when the current data set becomes full and is closed by CICS. The other data set is opened, message DFH4508 is issued, and journal activity to the data set continues without waiting for a reply to DFH4583. The operator can then schedule (concurrently with CICS) a batch program to copy the data set to an archive data set. When the copy is complete, the operator replies to DFH4583, indicating to CICS that it can reuse the first data set. If a reply is not received before CICS is ready to switch back again, CICS issues message DFH4584 to remind the operator that it is still waiting for a reply.

Tape journals:

For standard labeled tapes, the PAUSE parameter is unnecessary because MVS verifies that the correct volume is mounted. For unlabeled tapes, however, specify the PAUSE option, otherwise the volume at load point on the specified tape drive is automatically used for journal output.

RETRY

Output I/O errors are to be retried automatically on a new output volume before taking the action indicated by the CRUCIAL option.

RETRY indicates that if an I/O error is detected on output, journal control is to close the current data set (tape reel or disk data set), switch data sets, and try to write the block on the other data set. If the retry also fails (or if RETRY is not specified) a permanent I/O error condition will exist.

JTYPE={TAPE1|TAPE2|DISK1|DISK2|SMF}

Code this with the type of journal being defined.

TAPE1

A journal on one tape drive.

TAPE2

A journal on two tape drives.

DISK1

A journal on disk that has one data set to be reused when full.

DISK2

A journal on disk that has two data sets to be used alternately. You should specify this to minimize operator involvement during a takeover. For XRF, the system log must be on disk, and you must specify JTYPE=DISK2. Similarly, when automatic archiving has been specified, the journal must be on disk, and you must specify JTYPE=DISK2.

SMF

Journal records will be sent to SMF (system management facility) data sets, provided that the FORMAT=SMF operand is coded.

LABEL={NO|STANDARD}

Code this to indicate whether tapes with standard user labels are to be used for this journal.

NO

Unlabeled tapes will be used. If NO is specified, neither LAYOUT nor VOLCNT can be specified. If CICS is to be a participating subsystem in an IMS/ESA data-sharing environment, do *not* specify this option because it will cause CICS to abend.

STANDARD

Tapes with standard user labels will be used for this journal, and CICS will keep a record of the volumes available and used. (For CICS to be a participating subsystem in an IMS/ESA data-sharing environment, this option *must* be specified if using tape logs.)

LAYOUT={LINEAR|CYCLIC}

Code this to indicate, for LABEL=STANDARD only, whether the journal is to have linear or cyclic structure. CICS will keep a list of the volume serial numbers of the tapes to be used for the journal and will automatically request the operator to mount and demount volumes.

LINEAR

The journal will consist of a series of tape volumes. There is no limit to the number of tape volumes that can be in the journal.

CYCLIC

The journal will consist of a limited number of tape volumes which will be used in rotation, the oldest volume being overwritten when all volumes are full.

OPEN={INITIAL|DEFERRED}

Code this to indicate when the journal is to be opened.

INITIAL

The journal is to be opened for output by system initialization.

DEFERRED

The opening of the journal is to be deferred until after system initialization. This option can be used for journals opened by transactions that are executing under CICS or by programs that are specified in the program list table (PLT). This option can be used to save system and operator overhead when a journal is not needed for every CICS run.

You cannot use this option for the system log, or when the AUTOARCH option is also specified.

Note: The journal may be opened at any time in the run, by an application program.

SYSWAIT={STARTIO|ASIS}

Code this if I/O is to be initiated immediately on synchronizing requests, (PUT, (WRITE, WAIT), or WAIT) to this journal from CICS management modules. Note that this operand has no effect on user journaling

requests. For guidance on the performance implications of this operand, see the *CICS/ESA Performance Guide*.

STARTIO

I/O is to be initiated immediately on synchronizing requests from CICS management modules to the journal. This option has the same effect as STARTIO=YES coded on all such requests.

ASIS

The option coded in the STARTIO keyword in the macro request is to be honored for synchronizing requests to the journal from CICS management modules. In almost all cases, this will be STARTIO=NO. You should code SYSWAIT=ASIS only if the frequency of requests to the journal is so high that the device becomes overloaded.

VOLCNT=number

Code this to indicate, for LAYOUT=CYCLIC, the minimum number of tape volumes in the cycle.

For LAYOUT=LINEAR, "number" is the minimum number of tape volumes that are to be kept available for this journal. CICS will ask for this number of tape volumes to be identified as being for this journal before using them.

The minimum number that can be specified in either case is 2.

Note that for TAPE2 journals on labeled tapes, at least three volumes must always be supplied (to allow the overlapping of open and close operations).

This option only applies to standard labeled tapes.

DFHJCT example

Figure 45 illustrates the coding to create a JCT:

- The system log, allocated two disk data sets
- Journal 2, allocated one disk data set
- Journal 3, allocated two disk data sets
- Journal 4, allocated two tape drives
- Journal 5, allocated two tape drives.

Note that, for XRF, the system log **must** be on disk, and you must specify JTYPE=DISK2. Similarly, when automatic archiving has been specified, the journal must be on disk, and you must specify JTYPE=DISK2.

```

DFHJCT TYPE=INITIAL
*
DFHJCT TYPE=ENTRY,          SYSTEM LOG      *
      JFILEID=SYSTEM,      *
      JTYPE=DISK2,         *
      BUFSIZE=1500,        *
      JOUROPT=(RETRY,CRUCIAL,AUTOARCH), *
      ARCHJCL=DFH$ARCH
*
DFHJCT TYPE=ENTRY,          *
      JFILEID=2,           *
      JTYPE=DISK1,        *
      BUFSIZE=1500,       *
      JOUROPT=(RETRY,PAUSE)
*
DFHJCT TYPE=ENTRY,          *
      JFILEID=3,           *
      JTYPE=DISK2,        *
      JOUROPT=(RETRY,AUTOARCH), *
      ARCHJCL=DFH$ARCH,   *
      BUFSIZE=1000
*
DFHJCT TYPE=ENTRY,          (USER JOURNAL) *
      JFILEID=4,           *
      JTYPE=TAPE2,         *
      FORMAT=SMF,          *
      BUFSIZE=1500
*
DFHJCT TYPE=ENTRY,          *
      JFILEID=5,           *
      JTYPE=TAPE2,         *
      LABEL=STANDARD,     *
      LAYOUT=CYCLIC,      *
      VOLCNT=20,           *
      BUFSIZE=1500,       *
      JOUROPT=(RETRY,CRUCIAL)
*
DFHJCT TYPE=FINAL
END

```

Figure 45. Journal control table — example

Chapter 31. MCT — monitoring control table

The monitoring control table (MCT) defines the user data fields in performance class monitoring records, and describes how they are to be manipulated at the user-coded event monitoring points (EMPs). It also controls which system-defined performance class data fields are recorded. See the *CICS/ESA Performance Guide* for details of the MCT and performance.

The MCT is necessary only if you have coded user EMPs in your application programs, or if you need to exclude specific system-defined fields from being recorded. If no MCT is present the following defaults are assumed:

- All monitoring classes available (exception, performance, and SYSEVENT)
- All CICS/ESA system-defined data fields will be collected.

You code user EMPs in application programs using the EXEC CICS MONITOR command.

Elements of DFHMCT

The MCT consists of the following macro instructions:

- Control section — DFHMCT TYPE=INITIAL
- User event monitoring points — DFHMCT TYPE=EMP
- Control data recording — DFHMCT TYPE=RECORD
- End of monitoring control table — DFHMCT TYPE=FINAL (see page 249).

Control section — DFHMCT TYPE=INITIAL

The control section name for the MCT is established by the DFHMCT TYPE=INITIAL macro. This macro also creates the necessary linkage editor control statements for subsequent link-editing. The SYSEVENT class of transaction monitoring is always available.

Table 44. DFHMCT TYPE=INITIAL

	TABLE	TYPE=INITIAL [,SUFFIX=xx] [,SURROGATE=(YES NO)]
+		

? For the MCT, the suffix should be blank, two valid ? characters, or NO. For general information about TYPE=INITIAL macros, including the use of the SUFFIX # operand, see "TYPE=INITIAL" on page 248. Note that the # suffix, unless blank or NO, must be two characters for a # DFHMCT macro. Although the use of a single character # suffix does not lead to an error or message at the time of

coding the macro, single character suffixes, while acceptable # for other macros, are not acceptable to DFHMNDUP and to # other CICS components that work with the DFHMCT macros.

+ APAR PN71965

- + added the SURROGATE keyword (above), and the
- + following description of it.

+ SURROGATE=(YES|NO)

- + Indicates if the TERM field of the performance
- + monitoring record should contain the surrogate terminal
- + ID for a transaction routed task running in an AOR
- + directly connected to the TOR.

User event monitoring points — DFHMCT TYPE=EMP

The DFHMCT TYPE=EMP macro allows you to specify how the user data fields in performance class data records are to be added to or changed at each user event monitoring point. One TYPE=EMP macro must be coded for each user EMP at which user data is required.

The TYPE=EMP macro must be coded between the TYPE=INITIAL macro and the first TYPE=RECORD macro instruction.

Table 45. DFHMCT TYPE=EMP

	TABLE	TYPE=EMP ,CLASS=PERFORM ,ID={number (PP,number) entryname.number} [,CLOCK=(number,name1[,name2,...])] [,COUNT=(number,name1[,name2,...])] [,FIELD=(1,name)] [,PERFORM=(option[,...])]

TYPE=EMP

Indicates that this macro defines the user data to be collected at a user event monitoring point.

CLASS=PERFORM

Code this with the monitoring classes for which user data is to be collected at this user EMP. The value PERFORM must be coded. The corresponding PERFORM operand must also be coded.

ID={number|(PP,number)entryname.number}

Code this with the identifier of the user event monitoring point at which the user data defined in this macro is to be collected. Note that if one of the forms 'number' or (PP,number) is coded, a default entry name, "USER", is provided.

number

A decimal integer in the range 1 through 255. Identification numbers between 1 and 199 are available for user EMPs. Numbers between 200 and 255 are reserved for IBM program product EMPs and should be coded if you require to collect user data at EMPs that have been defined in the code of IBM program products.

(PP,number)

An IBM program product EMP identification number. It is equivalent to specifying an ID value of 199 + number. The value of 'number' is a decimal integer in the range 1 through 56.

entryname.number

Allows multiple use of number, a decimal integer in the range 1 through 255. Thus 'UNIQUE.3', 'DSN.3', and '3' are three different EMPs. A maximum of 98 entrynames can be specified against any particular number. Furthermore, any counts, clocks, or byte-offsets referred to by one of them will be different objects from those referred to by any other.

In the following descriptions, any reference to a constant means a hexadecimal constant of up to eight hexadecimal digits; any shorter string is padded on the left with zeros. For example, to add or subtract decimal 14, the constant would be coded as 0000000E or just E (no quotation marks are required).

Any reference to the fields DATA1 and DATA2 means the two binary fullwords supplied by the user EMP coded in the application program. These are specified by the DATA1 and DATA2 operands of the EXEC CICS MONITOR command for defining user EMPs. Depending on the options coded, the DATA1 and DATA2 fields can be interpreted as numbers, masks for performing logical operations, or pointers to further information.

Any reference to a number means a decimal integer in the range defined in the description of the option.

CLOCK=(number,name1[,name2,...])

Assigns an informal name to one or more clocks. The informal name of any clock will appear in its dictionary entry and will be available to a postprocessor for use as, for example, a column heading.

The character string, name1, will be assigned to the clock specified by number at MCT generation. If specified, name2 will be assigned to the clock number+1. Similarly, any subsequent names will be assigned to subsequent clocks. Any clock not named by this option will receive the entry name value from the ID operand (the default is USER).

Number must be in the range 1 through 256. The names specified must each be a character string up to 8 characters long. If any string contains one or more blanks or commas, it must be enclosed in quotes.

COUNT=(number,name1[,name2,...])

Assigns an informal name to one or more count fields. The informal name of any count field will appear in its dictionary entry and will be available to a postprocessor for use as, for example, a column heading.

The character string, name1, will be assigned to the count field specified by number at MCT generation. If specified, name2 will be assigned to the count field number+1. Similarly, any subsequent names will be assigned to subsequent count fields. Any count fields not named by this option will receive the entry name value from the ID operand (the default is USER).

Number must be in the range 1 through 256. The names specified must each be a character string up to eight characters long. If any string contains one or more blanks or commas, it must be enclosed in quotes.

FIELD=(1,name)

Assigns an informal name to the user byte-string field. The informal name of the user byte-string field will appear in its dictionary entry and will be available to a postprocessor for use as, for example, a column heading.

Name must be a character string up to 8 characters long. If it contains one or more blanks or commas, it must be enclosed in quotes.

PERFORM=(option[,...])

This operand must be coded when CLASS=PERFORM is coded. It specifies that information is to be added to or changed in the user fields of the performance class data record at this EMP.

The user fields for each user distinguished by a separate entry name in the ID operand can comprise:

1. Up to 256 counters
2. Up to 256 clocks, each made up of a 4-byte accumulator and 4-byte count
3. A byte string of up to 8192 bytes.

Note: The combined sizes of the objects (clocks, counts, and fields) implied in the specified options must not exceed 16384 bytes. If they do, assembly-time errors will occur. You can avoid this by using fewer objects, either by collecting less data, or by clustering references to clocks and counts to avoid implied, but unused, objects.

EMP warning

Defining user event monitoring point data extends the size of all CICS performance class monitoring records. Each CICS monitoring record will be the same size as the largest record; you should bear this in mind when specifying user data fields.

Actions will be performed on the user fields according to the options specified. PERFORM can be abbreviated to PER.

Valid options for the PERFORM operand are:

ADD CNT(number, {constant|DATA1|DATA2})

The value of the user count field specified by number is to be incremented by the specified constant or by the value of the field DATA1 or DATA2. Number is a decimal integer in the range 1 through 256.

SUBCNT(number, {constant|DATA1|DATA2})

The value of the user count field specified by number is to be decremented by the specified constant or by the value of the field DATA1 or DATA2. Number is a decimal integer in the range 1 through 256.

NACNT(number, {constant|DATA1|DATA2})

A logical AND operation is to be performed on the value of the user count field specified by number, using the specified constant or the value of the field DATA1 or DATA2. Number is a decimal integer in the range 1 through 256.

EXCNT(number, {constant|DATA1|DATA2})

A logical exclusive OR operation is to be performed on the value of the user count field specified by number, using the specified constant or the value of the field DATA1 or DATA2. Number is a decimal integer in the range 1 through 256.

ORCNT(number, {constant|DATA1|DATA2})

A logical inclusive OR operation is to be performed on the value of the user count field specified by number, using the specified constant or the value of the field DATA1 or DATA2. Number is a decimal integer in the range 1 through 256.

MLTCNT(number1, number2)

A series of adjacent user count fields are to be updated by adding the values contained in adjacent fullwords in an area addressed by the DATA1 field. To use this option, both the DATA1 and DATA2 fields must be passed from the user EMP.

The user count fields that are to be updated start at the field specified by number1. The number of user count fields that are updated is the smaller of the values of number2 and the DATA2 field. If the DATA2 field is zero, the value of number2 will be used. The series of adjacent fullwords used to add into the user count fields starts at the address specified in the DATA1 field. Successive fullwords will be added into successive user count fields.

Number1 and number2 are decimal integers in the range 1 through 256. The number of user counts generated is (number1 + number2 - 1). This value must also be in the range 1 through 256.

Note: Only one of the MLTCNT and MOVE options can be used in each DFHMCT TYPE=EMP macro.

SCLOCK(number)

The clock specified by number is to be started. The value of the 4-byte count in the user clock field will be incremented by 1 and flagged to show its running state. Number is a decimal integer in the range 1 through 256.

PCLOCK(number)

The clock specified by number is to be stopped. The 4-byte count in the user clock field will be flagged to indicate that the clock is now stopped. The accumulator is set to the sum of its contents before the previous SCLOCK and the elapsed period between that SCLOCK and this PCLOCK. Number is a decimal integer in the range 1 through 256.

SCPUCLK(number)

This option performs the same function as SCLOCK, but uses the CPU-time of the CICS main task instead of elapsed time.

PCPUCLK(number)

This option performs the same function as PCLOCK, but uses the CPU-time of the CICS main task instead of elapsed time.

MOVE(number3, number4)

A string of data is to be moved into the user byte-string field. To use this option, both the DATA1 and DATA2 fields must be passed from the user EMP.

The user byte-string field is updated starting at the offset specified by number3. The data to be moved starts at the address supplied in the DATA1 field. The maximum length of data that can be moved is given by number4 (in bytes), and the actual length of data that is to be moved is given by the value of the DATA2 field. If the value of DATA2 is zero, then the length of the data given by number4 is moved.

Number3 is a decimal integer in the range 0 to 8191, and number4 is a decimal integer in the range 1 to 8192. The maximum length of the user character field is (number3 + number4), and must be in the range 1 to 8192.

Note: Only one of the MLTCNT and MOVE options can be used in each DFHMCT TYPE=EMP macro instruction.

DELIVER

Performance class data accumulated for this task up to this point is delivered to the monitoring buffers. Any running clocks are stopped. The performance class section of the monitoring area for this task is reset to X'00', except for the key fields (transid, termid). Any clocks that were stopped by this option are restarted from zero for the new measurement period. The "high-water-mark" fields are reset to their current values.

Control data recording — DFHMCT TYPE=RECORD

The DFHMCT TYPE=RECORD macro identifies the performance class data fields which have been selected for monitoring.

Table 46. DFHMCT TYPE=RECORD

		,CLASS=PERFORM [,EXCLUDE={ALL (n1[,...])}] [,INCLUDE=(m1[,...])]
--	--	--

TYPE=RECORD

Indicates that monitoring data for selected performance class data fields will be recorded.

CLASS=PERFORM

This operand is necessary if selectivity of performance class data fields is in use.

PERFORM

Performance class data is to be recorded.
PERFORM can be abbreviated to PER.

EXCLUDE={ALL|(n1[,...])}

Code this to prevent one or more CICS fields from being reported by the monitoring facility. By default, all documented performance class fields are reported.

The EXCLUDE operand will always be honored before the INCLUDE operand, regardless of the order in which they are coded. (The INCLUDE operand is only relevant when the EXCLUDE operand is coded.)

ALL

This prevents all fields that are eligible for exclusion from being reported. Note that the following fields cannot be excluded:

1, 2, 4, 5, 6 and 89.

You can use the INCLUDE operand at the same time as EXCLUDE=ALL if you want to include some fields but exclude the majority.

Table 47 shows the fields that are eligible for exclusion. Each field has a group name associated with it, which identifies the group of fields to which it belongs. Each field also has its own numeric field identifier.

To exclude a group of fields you code the name of the group (a character string) as n1, for example, EXCLUDE=(DFHTASK).

To exclude a single field you code the numeric identifier of the field as n1, for example, EXCLUDE=(98,70). *Do not code leading zeros on numeric identifiers. Do not code numeric identifiers of fields that are ineligible for exclusion.*

You can code combinations of names and numeric identifiers, for example, EXCLUDE=(DFHFILE,DFHTERM,112,64).

Table 47 (Page 1 of 2). This table shows the data groups and fields that can be excluded/included

Group Name	Field Id	Description
DFHCICS	103	Transaction exception wait time
DFHCICS	112	Performance record type
DFHCICS	130	Transaction routing sysid
DFHDEST	41	TD get count
DFHDEST	42	TD put count
DFHDEST	43	TD purge count
DFHDEST	91	TD total count
DFHDEST	101	TD I/O wait time
DFHFEPI	150	FEPI Allocate count
DFHFEPI	151	FEPI Receive count
DFHFEPI	152	FEPI Send count
DFHFEPI	153	FEPI Start count
DFHFEPI	154	FEPI CHARS sent
DFHFEPI	155	FEPI CHARS received
DFHFEPI	156	FEPI Suspend time
DFHFEPI	157	FEPI Allocate time-out count
DFHFEPI	158	FEPI Receive time-out count
DFHFEPI	159	FEPI Total count
DFHFILE	36	FC get count
DFHFILE	37	FC put count
DFHFILE	38	FC browse count
DFHFILE	39	FC add count
DFHFILE	40	FC delete count
DFHFILE	63	FC I/O wait time
DFHFILE	70	FC access-method count
DFHFILE	93	FC total count
DFHJOUR	10	JC I/O wait time
DFHJOUR	58	JC put/write count
DFHMAPP	50	BMS MAP count
DFHMAPP	51	BMS IN count
DFHMAPP	52	BMS OUT count
DFHMAPP	90	BMS total count
DFHPROG	55	Program LINK count
DFHPROG	56	Program XCTL count
DFHPROG	57	Program LOAD count
DFHPROG	71	Program name
DFHPROG	113	Original abend code
DFHPROG	114	Current abend code
DFHPROG	115	Program load time
DFHSTOR	33	User-storage high-water-mark (UDSA)
DFHSTOR	54	User-storage get-count (UDSA)
DFHSTOR	87	Program-storage high-water-mark - total
DFHSTOR	95	User-storage-occupancy (bytes-ms) (UDSA)
DFHSTOR	105	User-storage get-count—above 16MB line (EUDSA)
DFHSTOR	106	User-storage high-water-mark—above 16MB line (EUDSA)
DFHSTOR	107	User-storage-occupancy (bytes-ms)—above 16MB line (EUDSA)

Table 47 (Page 2 of 2). This table shows the data groups and fields that can be excluded/included

Group Name	Field Id	Description
DFHSTOR	108	Program-storage high-water-mark-below 16MB line
DFHSTOR	116	User-storage high-water-mark-below 16MB line (CDSA)
DFHSTOR	117	User-storage get-count-below 16MB line (CDSA)
DFHSTOR	118	User-storage-occupancy (bytes-ms)-below 16MB line (CDSA)
DFHSTOR	119	User-storage high-water-mark-above 16MB line (ECDSA)
DFHSTOR	120	User-storage get-count-above 16MB line (ECDSA)
DFHSTOR	121	User-storage-occupancy (bytes-ms)-above 16MB line (ECDSA)
DFHSTOR	122	Program-storage high-water-mark (ERDSA)
DFHSTOR	139	Program-storage high-water-mark-above 16MB line (ECDSA)
DFHSTOR	142	Program-storage high-water-mark (ECDSA)
DFHSTOR	143	Program-storage high-water-mark (CDSA)
DFHSTOR	160	Program-storage high-water-mark (SDSA)
DFHSTOR	161	Program-storage high-water-mark (ESDSA)
DFHSTOR	162	Program-storage high-water-mark (RDSA)
DFHSYNC	60	Sync point count
DFHTASK	7	User-task dispatch time
DFHTASK	8	User-task CPU time
DFHTASK	14	User-task suspend time
DFHTASK	31	Task number
DFHTASK	59	IC put/initiate count
DFHTASK	64	Error flag field
DFHTASK	97	Network name of the originating terminal or system
DFHTASK	98	Unit-of-work id on the originating system
DFHTASK	102	User-task wait-for-dispatch time
DFHTASK	109	Transaction priority
DFHTASK	125	First dispatch delay time
DFHTASK	126	First dispatch delay time due to TRANCLASS
DFHTASK	127	First dispatch delay due to MXT
DFHTASK	129	Task ENQ delay time
DFHTASK	170	Resource Manager Interface-elapsed time
DFHTASK	171	Resource Manager Interface-suspended time
DFHTEMP	11	TS I/O wait time
DFHTEMP	44	TS get count
DFHTEMP	46	TS put auxiliary count
DFHTEMP	47	TS put main count
DFHTEMP	92	TS total count
DFHTERM	9	TC I/O wait time

Table 47 (Page 2 of 2). This table shows the data groups and fields that can be excluded/included

Group Name	Field Id	Description
DFHTERM	34	TC principal facility input messages
DFHTERM	35	TC principal facility output messages
DFHTERM	67	TC alternate facility input messages
DFHTERM	68	TC alternate facility output messages
DFHTERM	69	TC allocate count
DFHTERM	83	TC principal facility CHARS input
DFHTERM	84	TC principal facility CHARS output
DFHTERM	85	TC alternate facility CHARS input
DFHTERM	86	TC alternate facility CHARS output
DFHTERM	100	IR I/O wait time
DFHTERM	111	VTAM terminal LU name
DFHTERM	133	TC I/O wait time - LU6.1
DFHTERM	134	TC I/O wait time - LU6.2
DFHTERM	135	TC alternate facility input messages - LU6.2
DFHTERM	136	TC alternate facility output messages - LU6.2
DFHTERM	137	TC alternate facility CHARS input - LU6.2
DFHTERM	138	TC alternate facility CHARS output - LU6.2

INCLUDE=(m1[,...])

Code this to enable one or more CICS fields to be reported by the monitoring facility. By default, all documented performance class fields are reported, so this operand is relevant only if you code the EXCLUDE operand in the same macro.

The fields that are eligible to be coded for inclusion on this operand are the same as those that are eligible for exclusion. (See the description of the EXCLUDE operand.) Each field has a numeric field identifier associated with it. To include a field you code m1 as the numeric identifier of the field. You can code multiple numeric identifiers. *Do not code leading zeros on numeric identifiers. Do not code numeric identifiers of fields that are ineligible for exclusion, and that are therefore included by default.*

The EXCLUDE operand is always honored first. The INCLUDE operand, if coded, then overrides some of its effects. For example, coding:

```
EXCLUDE=DFHFILE,
INCLUDE=(37,93),
```

would secure the collection and reporting of file control PUTs and the total number of file control requests, while file control browse count and other file control fields would be excluded.

If you want to exclude the majority of fields, but include a few, you can code, for example:

```
EXCLUDE=ALL,
INCLUDE=(DFHTERM,97,98)
```

This is more convenient than coding individually all the fields you want to exclude.

DFHMCT example

| Three sample monitoring control tables are provided in
| CICS410.SDFHSAMP:

- | • For terminal-owning region (TOR) - DFHMCTT\$
- | • For application-owning region (AOR) - DFHMCTA\$
- | • For file-owning region (FOR) - DFHMCTF\$

| These samples show how to use the EXCLUDE and
| INCLUDE operands to reduce the size of the performance
| class record in order to reduce the volume of data written by
| CICS to SMF.

DFHMCT example

Figure 46 illustrates the coding to create a monitoring control table (MCT) for two user event monitoring points (EMPs).

```
DFHMCT TYPE=INITIAL
DFHMCT TYPE=EMP,
        ID=180,
        CLASS=PERFORM,
        PERFORM=(SCLOCK(1),ADDcnt(2,1)),
        ACCOUNT=ADDcnt(1,1)
DFHMCT TYPE=EMP,
        ID=181,
        CLASS=PERFORM,
        PERFORM=PCLOCK(1)
DFHMCT TYPE=FINAL
        END
```

*
*
*
*
*
*
*

Figure 46. Monitoring control table — example

Chapter 32. PLT — program list table

A program list table (PLT) contains a list of related programs. You may wish to generate several PLTs for one or more of the following reasons:

- To specify a list of programs that you wish to be executed in the second and/or third initialization stages of CICS startup. For more detail about the initialization stages, see the *CICS/ESA Recovery and Restart Guide*. For programming information about restrictions on using programs in the initialization stages, see the *CICS/ESA Customization Guide*. The selected list should be specified at initialization time by the PLTPI=xx system initialization parameter, where xx is the suffix of the PLT that contains the required list of programs.

For convenience, the list of programs selected for execution during initialization is referred to as the 'PLTPI' list.

- To specify a list of programs that you wish to be executed during the first and/or second quiesce stages of controlled shutdown. The selected list should be specified at initialization time by the PLTSD=xx system initialization parameter, where xx is the suffix of the PLT that contains the required list of programs.

The PLT specified in the PLTSD system initialization parameter can be overridden at shutdown time by the PLT option in the CEMT PERFORM SHUTDOWN command.

The shutdown PLT is normally loaded as CICS is being shutdown. However it is possible to use the same PLT for both initialization and shutdown, and under these circumstances the PLT will be loaded during initialization and CICS will not need to reload it during shutdown. It should be noted that if this is the case and the PLT is updated whilst CICS is operational, then a CEMT SET PROGRAM NEWCOPY command must be issued for the PLT to ensure that the updated version is used when CICS is shutdown.

For convenience, the list of programs selected for execution during shutdown is referred to as the 'PLTSD' list.

- To specify a list of programs that you wish to have enabled or disabled as a group by a master terminal ENABLE or DISABLE command. This use of PLTs means that a master terminal operator can enable or disable a set of programs with just one command, instead of using a separate command for each program.

Any number of PLTs can be generated for the above purposes, provided that:

1. Each PLT has a unique suffix
 2. Each program named in a PLT either has a program resource definition entry in the CSD, or is capable of being autoinstalled (that is, the appropriate SIT parameters have been specified for program autoinstall).
- ? **Note:** PLTs should not be defined as programs in the CSD.
?

PLTs must be placed in the DFHRPL library. However, CICS scans the LPA for phase 1 PLTPI programs if they are not already installed.

Elements of DFHPLT

The following macros are available to define the PLT entries:

- Control section — DFHPLT TYPE=INITIAL
- Entries in program list table — DFHPLT TYPE=ENTRY
- End of Program List Table — DFHPLT TYPE=FINAL. (See page 249.)

Control section — DFHPLT TYPE=INITIAL

Table 48. DFHPLT TYPE=INITIAL

	DFHPLT	TYPE=INITIAL [,SUFFIX=xx]
--	--------	------------------------------

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

TYPE=INITIAL

Generates the PLT control section.

Note that a program entry for each PLT generated must have a definition in the CSD.

SUFFIX=xx

Code this with the suffix character(s) that uniquely identify this particular table.

Note: The PLT suffix is referenced by:

- CEMT {INQUIRE|SET} PROGRAM CLASS(suffix)
- CEMT or EXEC CICS PERFORM SHUTDOWN PLT(suffix)
- system initialization parameters PLTPI and PLTSD keywords.

Entries in program list table — DFHPLT TYPE=ENTRY

Entries are specified in the PLT as follows:

	DFHPLT	TYPE=ENTRY ,PROGRAM=(program [,program,...])

TYPE=ENTRY

Indicates that one or more program names are to be listed in this table.

Note: As shown below, a TYPE=ENTRY macro is also needed to specify the PROGRAM=DFHDELIM entry.

PROGRAM=program name

Code this with a program name of up to eight characters. Each program must either have a definition in the CSD or must be capable of being autoinstalled (that is, the appropriate SIT parameters must be specified for program autoinstall). Undefined programs before the DFHDELIM statement are system autoinstalled.

For PLTPI and PLTSD lists, only **initial** programs should be named: other programs that are linked to by initial programs should not be listed (but must be defined or be capable of being autoinstalled). For programming information about restrictions on using PLT programs during initialization, see the *CICS/ESA Customization Guide*.

	DFHPLT	TYPE=ENTRY ,PROGRAM=DFHDELIM

PROGRAM=DFHDELIM

Code this to delimit the programs to run in the first or second passes of PLTPI or PLTSD.

Note that:

- Programs listed **before** the PROGRAM=DFHDELIM entry in a PLTPI are executed during the **second**

stage of initialization. These are to enable user exit programs needed during recovery. The user exit programs should be defined in the CSD. Failure to do so will cause difficulty in accessing the programs after CICS initialization is complete, for example in EXEC CICS DISABLE commands. However, note that the properties defined by RDO have no effect during the second stage of initialization.

- Programs listed **after** the PROGRAM=DFHDELIM entry in a PLTPI are executed during the **third** stage of initialization. If these programs are used to enable user exits, the user exit programs **must** also be defined in the CSD or must be capable of being autoinstalled.
- Programs listed **before** the PROGRAM=DFHDELIM entry in a PLTSD are executed during the **first** quiesce stage of shutdown.
- Programs listed **after** the PROGRAM=DFHDELIM entry in a PLTSD are executed during the **second** quiesce stage of shutdown.

Note: The DFHDELIM entry is not a program – it serves as a delimiter only.

Second stage initialization and second stage quiesce PLT programs do not require program resource definitions. If they are not defined they will be system autoinstalled (irrespective of the program autoinstall SIT options). This means that the autoinstall exit is not called to allow the definition to be modified. The programs are defined with the following attributes:

LANGUAGE(ASSEMBLER) STATUS(ENABLED) CEDF(NO)
 DATALLOCATION(BELOW) EXECKEY(CICS)
 EXECUTIONSET(FULLAPI)

Third stage initialization and first stage quiesce PLT programs can be defined using program autoinstall, depending upon the program autoinstall SIT options. See Chapter 11, "Autoinstall for programs, mapsets, and partitionsets" on page 123 for further information. If program autoinstall is not used, they must have program resource definitions in the CSD.

DFHPLT example

Figure 47 and Figure 48 illustrate the coding required to generate a PLT.

```

*
* LIST OF PROGRAMS TO BE EXECUTED SEQUENTIALLY DURING SYSTEM
* INITIALIZATION.
* REQUIRED SYSTEM INITIALIZATION PARAMETER: PLTPI=I1
*
  DFHPLT TYPE=INITIAL,          (DFHPLTI1 SHOULD ALSO BE DEFINED
    SUFFIX=I1                   BY RDO)
*
* The following programs are run in the first pass of PLTPI
*
  DFHPLT TYPE=ENTRY,PROGRAM=TRAQA EXECUTED DURING 2ND INIT. PHASE
  DFHPLT TYPE=ENTRY,PROGRAM=TRAQB (PROGRAMS SHOULD ALSO BE DEFINED
  DFHPLT TYPE=ENTRY,PROGRAM=TRAQC BY RDO)
*
  DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
*
*
* The following programs are run in the second pass of PLTPI
*
  DFHPLT TYPE=ENTRY,PROGRAM=TRASA EXECUTED DURING 3RD INIT. PHASE
  DFHPLT TYPE=ENTRY,PROGRAM=TRASB (PROGRAMS MUST ALSO BE DEFINED
  DFHPLT TYPE=ENTRY,PROGRAM=TRASC BY RDO)
  DFHPLT TYPE=FINAL
*
  END

```

Figure 47. PLTPI program list table — example

```

*
*
* LIST OF PROGRAMS TO BE EXECUTED SEQUENTIALLY DURING SYSTEM
* TERMINATION
* REQUIRED SYSTEM INITIALIZATION PARAMETER: PLTSD=T1
*
  DFHPLT TYPE=INITIAL,          (DFHPLTT1 MUST ALSO BE DEFINED
    SUFFIX=T1                   BY RDO)
*
* The following programs are run in the 1st pass of PLTSD
*
*
  DFHPLT TYPE=ENTRY,PROGRAM=TRARA EXECUTED DURING 1st QUIESCE PHASE
  DFHPLT TYPE=ENTRY,PROGRAM=TRARB (PROGRAMS MUST ALSO BE DEFINED
  DFHPLT TYPE=ENTRY,PROGRAM=TRARC BY RDO)
*
  DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
*
*
* The following programs are run in the 2nd pass of PLTSD
*
  DFHPLT TYPE=ENTRY,PROGRAM=TRAFA EXECUTED DURING 2nd QUIESCE PHASE
  DFHPLT TYPE=ENTRY,PROGRAM=TRAFB (PROGRAMS MUST ALSO BE DEFINED
  DFHPLT TYPE=ENTRY,PROGRAM=TRAFB BY RDO)
  DFHPLT TYPE=FINAL
*
  END

```

Figure 48. PLTSD program list table — example

Chapter 33. RST — recoverable service table

Note on terminology: In this chapter and in other IBM publications you will see the term DBCTL. This term refers to the CICS-IMS/ESA DBCTL (database control) interface.

The recoverable service table (RST) is used for CICS DBCTL XRF (extended recovery facility) support. It contains a description of the DBCTL configuration. On detection of a DBCTL failure, the **active** CICS uses the RST together with the MVS subsystem VERIFY to determine the existence of a suitable alternative DBCTL subsystem. The **alternate** CICS uses the RST to check for the presence of a DBCTL subsystem. For security reasons, the RST should be link-edited into a library authorized using APF. The RST is not loaded as part of the CICS nucleus.

DFHRST macro types

The following macros may be coded in a recoverable service table:

- DFHRST TYPE=INITIAL establishes the control section.
- DFHRST TYPE=RSE specifies the start of a recoverable service element (RSE). An RSE consists of a (non-empty) set of identifiers of equivalent DBCTL subsystems, the CICS applids associated with these DBCTL subsystems, and the application identifiers of the CICS systems which will use the DBCTL subsystems.
- DFHRST TYPE=SUBSYS specifies one of the DBCTL subsystems in an RSE.
- DFHRST TYPE=FINAL concludes the RST (see page 249).

Note: The RST to be used by the system must be defined in the system initialization table by the following SIT parameter:

	DFHSIT	...
		[,RST=]NO[xx]

Control section — DFHRST TYPE=INITIAL

The DFHRST TYPE=INITIAL macro establishes the recoverable service table control section.

	DFHRST	TYPE=INITIAL
		[,SUFFIX=xx]

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

Recoverable service elements — DFHRST TYPE=RSE

The DFHRST TYPE=RSE macro contains information about a recoverable service element and the CICS systems expected to connect to the DBCTL subsystems within the recoverable service element.

	DFHRST	TYPE=RSE
		[,CTLAPPLS=(applid1,applid2,...)]

TYPE=RSE

Indicates the start of a set of identifiers of equivalent DBCTL systems. An equivalent DBCTL subsystem must have the same database (DB) name.

CTLAPPLS=(applid1,applid2,...)

Specifies the application identifiers of the CICS systems that are authorized to restart the DBCTL subsystems in the RSE.

DBCTL subsystems — DFHRST TYPE=SUBSYS

The DFHRST TYPE=SUBSYS macro contains information about a specific DBCTL subsystem within an RSE. For each subsystem in an RSE there must be a DFHRST TYPE=SUBSYS macro.

	DFHRST	TYPE=SUBSYS
		,SYBSYSID=subsystem-identifier [,JOBNAME=(jobname1,jobname2,...)]

TYPE=SUBSYS

Defines one of the DBCTL subsystems in an RSE.

SUBSYSID=subsystem-identifier

Code this with the four-character name of the DBCTL subsystem-identifier. This identifier must be unique within the RST.

JOBNAME=(jobname1,jobname2,...)

Code this with the **MVS JOBNAME(S)** associated with the DBCTL subsystem identified in the SUBSYSID parameter statement. This is a form of security check. If it is necessary for CICS to cancel the DBCTL subsystem it is only authorized to do so if the appropriate MVS jobname associated with the DBCTL subsystem is one of those specified by the **JOBNAME** parameter.

DFHRST example

DFHRST example

Figure 49 illustrates the coding required to generate an RST.

```
DFHRST  TYPE=INITIAL,  
        SUFFIX=K1  
DFHRST  TYPE=RSE  
        ,CTLAPPLS=(applid1,applid2,applid3)  
DFHRST  TYPE=SUBSYS  
        ,SUBSYSID=CTL1  
        ,JOBNAME=(job1,job2,job3,job4)  
DFHRST  TYPE=SUBSYS  
        ,SUBSYSID=CTL2  
        ,JOBNAME=(job5,job6,job7,job8)  
DFHRST  TYPE=FINAL  
END
```

Figure 49. Recoverable service table — example

Chapter 34. SRT — system recovery table

The system recovery table (SRT) contains a list of abend codes that are intercepted. After a listed code is intercepted, CICS attempts to remain operational by causing the offending task to abend.

You can modify the default recovery action by writing your own recovery program. You do this by means of the XSRAB global user exit point within the System Recovery Program (SRP). (See the *CICS/ESA Customization Guide* for programming information about the XSRAB exit.)

Note that recovery is only attempted if a user task (**not** a system task) is in control at the time the abend occurs.

DFHSRT macro types

The following macros may be coded in a system recovery table:

- DFHSRT TYPE=INITIAL establishes the control section
- DFHSRT TYPE=SYSTEM|USER specifies the abend codes that are to be handled
- DFHSRT TYPE=FINAL concludes the SRT (see page 249).

Control section — DFHSRT TYPE=INITIAL

The DFHSRT TYPE=INITIAL macro generates the system recovery table control section.

DFHSRT	TYPE=INITIAL [,SUFFIX=xx]

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see "TYPE=INITIAL" on page 248.

Abend codes — DFHSRT TYPE=SYSTEM|USER

DFHSRT	TYPE=(SYSTEM USER) ,ABCODE=(abend-code,...) [,RECOVER={YES NO}]

TYPE={SYSTEM|USER}

Indicates the type of abend code to be intercepted.

SYSTEM

The abend code is an operating system abend code. This corresponds to an MVS Sxxx abend code.

USER

The abend code is a user (including CICS) abend code. This corresponds to an MVS Unnnn abend code.

ABCODE=(abend-code,...)

Code this with the abend code (or codes) to be intercepted.

If you code TYPE=SYSTEM, then this abend code must be three hexadecimal digits (xxx) representing the MVS system abend code Sxxx.

If you code TYPE=USER, then this abend code must be a decimal number (nnnn) representing the user part of the MVS abend code Unnnn. This is usually the same number as the CICS message that is issued before CICS tries to terminate abnormally. (See the *CICS/ESA Messages and Codes* manual.)

RECOVER={YES|NO}

This keyword is optional. If you omit **RECOVER**, the effect is the same as if you had coded **RECOVER=NO** (that is, the specified codes are removed from the SRT).

YES

Code this to add the specified codes to the SRT.

NO

Code this to remove the specified codes from the SRT.

Notes:

1. The following abend codes are intercepted automatically and recovery will be attempted.

001,002,013,020,025,026,030,032,033,034,035,
036,037,03A,03B,03D,100,113,137,213,214,237,
283,285,313,314,337,400,413,437,513,514,613,
614,637,713,714,737,813,837,913,A13,A14,B13,
B14,B37,D23,D37,E37

In addition, abend code 0F3 will be intercepted and recovery attempted. This abend code covers various machine check conditions. It also covers the Alternate Processor Retry condition that can occur only when running on a multiprocessor. CICS-supplied recovery code will attempt to recover from instruction-failure machine checks on the assumption that they are not permanent. It will also attempt to recover from Alternate Processor Retry conditions.

DFHSRT example

To obtain recovery for the standard abend codes above, the system recovery table (SRT) can be coded simply as follows:

```
DFHSRT TYPE=INITIAL
DFHSRT TYPE=FINAL
END
```

There is no need to list the standard codes individually.

2. If you want CICS to handle other errors, you can code the SRT as follows:

```
DFHSRT TYPE=INITIAL
DFHSRT TYPE=SYSTEM,or USER,
        ABCODE=(user or system codes),
        RECOVER=YES
DFHSRT TYPE=FINAL
END
```

3. If you do not want CICS to attempt recovery after one or more of the above standard abend codes occurs, specify the code(s) with **RECOVER=NO**, or without the **RECOVER** parameter.
4. Recovery is attempted only if a user task (**not** a system task) is in control at the time the abend occurs.

DFHSRT example

Figure 50 illustrates the coding required to generate a SRT.

```
DFHSRT TYPE=INITIAL,          *
        SUFFIX=K1
DFHSRT TYPE=SYSTEM,          *
        ABCODE=777,          *
        RECOVER=YES
DFHSRT TYPE=USER,            *
        ABCODE=(888,999),    *
        RECOVER=YES
DFHSRT TYPE=USER,            *
        ABCODE=020
DFHSRT TYPE=FINAL
END
```

Figure 50. System recovery table — example

Chapter 35. TCT — terminal control table

This chapter describes the terminal control table and the macro instructions you use to create it.

All VTAM connected terminals, intersystem communication links, and multiregion operation links must be defined using **resource definition online (RDO)**. See “Devices supported” on page 221 for details of VTAM terminals supported by RDO.

How this chapter is organized

This chapter begins with an introduction to the TCT and its macros, followed by the reference information for the TYPE=INITIAL and TYPE=GROUP macros. The chapter is then divided into sections, according to the telecommunications access method.

The sections are as follows:

- Logical device codes ... **page 302**
- Sequential devices ... **page 306**
- TCAM DCB interface ... **page 309**
- Remote terminals for transaction routing ... **page 321**

Introduction to the TCT

A CICS system can communicate with terminals, sequential devices, logical units, and other systems. The TCT defines each of the devices in the configuration. Each TCT entry defines the optional and variable features of the device to CICS, and specifies what optional and variable features of CICS are to be used.

CICS uses a **telecommunication access method** to communicate with terminals. This may be **VTAM** or **TCAM**; you may use one or both of these in your system. Note that logical units supporting logical device codes require DFHTCT macros, even though they are VTAM devices.

A **terminal** can be a telecommunication device, for example, an IBM 3279 Color Display Station, or a subsystem, for example an IBM 4700 Finance Communication System. Terminals can be local (channel-attached) or remote (link-attached).

You can use a **sequential device** to simulate a CICS terminal. You can define a card reader or punch, line printer, direct access storage device (disk drive), or magnetic tape drive as a sequential device.

A **logical unit (LU)** is a port through which a user of an SNA network gains access to the network facilities.

A **system** can be, for example, another CICS system, an IBM 8815 Scanmaster, an IBM Displaywriter, or an APPC/PC. Intercommunication with CICS systems can be:

- Between different processors (**intersystem communication** or **ISC**), using the LUTYPE 6.1 or LUTYPE 6.2 protocols, or using an intermediate system as an **indirect link**. (Intercommunication with non-CICS systems also uses ISC.)
- Within the same processor (**multiregion operation** or **MRO**), using interregion communication (IRC). (You can also use LUTYPE 6.2 ISC within the same processor.)

There are three ways in which you can create TCT entries, and install them in the TCT.

1. By coding **DFHTCT macros**, assembling them, and selecting the resulting TCT by using the TCT operand at system initialization. This way, TCT entries can be installed at system initialization only.
2. By using **resource definition online (RDO)** to create resource definitions in the CICS system definition (CSD) file. However, you **must** use RDO for VTAM-connected terminals (local and remote) and for MRO and ISC links and sessions. See Chapter 21, “TERMINAL” on page 183 for more information on defining terminals using RDO.
3. By using RDO and **autoinstall** (for VTAM-connected terminals only). See Chapter 9, “Autoinstall for VTAM terminals” on page 105 for information about autoinstall for terminals.

If you have terminals connected by other access methods, you **must** use DFHTCT macros, which are described here.

DFHTCT macro types

The DFHTCT macros you code depend on the device you are defining, and on the access method you are using. You always start with one of these:

DFHTCT TYPE=INITIAL,... (See page 300.)

There is a special macro to use when you assemble the TCT to migrate RDO-eligible definitions to the CSD:

DFHTCT TYPE=GROUP,... (See page 301.)

DFHTCT TYPE=INITIAL

You can define your devices in any order you want. Each device needs one or more macros, and these sometimes have to be in a particular order. We tell you when this is the case. The macros you need for each type of device or system are as follows:

VTAM terminals, MVS consoles.	For guidance, see Chapter 21, "TERMINAL" on page 183.
Logical device codes.	DFHTCT TYPE=LDC,... DFHTCT TYPE=LDCLIST,...
TCAM terminals, sequential devices.	DFHTCT TYPE=SDSCI,... DFHTCT TYPE=LINE,... DFHTCT TYPE=TERMINAL,...
Remote terminals, for transaction routing.	DFHTCT TYPE=REMOTE,... or: DFHTCT TYPE=REGION,... DFHTCT TYPE=TERMINAL,...

At the very end of your macros you code:

```
DFHTCT TYPE=FINAL
END (This macro is described on page 249.)
```

Note on SYSIDNT and TRMIDNT operands: CICS accepts both uppercase and lowercase characters for SYSIDNT and TRMIDNT, but the lowercase characters are not checked for duplication. Assembling a TCT containing lowercase SYSIDNT or TRMIDNT will result in an MNOTE.

If you want duplicate checking, you should use only uppercase characters for SYSIDNT and TRMIDNT.

Note on assembling the TCT: The assembly and link-edit of a TCT will lead to the creation of two separate load modules. Assembly of a suffixed TCT (source name DFHTCTxx) produces a single text file. However, when this is link-edited into a load library, two members are created:

1. DFHTCTxx, which contains the non-RDO-eligible definitions in control block format
2. DFHRDTxx, which contains the RDO-eligible (VTAM terminal and system) definitions in RDO command format.

This will happen, *whether or not you intend to use RDO*. You need to be aware of the existence of these two tables if you have to copy or move assembled TCT tables between load libraries.

If you reassemble the TCT after starting CICS, any changes will be picked up at a warm or emergency start.

Control section — DFHTCT TYPE=INITIAL

You code one DFHTCT TYPE=INITIAL macro before all the macros that define your resources. The TYPE=INITIAL macro has two purposes:

1. To establish the area of storage into which the TCT is assembled
2. To specify information that applies to the whole TCT, or to the individual non-VTAM entries (and any VTAM-LDC definitions).

DFHTCT	TYPE=INITIAL [,ACCMETH={([VTAM,]NONVTAM)} [,ERRATT={NO}([LASTLINE] [,INTENSIFY] [,BLUE RED PINK GREEN TURQUOISE YELLOW NEUTRAL}} [,BLINK REVERSE UNDERLINE}})] [,MIGRATE={YES COMPLETE} [,SUFFIX=xx]
	<u>Intercommunication Only:</u> [,SYSIDNT=name]

Note: For general information about TYPE=INITIAL macros, see "TYPE=INITIAL" on page 248.

ACCMETH={([VTAM,]NONVTAM)}

This specifies the access methods required in the running CICS system.

VTAM

Specify this if you are using VTAM as an access method, even though you must define VTAM devices with the CEDA transaction.

NONVTAM

Specify this if you are using telecommunications access methods other than VTAM. These access methods include BSAM (for sequential devices) and TCAM.

Note: The default is to assume *both* VTAM and NONVTAM.

ERRATT={NO}([LASTLINE],[INTENSIFY]

[,{BLUE|RED|PINK|GREEN|TURQUOISE|YELLOW|NEUTRAL}}[,BLINK|REVERSE|UNDERLINE}})]

Indicates the way error messages are displayed on all 3270 display screens. You can either leave it to default to NO, or specify any combination of LASTLINE, INTENSIFY, one of the colors, and one of the highlights. Specifying INTENSIFY, one of the colors, or one of the highlights forces LASTLINE.

Any attributes that are not valid for a particular device will be ignored.

NO

Any error message is displayed at the current cursor position and without any additional attributes.

LASTLINE

Any error message is displayed starting at the beginning of the line nearest the bottom of the screen, such that the message will fit on the screen.

Warning: If messages are received in quick succession, they will overlay one another. The earlier messages may disappear before the operator has read them.

INTENSIFY

Error messages are intensified, and are placed on the last line of the screen.

BLUE|RED|PINK|GREEN|TURQUOISE|YELLOW|NEUTRAL

Error messages are shown in the color specified, and on the last line of the screen.

BLINK|REVERSE|UNDERLINE

Error messages are highlighted, and on the last line of the screen.

MIGRATE={YES|COMPLETE}

This operand controls the building of TCT entries for VTAM devices that are **eligible** for resource definition online (RDO). The only way RDO-eligible resources may be moved to the CSD from the macro source is to use DFHCSDUP, as described under YES below. For information about migrating macro definitions to the CSD, see the *CICS/ESA Migration Guide*.

YES

YES indicates that you want to generate the necessary data to migrate your RDO-eligible resources. The records generated from the macro source are designed to be used as input to the DFHCSDUP utility program. An MNOTE warning message is issued for each RDO-eligible resource. The DFHCSDUP MIGRATE command will convert them into resource definitions on the CSD. These can then be managed using RDO.

COMPLETE

Use of COMPLETE means that TCT entries are not generated from the macro source for any RDO-eligible devices. For each one, the assembly produces an MNOTE. This means that you can keep your TCT macro source code after you have migrated your definitions.

If you continue to assemble a TCT for resources that are not eligible for RDO, you should continue to use MIGRATE=COMPLETE.

SYSIDNT={CICS|name}

This 1- to 4-character name is a private name that the CICS system uses to identify itself. If you use DFHTCT TYPE=REGION macros to define remote terminals, you must code this operand. It is used to determine whether a remote or a local TCT terminal entry will be generated from each TYPE=REGION macro following the TYPE=REGION macro. If the SYSIDNT on the TYPE=REGION macro is the same as the SYSIDNT on the TYPE=INITIAL macro, a local definition will be generated. If the SYSIDNT on the TYPE=REGION macro is different from the SYSIDNT on the TYPE=INITIAL macro, a remote definition will be generated.

The value you code for this operand is used to define the name of the local region during assembly of the TCT only. You must also define the name of the local region, for execution purposes, using the SYSIDNT parameter of the DFHSIT macro or as an initialization override.

Migrating TCT definitions — DFHTCT TYPE=GROUP

Use this macro to name the groups into which TCT definitions will be put when you migrate to resource definition online. This macro can appear as many times as required and at any point in the macro source. Each time it appears, it defines the CSD group into which subsequent definitions will be put until the next DFHTCT TYPE=GROUP macro occurs.

Table 59. DFHTCT TYPE=GROUP

	DFHTCT	TYPE=GROUP [,GROUP=name]
--	--------	-----------------------------

GROUP=name

Code this with the name of the group to which subsequent definitions will be migrated. The name can be up to eight alphanumeric characters, but must not begin with DFH. The default name is TCTxx where xx is the value coded for SUFFIX in the DFHTCT TYPE=INITIAL macro. If an error is found, the existing group name continues.

If a group with the name you specify does not already exist, it will be created. If it does exist, subsequent definitions will be added to it.

For details of how to migrate your TCT to the CSD, see Appendix D, "Migrating the TCT to the CSD" on page 389.

Logical device codes

Certain types of logical unit may be used to gain access to more than one resource within a subsystem. For example, a card punch device may be attached to a 3770 logical unit: the CICS application program can direct punch output, through BMS, via the 3770 to the card punch device. The facility provided by CICS to permit communication to devices within logical units of this type is the logical device code (LDC).

Although these are VTAM units, they require macro definition, unlike other VTAM devices.

The logical units that support LDCs are:

- 3601 logical unit
- 3770 batch logical unit
- 3770 batch data interchange logical unit
- 3790 batch data interchange logical unit
- LUTYPE 4 logical unit.

To reference such a device in a CICS application program, or in the CMSG transaction for message switching, you specify an LDC mnemonic which CICS will translate into a numeric LDC value. When CICS sends an output data stream to the logical unit, it includes the LDC value in the function management header (FMH). When the logical unit receives the data stream, it uses the LDC value to determine which component is to receive the output, or to perform some standard action.

Each LDC mnemonic to be referenced must be defined in the TCT, optionally with its associated LDC value and certain device characteristics for use by BMS functions. Such LDC information is contained in either the **system LDC table**, or in an **extended local LDC list**. You code the following DFHTCT macros to specify the system LDC table or an extended local LDC list:

- Code DFHTCT TYPE=LDC macro(s) to generate entries in the system LDC table. You may generate certain default LDC entries provided by CICS. For example,

```
DFHTCT TYPE=LDC,LDC=SYSTEM
```

will generate the following entries in the system LDC table:

Table 60. System LDC table entries

LDC mnemonic	LDC value	Device	Pagesize (row,column)
DS	1	3604 Keyboard Display	6,40
JP	2	3610 Document Printer	1,80
PB	3	Passbook and Document Printer	1,40
LP	4	3618 Administrative Line Printer	50,80
MS	5	3604 Magnetic Stripe Encoder	1,40
CO	0	Console medium or default print data set group	
R1	32	Card input medium	1,80
H1	32	Card output medium	1,80
P1	48	Print medium or print data set group	50,80
W1	128	Word processing medium 1	50,80
W2	144	Word processing medium 2	50,80
W3	160	Word processing medium 3	50,80
W4	192	Word processing medium 4	50,80

You may also define LDCs specifically to add LDC entries to the system LDC table. For example,

```
DFHTCT TYPE=LDC,
LDC=XX,
DVC=BLUPRT,
PGESIZE=(12,80),
PGESTAT=PAGE
DFHTCT TYPE=LDC,
LDC=YY,
DVC=BLUPCH,
PGESIZE=(1,80),
PGESTAT=AUTOPAGE
```

will add the following entries to the system LDC table:

Table 61. System LDC table entries defined by LDCs

LDC mnemonic	LDC value	Device	Pagesize (row,column)	PGESTAT
XX	48	Batch LU printer	12,80	PAGE
YY	32	Batch LU card output	1,80	AUTOPAGE

- Instead of the system LDC table, you may code the following series of DFHTCT TYPE=LDC macros to create an extended local LDC list. Default entries may also be generated. For example,

```
LDC1 DFHTCT TYPE=LDC,LOCAL=INITIAL
* the next line generates default CO,R1,H1,P1 LDCs
DFHTCT TYPE=LDC,LDC=BCHLU
DFHTCT TYPE=LDC,DVC=BLUPRT,LDC=AA,PAGESIZE=(6,30)
DFHTCT TYPE=LDC,DVC=BLUPCH,LDC=BB,PAGESIZE=(1,80)
DFHTCT TYPE=LDC,DVC=BLUCON,LDC=CC,PAGESIZE=(1,132)
DFHTCT TYPE=LDC,LOCAL=FINAL
```

will generate an extended local LDC list named LDC1 containing the following entries:

LDC mnemonic	LDC value	Device	PAGESIZE (row,column)
CO	0	Console medium or default printer data set group	
R1	32	Card input medium	1,80
H1	32	Card output medium	1,80
P1	48	Print medium or print data set group	50,80
AA	48	BLUPRT batch LU printer	6,30
BB	32	BLUPCH batch LU card output	1,80
CC	0	BLUCON batch LU console printer	1,132

When you are defining a logical unit in the TCT, you can specify its LDCs in either of two ways:

1. Code a DFHTCT TYPE=LDCLIST macro to define a local list of LDC mnemonics (and optionally their LDC values); for example,

```
LDC2 DFHTCT TYPE=LDCLIST,
LDC=(DS,JP,PB=5,LP,MS)
```

In the DFHTCT TYPE=TERMINAL macro defining the logical unit, you specify in the LDC operand the name of the local list as defined by the DFHTCT TYPE=LDCLIST macro. For example:

```
DFHTCT TYPE=TERMINAL,
TRMTYPE=3600,
LDC=LDC2, ...
```

has associated the LDCs DS, JP, PB, LP, and MS with the 3601 logical unit which you are defining. The LDC values may either be specified in the local list, or are obtained from the system LDC table. If BMS uses these LDC mnemonics, their page size and page status must also be available from the system LDC table.

Note: A local list defined by a DFHTCT TYPE=LDCLIST macro may be shared by a number of 3601, LUTYPE 4 and batch logical units.

2. In the DFHTCT TYPE=TERMINAL macro defining the logical unit, you specify in the LDC operand the name of an extended local LDC list. For example:

```
LDC1 DFHTCT TYPE=LDC,LOCAL=INITIAL
DFHTCT TYPE=LDC,LDC=BCHLU
DFHTCT TYPE=LDC,DVC=BLUPRT,LDC=AA,PAGESIZE=(6,30)
DFHTCT TYPE=LDC,DVC=BLUPCH,LDC=BB,PAGESIZE=(1,80)
DFHTCT TYPE=LDC,DVC=BLUCON,LDC=CC,PAGESIZE=(1,132)
DFHTCT TYPE=LDC,LOCAL=FINAL
DFHTCT TYPE=TERMINAL,TRMTYPE=BCHLU,LDC=LDC1, ...
```

has associated the LDCs CO, R1, H1, P1, AA, BB, CC with the batch logical unit which you are defining. Their LDC values and device characteristics for BMS functions are described in the extended local LDC list which is named LDC1.

When CICS requests an output or message switching operation using a particular LDC mnemonic for a logical unit, resolution of the mnemonic is attempted from the list (whichever form) as specified by the LDC operand of the DFHTCT TYPE=TERMINAL macro. If the LDC is not located in the local list or in the extended local list, the LDC specified is not valid for that terminal entry. In this case, X'00' is inserted in the logical device code portion of the FMH, and no destination name is inserted.

When a BMS function is requested for an LDC, and the LDC mnemonic is successfully resolved, the device characteristics (for example, device name and destination name) are accessed for the BMS function. If the LDC is in an extended local LDC list, these characteristics lie in the located extended local list entry. Otherwise, the system LDC table is searched for the LDC and the associated device characteristics.

Logical device codes — DFHTCT TYPE=LDC macro

The DFHTCT TYPE=LDC macro may only be used with 3600, LUTYPE4, 3770 batch logical unit, and 3770/3790 batch data interchange logical units.

You are responsible for setting up the LDC structure to be used with the terminal.

The expansion of this macro is the same, regardless of where it is specified in the TCT definition.

DFHTCT	TYPE=LDC
	[,DSN=destination-name] [,DVC=(device-type,sub-address)] [,LDC={SYSTEM LUTYPE4 3600 BCHLU}(aa[=nnn])] [,LOCAL={INITIAL FINAL}] [,PAGESIZE=(row,column)] [,PGESTAT={AUTOPAGE PAGE}]

DFHTCT TYPE=LDC: VTAM non-3270

name

Code this with the name of the extended local LDC list. It should be the same as that specified in the LDC operand of the DFHTCT TYPE=TERMINAL macro, and is only required if LOCAL=INITIAL is coded.

TYPE=LDC

Code this if an LDC is being defined to the system LDC table or to the extended local LDC list.

DSN=destination-name

Code this with the name to be used by BMS for destination selection for the batch data interchange logical unit. See the relevant CICS subsystem guides for further information on destination selection.

DVC=(device-type,sub-address)

Code this with the device type associated with the LDC to be used for a BMS request. This operand can only be coded in conjunction with the LDC=aa[=nnn] operand.

device-type

May be coded as follows:

Device type	Explanation
3604	Keyboard display
3610	Cut-forms document printer or journal printer (including the document/journal printer of a 3612)
3612	Passbook portion of a 3612
3618	Currently selected carriage
3618P	Primary carriage
3618S	Secondary carriage
3618B	Both carriages
BLUCON	Batch logical unit console printer
BLUPRT	Printer component of a batch logical unit
BLURDR	Card input component of a batch logical unit
BLUPCH	Card output component of a batch logical unit
WPMED1	Word processing medium 1
WPMED2	Word processing medium 2
WPMED3	Word processing medium 3
WPMED4	Word processing medium 4

The device types BLUPRT, BLURDR, BLUPCH, and BLUCON are devices attached to a batch, batch data interchange, or LUTYPE4 logical unit.

The WPMED1, 2, 3, and 4 options apply to LUTYPE4 logical units only. The component to which these options apply is defined by the particular type 4 logical unit implementation.

sub-address

Code this with the media sub-address. The range is 0 through 15, with a default of 0. A value of 15 indicates any sub-address. The sub-address differentiates between two units of the same device type (for example, BLUPRT,0 and BLUPRT,1),

which could be two print components attached to one logical unit.

LDC={SYSTEM|LUTYPE4|3600|BCHLU|(aa[=nnn])}

Code this with the LDC mnemonic and numeric value to be defined. Only the LDC=aa[=nnn] option can be used in conjunction with the DVC, PGESIZE, and PGESTAT operands.

SYSTEM

The following system-default LDCs for 3600, batch, and LUTYPE4 logical units are to be established:

LDC mnemonic	LDC value	Device	Pagesize (row, column)
DS	1	3604 Keyboard Display	6,40
JP	2	3610 Document Printer	1,80
PB	3	Passbook and Document Printer	1,40
LP	4	3618 Administrative Line Printer	50,80
MS	5	3604 Magnetic Stripe Encoder	1,40
CO	0	Console medium or default print data set group	
R1	32	Card input medium	1,80
H1	32	Card output medium	1,80
P1	48	Print medium or print data set group	50,80
W1	128	Word processing medium 1	50,80
W2	144	Word processing medium 2	50,80
W3	160	Word processing medium 3	50,80
W4	192	Word processing medium 4	50,80

LUTYPE4

System-default LDC mnemonics are to be established for an LUTYPE4 (word processing) logical unit. These consist of the CO, R1, P1, H1, W1, W2, W3, and W4 mnemonics, the corresponding LDC values, and the appropriate page sizes.

3600

System-default LDC mnemonics for the 3600 are to be established. These consist of the DS, JP, PB, LP, and MS mnemonics, the corresponding LDC values, and the appropriate page-size and page-status.

BCHLU

System-default LDC mnemonics for a batch logical unit are to be established. These consist of the CO, R1, P1, and H1 mnemonics, the corresponding LDC

values, and the appropriate page-size and page-status.

aa The two-character mnemonic to be used for this LDC.

nnn

The numeric value to be associated with the LDC in the system or extended local LDC list. The value in the system list is used as a default value for this LDC if a value is not found in a local LDC list (that is not extended) associated with a TCTTE. A value must be specified for 3600 devices. A value need not be specified for batch, batch data interchange, or LUTYPE4 logical units, but if one is specified it must correspond to the LDC value for the device type.

LOCAL={INITIAL|FINAL}

An extended local LDC list is to be generated.

INITIAL

This is the start of an extended local LDC list.

FINAL

This is the end of an extended local LDC list.

Note: LOCAL=INITIAL or FINAL may not be coded in the same DFHTCT TYPE=LDC macro as other operands. All DFHTCT TYPE=LDC entries coded after LOCAL=INITIAL and before LOCAL=FINAL will form part of one extended local LDC list; the entries coded outside the structure of this group will be added to the system LDC table. See the extended local LDC list example below.

The following is an example of an extended local LDC list:

```

DFHTCT TYPE=TERMINAL,TRMIDNT=BTCH, *
TRMTYPE=BCHLU,ACCMETH=VTAM,LDC=LDCA *
LDCA DFHTCT TYPE=LDC,LOCAL=INITIAL
DFHTCT TYPE=LDC,DVC=BLUPRT,LDC=AA, *
PGESIZE=(6,30)
DFHTCT TYPE=LDC,DVC=BLUPCH,LDC=BB, *
PGESIZE=(1,80)
DFHTCT TYPE=LDC,DVC=BLUON,LDC=CC, *
PGESIZE=(1,132),PGESTAT=AUTOPAGE
DFHTCT TYPE=LDC,LOCAL=FINAL
    
```

PGESIZE=(row,column)

Code this with the logical page size to be used with this LDC when BMS requests are processed.

row x column must not exceed 32767.

PGESTAT={AUTOPAGE|PAGE}

Indicates whether the device is to use autopaging or not. Autopaging means that BMS multiple page messages are printed continuously, without operator intervention. This is what is normally required for a printer. (Contrast the requirement for multiple page messages, displayed

on a 3270-type display, when the operator wants to finish reading a page, before requesting the next page to be delivered.)

Only BMS SEND commands with the PAGING option use autopaging. BMS SEND with TERMINAL or SET, does not use autopaging.

AUTOPAGE

Specify this for printers.

PAGE

Specify this for displays.

If the default PGESIZE or PGESTAT values provided by the LDC operand are to be overridden, a specific LDC should be coded with the mnemonic to be overridden. This overriding LDC must be coded in the LDC table prior to the LDC operand being coded.

PGESTAT=AUTOPAGE may be used to override the PGESTAT specification in DFHTCT TYPE=TERMINAL.

Logical device codes — DFHTCT TYPE=LDCLIST

The DFHTCT TYPE=LDCLIST macro, which may be used with 3600, LUTYPE4, and batch logical units, allows you to build a common list of logical device codes (LDCs) to be shared by more than one TCTTE.

You are responsible for setting up the LDC structure to be used with the terminal.

The expansion of this macro is the same, regardless of where it is coded in the TCT definition.

Table 66. DFHTCT TYPE=LDCLIST		
listname	DFHTCT	TYPE=LDCLIST ,LDC=(aa [=nnn][,bb[=nnn][,cc[=nnn],...])

listname

Is the required name of the LDC list. This name is referenced by TCTTEs through the LDC operand in DFHTCT TYPE=TERMINAL.

TYPE=LDCLIST

An LDC list is being defined.

LDC=(aa[=nnn][,bb[=nnn]][,cc[=nnn]][,...])

Code this with the LDCs (mnemonics and, optionally, the LDC numeric value) in this list.

(aa[=nnn][,bb[=nnn]] [,cc[=nnn]][,...])

Generates the LDCs in the list.

aa,bb,cc...

The two-character mnemonics of the LDCs in this list.

DFHTCT examples: LDC

nnn

A decimal value in the range 1 through 255 to be associated with an LDC. If a value is not specified, the system default value from the table defined by the DFHTCT TYPE=LDC macro, is used for this LDC. This value need not be coded for a batch or LUTYPE4 logical unit, but if it is, it must correspond to the LDC value for the device. LDCs for devices attached to a batch or LUTYPE4 logical unit are listed under the LDC parameter of the DFHTCT TYPE=LDC macro.

LDC TCT examples

The IBM 3770 Data Communication System and the IBM 6670 Information Distributor: The *CICS/OS/VS IBM 3767/3770/6670 Guide* provides information for CICS users who intend to install a CICS system that communicates with an IBM 3767 Communication Terminal, a 3770 Communication System, or a 6670 Information Distributor.

LDCs for 3770 batch logical unit TCT example

DFHTCT	TYPE=LDC,	*
	LDC=XX,	*
	DVC=BLUPRT,	*
	PGESIZE=(12,80),	*
	PGESTAT=PAGE	
DFHTCT	TYPE=LDC,	*
	LDC=YY,	*
	DVC=BLUPCH,	*
	PGESIZE=(1,80),	*
	PGESTAT=AUTOPAGE	
DFHTCT	TYPE=LDC,	*
	LDC=SYSTEM	

Figure 51. LDCs for 3770 batch logical unit TCT example

6670 LUTYPE 4 TCT example You must use RDO to define the terminal. For guidance, refer to the LDCLIST keyword of the TYPETERM option on page 231.

LDCS	DFHTCT	TYPE=LDC,LDC=SYSTEM	
LDC1	DFHTCT	TYPE=LDC,LOCAL=INITIAL	
	DFHTCT	TYPE=LDC,DVC=(BLUCON,01),	*
		PROFILE=DEFAULT,LDC=PC,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(BLUPRT,02),	*
		PROFILE=BASE,LDC=PP,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(BLUPRT,08),	*
		PROFILE=BASE,LDC=P8,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(BLUPRT,08),	*
		PROFILE=DEFAULT,LDC=DP,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(BLUPCH,03),	*
		PROFILE=JOB,LDC=PM,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(BLUPCH,03),	*
		PROFILE=DEFAULT,LDC=DM,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(WPMED1,04),	*
		PROFILE=WPRAW,LDC=P1,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(WPMED1,04),	*
		PROFILE=DEFAULT,LDC=D1,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(WPMED2,05),	*
		PROFILE=OI11,LDC=P2,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(WPMED2,05),	*
		PROFILE=DEFAULT,	*
		LDC=D2,PGESIZE=(50,80),	*
		PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(WPMED3,06),	*
		PROFILE=OI12,	*
		LDC=P3,PGESIZE=(50,80),	*
		PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,DVC=(WPMED4,07),	*
		PROFILE=OI13,LDC=P4,	*
		PGESIZE=(50,80),PGESTAT=AUTOPAGE	
	DFHTCT	TYPE=LDC,LOCAL=FINAL	

Figure 52. 6670 Lutype 4 TCT example

Sequential devices

CICS uses BSAM to control sequential devices such as card readers, line printers, magnetic tape units, and DASD to simulate terminals. Only unblocked data sets can be used with BSAM.

These "sequential terminals" may be used before actual terminals are available, or during testing of new applications.

| To define a sequential device, code the following macro instructions:

```
|           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.

The end of data indicator (EODI) for sequential devices may be altered using the SIT parameter EODI.

JCL for sequential devices

The DDNAME operands on the DFHTCT TYPE=SDSCI macros specify the ddname of the DD statements which you must provide in the CICS startup job stream:

```
//indd  DD ...           input data set
//outdd DD ...           output data set
```

where indd is the data set where input from the simulated terminal is submitted, and outdd is the data set where output to the simulated terminal is sent.

Sequential devices — DFHTCT TYPE=SDSCI

Table 67. DFHTCT TYPE=SDSCI		
DFHTCT	TYPE=SDSCI	
	,DEVICE=device	
	,DSCNAME=name	
	[,BLKSIZE=length]	
	[,DDNAME={name-in-DSCNAME	
	name}]	
	[,MACRF={([R],[W])}]	
	[,RECFM={U F V}]	

BLKSIZE=length

Code this with the maximum length (in bytes) of a block.

The default is BLKSIZE=0. If this operand is omitted, the block size can be specified in the data definition (DD) statement associated with the data set. A more detailed explanation of this operand is given in the

MVS/ESA Data Administration: Macro Instruction Reference manual.

DDNAME={name-in-DSCNAME|name}

Supplies the name of the data definition (DD) statement associated with a particular data set (line group). If this operand is omitted, the DSCNAME becomes the DDNAME.

DEVICE=device

One of the following values may be coded:

- For card readers:
{1442|2501|2520|2540|2560|2596| 3505|3525|5425}
- For line printers:
{1403|1404|1443|1445|3203|3211|5203}
- For disk (DASD):
{2314|3330|3340|3350|DASD|DISK}
- For tapes: **TAPE.**

The TAPE specification generates tape work files for both the input and the output data sets. Note that if an input tape with an expired label is used, the header may be rewritten, causing the first data records to be destroyed.

DSCNAME=name

The name of either the input or the output data set. If you are defining the input data set, ISADSCN on the DFHTCT TYPE=LINE macro must match the name that you specify: if you are defining the output data set, OSADSCN on the DFHTCT TYPE=LINE macro must match it.

MACRF={([R],[W])}

Code this with the way in which access to the sequential device is to be gained.

R Indicates the READ macro.

W Indicates the WRITE macro.

The default is MACRF=R for a card reader and MACRF=W for a line printer. For other sequential devices, MACRF=R or MACRF=W must be coded.

RECFM={U|F|V}

Code this with the record format for the DCB.

U Indicates undefined records. This option must be coded for DEVICE=1403 or 3211, or if you are using DASD for sequential terminal output (that is, if DEVICE=DASD and MACRF=W).

F Indicates fixed-length records.

V Indicates variable-length records.

If this operand is omitted, the record format can be specified in the data definition (DD) statement associated with the sequential data set.

Sequential devices — DFHTCT TYPE=LINE

DFHTCT	TYPE=LINE
	,ACCMETH={SAM BSAM SEQUENTIAL} ,INAREAL=length ,ISADSCN=input-name ,OSADSCN=output-name ,TRMTYPE={U/R CRLP DASD TAPE} [,LINSTAT='OUT OF SERVICE'] [,TCTUAL={0 length}]

ACCMETH={SAM|BSAM|SEQUENTIAL}

Specify SAM, BSAM, or SEQUENTIAL — they are equivalent in CICS.

INAREAL=length

Code this with the message input area length. The value should be equal to the length of the longest initial logical record of a transaction that may include multiple physical records.

ISADSCN=name

The name of the input data set. The TYPE=SDSCI DSCNAME operand for the input data set must match this.

LINSTAT='OUT OF SERVICE'

The line is to be initiated with an 'out of service' status. The default is 'in service'.

OSADSCN=name

The name of the output data set. The TYPE=SDSCI DSCNAME operand for the output data set must match this.

TCTUAL={0|length}

Indicates the length, in bytes (0 through 255), of the user area (the process control information field or PCI) for all terminal entries (TCTTEs) associated with this line. It should be made as small as possible. The TCT user area is initialized to zeros at system initialization. If fields of different (variable) lengths are desired, the TCTUAL value can be specified in one or more TYPE=TERMINAL macro instructions for terminals associated with this line.

TRMTYPE={U/R|CRLP|DASD|TAPE}

Indicates the sequential device type:

U/R	Any reader or printer
CRLP	A card reader and a line printer
DASD	A direct access storage device
TAPE	A magnetic tape device.

Sequential devices — DFHTCT TYPE=TERMINAL

DFHTCT	TYPE=TERMINAL
	[,LPLEN={120 value}] [,PGESIZE=9 lines,columns]) [,TCTUAL={number-specified-in-TYPE=LINE number}] [,TRANSID=transaction-identification-code] [,TRMPRTY={0 number}] [,TRMSTAT={TRANSACTION status[,s,...]}] [,USERID=userid]

LPLEN={120|value}

Controls the length of the print line for SAM output line printers. If no NL symbols are found in a segmented write, the print line length is the LPLEN value. The default is LPLEN=120.

PGESIZE=(lines,columns)

The default page size for a 1403 or CRLP terminal is (12,80). You must code PGESIZE if BMS is required for a device that has TRMTYPE=DASD specified. In this case, you must specify the number of lines and columns you wish to use. These two values multiplied together must equal the value specified for INAREAL.

lines × columns must not exceed 32767.

TCTUAL={number-specified-in-TYPE=LINE |number}

Indicates the length, in bytes (0 through 255), of the user area (the process control information field or PCI) for the terminal entry (TCTTE) associated with this terminal. It should be made as small as possible. The TCT user area is initialized to zeros at system initialization.

Use the TCTUAL operand of the DFHTCT TYPE=TERMINAL macro if fields of different (variable) lengths are desired for terminals associated with this line. In any case, the PCI field is generated for each terminal after the last terminal entry of the last line. The address of the PCI field is located at TCTTECIA; the length is located at TCTTECIL.

TRANSID=transaction-identification-code

Code this with a 1- to 4-character transaction code. This code specifies a transaction that is to be initiated each time input is received from the terminal when there is no active task.

If a TRANSID is not specified in the TCTTE, the TRANSID in a RETURN command from the previous transaction will be used. Otherwise, the first one to four characters of the data passed in the TIOA are used as the transaction code. A delimiter is required for transaction identifications of less than four characters.

TRMIDNT=name

Code this with a unique 4-character symbolic identification of each terminal. The identification supplied will be left-justified and padded with blanks to four characters if less than four characters are supplied.

The value CERR is reserved, as this is the identification generated for the error console.

TRMPRTY={0|number}

Establishes the terminal priority. This decimal value (0 through 255) is used in establishing the overall transaction processing priority. (Transaction processing priority is equal to the sum of the terminal priority, transaction priority, and operator priority, and must not exceed 255.)

TRMSTAT={TRANSACTION|(status,...)}

Code this with the types of activity that may occur at a given terminal. This terminal status is initially set in the TCTTE and is a combination of the processing status and the service status. The default is TRMSTAT=TRANSACTION.

TRANSACTION

A terminal with TRANSACTION status is used in the processing of transactions such as inquiries or order entries. A display station or a hard-copy terminal, to which no messages are sent without a terminal request, and through which transactions are entered, is a TRANSACTION terminal.

INPUT

Indicates a terminal that can send messages to, but cannot receive messages from, CICS.

Note: System messages may be routed to an input terminal under conditions such as invalid transaction identification and ATP batch count. This causes DFHTACP to be scheduled. To handle this situation, you should code a DFHTEP to perform any user-required action.

'OUT OF SERVICE'

Indicates a terminal that can neither receive messages nor transmit input. Such terminals are not polled by CICS. The 'OUT OF SERVICE' parameter can be used in combination with any status setting.

All terminals except the master terminal can be designated as 'OUT OF SERVICE'. When appropriate, the terminals can be placed in service by the master terminal and polling will be resumed.

RECEIVE

Indicates a terminal to which messages are sent but from which no input is allowed. An example of this type of terminal is one that is located in a remote location, such as a warehouse, and is unattended, but may receive messages. Automatic transaction initiation is implemented as for TRANSCEIVE, below.

TRANSCEIVE

A terminal with TRANSCEIVE status is a TRANSACTION terminal to which messages are sent automatically. The automatic transaction initiation, either by transient data control or interval control, sets a condition in an appropriate terminal control table terminal entry. If the terminal status is TRANSCEIVE and if there is no transaction at the terminal, terminal control initiates the user-defined task. This task is expected to send messages to the terminal.

USERID=userid

Code this to specify a user identifier for devices such as printers that are unable to sign on using CESN. (You can also specify USERID for a display device, in which case the display is permanently signed on. Operators are unable to sign on.) You must code this operand if you want to use preset security with this device. All access to protected resources depends on USERID.

The userid is referred to in security error messages, security violation messages, and the audit trail. It must be defined to the security manager.

userid must be a unique 1- to 8-character user identification. (A-Z 0-9 # \$ and @ are acceptable characters.)

TCAM DCB interface

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

Each TCAM communication line is associated with two sequential queues: the input process queue and the output process queue. CICS routes messages for terminals connected using TCAM 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) to process messages on the TCAM queues. CICS leaves it to the TCAM MCP to poll and address terminals, translate code, and control lines. Therefore, a number of DFHTCT operands are irrelevant for TCAM. This book describes how to define CICS terminals.

TCAM — DFHTCT TYPE=SDSCI, LINE, TERMINAL

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. A queue is treated like a communication line by CICS, and each one must be described by a DFHTCT TYPE=LINE macro. The DFHTCT TYPE=LINE macro will generate a TCT line entry (TCTLE), of which there is one for each queue.

DFHTCT TYPE=SDSCI: TCAM

Each TCAM terminal must be described in a DFHTCT TYPE=TERMINAL macro. The DFHTCT TYPE=TERMINAL macro will generate a TCT terminal entry (TCTTE), of which there is one for each terminal. To avoid duplication of the TCTTEs for both the input queue and the output queue, all the terminals are described immediately following the DFHTCT TYPE=LINE macro for the output queue. Although attached to the output TCTLE, these TCTTEs will be used for both input and output processing. One dummy TCTTE must also be generated 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. Using this identification as a search argument, the corresponding TCTTE can be located by CICS by comparing against the NETNAME value for each TCTTE.

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

Using the POOL feature (by specifying POOL=YES on the DFHTCT TYPE=LINE macro), it is possible to 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. This POOL feature necessarily imposes a number of restrictions and should be thoroughly understood before being implemented.

TCAM devices — DFHTCT TYPE=SDSCI

Table 70. DFHTCT TYPE=SDSCI		
DFHTCT	TYPE=SDSCI	
	,BLKSIZE=length	
	,DDNAME={name-in-DSCNAME name}	
	,DEVICE=TCAM	
	,DSCNAME=name	
	,MACRF=([R],[W])	
	,RECFM=U	
	[,OPTCD={W WU WC WUC U C UC	
	[,SYNAD=symbolic-name[

BLKSIZE=length

Code this with the maximum length (in bytes) of a block. For TCAM queues, the block size value must specify the maximum length that any CICS application program will require to be written in one request. Note that CICS application programs include the master terminal command. A block size of at least 2024 bytes (one screen size plus attribute bytes) should be specified.

DDNAME={name-in-DSCNAME|name}

Supplies the name of the data definition (DD) statement associated with a particular data set (line group). If this

operand is omitted, the DSCNAME becomes the DDNAME.

DEVICE=TCAM

This causes CICS to generate the appropriate data set control information to handle the TCAM input or output process queue.

DSCNAME=name

Code this with the symbolic data set control name associated with the data set control information and its related communication line(s).

The same DSCNAME must be specified on the DFHTCT TYPE=SDSCI macro and on its related DFHTCT TYPE=LINE macro(s).

MACRF=([R],[W])

Code this with how access to the TCAM device is to be gained.

R indicates the READ macro.

W indicates the WRITE macro.

OPTCD={W|WU|WC|WUC|U|C|UC}

Code this with the optional fields for the TCAM work unit.

C Code this if a one-byte field in the work area, called the position field, indicates whether the work unit being handled is the first, an intermediate, or the last segment of the message, and, on input, whether a record delimiter has been detected in the data.

U Code this if the work unit to be handled is either a message or a message segment that is not a record. If U is omitted, the work unit is assumed to be a record.

W For input, specifies that TCAM is to place the name of the source of each message in an eight-byte origin field in the work area. If OPTCD=W is omitted, you must ensure that a source name is placed in the origin field.

For output, specifies to TCAM that the name of the destination of the message will be placed in an eight-byte destination field in the work area before a TCAM WRITE macro is executed. CICS always inserts a destination name. If OPTCD=W is omitted, you must provide for interpretation of the destination field.

For details of the OPTCD operand, see the *ACF/TCAM Version 3 Application Programming* manual.

RECFM=U

Code this with the record format for the DCB.

U This indicates undefined records. This option must be coded for TCAM queues used by CICS.

SYNAD=symbolic-name

Specifies the address of a subroutine that is to be given control if message processing is used, if the work unit is larger than the work area, or if OPTCD=C is not coded.

For input queues, a user-written SYNAD routine can be specified for which an EXTRN is generated. If SYNAD is not coded, a CICS generated SYNAD routine is provided. If CICS SYNAD is used and the exit occurs:

1. Message DFH4000 is issued.
2. The DCB is closed.
3. The DCB is reopened.
4. Data is truncated to the specified block size and is passed to the CICS application program.

TCAM devices — DFHTCT TYPE=LINE

Table 71. DFHTCT TYPE=LINE		
DFHTCT	TYPE=LINE	
	,ACCMETH=TCAM	
	,DSCNAME=name	
	,INAREAL=length	
	,TRMTYPE=type	
	[,NPDELAY={0 number}]	
	[,OUTQ=symbolic-name]	
	[,POOL=YES]	
	[,QUEUEID=hexadecimal-number]	
	[,SIGNOFF={YES NO}]	
	[,TCTUAL={0 length}]	
	[,TRMMODL=number]	
	<u>TCAM SNA Only</u>	
	[,TCAMFET=SNA]	

ACCMETH=TCAM

Indicates that TCAM is to be used.

DSCNAME=name

Code this with the symbolic data set control name associated with the data set control information and its related communication line(s).

The same DSCNAME must be specified on the DFHTCT TYPE=SDSCI macro.

INAREAL=length

Code this with the message input area length.

- For an input line entry, the INAREAL value must be equal to or greater than the corresponding TCAM PCB buffer size.
- For an output line entry, the INAREAL value must be equal to or less than the corresponding TCAM PCB buffer size.

NPDELAY={0|number}

When used with a TCAM line, this parameter specifies the time interval that is to expire before control is passed to DFHTEP when a CICS task is not ready to accept a record from an input process queue.

If the CICS task issues a read before the time interval expires, processing continues normally and DFHTEP is not notified. The default value is zero.

OUTQ=symbolic-name

This is required in all TCAM input process queue terminal control table line entries. The symbolic name identifies the corresponding TCAM output process queue TCTLE. Multiple input process queues may reference the same output process queue.

POOL=YES

Coding POOL=YES on the TCAM output process queue indicates that the TCAM POOL feature is supported for that TCAM line.

QUEUEID=hexadecimal-number

Specifies a unique user ID for the TCAM process queue. The ID is an unframed, 1-byte hexadecimal number (00 to FF) that is placed in the input and output line entry at TCTLEQID to provide queue identification while executing a user exit.

SIGNOFF={YES|NO}

Code this to control whether the terminal should be signed off if the inactivity time limit set for the terminal user, as defined to the external security manager, is exceeded. (If you are using RACF as your security manager, the time limit for each operator is set with the TIMEOUT operand of the ADDUSER command.)

YES

The terminal will be signed off if the time limit is exceeded.

NO

The terminal will not be signed off if the time limit is exceeded.

TCAMFET=SNA

This must be coded if TCAM SNA devices are to be used on this line. The same DFHTCT TYPE=LINE macro must not include specifications for TCAM SNA and non-SNA devices. Specifying TCAMFET=SNA allows TCAM SNA devices to be used in conjunction with the TRMTYPE/SESTYPE combination of operands in DFHTCT TYPE=TERMINAL to generate logical units. This operand is required if SNA support (for example, FMH) is to be supplied by BMS or by DFHDIP.

TCAMFET=SNA and TRMTYPE=TCAM are mutually exclusive on the same DFHTCT TYPE=LINE macro.

TCTUAL={0|length}

Indicates the length, in bytes (0 through 255), of the user area (the process control information field or PCI) for all terminal entries (TCTTEs) associated with this line. It should be made as small as possible. The TCT user area is initialized to zeros at system initialization. If fields of different (variable) lengths are desired, the TCTUAL value can be specified in one or more TYPE=TERMINAL macro instructions for terminals associated with this line.

DFHTCT TYPE=TERMINAL: TCAM

TRMMODL=number

Sets the default terminal model for terminals on this line. See the description of TRMMODL for DFHTCT TYPE=TERMINAL.

TRMTYPE=type

In a non-SNA environment, you can specify TRMTYPE=TCAM to define a TCAM-only terminal associated with the line. This allows terminals supported by TCAM to use the TCAM interface through CICS. Device dependent editing must be handled by your message control program if a TCAM terminal type is specified. CICS systems programs only insert NL characters. When using TRMTYPE=TCAM, the following parameters are required on the DFHTCT TYPE=LINE macro: ACCMETH=TCAM, DSCNAME, and INAREAL. QUEUEID, NPDELAY, and OUTQ are optional.

If either 3270 data stream or 2260 support is required under TCAM, the appropriate 3270 terminal type (for example, L3270, L3270P) or the appropriate 2260 terminal type (for example, L2260) must be specified in the TRMTYPE operand. This will enable BMS to generate the correct data stream. TRMTYPE=TCAM should be used for all other terminals that require EBCDIC support. BMS will supply new line editing for those terminals specified in this way.

TRMTYPE=TCAM and TCAMFET=SNA are mutually exclusive on the DFHTCT TYPE=LINE macro.

In an SNA environment, you must specify TCAMFET=SNA on the DFHTCT TYPE=LINE macro, and code the correct TRMTYPE (see TRMTYPE=type and SESTYPE=session-type on page 317) on the DFHTCT TYPE=LINE macro. Code TRMTYPE=3767 for a TCAM-SNA device type 3767. TRMTYPE is also mandatory on the DFHTCT TYPE=LINE macro when a 3600 TCAM+SNA device is associated with the line.

TCAM devices — DFHTCT TYPE=TERMINAL

Table 72. DFHTCT TYPE=TERMINAL	
DFHTCT	TYPE=TERMINAL
	,TRMIDNT=name
	[,ACCMETH=TCAM]
	[,ALTPGE=(lines,columns)]
	[,ALTSCRN=(lines,columns)]
	[,ALTSFX=number]
	[,BUFFER=buffer-size]
	[,DEFSCRN=(lines,columns)]
	[,ERRATT=
	{NO ([LASTLINE],[,INTENSIFY]
	[, {BLUE RED PINK GREEN
	TURQUOISE YELLOW NEUTRAL}]
	[, {BLINK REVERSE
	UNDERLINE}}}]
	[,FEATURE=(feature[,feature],...)]
	[,FF={NO YES}]
	[,LASTTRM=LINE]

Table 72. DFHTCT TYPE=TERMINAL

	[,NETNAME=
	{name-specified-in-TRMIDNT name}}
	[,PGESIZE=(lines,columns)]
	[,PGESTAT={AUTOPAGE PAGE}]
	[,SIGNOFF={YES NO}]
	[,TCTUAL=
	{number-specified-in-TYPE=LINE
	number}]
	[,TIOAL={value (value1,value2)}]
	[,TRANSID=
	transaction-identification-code]
	[,TRMMODL=
	{number-specified-in-TYPE=LINE
	number}]
	[,TRMPRTY={0 number}]
	[,TRMSTAT=
	{TRANSACTION (status[,status])}
	[,TRMTYPE=
	{type-specified-in-TYPE=LINE type}]
	[,USERID=userid]
	<u>TCAM SNA Only</u>
	[,BMSFEAT=
	{([NOROUTE],[,NOROUTEALL]
	[,OBFMT],[,OBOPID])}]
	[,HF={NO YES}]
	[,LDC={listname
	{(aa[=nnn],bb[=nnn],
	cc[=nnn],...)}]
	[,SESTYPE=session-type]
	[,VF={NO YES}]

ACCMETH=TCAM

Indicates that the access method is TCAM.

ALTPGE=(lines,columns)

Indicates the page size to be used by BMS for this terminal entry when ALTSCRN has been selected as the screen size. The default is the value specified in ALTSCRN.

Unexpected results will occur if the columns value of ALTPGE is different from that of ALTSCRN. The lines value of ALTPGE can usefully be less than that of ALTSCRN, perhaps to reserve the bottom line of the screen for error messages (see the ERRATT operand).

lines × columns must not exceed 32767.

ALTSCRN=(lines,columns)

Defines the 3270 screen size to be used for a transaction whose RDO PROFILE definition has the attribute SCRNSIZE(ALTERNATE). The default is the value specified in the DEFSCRN operand.

The values that can be specified are:

Device	Alternate screen size	
3276-1,	3278-1	(12,80)
3276-2,	3278-2	(24,80)
3276-3,	3278-3	(32,80)
3276-4,	3278-4	(43,80)
	3278-5	(27,132)
3279-2A,	3279-2B	(24,80)
3279-3A,	3279-3B	(32,80)

For 3287 and 3289 printers, the value specified must equal the buffer size of the particular device.

Note: There is no validity checking performed on the screen size selected, and incorrect sizes may lead to unpredictable results.

For BSC devices, both the alternate and default screen sizes are determined by the device hardware. The alternate screen size is the maximum screen size.

For the 3290 display, both the default and alternate screen sizes are determined by the customer setup procedure. See the *IBM 3290 Information Panel Description and Reference* manual for further information.

If dual screen sizes are used, you can make CICS transactions use the alternate screen size by specifying SCRNSIZE(ALTERNATE) in the RDO PROFILE definition. If an application consists of pseudo conversations containing several linked transactions, you should provide PROFILE definitions for each of these transactions specifying SCRNSIZE(ALTERNATE) if the application uses alternate screen size.

ALTSFX=number

Code this with a one-character numeric suffix (specified in the SUFFIX operand of the application programmer's DFHMSD TYPE={DSECT|MAP} macro). This suffix will be appended by BMS to map set names if the screen size being used is the same value as the alternate screen size, that is, if SCRNSIZE(ALTERNATE) is specified in the transaction PROFILE definition (using RDO) or if the default and alternate screen size are the same. In this case, BMS map set selection routines will attempt to load the map set with the suffix specified in the ALTSFX operand. If there is no such map set, or if the transaction uses default screen size, BMS will try to load a map set suffixed with M or L and, if this load fails, BMS will try to load an unsuffixed map set version.

BMSFEAT=(**[NOROUTE][,NOROUTEALL][,OBFMT][,OBOPID]**)

Required for TCAM SNA only. Indicates which BMS features will be used for this TCTTE.

NOROUTE

BMS is not to produce routed data for this terminal.

NOROUTEALL

BMS is not to include this terminal in the list of terminals to receive data in response to a BMS route request to all devices. This operand is never required, but may be coded for any terminal.

OBFMT

BMS is to support outboard formatting for this terminal. See the *CICS/OS/VS IBM 3650/3680 Guide* for more details.

OBOPID

The outboard operator identifiers will be used by CICS in order to support the BMS routing facilities required for this terminal.

BUFFER=buffer-size

Code this with the size of the receive buffer for the indicated terminal. The default value is BUFFER=0.

For 3600 BSC devices, BUFFER represents the maximum data length that the workstation may receive. The value specified must not be greater than the size of the host read buffers of either the 3601 or of the workstation.

The unit of transmission from CICS to the BSC 3601 is a segment. If an application program issues a WRITE for a message longer than the value in BUFFER, CICS will break the message down into as many segments as necessary. The length of each segment (except the last) is equal to the value of BUFFER. The length of the last or only segment (in the case of a single segment message) is the length of the data still to be sent.

DEFSCRN=(**(lines,columns)**)

Defines the 3270 screen size or 3270 printer page size to be used on this device when attached to a transaction or used by BMS for which SCRNSIZE(DEFAULT) has been specified in the RDO PROFILE definition. The default is the value associated with the appropriate option in the TRMMODL operand, and is valid for dual screen-size terminals only.

The values that may be specified in the DEFSCRN operand are:

Device	Screen size	
3278-1		(12,40)
3278-2		(24,80)
3276-3,	3278-3	(24,80)
3276-4,	3278-4	(24,80)
	3278-5	(24,80)
3279-2A,	3279-2B	(24,80)
3279-3A,	3279-3B	(24,80)

For BSC devices, both default and alternate screen sizes are determined by the terminal hardware. The default screen size is (24,80), except for the 3278-1 where it is (12,40).

DFHTCT TYPE=TERMINAL: TCAM

**ERRATT={NO|([LASTLINE|,INTENSIFY|
[,({BLUE|RED|PINK|GREEN|TURQUOISE|
YELLOW|NEUTRAL})|
[,({BLINK|REVERSE|UNDERLINE})])}}**

This indicates the attributes that are to be associated with error messages that are displayed on this 3270 screen. This will override the value of the ERRATT operand coded in a DFHTCT TYPE=INITIAL or TYPE=LINE macro.

For more details, see "Control section — DFHTCT TYPE=INITIAL" on page 300.

FEATURE=(feature[,feature],...)

Code this with the applicable features for the 3270 terminals.

Further information on the features for 3270 devices can be found in the *CICS/OS/VS IBM 3270 Data Stream Device Guide*.

+
+

AUDALARM

Code this for the audible alarm feature for a 3270 terminal.

BTRANS

Code this for the background transparency feature.

COLOR

Code this for the extended color feature on a 3270 terminal. This feature allows colors to be selected for each field or character.

COPY

Code this if the Copy Feature for a 3270 display or printer is included in the 3270 control unit.

EXTDS

Code this for 3270 data stream extensions. This option is implied if any one of the COLOR, HILIGHT, PS, BTRANS, SOSI, OUTLINE, QUERYALL, QUERYCOLD, or VALIDATION (3270 only) options is coded.

HILIGHT

Code this for the 3270 extended highlight facility, which enables fields or characters to be displayed in reverse-video, underline mode, or blink (3270 only).

KATAKANA

Code this for Katakana support. Katakana terminals do not have a lowercase character set, but a Katakana character set instead. When coded for a Katakana terminal, all lowercase characters directed to the terminal from the following transactions will be translated to uppercase:

CBRC	CEDC
CDBC	CEDF
CDBI	CEMT
CEBR	CEOT
CECI	CESN
CEDA	CEST
CEDB	

MSRCNTRL

Code this if the terminal, an 8775 or 3643, has a magnetic slot reader. This option is not valid for SCS printers. If coded for an SCS printer, an error message will be raised but the option will not be generated.

OUTLINE

Code this for the field outlining feature of the 5550.

PARTNS

Code this if the terminal, a 3290, is to support partitions.

PS Code this for the 3270 programmed symbol (PS) facility. The facility enables up to six 191-character sets, with customer-defined and program-loaded fonts and codes, to be stored and accessed.

PTRADAPT

Code this for the 3275 printer adapter feature and corresponding 3284 Printer Model 3 on the 3275 Display Station. This feature makes the 3284 eligible for print requests through the program access key from the host 3275. A separate DFHTCT TYPE=TERMINAL macro cannot be coded for the 3284 Printer Model 3, because this printer shares the buffer of the 3275 Display Station.

Note: If output is created on the screen by DFHBMS or DFHMSD macro instructions with CTRL=PRINT, by BMS requests with the NLEOM option, or by the CMSG command, the contents of the screen are automatically copied to a 3270 printer, whether the program access key was hit or not.

QUERYALL

Code this for CICS to determine the features the terminal has, each time the terminal is connected. The features that can be determined are:

BTRANS
COLOR
EXTDS
HILIGHT
MSRCNTRL
OUTLINE
PARTNS
PS
SOSI
VALIDATION

Any features you code explicitly will be overridden by the values obtained by the QUERY.

You will find further guidance about using the SNA QUERY function, on page 217.

QUERYCOLD

Code this for CICS to determine the features the terminal has when the terminal is connected for the first time after a cold start. The features that can be determined are:

BTRANS
 COLOR
 EXTDS
 HILIGHT
 MSRCNTRL
 OUTLINE
 PARTNS
 PS
 SOSI
 VALIDATION

Any features you code explicitly will be overridden by the values obtained by the QUERY.

You will find further guidance about using the SNA QUERY function, on page 217.

SOSI

Code this to indicate that the device supports mixed EBCDIC and DBCS fields.

TRANSPARENCY

Code this if terminal data is not to be translated on a read or write, allowing the sending or receiving of all 256 bit combinations in a byte. This applies to 3270 devices with extended data stream support, and is implied by FEATURE=EXTDS. This is itself implied by setting any extended attribute feature.

UCTRAN

Code this for translation of lowercase data to uppercase in 3270 input data streams. Translation can be overridden in the application program by coding the ASIS option on specific RECEIVE requests.

VALIDATION

Code this for the 3290 device validation feature, which allows fields to be defined as MANDATORY FILL or MANDATORY ENTER.

Information in the TCTTE provided by coding the following features will not be used by CICS, but may be of use to applications programs requiring information on the features available on particular devices.

3270E

The device is one of the 3270 range having the alternate screen size facility (3276, 3278, 3279, 3287 or 3289). This option may not be coded for a 3287 printer attached to a 3271 or 3272 control unit.

APLKYBD

The 3270 device has the APL keyboard feature.

APLTEXT

The 3270 device has the APL text feature. This option may not be coded for a 3288 printer (with or

without the TEXTPRINT option, below). The APLTEXT feature is used in conjunction with the TEXTKYBD and APLKYBD options.

DKKYBD

Code this for the typewriter keyboard and/or operator console keyboard for a 3270 display. Both uppercase and lowercase data can be transmitted with either of these keyboards.

SELCTPEN

Code this for the selector pen feature for a 3270 display.

TEXTKYBD

The 3270 device has the text-keyboard feature.

TEXTPRINT

A 3288 printer has the text-print feature. This option may be used in conjunction with the 3270E option to indicate that the text-print feature will be used on a 3289 printer.

FF={NO|YES}

Indicates whether the terminal supports forms feed (FF). If FF=YES is coded, BMS will use this character when formatting output documents.

For 3270 displays and printers, FF=YES must be used in conjunction with the FORMFEED option in the BMS SEND commands. Use of form feed on display devices provides for a skip to a new page when the screen data is copied to a printer.

HF={NO|YES}

Required for TCAM SNA only. Indicates whether the horizontal forms feature is to be supported by the batch, batch data interchange, interactive, or LUTYPE4 logical unit. HF=NO will override the HTAB=(tab,...) parameter in the DFHMSD macro.

LASTTRM={LINE|POOL}

TYPE=TERMINAL

Indicates a "last terminal" condition.

LINE

Applies to TCAM nonswitched line processing.

POOL

Applies to TCAM switched-line processing. It must be coded to identify the last terminal in the pool.

LDC=listname

Required for TCAM SNA only. This TCTTE points to a list of logical device codes (LDCs). The list specifies which LDCs are valid for this logical unit and, optionally, which device characteristics are valid for each LDC. The first LDC generated in this list is the default when CICS must choose a default LDC for a logical unit. For more information see "Logical device codes" on page 302.

DFHTCT TYPE=TERMINAL: TCAM

listname

Code this with the name of the local LDC list or extended local LDC list to be associated with this logical unit. (This LDC list is generated by a DFHTCT TYPE=LDCLIST macro or by a series of DFHTCT TYPE=LDC macros for the extended local LDC list.)

NETNAME={name-specified-in-TRMIDNT|name}

The name must be the same as that used in the TCAM TERMINAL macro.

If you code a netname, ensure that it is unique to this terminal.

PGESIZE=(lines,columns)

Indicates the default page size for this terminal.

lines

Indicates the number of lines in the page.

columns

Indicates the number of characters in each line.

lines × columns must not exceed 32767.

If PGESIZE is not coded, the following defaults will be used:

TW33, TW35	(12,80)
3270 display model 1	(12,40)
3270 display model 2	(24,80)
3270 printer	(12,80)

For 3270 devices, if PGESIZE is not coded, it defaults to the DEFSCRN value.

PGESTAT={AUTOPAGE|PAGE}

Indicates whether the device is to use autopaging or not. Autopaging means that BMS multiple page messages are printed continuously, without operator intervention. This is what is normally required for a printer. (Contrast the requirement for multiple page messages, displayed on a 3270-type display, when the operator wants to finish reading a page, before requesting the next page to be delivered.)

Only BMS SEND commands with the PAGING option use autopaging. BMS SEND with TERMINAL or SET, does not use autopaging.

AUTOPAGE

Specify this for printers.

PAGE

Specify this for displays.

SESTYPE=session-type

See TRMTYPE=type and SESTYPE=session-type following.

SIGNOFF={YES|NO}

Code this to control whether the terminal should be signed off if the inactivity time limit set in the external security manager is exceeded. (If you are using RACF as your security manager, the time limit for each

operator is set with the TIMEOUT operand of the ADDUSER command.)

YES

The terminal will be signed off if the time limit is exceeded.

NO

The terminal will not be signed off if the time limit is exceeded.

TCTUAL={number-specified-in-TYPE=LINE|number}

Indicates the length, in bytes (0 through 255), of the user area (the process control information field or PCI) for the terminal entry (TCTTE) associated with this terminal. It should be made as small as possible. The TCT user area is initialized to zero at system initialization.

Use the TCTUAL operand of the DFHTCT TYPE=TERMINAL macro if fields of different (variable) lengths are desired for terminals associated with this line. In any case, the PCI field is generated for each terminal after the last terminal entry of the last line. The address of the PCI field is located at TCTTECIA; the length is located at TCTTECIL.

TIOAL=value

Indicates the terminal input/output area length to be passed to a transaction (that is, the length of the input area of the terminal), and should be set to a value that is slightly larger than the average input message length for the terminal. The default value is the message input area length generated from the INAREAL operand of the associated line entry. You can let it default for local 3270 terminals, but you should not omit the TIOAL parameter for remote 3270 terminals.

Note: If automatic transaction initiation is used, the minimum TIOAL that can be specified is one byte.

TRMIDNT=name

Code this with a unique 4-character symbolic identification of each terminal. The identification supplied will be left-justified and padded with blanks to four characters if less than four characters are supplied.

The name and the destination identification in the destination control table, when applicable to terminal destinations, must be the same.

The value CERR is reserved, as this is the identification generated for the error console.

TRANSID=transaction-identification-code

When using TCAM, TRANSID applies only to TCTLEs associated with the TCAM output queue.

TRMMODL={model-specified-in-TYPE=LINE|number}

When coded on the DFHTCT TYPE=LINE macro, the TRMMODL parameter sets the default model number of the terminals associated with this line.

If models vary on a line, the DFHTCT TYPE=TERMINAL macro can have a TRMMODL parameter associated with it which is different from that specified in the DFHTCT

TYPE=LINE macro. This will override the DFHTCT TYPE=LINE macro instruction for that DFHTCT TYPE=TERMINAL macro.

This operand must be coded on either the DFHTCT TYPE=LINE or TYPE=TERMINAL macro if you are defining a component of the 3270 Information Display System.

- 1 Code this for the 3270 Model 1 displays and printers with a default screen or buffer size of 12x40 (480 bytes or characters) (for example, 3277 Model 1). TRMMODL=1 is the default for 3270 Model 1 printers and displays. You should also code this for 3275 Display Station Model 11 support.
- 2 Code this for the 3270 displays and printers with a default screen or buffer size of 24x80 (1920 bytes or characters) (for example, 3278 Model 2). TRMMODL=2 is the default for the 3286 printer in 3270 compatibility mode. You should also code this for 3275 Display Station Model 12 support.

TRMPRTY={0|number}

Establishes the terminal priority. This decimal value (0 through 255) is used in establishing the overall transaction processing priority. (Transaction processing priority is equal to the sum of the terminal priority, transaction priority, and operator priority, and must not exceed 255.)

TRMSTAT={TRANSACTION|(status,...)}

Code this with the types of activity that may occur at a given terminal. This terminal status is initially set in the TCTTE and is a combination of the processing status and the service status. The default is TRMSTAT=TRANSACTION.

TRANSACTION

A terminal with TRANSACTION status is used in the processing of transactions such as inquiries or order entries. A display station or a hard-copy terminal to which no messages are sent without a terminal request and through which transactions are entered is a TRANSACTION terminal.

INPUT

Indicates a terminal that can send messages to, but cannot receive messages from, CICS.

Note: System messages may be routed to an input terminal under conditions such as invalid transaction identification and ATP batch count. This causes DFHTACP to be scheduled. To handle this situation, you should code a DFHTEP to perform any user-required action.

'OUT OF SERVICE'

Indicates a terminal that can neither receive messages nor transmit input. Such terminals are not polled by CICS. The 'OUT OF SERVICE'

parameter can be used in combination with any status setting.

All terminals except the master terminal can be designated as 'OUT OF SERVICE'. When appropriate, the terminals can be placed in service by the master terminal and polling will be resumed.

RECEIVE

Indicates a terminal to which messages are sent but from which no input is allowed. An example of this type of terminal is one that is located in a remote location, such as a warehouse, and is unattended, but may receive messages. Automatic transaction initiation is implemented as for TRANSCEIVE, below.

TRANSCEIVE

A terminal with TRANSCEIVE status is a TRANSACTION terminal to which messages are sent automatically. The automatic transaction initiation, either by transient data control or interval control, sets a condition in an appropriate terminal control table terminal entry. If the terminal status is TRANSCEIVE and if there is no transaction at the terminal, terminal control initiates the user-defined task. This task is expected to send messages to the terminal.

TRMTYPE=type and SESTYPE=session-type

In a non-SNA environment, the TRMTYPE operand can be used to specify the terminal type:

- If the terminal type has not already been specified in the DFHTCT TYPE=LINE macro, or
- To override the type specified in that macro.

In an SNA environment, support for TCAM SNA logical units is generated by coding the appropriate TRMTYPE/SESTYPE combinations. However, you should note that TCAM does not support the IBM 3650 Retail Store System or the IBM 3600 Finance Communication System.

USERID=userid

Code this to specify a user identifier for devices such as printers that are unable to sign on using CESN. (You can also specify USERID for a display device, in which case the display is permanently signed on. Operators are unable to sign on.) You must code this operand if you want to use preset security with this device. All access to protected resources depends on USERID.

The userid is referred to in security error messages, security violation messages, and the audit trail. It must be defined to the security manager.

userid must be a unique 1- to 8-character user identification. (A-Z 0-9 # \$ and @ are acceptable characters.)

DFHTCT example: TCAM

VF={NO|YES}

Required for TCAM SNA only. Indicates whether the vertical form feature is to be supported by the batch, batch data interchange, interactive, or LUTYPE4 logical unit. VF=NO will override the VTAB=(tab,...) parameter in the BMS map definition.

TCAM TCT example

This example defines a local line with two 2260 terminals, and a remote line with three 3270 terminals.

Note: The TCTTEs for the local 2260 terminals and the remote 3270 terminals are used for both input and output processing.

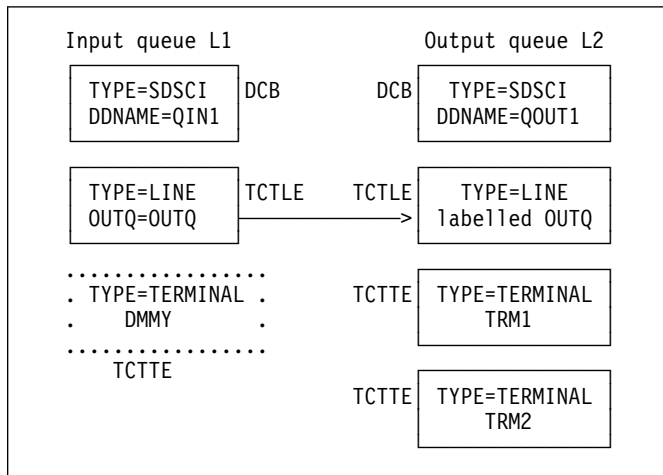


Figure 53. Local line with two 2260 terminals

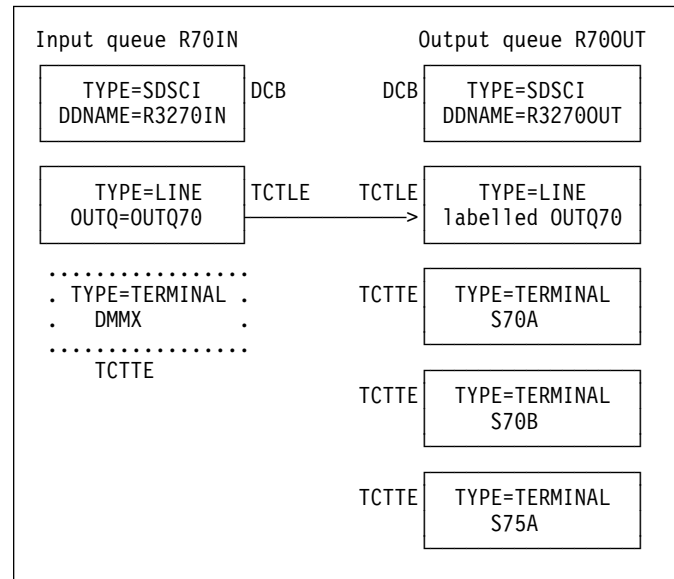


Figure 54. Remote line with three 3270 terminals

Figure 55 shows the coding for the configuration shown in Figure 53 and Figure 54 on page 318.

	DFHTCT	TYPE=INITIAL, SUFFIX=TV		*
	DFHTCT	TYPE=SDSCI, DEVICE=TCAM, DSCNAME=L1, DDNAME=QIN1, OPTCD=WU, MACRF=R, RECFM=U, BLKSIZE=500	DCB for input queue for local 2260 line	*
	DFHTCT	TYPE=SDSCI, DEVICE=TCAM, DSCNAME=L2, DDNAME=QOUT1, OPTCD=WU, MACRF=W, RECFM=U, BLKSIZE=500	DCB for output queue for local 2260 line	*
	DFHTCT	TYPE=LINE, ACCMETH=TCAM, QUEUEID=F1, INAREAL=500, TRMTYPE=L2260, DSCNAME=L1, OUTQ=OUTQ	Input queue for local 2260 line	*
	DFHTCT	TYPE=TERMINAL, TRMIDNT=DMMY, TRMPRTY=32, LASTTRM=LINE	Dummy	*
OUTQ	DFHTCT	TYPE=LINE, ACCMETH=TCAM, QUEUEID=F0, INAREAL=500, TRMTYPE=L2260, DSCNAME=L2	Output queue for local 2260 line	*
	DFHTCT	TYPE=TERMINAL, TRMIDNT=TRM1, TRMPRTY=32	First local 2260 terminal on the line	*
	DFHTCT	TYPE=TERMINAL, TRMIDNT=TRM2, LASTTRM=LINE, TRMPRTY=32	Second and last local 2260 terminal on the line	*

Figure 55 (Part 1 of 2). Coding for TCAM TCT example

DFHTCT example: TCAM

```

DFHTCT TYPE=SDSCI,          DCB for input queue for      *
      DEVICE=TCAM,          remote 3270 line      *
      DSCNAME=R70IN,        *
      DDNAME=R3270IN,      *
      OPTCD=WU,            *
      MACRF=R,             *
      RECFM=U,             *
      BLKSIZE=500          *

DFHTCT TYPE=SDSCI,          DCB for output queue for      *
      DEVICE=TCAM,          remote 3270 line      *
      DSCNAME=R70OUT,      *
      DDNAME=R3270OUT,    *
      OPTCD=WU,            *
      MACRF=W,             *
      RECFM=U,             *
      BLKSIZE=500          *

DFHTCT TYPE=LINE,          Input queue for remote 3270 *
      ACCMETH=TCAM,        line *
      INAREAL=500,         *
      DSCNAME=R70IN,      *
      OUTQ=OUTQ70,        *
      TRMTYPE=3277        *

DFHTCT TYPE=TERMINAL,      Dummy *
      TRMIDNT=DMMX,       *
      TRMPRTY=32,         *
      LASTTRM=LINE        *

OUTQ70 DFHTCT TYPE=LINE,    Output queue for remote 3270 *
      ACCMETH=TCAM,        line *
      INAREAL=500,         *
      DSCNAME=R70OUT,     *
      TRMTYPE=3277        *

DFHTCT TYPE=TERMINAL,      First remote 3270 terminal *
      TRMPRTY=32,         on the line *
      TRMIDNT=S70A        *

DFHTCT TYPE=TERMINAL,      Second remote 3270 terminal *
      TRMPRTY=32,         on the line *
      TRMIDNT=S70B        *

DFHTCT TYPE=TERMINAL,      Third and last remote 3270 *
      TRMPRTY=32,         terminal on the line *
      TRMIDNT=S75A,      *
      LASTTRM=LINE        *

DFHTCT TYPE=FINAL
END

```

Figure 55 (Part 2 of 2). Coding for TCAM TCT example

Remote terminals for transaction routing

CICS can communicate with other systems that have similar communication facilities. We have called this sort of communication **CICS intercommunication**. You can read about it in the *CICS/ESA Intercommunication Guide*.

We sometimes refer to the **local** system and the **remote** system. When you are concerned with resource definition, the system where the TCT is installed is the local system. The system that is being defined in the TCT is the remote system.

Transaction routing enables terminals in one CICS system to invoke transactions in another CICS system. You can use transaction routing between systems connected by MRO or by an LUTYPE 6.2 link. There are two possible methods of defining the terminals using macros. (There is another method, only possible using RDO, called 'shipping terminal definitions'. See "Terminals for transaction routing" on page 188.) The two macro methods are described below.

Creating remote definitions for terminals for transaction routing

The two methods of creating remote definitions for terminals to be used for transaction routing are:

- Method 1:

```
DFHTCT TYPE=REGION, ... (one for each region)

DFHTCT TYPE=SDSCI, ... (for non-VTAM terminals
                        only; ignored for remote
                        definitions)

DFHTCT TYPE=LINE, ... (for non-VTAM terminals
                       only)

DFHTCT TYPE=TERMINAL, ... (for non-VTAM: one for
                           each terminal)
```

- Method 2:

```
DFHTCT TYPE=REMOTE, ... (one for each terminal)
```

In method 1, you can use copybooks to include the same source code in the TCTs for local and remote systems. The information not needed (that is, the whole of the TYPE=SDSCI macro, and some of the TYPE=LINE and TYPE=TERMINAL macros) is discarded for remote entries.

Method 2 employs a single DFHTCT TYPE=REMOTE macro.

CICS decides whether to create a remote or a local definition on the basis of the SYSIDNT operand, either on the TYPE=REGION, or on the TYPE=REMOTE macro. This is compared with the SYSIDNT operand in DFHTCT TYPE=INITIAL. If they are the same, the definition(s) will be local. If they are different, the definition(s) will be remote.

These terminals cannot use transaction routing and therefore cannot be defined as remote:

- IBM 7770 or 2260 terminals
- MVS system consoles
- Pooled TCAM terminals
- Pooled 3600 or 3650 Pipeline Logical Units.

Note: BTAM is not supported in this release. To gain access to this release of CICS from BTAM terminals, define the BTAM terminals in an earlier release of CICS, and use transaction routing to gain access to the current release. BTAM terminals must be defined as remote resources in this release, and as local resources in the earlier release. More information about the definition of local BTAM terminals can be found in the documentation for earlier releases of CICS.

Remote terminals, method 1 — DFHTCT TYPE=REGION

The DFHTCT TYPE=REGION macro introduces information about the named region. The information consists of DFHTCT TYPE=LINE and TYPE=TERMINAL macros. These macros must follow the DFHTCT TYPE=REGION macro. For a remote region, the DFHTCT TYPE=LINE macro will not generate a TCT line entry (TCTLE). Every terminal that will participate in transaction routing must be defined. Only certain DFHTCT macro types and operands are relevant in remote region definitions; all others will be ignored. The operands that are relevant are those listed in "Remote terminals, method 2 — DFHTCT TYPE=REMOTE" on page 322.

Table 73. DFHTCT TYPE=REGION

	DFHTCT	TYPE=REGION ,SYSIDNT={name LOCAL}

SYSIDNT={name|LOCAL}

Indicates the 4-character name of the system or region whose information starts or resumes here.

SYSIDNT=LOCAL can be specified to indicate that the TYPE=TERMINAL definitions following it refer to the home region, as do all definitions preceding the first DFHTCT TYPE=REGION macro. The name of the home region (that is, the region in which this terminal control table will be used) is the value of the SYSIDNT operand of the DFHTCT TYPE=INITIAL macro. The name can instead be that of a previously defined MRO link or ISC link.

Remote terminals, method 1 — DFHTCT TYPE=TERMINAL

Note: The DFHTCT TYPE=LINE macro and the additional operands of the DFHTCT TYPE=TERMINAL macro are valid, but are ignored if the SYSIDNT operand on the preceding DFHTCT TYPE=REGION macro indicates a remote region. (For details of the DFHTCT TYPE=LINE and DFHTCT TYPE=TERMINAL macros, see “Sequential devices” on page 306 and “TCAM — DFHTCT TYPE=SDSCI, LINE, TERMINAL” on page 309.)

DFHTCT	TYPE=TERMINAL ,ACCMETH=access-method ,SYSIDNT=name ,TRMIDNT=name ,TRMTYPE=terminal-type ,[,RMTNAME= {name-specified-in-TRMIDNT}]

ACCMETH=access-method

Code this with the access method of the remote terminal.

RMTNAME={name-specified-in-TRMIDNT}name

Specifies the 1- to 4-character name by which the terminal is known in the system or region that owns the terminal. (That is, in the TCT of the **other** system.) If this operand is omitted the name in the TRMIDNT operand is used.

SYSIDNT=name

Indicates the 4-character name of the system or region that owns this terminal. This may be the local system or region (that is, the name defined in the TYPE=INITIAL macro), in which case the TCT entry created will be a local definition. It may be the name of a different system or region, in which case the TCT entry created will be a remote definition. This SYSIDNT must be the same as the SYSIDNT on the TYPE=REGION macro that precedes this macro.

TRMIDNT=name

Specifies the 1- to 4-character name by which the terminal is known in **this** system (that is, in the local system that owns this TCT, and that owns the transactions).

TRMTYPE=terminal-type

Code this with the terminal type. For details, see “Sequential devices” on page 306 and “TCAM — DFHTCT TYPE=SDSCI, LINE, TERMINAL” on page 309.

Remote terminals, method 2 — DFHTCT TYPE=REMOTE

Terminal entries for remote systems or regions can be defined to CICS using the DFHTCT TYPE=REMOTE macro as an alternative to defining them using DFHTCT TYPE=TERMINAL macro instructions in conjunction with a DFHTCT TYPE=REGION macro.

The expansion of the DFHTCT TYPE=REMOTE macro is independent of the region currently referenced.

Note: If the SYSIDNT operand indicates that the terminal is owned by the **home** region then all the operands of the DFHTCT TYPE=TERMINAL macro become valid on the DFHTCT TYPE=REMOTE macro and have the same meaning as for TYPE=TERMINAL. However, if (as is normally the case) the SYSIDNT operand indicates a remote region, the additional operands of DFHTCT TYPE=TERMINAL are valid on the DFHTCT TYPE=REMOTE macro, but are ignored. (For details of the DFHTCT TYPE=TERMINAL macro, see “Sequential devices” on page 306 and “TCAM — DFHTCT TYPE=SDSCI, LINE, TERMINAL” on page 309.)

DFHTCT	TYPE=REMOTE ,ACCMETH=access-method ,SYSIDNT=name ,TRMIDNT=name ,TRMTYPE=terminal-type ,[,RMTNAME= {name-specified-in-TRMIDNT}]

ACCMETH=access-method

Code this with the access method of the remote terminal.

RMTNAME={name-specified-in-TRMIDNT}name

Specifies the 1- to 4-character name by which the terminal is known in the system or region that owns the terminal. (That is, in the TCT of the **other** system.) If this operand is omitted the name in the TRMIDNT operand is used.

SYSIDNT=name

Specifies the name of the system or region that owns this terminal. The name must be the same as that used in the SYSIDNT operand of a previous TYPE=SYSTEM macro, or the TYPE=INITIAL macro.

TRMIDNT=name

Specifies the 1- to 4-character name by which the terminal is known in **this** system, that is, in the local system that owns this TCT, and that owns the transactions.

TRMTYPE=terminal-type

Code this with the terminal type. For details, see “Sequential devices” on page 306 or “TCAM — DFHTCT TYPE=SDSCI, LINE, TERMINAL” on page 309.

CICS terminals list

This release of CICS is able to communicate with almost all previously supported terminals, either *directly* or *indirectly*, as described below.

New or current terminals will be directly supported by CICS/ESA 4.1 if they conform to the VTAM interface.

Because of the removal of support for BTAM and TCAM(ACB), certain device types are not able to connect directly to CICS/ESA 4.1. CICS/ESA 4.1 provides support for such devices indirectly, through transaction routing from an earlier release of CICS.

The BTAM terminals affected are listed in Table 76 and described in more detail in "Details of BTAM terminals supported" on page 325.

If you are a user of TCAM/ACB you have a choice between:

1. Migrating your TCAM/ACB terminals to ACF/VTAM V3R3 at the same time as migrating to CICS/ESA 4.1.

Note: All terminals supported by TCAM/ACB are also supported by VTAM.

2. Leaving your TCAM/ACB terminals on a coexisting earlier CICS system and transaction routing to this release of CICS for the lifetime of the TCAM/ACB terminals.

Table 76 summarizes how terminals are supported in CICS/ESA 4.1.

Directly supported by CICS/ESA 4.1 using VTAM	Supported using transaction routing through earlier CICS; (BTAM Terminals).
*3101 Display Terminal *3230 Printer 3268 Printer *3270 Information Display System 3270 PC 3270 PC/G 3270 PC/GX 3287 Printer *3600 Finance Communication System 3630 Plant Communication System 3640 Plant Communication System 3650 Retail Store System 3680 Programmable Store System 3730 Distributed Office Communication System *3767 Communication Terminal *3770 Data Communication System 3790 Communication System 4300 Processors *4700 Finance Communication System *5280 Distributed Data System *5520 Administrative System	1050 Data Communication System 2740 Communication Terminal 2741 Terminal controller 2770 Data Communication System 2780 Data Transmission Terminal 2980 General Banking Terminal System 3660 Supermarket System 3735 Programmable Buffered Terminal 3740 Data Entry System 3780 Data Communications Terminal 5100 Portable Computer 5110 Portable Computer 5230 Data Collection System 5260 Retail System Communicating Magnetic Card Selectric Typewriter (CMCST) Office System/6 Series/1 System/3 System/7 System/23
5550 Administrative System *5937 Rugged Terminal *6670 Information Distributor 8100 Information System 8775 Display Terminal 8815 Scanmaster *Displaywriter *Personal Computer, PS/2, PS/55 *System/32 *System/34 *System/36 *System/38	

Table 76 (Page 2 of 2). Summary of terminal support. List of IBM terminals and system types and how they are supported by CICS/ESA 4.1.

Directly supported by CICS/ESA 4.1 using VTAM	Supported using transaction routing through earlier CICS; (BTAM Terminals).
AS/400 *System/370 (inc 303x, 308x, and 3090 processors) Teletypewriter Exchange Service (TWX 33/35) World Trade Typewriter Terminal (WTTY)	
Note: * These terminals are also supported by BTAM.	

Details of BTAM terminals supported

This section provides a detailed account of the BTAM terminals supported by CICS/ESA 4.1. For information on how to connect them to CICS/ESA 4.1, using transaction routing, see "Remote terminals for transaction routing" on page 321. The following abbreviations have been used:

local	channel or adapter attached
s/s	start/stop transmission
SDLC	synchronous data link control
sw	switched
BSC	binary synchronous
nonsw	non-switched communications.

Table 77 (Page 1 of 2). IBM BTAM terminals. Detailed list of BTAM terminals and subsystems supported by transaction routing from an earlier release of CICS.

Terminal /System Type	Units	Attachment	Notes
1050	1051,1052,1053,1056	s/s sw or nonsw	
2740		s/s sw or nonsw	
2741		s/s sw or nonsw	
2770	2772,0545,1053,2213,2265,2502	BSC sw or nonsw	1 on page 326
2780		BSC sw or nonsw	2 on page 326
2980	2972,2980	BSC nonsw	3 on page 326
3660	3651,3661	BSC sw	4 on page 326
3735		BSC sw	
3740	3741	BSC sw or nonsw	6 on page 326
3780		BSC, sw or nonsw	7 on page 326
5100		s/s supported as 2741	4 on page 326
5110		BSC attached as 2770	4 on page 326

Table 77 (Page 2 of 2). IBM BTAM terminals. Detailed list of BTAM terminals and subsystems supported by transaction routing from an earlier release of CICS.

Terminal /System Type	Units	Attachment	Notes
5230	5231	BSC supported as 3741	
5260	5265	BSC attached as 3741	
CMCST		Supported as 2741	
Office System/6	6640,6670	Supported as 2770	
Series/1		Attached as System/3; supported as 3650 Pipeline (VTAM) or 3790 (full function LU)	4 on page 326
System/3	5406,5408,5410,5412,5415	BSC sw or nonsw	4 on page 326
System/7	5010	s/s or BSC, sw or nonsw	4 on page 326 4 on page 326, 5 on page 326
System/23		Attached as 3741 (BSC)	
<p>Notes:</p> <ol style="list-style-type: none"> 1. CICS 2770 support includes optional 2772 features #3650 (EBCDIC Transparency), #9936 (WACK response), #1490 (Buffer Expansion), #1910 (Conversational Mode), #1340 (Automatic Answering), #4610 (Identification), #6310 (Security Identification), and #5010 (Multipoint). 2. 6-bit transcode is not supported. Support includes optional features #8030 (EBCDIC transparency) and #1340 (Automatic Answering), or #5020 (Multipoint). 3. 2980 support is for 2972 Model 8 (RPQ 858160) or Model 11 (RPQ 8582311) with 2980 Model 1 (RPQ 835504), Model 2 (RPQ 835505) or Model 4 (RPQ 858147), including options RPQ 858188 (Auditor key for Model 2) and RPQ 858165 (Buffer Expansion). 4. Devices and features supported by a system or programmable controller are generally transparent to CICS. In some cases CICS provides specific device support, in which case the units are listed. 5. Non-switched as a multipoint device: System/7 Remote IPL is supported. Switched and as a Point-to-Point Device: Remote IPL is not supported. 6. 3740 support includes optional features #7850 (Terminal Identification), #1685 (Multipoint), #5450 (Operator ID Card Reader), and #1680 (Expanded Communications). 7. 3780 support includes features #3601 (EBCDIC Transparency), #9936 (WACK), #5010 (Multipoint), or #7651 (Switched), #1601 (Component Selection). 			

The terminals listed above are all BTAM and are not supported by VTAM. There are other terminals which are supported by both BTAM and VTAM, and which can be used for transaction routing. For a detailed list of all VTAM terminals, see the "Devices supported" on page 221. Some BTAM terminals can not be used for transaction routing from an earlier release of CICS to CICS/ESA 4.1. The terminals are:

2260
2265
7770

VTAM terminals

For a detailed list of VTAM-supported terminals, and how to define them to CICS, see “Devices supported” on page 221.

VTAM terminals

Chapter 36. TLT — terminal list table

A terminal list table (TLT) generated by the DFHTLT macro instruction allows terminal and operator identifications to be grouped logically. A TLT:

- Is *mandatory* for use by CEST (the supervisor terminal transaction), to define and limit the effective range of the operation. For example:

```
CEST SET TERMINAL(*) SUPRID(CG) OUTSERVICE
```

will set all terminals defined in DFHTLT CG out of service.

- May be used by CEST or CEMT (the master terminal transaction) to apply an operation to a predetermined group of terminals. (For a CEST operation, this TLT must define a subset of the TLT specified by SUPRID.) For example:

```
CEST SET TERMINAL(*) SUPRID(CG) CLASS(EM) INSERVICE
CEMT SET TERMINAL(*) CLASS(EM) INSERVICE
```

will both set all terminals defined in DFHTLT EM in service.

- May be used singly or in combination with other TLTs to provide predefined destinations for message switching. For example:

```
CMMSG ROUTE=.PG,'PRODUCTION MEETING AT 11.00 IN
ROOM 2124',SEND
```

will send a message to all terminals or operators defined in DFHTLT PG.

The same TLT can be used for message switching and for supervisory or master terminal functions. For example, a TLT that defines the terminals that are under control of a supervisory terminal, could also be used as a destination list for sending messages to those terminals.

For some logical units, logical device code (LDC) mnemonics (that may be associated with each table entry), are used for message switching and are ignored for master and supervisory terminal operations.

In an intercommunication network, all the terminals in a terminal list table must be owned by the system on which the table is used.

Elements of DFHTLT

The following macros define the TLT entries:

- Control section — DFHTLT TYPE=INITIAL
- Entries in terminal list table — DFHTLT TYPE=ENTRY
- End of terminal list table — DFHTLT TYPE=FINAL (see page 249).

Control section — DFHTLT TYPE=INITIAL

The entry point and the address of the start of the terminal list table being defined are established by the DFHTLT TYPE=INITIAL macro.

Table 78. DFHTLT TYPE=INITIAL

	DFHTLT	TYPE=INITIAL [.LDC=aa] [.SUFFIX=xx]
--	--------	---

Note: For general information about TYPE=INITIAL macros, see “TYPE=INITIAL” on page 248.

LDC=aa

Code this with a 2-character logical device code (LDC) mnemonic. This is associated with every logical unit identification, except for those for which an LDC mnemonic is specified on a DFHTLT TYPE=ENTRY macro.

SUFFIX=xx

The module name of the TLT is DFHTLTxx, where xx is a 1- or 2-character suffix. This provides unique identification for each TLT used. Because the names TLTB A, TLTB B, TLTB C, and TLTE A are used within the TLT, suffixes BA, BB, BC, and EA must not be used.

A TLT must have a suffix to be used by the message switching transaction, CMMSG.

Entries in terminal list table — DFHTLT TYPE=ENTRY

Entries are coded in the TLT as follows:

Table 79. DFHTLT TYPE=ENTRY

	DFHTLT	TYPE=ENTRY ,TRMIDNT=([termid-1[*ldc-1]]/opid-1 [,termid-2[*ldc-2]/opid-2],...)
--	--------	--

TYPE=ENTRY

Code this if one or more entries are to be generated in this table, up to a maximum of 1000 entries.

TRMIDNT=([termid-1[*ldc-1]] [/opid-1],[termid-2[*ldc-2]]/opid-2],...)

Code this with a list of start-stop and BSC terminal, logical unit, and operator identifications. A logical unit identification can be qualified by an LDC mnemonic.

termid

Indicates a 1- to 4-character start-stop or BSC terminal or logical unit identification.

DFHTLT example

Note: A 3614 attached to a communications controller may be used in master or supervisory terminal operations but should not be used in message switching operations. (A 3614 is not valid for a message destination.)

ldc

Indicates a 2-character LDC mnemonic, which must be preceded by an asterisk (*) and is only used following the 'termid' parameter.

opid

Indicates a 1- to 3-character operator identification that must be preceded by a slash (/).

Any terminal or operator identification specified should also be specified in the TRMIDNT operand of the DFHTCT macro and in your external security manager, respectively. (If you employ RACF, you will use the OPIDENT operand of the ADDUSER command to record the identification for each operator.) Any LDC mnemonic specified should also be specified in the LDC operand of the DFHTCT TYPE=LDC and DFHTCT TYPE=TERMINAL macros.

Supervisory and master terminal functions use all terminal and logical unit identifications included in the TLT, but ignore all references to LDC mnemonics and operator identifications.

DFHTLT example

Figure 56 illustrates coding to create a terminal list table.

```
Example 1

DFHTLT TYPE=INITIAL, *
        SUFFIX=AA
DFHTLT TYPE=ENTRY, *
        TRMIDNT=(NYC,CHI,LA,WDC)
DFHTLT TYPE=ENTRY, *
        TRMIDNT=SF
DFHTLT TYPE=ENTRY, *
        TRMIDNT=(BSTN/OP1,ATL/OP5,/OP9,DNVR)
DFHTLT TYPE=ENTRY, *
        TRMIDNT=/OP6
DFHTLT TYPE=FINAL
END

Example 2

DFHTLT TYPE=INITIAL, *
        SUFFIX=XX
DFHTLT TYPE=ENTRY, *
        TRMIDNT=(NYC,T361*LP,T362*LP/OP1)
DFHTLT TYPE=ENTRY, *
        TRMIDNT=(T363/OP2,T364/OP5,T365)
DFHTLT TYPE=FINAL
END
```

Figure 56. Terminal list table — example

Chapter 37. TST — temporary storage table

APAR PQ00933

introduced the TYPE=LOCAL macro and the null DATAID=(), and made associated changes to this section.

The temporary storage table (TST) is a list of generic names (or prefixes) used to identify sets of temporary storage queues. Generic names are formed from the leading characters of the appropriate queue names, and can be up to seven characters long.

- The generic names coded on a DFHTST TYPE=RECOVERY macro identify queues for which CICS provides backout of changes in the event of transaction failure or protection against system failure.
- The generic name coded on a DFHTST TYPE=REMOTE macro identifies queues for which CICS routes the temporary storage request to a remote CICS region, unless the remote system name (SYSIDNT) is the same as that of the local CICS. If SYSIDNT is the same name as the local CICS, the queues specified by the DATAID option are treated by CICS as local queues.
- The generic name coded on a DFHTST TYPE=LOCAL macro identifies queues as local queues that reside in the CICS region in which the TST is installed.
- The generic name coded on a DFHTST TYPE=SECURITY macro identifies queues for which resource security checking is required.

Note: DATAIDs using all eight characters define unique temporary storage queue names.

Choose a naming convention for queue names that enables you to define many queues with only a few generic names. This reduces considerably the task of TST definition.

APAR PN92338

added this note

Note: CICS searches the TST for the first prefix that satisfies the particular search criteria. For example, if CICS searches for temporary storage queue ABCDEFGH, and the TST contains prefix A followed by prefix AB, A is selected. To avoid this, define the less-generic entries to the TST before any more-generic entries, so that the first to be found is the least generic of all possible matches.

Note that when CICS is looking for DATAIDs to match against a TS queue name, it searches only the types of entry in which it is interested for that particular search. CICS searches:

- Local *and* remote entries when determining whether a queue is remote. Thus, local and

remote entries are regarded as one search category when CICS is matching a queue name against generic names.

- Recovery and remote entries when determining whether a queue is recoverable. However, if the leading characters of a queue name match **both** TYPE=RECOVERY and TYPE=REMOTE generic names, TYPE=REMOTE takes precedence, and the recovery option must be redefined in the local region in which the queue resides.
- Security entries only when determining whether a queue is subject to security.

When a task modifies temporary storage data designated as recoverable, the data is protected from modification by a concurrent task by enqueueing on the queue name. The queue name is not dequeued until the task terminates or issues a task syncpoint request to designate the end of a logical unit of work. At this time a log record is written to the system log data set to provide external information sufficient to recover the data if the system subsequently terminates abnormally.

Elements of DFHTST

The following macros are available to define the TST entries:

- Control section — DFHTST TYPE=INITIAL
- Recoverable temporary storage — DFHTST TYPE=RECOVERY
- Local temporary storage—DFHTST TYPE=LOCAL
- Remote temporary storage — DFHTST TYPE=REMOTE
- Temporary storage security checking — DFHTST TYPE=SECURITY
- End of temporary storage table — DFHTST TYPE=FINAL (see page 249).

Control section — DFHTST TYPE=INITIAL

The entry point and the beginning address for the temporary storage table being defined are established by the DFHTST TYPE=INITIAL macro.

Table 80. DFHTST TYPE=INITIAL

	DFHTST	TYPE=INITIAL [,SUFFIX=xx] [,TSAGE={0 number}]
--	--------	---

For general information about TYPE=INITIAL macros, including the use of the SUFFIX operand, see “TYPE=INITIAL” on page 248.

TSAGE={0|number}

Defines the aging limit of temporary storage data used by the temporary storage recovery program (DFHTSRP) during emergency restart of CICS. Data that is older than the specified limit will not be recovered. The value is specified in days with a maximum value of 512. A value of zero indicates that no data is to be purged on this basis.

- “\$\$” refers to temporary storage queues used by the BMS CMSG transaction when the PROTECT=YES option is specified on a START TRANSID command.

Recoverable temporary storage — DFHTST TYPE=RECOVERY

The generic names used to specify temporary storage queues for which recovery is applicable, are specified by the DFHTST TYPE=RECOVERY macro.

Table 81. DFHTST TYPE=RECOVERY

DFHTST	TYPE=RECOVERY ,DATAID=(<i>character-string</i> [, <i>character-string</i> ,...])()
--------	---

TYPE=RECOVERY

Code this to identify the temporary storage queue names that are recoverable. If a temporary storage queue name is such that it is defined by both a remote **and** a recovery DATAID, it is considered to be remote. Recoverability can be specified only in the CICS region in which the queue is local.

Note: TYPE=ENTRY is retained for compatibility with previous releases, and means exactly the same as TYPE=RECOVERY.

+ DATAID=(*character-string*[,*character-string*,...])()

Code this with one or more alphanumeric TS queue names that you want to be recoverable, where each name can be up to 8 characters in length. (See page 331 for information about generic names and matching criteria.)

character-string

Each character string can represent a generic queue name, or a unique TS queue name. Generic names are specified using 1 to 7 leading characters of TS queue names. DATAIDs that use all 8 characters define unique queues names.

Multiple names must be enclosed in parentheses, and separated by a comma. You can omit the parentheses if you specify only one name. Some CICS-generated TS queue names that you should consider for recovery are:

- “DF” refers to temporary storage queues used by CICS interval control for START commands with data, but which do not specify a REQID.
- “***” refers to temporary storage queues used by the BMS ROUTE command, and to those commands that use the PAGING operand.

0

This special (null) operand, without any value between the parentheses, is interpreted by CICS as specifying any queue that is not more explicitly specified by other DATAIDs.

Notes:

1. If a TST is generated with no TYPE=RECOVERY entries, no recovery processing is performed.
2. If an EXEC CICS START command is issued with any of the FROM, RTRANSID, RTERMID, or QUEUE parameters specified, and a REQID is not specified, CICS generates request identifications starting with the prefix “DF”. If recovery is required for these requests, the TST should be generated with the corresponding generic name.
3. All temporary storage queues used by restartable transactions (those defined with RESTART(YES) in the transaction resource definition) should be made recoverable (including those with the default DF prefix).
4. Only data on auxiliary storage can be made recoverable. Data written to main storage is not recoverable, regardless of any recovery options that you may specify.

+ Local temporary storage—DFHTST TYPE=LOCAL

The DFHTST TYPE=LOCAL macro defines temporary storage queue names that reside in the local CICS region in which the TST is installed. This macro enables you to define local queues without knowing the SYSIDNT (see the +SYSIDNT option on the DFHTST TYPE=REMOTE macro for more information).

Used in conjunction with the all-generic DATAID specified on the TYPE=REMOTE macro for remote queues, this macro can help you to simplify greatly the task of defining local and remote queues.

Table 82. DFHTST TYPE=LOCAL

DFHTST	TYPE=LOCAL ,DATAID=(<i>character-string</i> [, <i>character-string</i> ,...])()
--------	--

+ TYPE=LOCAL

Indicates that this TST entry defines a set of local temporary storage queues.

+ **DATAID=(character-string[,character-string,...])()**
 + Code this with one or more alphanumeric TS queue
 + names, where each name can be up to 8 characters in
 + length.

+ *character-string*
 + Each character string can represent a generic
 + queue name, or a unique TS queue name.
 + Typically, generic names are specified using 1 to 7
 + leading characters of TS queue names. DATAIDs
 + that use all 8 characters define unique queue
 + names.

+ Multiple names must be enclosed in parentheses,
 + and separated by a comma. You can omit the
 + parentheses if you specify only one name.

+ **()**
 + This special (null) operand, without any value
 + between the parentheses, is interpreted by CICS as
 + specifying any queue that is not more explicitly
 + specified by other DATAIDs. You can use this as a
 + catch-all in the following way:

- + • If certain queues, which reside in another
 + region, are specified on a TYPE=REMOTE
 + macro with suitable generic DATAIDs, you can
 + define all other queues as local by specifying
 + DATAID=() on the TYPE=LOCAL macro.

+ This null option on the TYPE=LOCAL macro is
 + mutually exclusive with DATAID=() on the
 + TYPE=REMOTE macro, and the TST macro returns
 + an assembly error if it is specified on both local and
 + remote entries. Thus, if you specify DATAID=() on
 + local TS queue entries, the TYPE=LOCAL macros
 + must follow all TYPE=REMOTE macros.

Remote temporary storage — DFHTST TYPE=REMOTE

The DFHTST TYPE=REMOTE macro defines temporary
 + storage queue names that reside in remote CICS regions
 + when CICS intercommunication facilities are being used.

Table 83. DFHTST TYPE=REMOTE

	DFHTST	TYPE=REMOTE ,DATAID=(<i>character-string</i> [, <i>character-string</i> ,...])() ,SYSIDNT=name [,RMTNAME= <i>character-string</i>]
--	--------	--

TYPE=REMOTE

Indicates that this TST entry defines a set of remote
 temporary storage queues.

+ **DATAID=(character-string[,character-string,...])()**
 + Code this with one or more alphanumeric TS queue
 + names, where each name can be up to 8 characters in
 + length. Use 1 to 7 leading characters from the leading

+ characters of queue names to form generic names of
 + those queues for which requests are to be routed to a
 + remote region. (See page 331 for information about
 + generic names and matching criteria.)

+ **Note:** You cannot use the list form of the DATAID
 + operand when RMTNAME is specified. If you
 + specify the RMTNAME parameter, the syntax for
 + DATAID is DATAID=*character-string*.

+ *character-string*
 + Each character string can represent a generic
 + queue name, or a unique TS queue name.
 + Typically, generic names are specified using 1 to 7
 + leading characters of TS queue names. The
 + generic names are those used by application
 + programs in the region in which this TST is installed.

+ Multiple names must be enclosed in parentheses,
 + and separated by a comma. You can omit the
 + parentheses if you specify only one name.

+ **()**
 + This special operand, without any value between the
 + parentheses, is interpreted by CICS as specifying
 + any queue that is not more explicitly specified by
 + other DATAIDs. You can use this as a catch-all in
 + the following way:

- + • If the queues with names beginning with letters
 + L, M, and N are local, and these are specified
 + on a TYPE=LOCAL macro with suitable generic
 + DATAIDs, you can define all other queues as
 + remote by specifying DATAID=() on the
 + TYPE=REMOTE macro, as follows:

```

DFHTST TYPE=LOCAL,      *
DATAID=(L,M,N)

*
DFHTST TYPE=REMOTE,    *
DATAID=()
    
```

+ The DATAID=() option on the TYPE=REMOTE macro is
 + mutually exclusive with DATAID=() on the TYPE=LOCAL
 + macro, and the TST macro returns an assembly error if
 + it is specified on both local and remote entries.

+ DATAID=() must be the last entry in a set of local and
 + remote entries. Thus, if you use DATAID=() on remote
 + TS queue entries, the TYPE=REMOTE macros must
 + follow any TYPE=LOCAL macros.

SYSIDNT=name

+ Identifies the region in which the remote temporary
 + storage queues reside. For a remote queue owned by
 + another CICS region, the 4-character alphanumeric
 + name specified must be the same as a REMOTENAME
 + specified in the CONNECTION definition.

+ You can use this parameter to specify the name of the
 + local region in which the TST is installed. When the
 + SYSIDNT operand matches the SYSIDNT specified on
 + the system initialization parameter, the TS queues that
 + match the DATAIDs are treated as local queues.

DFHTST TYPE=SECURITY

RMTNAME=character-string

Code this with the 1- to 8-character prefix that is to be used by CICS to replace that specified in the DATAID operand when a reference to the temporary storage queue is transmitted to a remote system or region. This operand defaults to the character string specified in the DATAID operand. The length of the character string specified in this operand must be the same as that in the DATAID operand. This mechanism allows access to a temporary storage queue in the remote system with the same name as one in the local system.

Temporary storage security checking — DFHTST TYPE=SECURITY

The DFHTST TYPE=SECURITY macro indicates that security checking is required for the temporary storage queues specified in the TST.

Table 84. DFHTST TYPE=SECURITY

DFHTST	TYPE=SECURITY
	,DATAID=(<i>character-string</i> [, <i>character-string</i> ,...])()

TYPE=SECURITY

Indicates that this TST entry defines a set of temporary storage queues that require security checking. You are recommended to code this operand for each temporary storage queue that can be browsed by the CEBR transaction.

+ **DATAID=(*character-string*[,*character-string*,...])()**

+ Code this with one or more alphanumeric TS queue names, where each name can be up to 8 characters in

+ length. Use 1 to 7 leading characters from the leading characters of queue names to form generic names of those queues that are subject to security checking. (See page 331 for information about generic names and matching criteria.)

Notes:

1. When this macro is used, a suitable profile (see the *CICS/ESA CICS-RACF Security Guide* for information about profiles) must be defined to the external security manager to control access to the TSQ. Otherwise, the macro will not have the intended effect.
2. The full TSQ name, rather than the DATAID, is passed to the security manager. (CICS/ESA 3.3 and earlier releases passed the DATAID.)

character-string

Each character string can represent a generic queue name, or a unique TS queue name.

Typically, generic names are specified using 1 to 7 leading characters of TS queue names. The generic names are those used by application programs in the region in which this TST is installed.

Multiple names must be enclosed in parentheses, and separated by a comma. You can omit the parentheses if you specify only one name.

()

This null operand, without any value between the parentheses, is interpreted by CICS as specifying any queue that is not more explicitly specified by other DATAIDs.

DFHXLT example

<pre> DFHXLT TYPE=INITIAL, SUFFIX=IN * * * DFHXLT TYPE=ENTRY,TASKREQ=PF5 * * * DFHXLT TYPE=ENTRY,TRANSID=(USR1,USR2) DFHXLT TYPE=ENTRY,TRANSID='AA,1' DFHXLT TYPE=ENTRY,TRANSID='AA,2' DFHXLT TYPE=FINAL END DFHXLT TYPE=INITIAL, SUFFIX=G1 * * DFHXLT TYPE=ENTRY,TRANSID=(TSSA,TSRA) DFHXLT TYPE=ENTRY,TRANSID=(TDSA,TDRA) DFHXLT TYPE=ENTRY,TRANSID=ICSA DFHXLT TYPE=FINAL END </pre>	<pre> LIST OF TRANSACTIONS * THAT WILL BE ACCEPTED DURING THE FIRST QUIESCE PHASE OF SYSTEM TERMINATION. (TASKREQ MUST ALSO BE DEFINED IN THE CSD AND INSTALLED IN THE RUNNING CICS SYSTEM. AN ENTRY FOR THE XLT MUST BE MADE IN THE CSD.) LIST OF LOGICALLY RELATED* TRANSIDS TO BE ENABLED OR DISABLED BY MASTER TERMINAL. (TRANSIDS MUST ALSO BE DEFINED IN THE CSD AND INSTALLED IN THE RUNNING CICS SYSTEM.) </pre>
--	---

Figure 58. Transaction list table — example

Appendix A. Obsolete attributes retained for compatibility

The attributes described in this chapter are not valid in CICS/ESA 4.1, but are supported to provide CSD compatibility for earlier releases of CICS where they are still valid. See "Compatibility mode (CSD sharing)" on page 19 for more information on compatibility mode.

Table 87 on page 345 shows which resource or resources each attribute is associated with, and which release or releases it was supported in.

BINDPASSWORD(password) (APPC only)

Enter a password of up to 16 hexadecimal characters (0-9, A-F). A password of less than 16 characters will be padded on the right with hexadecimal zeros.

CICS masks the password you supply to avoid unauthorized access. You should therefore find a safe way of recording the password.

If you supply a password, an identical password must be supplied in the remote system to ensure bind-time security, allowing a connection to be established.

EXTSEC({NO|YES})

Indicates whether an external security manager (for example, RACF) is to be used for transaction security or resource security checking.

NO

Only the security facilities provided by CICS will be used by this transaction.

YES

An external security manager may be used by this transaction.

INSERVICE({YES|NO})

Indicates whether the session(s) can be used for communication. This attribute applies only to LUTYPE 6.1 ISC sessions. It is invalid for LUTYPE 6.2, and is ignored for MRO sessions. For MRO the status (in service or out of service) is determined by the status of the corresponding MRO CONNECTION.

YES

Transactions may be initiated and messages may automatically be sent across the session(s).

NO

The session(s) can neither receive messages nor transmit input.

OPERID(code)

The 3-character operator identifier associated with the sessions. You should use OPERID if you are not specifying SECURITYNAME on the CONNECTION definition. Specifying OPERID is the **only** way of having an operator identifier if you have preset security (by specifying OPERRSL and OPERSECURITY).

OPERPRIORITY({0|number})

The operator priority code to be used to determine the task processing priority for each transaction attached to the sessions. The code may be any value from 0 through 255. You should use OPERPRIORITY if you are not specifying SECURITYNAME on the CONNECTION definition. Specifying OPERPRIORITY is the **only** way of having an operator priority code if you have preset security (by specifying OPERRSL and OPERSECURITY).

OPERRSL({0|number[,...]})

The resource security key for these sessions.

number[,...]

The preset resource security keys for these sessions. The OPERRSL keys are checked to see that they include the resource RSL value, by transactions that request RSL checking (RSLC(YES)). They are referenced for function shipping and distributed transaction processing requests. The OPERRSL keys comprise one or

more decimal values from 1 through 24. You can specify more than one value as an inclusive range, using a dash, for example: 5-12 , or as a series of numbers separated by commas, for example: 5,6,7,8,9,10,11,12. These two examples are equivalent. You can use dashes and commas in the same specification if you need to.

You should specify OPERRSL keys if you are not specifying SECURITYNAME on the CONNECTION definition. However, you should be aware that if you specify OPERRSL keys for the sessions, you cannot have a sign-on, using SECURITYNAME, when the link is established. (For more information, see the *CICS/ESA Intercommunication Guide*.) Note that the OPERRSL keys give access only to resources with the RSL values actually specified in the OPERRSL keys, not to resources with lower RSL values.

- 0** The sessions have no OPERRSL keys specified and will not have access to any resources through transactions with RSLC(YES), except resources with RSL(PUBLIC).

OPERSECURITY({1|number[,...]})

The preset transaction security keys for the device. The transaction security keys are checked to see that they include the security value (TRANSEC) for a transaction about to be attached. They are referenced for function shipping and distributed transaction processing requests.

The security keys comprise one or more decimal values from 1 through 64. You can specify these values in the same way as for OPERRSL, above. In addition to the values you specify, a value of 1 will also be assumed. The default value of 1 gives access to all unsecured transactions, because the default TRANSEC value is 1. For example: 5-10,12 is translated into: 1,5,6,7,8,9,10,12.

You should use OPERSECURITY if you are not specifying SECURITYNAME on the CONNECTION definition. However, you should be aware that if you specify OPERSECURITY keys for the sessions, you cannot have a sign-on, using SECURITYNAME, when the link is established. (For more information, see the *CICS/ESA Intercommunication Guide*.)

PRIMEDSIZE({0|value})

The primed storage allocation size in bytes.

- 0** CICS will take care of the storage for the control blocks.

Note: You should leave PRIMEDSIZE as 0 if this TRANSACTION definition has been migrated with ANTICPG=YES.

value

This value must not exceed 65520 bytes and, if specified at all, must include an allowance of 2800 bytes for CICS control blocks, and an allowance for the size of the TWA.

Storage acquired by a GETMAIN within the primed storage area is never freed (that is, the corresponding FREEMAIN is ignored).

Note that storage accounting areas within the primed storage allocation are doubleword-aligned, instead of the normal double-doubleword-aligned.

RECOVNOTIFY({NONE|MESSAGE|TRANSACTION})

This specifies whether, and how, the terminal user is notified that an XRF takeover has occurred. The terminal user may have to do something, such as sign on again, after a takeover. If so, they need to be notified.

NONE

There is no notification that a takeover has occurred.

MESSAGE

A message is displayed on the screen to say that the system has recovered. There are two BMS maps, DFHXRC1 and DFHXRC2, in map set DFHXMSG for the message. MESSAGE, rather than TRANSACTION, minimizes the takeover time.

The terminal must be defined with the ATI(YES) option, and must be capable of displaying a BMS map.

TRANSACTION

A transaction is initiated at the terminal. The name of the transaction is specified by the RMTRAN system initialization parameter. (The default transaction for this is the one specified in the GMTRAN system initialization parameter: the good-morning transaction.) TRANSACTION is more versatile than MESSAGE.

The terminal must be defined with ATI(YES).

RESSECNUM({0|value|PUBLIC})

The resource security value to be associated with this file. This attribute is used when an EXEC command is executed within a transaction that has been defined with RESSEC(YES), and the command is attempting to reference this file.

0 This means that any transaction defined with RESSEC(YES) will not be allowed access to the file.

value

The resource security value, in the range 1 through 24. When a transaction defined with RESSEC(YES) attempts to reference this file, the value is checked against the keys derived from RESSECKEYS either in the sign-on table, or from the TERMINAL definition. If one of these keys matches the RESSECNUM value, the transaction is allowed access to the file.

PUBLIC

Any transaction is allowed access to the file, regardless of whether security checking is specified or not.

RSL

Enter the resource security value to be associated with this resource. This operand is used when an EXEC command is executed within a transaction that has been defined with RSLC(YES), and the command is attempting to reference the partition set.

0 This means that any transaction defined with RSLC(YES) will not be allowed access to the partition set.

value

The resource security value, in the range 1 through 24. When a transaction defined with RSLC(YES) attempts to reference this partition set, the value is checked against the keys derived either from the RSLKEY in the sign-on table, or from the OPERRSL on the TERMINAL definition. If one of these keys matches the RSL value, the transaction is allowed access to the partition set.

PUBLIC

Any transaction is allowed access to the partition set, regardless of whether no security checking or RSL checking is specified. However, if an external security manager is in force, it checks access authorities no matter what RSL value (including PUBLIC) has been defined for the resource.

TCLASS({NO|value})

The class associated with the task.

NO

No class is assigned to the task.

value

A value (from decimal 1 to 10) of the class associated with a task.

Note: You must not specify a TCLASS for CICS-supplied transactions because their initiation could be inhibited if the class threshold was reached.

TRANSACTION(name)

This allows the one transaction specified to be initiated from this device.

The name can be up to 4 characters in length. The acceptable characters are: A-Z a-z 0-9 \$ @ # . / - _ % & ¢ ? ! : | " = ~ , ; < and >.

If you code this operand for a 3270 display, the only CICS functions the operator will be able to invoke – other than this transaction – are paging commands and print requests.

TRANSEC

The transaction security value, in the range 1 through 64. When a user attempts to initiate the transaction, or when it is automatically initiated (through transient data or interval control), this value is matched against the user's security keys defined in the DFHSNT SCTYKEY operand or, if the user is not signed on, the security keys defined in OPERSECURITY on the TERMINAL definition. If the TRANSEC value is present in the security keys, the transaction is initiated.

Because all users and terminals have a security key of 1, any transaction with the default TRANSEC value of 1 is an unsecured transaction, and as such, it can be initiated by any user on the CICS system, whether they are signed on or not.

Obsolete attributes — when were they supported?

Table 87 shows which resource or resources each attribute is associated with, and which release or releases it was supported in.

<i>Table 87. Obsolete attributes and their valid releases of CICS</i>		
Attribute	Resource type	Supported in
BINDPASSWORD	CONNECTION TERMINAL	CICS/ESA 3.3 CICS/ESA 3.2.1 CICS/ESA 3.1.1 CICS/MVS 2.1
EXTSEC	TRANSACTION	CICS/MVS 2.1
INSERVICE	SESSION	CICS/ESA 3.2.1 CICS/ESA 3.1.1 CICS/MVS 2.1.2
OPERID	SESSION TERMINAL	CICS/ESA 3.1.1 CICS/MVS 2.1.2
OPERPRIORITY	SESSION TERMINAL	CICS/ESA 3.1.1 CICS/MVS 2.1
OPERRSL	SESSION TERMINAL	CICS/MVS 2.1
OPERSECURITY	SESSION TERMINAL	CICS/MVS 2.1
PRIMEDSIZE	TRANSACTION	CICS/MVS 2.1
RECOVNOTIFY	SESSION	CICS/ESA 3.3 CICS/ESA 3.2.1 CICS/ESA 3.1.1 CICS/MVS 2.1
RESSECNUM	FILE	CICS/ESA 3.3 CICS/ESA 3.2.1 CICS/ESA 3.1.1
RSL	MAPSET PARTITIONSET PROGRAM TRANSACTION	CICS/MVS 2.1
TCLASS	TRANSACTION	CICS/ESA 3.3 CICS/ESA 3.2.1 CICS/ESA 3.1.1 CICS/MVS 2.1
TRANSACTION	SESSION	CICS/MVS 2.1
TRANSEC	TRANSACTION	CICS/ESA 3.1.1 CICS/MVS 2.1

Appendix B. CICS-supplied resource definitions, groups, and lists

This chapter lists the resource definitions, groups, and lists supplied by IBM.

You initialize the CSD by using the DFHCSDUP INITIALIZE command. (This command has no operands.) Following initialization, the CSD contains two categories of resource definition groups.

1. Groups named in DFHLIST, essential for using RDO and other CICS-supplied transactions
2. Groups of definitions for the sample application programs, which are described in the *CICS/ESA Sample Applications Guide*.

This chapter also lists the CICS transactions, the TYPETERM definitions, model TERMINAL definitions, and PROFILE definitions that are supplied by IBM. These definitions are in four groups:

- DFHTYPE — TYPETERM definitions
- DFHTERM — model TERMINAL definitions for automatic installation
- DFHISC — PROFILE definitions for intersystem communication sessions
- DFHSTAND — PROFILE definitions.

DFHLIST definitions

<i>Table 88 (Page 1 of 3). DFHLIST resource definitions</i>				
Group Name	Description	Programs (Map sets, Profiles, Terminals, Tranclasses or Typeterms as indicated)		Transactions
DFHAKP	Activity keypoint program for recovery/restart	DFHAKP		CSKP
DFHBACK	Dynamic backout	DFHDBP1\$	DFHDBP2\$	-
DFHBMS	Basic mapping support	DFHTPQ	DFHTPR	CSPG CSPQ CSPS
DFHCONS	Write to CPU console	DFHCWTO		CWTO
DFHDBCTL	DBCTL transactions	DFHDBAT	DFHDBCON	CDBC CDBD
		DFHDBCT	DFHDBDI	CDBI CDBN
		DFHDBDSC	DFHDBIQ	CDBO CDBT
		DFHDBME	DFHDBUEX	
		Map set:		
		DFHDBIE	DFHDBNE	
DFHDB2	DB2 support	DSNCUEXT	DSN2COM0	DSNC
		DSN2COM1	DSN2COM2	
		DSN2EDF1	DSN2EXT1	
		DSN2EXT2	DSN2MSG0	
		DSN2STOP	DSN2STRT	
DFHDLI	DL/I data sharing	DFHBRCP	DFHDLG	CBRC CSGX
		DFHDLS	DFHEBRCT	CSSX
DFHEDF	Execution diagnostic facility	DFHDBMS	DFHDBTI	CEBR CEDF
		DFHEDFBR	DFHEDFD	
		DFHEDFP	DFHEDFR	
		DFHEDFX	DFHEIGDS	
		DFHEITAB		
		Map sets:		
		DFHEDFM		
DFHEDP	EXEC DLI user exit	DFHEDP		-
DFHFE	FE terminal test facility	DFHFEP	DFHTRAP	CSFE
DFHFEPI	Front End Programming Interface	DFHEITSZ	DFHSZRMP	CSZI
DFHHARDC	3270 Printer - VTAM	DFHP3270		CSPP
DFHINQUI	Command definition	DFHEITBS		-
DFHINTER	Command interpreter	DFHECID	DFHECIP	CECI CECS
		DFHECSP		
DFHISC	Intersystem communication	DFHCRNP	DFHCRP	CLS1 CLS2
		DFHCRQ	DFHCRR	CLS3 CMPX
		DFHCRS	DFHCRSP	CRSQ CRSR
		DFHCRT	DFHMIRS	CRTE CSMI
		DFHDYP	DFHLUP	CSM1 CSM2
		DFHMXP	DFHRTC	CSM3 CSM5
		DFHRTE	DFHCCNV	CSNC CSSF
		Profiles:		CXRT CVMI
		DFHCICSF	DFHCICSR	
		DFHCICSS		
DFHJRNL	Journaling	DFHJCBSP	DFHJCC	-
		DFHJCEOV	DFHJCI	
		DFHJCIOE	DFHJCKOJ	
		DFHJCO	DFHJCSDJ	
DFHMISC	Miscellaneous programs	DFHNEP	DFHPEP	-
		DFHRTY		
DFHMSWIT	Message switching program	DFHMSP		CMSG
DFHOPCLS	Dynamic open/close program	DFHFCU		CSFU

Table 88 (Page 2 of 3). DFHLIST resource definitions

Group Name	Description	Programs (Map sets, Profiles, Terminals, Tranclasses or Typeterms as indicated)	Transactions
DFHOPER	Operator programs	DFHCETRA DFHCETRB DFHCETRC DFHCETRD DFHEITMT DFHEITOT DFHEITST DFHEMTA DFHEMTD DFHEMTP DFHEOTP DFHESTP Map sets: DFHCTRH DFHCTRM	CEMT CEOT CEST CETR
DFHPGAIP	Autoinstall for programs	DFHPGADX DFHPGAHX DFHPGALX DFHPGAOX DFHPGAPG Map sets: DFHPGAMP Partition sets: DFHPGAPT	-
Note: Group DFHPLI is withdrawn. For PL/I Version 2 or later, refer to the PL/I installation manual, for example, <i>OS PL/I Version 2 Installation and Customization under MVS Guide</i> , for a complete list of the required program entries.			
DFHRMI	Resource manager interface	DFHRMSY	CRSY
DFHRSEND	VTAM resend program	DFHZRSP	CSRS
DFHRSPLG	VTAM response logging program	DFHZRLG	CSLG
DFHSIGN	Sign-on/sign-off programs and table	DFHSFP DFHSNP DFHSNT Map sets: DFHSNLE DFHSNSE	CESF CESN
DFHSPI	Resource definition online	DFHAMP DFHDMP DFHEDAD DFHEDAP DFHEITSP DFHPUP DFHTBS DFHTOR DFHZATA DFHZATD DFHZATDX DFHZATR DFHZATS DFHZCQ DFHZCTDX DFHZDTDX DFHZPTDX	CATA CATD CATR CATS CDTS CEDA CEDA CEDB CEDC CFTS CITS CMTS
DFHSTAND	Standard CICS application programs	DFHACP DFHCXCU DFHPSIP DFHQRY DFHSTP DFHTACP DFHTEP DFHTEPT DFHZXCU DFHZXRE DFHZXST Profiles: DFHCICSA DFHCICSE DFHCICST DFHCICSV DFHCICSP Map sets: DFHXMSG	CQRY CSAC CSTE CXCU CXRE
DFHTCL	Compatibility TRANCLASS definitions	Tranclasses DFHTCL00 DFHTCL01 DFHTCL03 DFHTCL04 DFHTCL05 DFHTCL06 DFHTCL07 DFHTCL08 DFHTCL09 DFHTCL10	-

CICS-supplied definitions

Table 88 (Page 3 of 3). DFHLIST resource definitions			
Group Name	Description	Programs (Map sets, Profiles, Terminals, Tranclasses or Typeterms as indicated)	Transactions
DFHTERM	Model TERMINAL definitions	Terminals: LU2 LU3 APPC SCSP 3270 3284 L0E2 L0M2 L0M3 L0M4 L0M5 L2E2 L2M2 L2E3 L2M3 L2E4 L2M4 L2M5	-
DFHTYPE	TYPETERM definitions	Typeterms: DFHCONS DFHLU2 DFHLU3 DFHLU62T DFHSCSP DFH3270 DFH3270P DFHLU0E2 DFHLU0M2 DFHLU0M3 DFHLU0M4 DFHLU0M5 DFHLU2E2 DFHLU2M2 DFHLU2E3 DFHLU2M3 DFHLU2E4 DFHLU2M4 DFHLU2M5	-
DFHVTAM	VTAM programs	DFHGMM DFHZNAC DFHZNEP	CSGM CSNE
DFHVTAMP	VTAM terminal control print key function	DFHCPY DFHEXI DFHPRK DFHRKB	CSCY CSPK CSRK
Note: The DFHTYPE typeterm definitions match the VTAM-supplied LOGMODE definitions. If you are not using the supplied VTAM LOGMODES, you may need to modify the DFHTYPE TYPETERM definitions. For programming information on VTAM LOGMODE definitions, see the <i>CICS/ESA Customization Guide</i> .			

CICS-supplied groups not in DFHLIST

Table 89. Resource definitions outwith DFHLIST			
Group Name	Description	Programs (Map set or File as indicated)	Transactions
DFHCMAC	CICS online messages and codes	DFHCMAC Map set: DFHCMCM File: DFHCMACD	CMAC (alias CHLP)
Note: DFHCMAC is not included in DFHLIST for reasons of compatibility with earlier releases. If you want to use the online messages and codes transaction, you must add DFHCMAC to your start-up group list.			
DFHMISC3	Miscellaneous group	DFHNET	-
Note: DFHMISC3 is not included in DFHLIST for reasons of compatibility with earlier releases. If you require it in your system, you must add it to your start-up group list.			

CICS-supplied compatibility groups

<i>Table 90 (Page 1 of 2). Group definitions supplied for compatibility with earlier releases</i>				
Group Name	Description	Programs (Map set or File as indicated)		Transactions
DFHCOMP1	Required for compatibility with releases 1.7 and 2.1	DFHCCMF DFHDMP DFHEMB DFHEMD DFHEMF DFHEMH DFHFCS DFHMTPB DFHMTPD DFHMTPF DFHRTY DFHSTLK DFHSTSP DFHSTTR DFHTRNSM DFHVAP DFSHLPI	DFHCMON DFHEMA DFHEMC DFHEME DFHEMG DFHEMI DFHMTPA DFHMTPC DFHMTPD DFHMTPG DFHSTKC DFHSTPD DFHSTTD DFHTAJP DFHTRNSN DFHZCQ	CAUT CCMF CDTS CECI CECS CEDA CEDB CEDC CEOT CEST CFTS CITS CMSC CMTS CQRY CRSR CSCY CSFE CSIR CSMT CSNC CSOT CSPK CSPP CSRK CSSC CSSF CSSN CSST CSTA CSTT CVST CWTO CXCU 8888 9999
DFHCOMP2	Required for compatibility with releases 1.7, 2.1, and 3.1	DFHECID DFHECSP DFHEIPRT DFHEIPSH DFHEIQDS DFHEIQIR DFHEIQMT DFHEIQSC DFHEIQSK DFHEIQSP DFHEIQST DFHEIQSX DFHEIQVT DFHESE DFHEITAB DFHETRX DFHOCP IBMBCCLA IBMBEOCA IBMBETBA IBMBETIA IBMBETPA IBMBETTA IBMFESMA IBMFKCSA IBMFKPTA IBMFKCA IBMFPGDA IBMFSTVA	DFHECIP DFHEDCP DFHEIPSE DFHEIQDN DFHEIQDU DFHEIQMS DFHEIQSA DFHEIQSJ DFHEIQSM DFHEIQSQ DFHEIQSV DFHEIQTR DFHEOP DFHESN DFHEITBS DFHMATOC DFHUAKP IBMBCCRA IBMBETAA IBMBETCA IBMBETOA IBMBETQA IBMFEFCA IBMFESNA IBMFKMRA IBMFKTBA IBMFKTRA IBMFPMRA	CSNC
DFHCOMP3	Required for compatibility with releases 1.7, 2.1, 3.1, and 3.2	DFHAKP DFHCHS DFHDBP2\$ DFHDLS DFHFUCU DFHJCI DFHJCKOJ DFHMIR DFHNET DFHSTP	DFHBRCP DFHDBAT DFHDLG DFHEBRCT DFHJCC DFHJCIOE DFHJCO DFHMIRVM DFHRTY DFHTACP	CATS CEMT CEOT CEST CPMI CRSR CSFE CSJC CSMI CSM1 CSM2 CSM3 CSM5 CSNE CSPP CSPQ CSPS CSRS CVMI

CICS-supplied definitions

Group Name	Description	Programs (Map set or File as indicated)		Transactions
DFHCOMP4	Required for compatibility with release 3.3	DFHCRP DFHNEP DFHSNT	DFHCSSC DFHRTY	CEDF CLS1 CLS3 CSPG CSSC

If, after upgrading a CSD to CICS/ESA 4.1, you plan to share the CSD with earlier releases of CICS, you must include the appropriate DFHCOMPx compatibility groups in your startup group list. Table 91 shows you which DFHCOMP groups you need to include for the earlier releases.

Release the CSD is being shared with	The CICS/ESA release level of the CSD			
	4.1	3.3	3.2.1	3.1.1
3.3	DFHCOMP4	None	None	None
3.2.1	DFHCOMP4 DFHCOMP3	DFHCOMP3	None	None
3.1.1	DFHCOMP4 DFHCOMP3 DFHCOMP2	DFHCOMP3 DFHCOMP2	DFHCOMP2	None
CICS/MVS 2.1.2 and earlier	DFHCOMP4 DFHCOMP3 DFHCOMP2 DFHCOMP1	DFHCOMP3 DFHCOMP2 DFHCOMP1	DFHCOMP2 DFHCOMP1	DFHCOMP1

Note: It is important that you install these groups in the order shown. For example, in a CICS/MVS 2.1.2 region, you must append the compatibility groups DFHCOMP4 followed by DFHCOMP3, DFHCOMP2, and DFHCOMP1, at the end of your group list.

The sample application program groups

These resource definitions are needed to run the sample application programs supplied with CICS. The groups are not named in DFHLIST.

<i>Table 92 (Page 1 of 4). CICS sample applications – resource definitions</i>			
Group Name	Language	Description	Resources Defined
DFH\$ACCT	COBOL	<i>CICS Application Programming Primer (VS COBOL II) samples</i>	Map set: ACCTSET Programs: ACCT00 ACCT01 ACCT02 ACCT03 ACCT04 Transactions: ACCT ACEL ACLG AC01 AC02 AC03 AC05 AC06
DFH\$AFLA	Assembler	FILEA sample applications	Map sets: DFH\$AGA DFH\$AGB DFH\$AGC DFH\$AGD DFH\$AGK DFH\$AGL Programs: DFH\$AALL DFH\$ABRW DFH\$ACOM DFH\$AMNU DFH\$AREN DFH\$AREP Transactions: AADD ABRW AINQ AMNU AORD AORQ AREP AUPD
DFH\$BMSP	COBOL & PL/I	BMS partition support applications	Partitionset: DFH0PS Map sets: DFH0CGP DFH\$PGP Programs: DFH0CPKO DFH0CPLA DFH\$PPKO DFH\$PPLA Transactions: PPKO PPLA XPKO XPLA
DFH\$CFLA	COBOL	FILEA sample applications	Map sets: DFH0CGA DFH0CGB DFH0CGC DFH0CGD DFH0CGK DFH0CGL Programs: DFH0CALL DFH0CBRW DFH0CCOM DFH0CMNU DFH0CREN DFH0CREP Transactions: ADDS BRWS INQY MENU OREN OREQ REPT UPDT
DFH\$CNLS		Sample console definitions	Typeterms: DFH\$JCLC DFH\$CONS Terminals: CJCL CNLS CN02

Sample programs

Table 92 (Page 2 of 4). CICS sample applications – resource definitions			
Group Name	Language	Description	Resources Defined
DFH\$CTXT	COBOL	CUA text model application	Files: DFH0FCAI DFH0FCUS DFH0FHLP Map sets: DFH0AB DFH0ABT DFH0BRW DFH0DEL DFH0FPD DFH0HLP DFH0HP DFH0HPD DFH0LST DFH0NEW DFH0OPN DFH0PRT DFH0SAS DFH0T1 DFH0UPD Programs: DFH0VAB DFH0VABT DFH0VBRW DFH0VDEL DFH0VDQ DFH0VHLP DFH0VHP DFH0VLIO DFH0VLST DFH0VNEW DFH0VOL DFH0VOPN DFH0PRT DFH0VRIO DFH0VSAS DFH0VTBL DFH0VT1 DFH0VUPD Profiles: DFH\$CUA2 Transactions: AC2A AC2C AC2D AC2E AC2F AC20 AC21 AC22 AC23 AC24 AC25 AC26 AC27 AC28 DELQ
DFH\$DFLA	C/370	FILEA sample applications	Map sets: DFH\$DGA DFH\$DGB DFH\$DGC DFH\$DGD DFH\$DGK DFH\$DGL Programs: DFH\$DALL DFH\$DBRW DFH\$DCOM DFH\$DMNU DFH\$DREN DFH\$DREP Transactions: DADD DBRW DINQ DMNU DORD DORQ DREP DUPD
DFH\$DLIV		IMS installation verification procedure	Programs: DFH\$DLAC DFH\$DLAE DFH\$DLCC DFH\$DLCE DFH\$DLPC DFH\$DLPE Transactions: ASMC ASME COBC COBE PLIC PLIE
DFH\$EXCI		EXCI batch call interface samples	Connections: EXCG EXCS Program: DFH\$AXCS Sessions: EXCG EXCS Transaction: EXCI
DFH\$FILA		FILEA samples	File: FILEA

Table 92 (Page 3 of 4). CICS sample applications – resource definitions

Group Name	Language	Description	Resources Defined
DFH\$ICOM	Assembler	Intersystem communication (ISC)	Map sets: DFH\$IGB DFH\$IGC DFH\$IGS DFH\$IGX DFH\$IG1 DFH\$IG2 Programs: DFH\$ICIC DFH\$IFBL DFH\$IFBR DFH\$IMSN DFH\$IMSO DFH\$IQRD DFH\$IQRL DFH\$IQRR DFH\$IQXL DFH\$IQXR Transactions: ICIC IFBL IFBR IMSN IMSO IQRD IQRL IQRR IQXL IQXR
DFHMROAR		Starter MRO systems	Connections: CICD CICT Sessions: CICSRD CICSRT
DFHMRODR		Starter MRO systems	Connections: CICA Sessions: CICSRA
DFHMROFA		Starter MRO systems	File: FILEA Map sets: DFH\$AGA DFH\$AGB DFH\$AGC DFH\$AGD DFH\$AGK DFH\$AGL Programs: DFH\$AALL DFH\$ABRW DFH\$ACOM DFH\$AMNU DFH\$AREN DFH\$AREP Transactions: AADD ABRW AINQ AMNU AORD AORQ AREP AUPD
DFHMROFD		Starter MRO systems	File: FILEA
DFHMROFT		Starter MRO systems	Transactions: AADD ABRW AINQ AMNU AORD AORQ AREP AUPD
DFHMROTR		Starter MRO systems	Connections: CICA Sessions: CICSRA
DFHAI62		Starter APPC connections	Program: DFHZATDY Connections: CBPS CBSS CCPS Sessions: CBPS CBSS CCPS

Sample programs

Table 92 (Page 4 of 4). CICS sample applications – resource definitions			
Group Name	Language	Description	Resources Defined
DFH\$PFLA	PL/I	FILEA sample applications	Map sets: DFH\$PGA DFH\$PGB DFH\$PGC DFH\$PGD DFH\$PGK DFH\$PGL Programs: DFH\$PALL DFH\$PBRW DFH\$PCOM DFH\$PMNU DFH\$PREN DFH\$PREP Transactions: PADD PBRW PINQ PMNU PORD PORQ PREP PUPD
DFH\$UTIL	Assembler	Transient data utility dynamic allocation	Programs: DFH\$TDWT DFH99 Transactions: ADYN TDWT
DFH\$VTAM		Terminal definitions	Typeterms: DFH\$L77 DFH\$L78 DFH\$L79 DFH\$L86 Terminals: L77C L77D L78A L79A L86A
DFH\$SXP		Message domain exits	Programs: DFH\$SXP1 DFH\$SXP2 DFH\$SXP3 DFH\$SXP4 DFH\$SXP5 DFH\$SXP6
DFH\$STAT		Statistics	Program: DFH\$STED
DFH\$SIGN		Some kind of transactions	Map sets: DFH\$NLE DFH\$NSE Programs: DFH\$CEGN DFH\$CESC DFH\$SFP DFH\$SNP Transactions: CEGN CESC CESF CESN

CICS transactions supplied by IBM

The CICS transactions supplied by IBM are listed here in alphabetic order, together with the name of the program invoked by the transaction.

Table 93 (Page 1 of 4). CICS transactions supplied by IBM

Transaction	Group	Program
AADD	DFH\$AFLA	DFH\$AALL
AADD	DFHMROFA	DFH\$AALL
AADD	DFHMROFT	
ABRW	DFH\$AFLA	DFH\$ABRW
ABRW	DFHMROFA	DFH\$ABRW
ABRW	DFHMROFT	
ACCT	DFH\$ACCT	ACCT00
ACEL	DFH\$ACCT	ACCT03
ACLG	DFH\$ACCT	ACCT03
AC01	DFH\$ACCT	ACCT01
AC02	DFH\$ACCT	ACCT02
AC03	DFH\$ACCT	ACCT03
AC05	DFH\$ACCT	ACCT03
AC06	DFH\$ACCT	ACCT03
AC2A	DFH\$CTXT	DFH0VSAS
AC2C	DFH\$CTXT	DFH0VHLP
AC2D	DFH\$CTXT	DFH0VAB
AC2E	DFH\$CTXT	DFH0VHP
AC2F	DFH\$CTXT	DFH0VABT
AC20	DFH\$CTXT	DFH0VT1
AC21	DFH\$CTXT	DFH0VOL
AC22	DFH\$CTXT	DFH0VOPN
AC23	DFH\$CTXT	DFH0VLST
AC24	DFH\$CTXT	DFH0VNEW
AC25	DFH\$CTXT	DFH0VBRW
AC26	DFH\$CTXT	DFH0VUPD
AC27	DFH\$CTXT	DFH0VDEL
AC28	DFH\$CTXT	DFH0VPRT
ADDS	DFH\$CFLA	DFH0CALL
ADYN	DFH\$UTIL	DFH99
AINQ	DFH\$AFLA	DFH\$AALL
AINQ	DFHMROFA	DFH\$AALL
AINQ	DFHMROFT	
AMNU	DFH\$AFLA	DFH\$AMNU
AMNU	DFHMROFA	DFH\$AMNU
AMNU	DFHMROFT	
AORD	DFH\$AFLA	DFH\$AREN
AORD	DFHMROFA	DFH\$AREN
AORD	DFHMROFT	
AORQ	DFH\$AFLA	DFH\$ACOM
AORQ	DFHMROFA	DFH\$ACOM
AORQ	DFHMROFT	
AREP	DFH\$AFLA	DFH\$AREP
AREP	DFHMROFA	DFH\$AREP
AREP	DFHMROFT	
ASMC	DFH\$DLIV	DFH\$DLAC
ASME	DFH\$DLIV	DFH\$DLAE
AUPD	DFH\$AFLA	DFH\$AALL
AUPD	DFHMROFA	DFH\$AALL
AUPD	DFHMROFT	
BRWS	DFH\$CFLA	DFH0CBRW
CATA	DFH\$SPI	DFHZATA
CATD	DFH\$SPI	DFHZATD
CATR	DFH\$SPI	DFHZATR
CATS	DFHCOMP3	DFHZATS
CAUT	DFHCOMP1	DFHSTSP
CBRC	DFHDLI	DFHBRCP

Table 93 (Page 2 of 4). CICS transactions supplied by IBM

Transaction	Group	Program
CCMF	DFHCOMP1	DFHCCMF
CDBC	DFHDBCTL	DFHDBME
CDBD	DFHDBCTL	DFHDBDI
CDBI	DFHDBCTL	DFHDBIQ
CDBN	DFHDBCTL	DFHDBCON
CDBO	DFHDBCTL	DFHDBCT
CDBT	DFHDBCTL	DFHDBDSC
CDTS	DFHSPI	DFHZATS
CEBR	DFHEDF	DFHEDFBR
CECI	DFHINTER	DFHECIP
CECS	DFHINTER	DFHECSP
CEDA	DFHSPI	DFHEDAP
CEDB	DFHSPI	DFHEDAP
CEDC	DFHSPI	DFHEDAP
CEDF	DFHEDF	DFHEDFP
CEGN	DFHSIGN	DFHCEGN
CEHP	DFHISC	DFHCHS
CEHS	DFHISC	DFHCHS
CEID	DFHDCE	DFHCEID
CEMT	DFHOPER	DFHEMTP
CEOT	DFHOPER	DFHEOTP
CESC	DFHSIGN	DFHCESC
CESF	DFHSIGN	DFHSFP
CESN	DFHSIGN	DFHSNP
CEST	DFHOPER	DFHESTP
CEST	DFHCOMP1	DFHESTP
CEST	DFHCOMP3	DFHESTP
CETR	DFHOPER	DFHCETRA
CFTS	DFHSPI	DFHZATS
CITS	DFHSPI	DFHZATS
CLS1	DFHISC	DFHZLS1
CLS2	DFHISC	DFHLUP
CLS3	DFHISC	DFHLUP
CLS4	DFHISC	DFHCLS4
CMAC	DFHCMAC	DFHCMAC
CMPX	DFHISC	DFHMXP
CMSG	DFHMSWIT	DFHMSP
CMTS	DFHSPI	DFHZATS
COBC	DFH\$DLIV	DFH0DLCC
COBE	DFH\$DLIV	DFH0DLCE
CPMI	DFHISC	DFHMIRS
CQRY	DFHSTAND	DFHQRY
CRSQ	DFHISC	DFHCRQ
CRSR	DFHISC	DFHCRS
CRSY	DFHRMI	DFHRMSY
CRTE	DFHISC	DFHRTE
CRTX	DFHISC	#####
CSAC	DFHSTAND	DFHACP
CSCY	DFHVTAMP	DFHCPY
CSFE	DFHFE	DFHFEP
CSFU	DFHOPCLS	DFHFCU
CSGM	DFHVTAM	DFHGMM
CSGX	DFHDLI	DFHDLG
CSIR	DFHCOMP1	DFHCRR
CSJC	DFHCOMP3	DFHJCbsp
CSKP	DFHAKP	DFHAKP
CSLG	DFHRSPLG	DFHZRLG
CSMI	DFHISC	DFHMIRS
CSMT	DFHCOMP1	DFHMTPA
CSM1	DFHISC	DFHMIRS
CSM2	DFHISC	DFHMIRS
CSM3	DFHISC	DFHMIRS
CSM5	DFHISC	DFHMIRS

Table 93 (Page 3 of 4). CICS transactions supplied by IBM

Transaction	Group	Program
CSNC	DFHISC	DFHCRNP
CSNC	DFHCOMP1	DFHCRNP
CSNC	DFHCOMP2	DFHCRNP
CSNE	DFHVTAM	DFHZNAC
CSOT	DFHCOMP1	DFHMTPA
CSPG	DFHBMS	DFHTPR
CSPK	DFHVTAMP	DFHPRK
CSPP	DFHHARDC	DFHP3270
CSPQ	DFHBMS	DFHTPQ
CSPS	DFHBMS	DFHTPS
CSRK	DFHVTAMP	DFHRKB
CSRS	DFHRSEND	DFHZRSP
CSSC	DFHSIGN	DFHCSSC
CSSF	DFHISC	DFHRTC
CSSN	DFHCOMP1	DFHNSP
CSST	DFHCOMP1	DFHMTPA
CSSX	DFHDLI	DFHDLS
CSTA	DFHCOMP1	DFHTAJP
CSTE	DFHSTAND	DFHTACP
CSTT	DFHCOMP1	DFHSTKC
CSZI	DFHFEPI	DFHSZRMF
CVMI	DFHISC	DFHMIRS
CVST	DFHCOMP1	DFHVAP
CWTO	DFHCONS	DFHCWTO
CXCU	DFHSTAND	DFHCXCU
CXRE	DFHSTAND	DFHZXRE
CXRT	DFHISC	DFHCRT
DADD	DFH\$DFLA	DFH\$DALL
DBRW	DFH\$DFLA	DFH\$DBRW
DELQ	DFH\$CTXT	DFH0VDQ
DINQ	DFH\$DFLA	DFH\$DALL
DMNU	DFH\$DFLA	DFH\$DMNU
DORD	DFH\$DFLA	DFH\$DREN
DORQ	DFH\$DFLA	DFH\$DCOM
DREP	DFH\$DFLA	DFH\$DREP
DSNC	DFHDB2	DSN2COM1
DUPD	DFH\$DFLA	DFH\$DALL
EXCI	DFH\$EXCI	DFH\$MIRS
ICIC	DFH\$ICOM	DFH\$ICIC
IFBL	DFH\$ICOM	DFH\$IFBL
IFBR	DFH\$ICOM	DFH\$IFBR
IMSN	DFH\$ICOM	DFH\$IMSN
IMSO	DFH\$ICOM	DFH\$IMSO
INQY	DFH\$CFLA	DFH0CALL
IQRD	DFH\$ICOM	DFH\$IQRD
IQRL	DFH\$ICOM	DFH\$IQRL
IQRR	DFH\$ICOM	DFH\$IQRR
IQXL	DFH\$ICOM	DFH\$IQXL
IQXR	DFH\$ICOM	DFH\$IQXR
MENU	DFH\$CFLA	DFH0CMNU
OREN	DFH\$CFLA	DFH0CREN
OREQ	DFH\$CFLA	DFH0CCOM
PADD	DFH\$PFLA	DFH\$PALL
PBRW	DFH\$PFLA	DFH\$PBRW
PINQ	DFH\$PFLA	DFH\$PALL
PLIC	DFH\$DLIV	DFH\$DLPC
PLIE	DFH\$DLIV	DFH\$DLPE
PMNU	DFH\$PFLA	DFH\$PMNU
PORD	DFH\$PFLA	DFH\$PREN
PORQ	DFH\$PFLA	DFH\$PCOM
PPKO	DFH\$BMSP	DFH\$PPKO
PPLA	DFH\$BMSP	DFH\$PPLA
PREP	DFH\$PFLA	DFH\$PREP

CICS transactions

Table 93 (Page 4 of 4). CICS transactions supplied by IBM

Transaction	Group	Program
PUPD	DFH\$PFLA	DFH\$PALL
REPT	DFH\$CFLA	DFH0CREP
TDWT	DFH\$UTIL	DFH\$TDWT
UPDT	DFH\$CFLA	DFH0CALL
XPKO	DFH\$BMSP	DFH0CPKO
XPLA	DFH\$BMSP	DFH0CPLA

TYPETERM definitions in group DFHTYPE

The CICS-supplied CSD group DFHTYPE contains the following TYPETERM definitions:

DFHCONS	MVS console
DFHLU2	SNA logical unit type 2 (3270 displays)
DFHLU3	SNA logical unit type 3 (3270 printers)
DFHSCSP	SNA logical unit type 1 (3270 SCS printers)
DFH3270	Locally attached (non-SNA) 3270 displays
DFH3270P	Locally attached (non-SNA) 3270 printers
DFHLU62T	SNA logical unit type 6.2 (APPC) single session terminal.
DFHLU0E2	Non-SNA model 2 with extended data stream (Query)
DFHLU0M2	Non-SNA model 2
DFHLU0M3	Non-SNA model 3
DFHLU0M4	Non-SNA model 4
DFHLU0M5	Non-SNA model 5
DFHLU2E2	SNA LU type 2 model 2 with extended data stream (Query)
DFHLU2M2	SNA LU type 2 model 2
DFHLU2E3	SNA LU type 2 model 3 with extended data stream (Query)
DFHLU2M3	SNA LU type 2 model 3
DFHLU2E4	SNA LU type 2 model 4 with extended data stream (Query)
DFHLU2M4	SNA LU type 2 model 4
DFHLU2M5	SNA LU type 2 model 5
DFHLU2E5	SNA LU type 2 model 5 with extended data stream (Query)

The following figures show the contents of each definition.

TYPETERM(DFHCONS)	GROUP(DFHTYPE)	
DEVICE(CONSOLE)		
PAGESIZE(1,124)	AUTOPAGE(NO)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(NONE)
UCTRAN(YES)		

Figure 59. DFHCONS TYPETERM definition - MVS console

TYPETERM(DFHLU2)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(YES)	UCTRAN(YES)
SENDSIZE(1536)	RECEIVESIZE(256)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	ATI(YES) TTI(YES)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(YES)
LOGONMSG(YES)	QUERY(ALL)	CREATESESS(NO)

Figure 60. DFHLU2 TYPETERM definition - SNA LU type 2. This definition is for a 3278 Model 2 display. It is suitable for the following devices: 3178, 3179, 3277, 3278, 3279, 3290, 3270PC, 3270PC/G, 3270PC/GX, 8775, and 5550.

Supplied TYPETERMs

TYPETERM(DFHLU3)	GROUP(DFHTYPE)	
DEVICE(LUTYPE3)	TERMMODEL(2)	
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(YES)
BRACKET(YES)	BUILDCHAIN(NO)	ROUTEDMSGS(ALL)
SENDSIZE(256)	RECEIVESIZE(256)	IOAREALEN(512,0)
EXTENDEDDES(YES)	QUERY(ALL)	ATI(YES) TTI(YES)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(YES)
LOGONMSG(NO)		CREATESESS(NO)

Figure 61. DFHLU3 TYPETERM definition - SNA LU type 3. This definition is for a 3287 printer. It is suitable for the following devices: 3262, 3268, 3284, 3286, 3287, 3288, 3289, and 5550.

TYPETERM(DFHSCSP)	GROUP(DFHTYPE)	
DEVICE(SCSPRINT)		
PAGESIZE(24,80)	AUTOPAGE(YES)	
BRACKET(YES)	BUILDCHAIN(NO)	ROUTEDMSGS(ALL)
SENDSIZE(256)	RECEIVESIZE(256)	IOAREALEN(512,0)
EXTENDEDDES(YES)	QUERY(ALL)	ATI(YES) TTI(YES)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(YES)
LOGONMSG(NO)		CREATESESS(NO)

Figure 62. DFHSCSP TYPETERM definition - SNA LU type 1. This definition is for a 3287 printer. It is suitable for the following devices: 3262, 3268, 3287, 3289, and 5550.

TYPETERM(DFH3270)	GROUP(DFHTYPE)	
DEVICE(3270)	TERMMODEL(2)	
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
BRACKET(YES)	BUILDCHAIN(NO)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(YES)	UCTRAN(YES)
SENDSIZE(0)	RECEIVESIZE(0)	IOAREALEN(512,0)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	ATI(YES) TTI(YES)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(YES)
LOGONMSG(YES)	QUERY(ALL)	CREATESESS(NO)

Figure 63. DFH3270 TYPETERM definition - locally attached (non-SNA). This definition is for a 3278 Model 2 display. It is suitable for the following devices: 3178, 3179, 3277, 3278, 3279, and 3290.

TYPETERM(DFH3270P)	GROUP(DFHTYPE)	
DEVICE(3270P)	TERMMODEL(2)	
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(YES)
BRACKET(YES)	BUILDCHAIN(NO)	ROUTEDMSGS(ALL)
SENDSIZE(0)	RECEIVESIZE(0)	IOAREALEN(512,0)
EXTENDEDDES(YES)	QUERY(ALL)	ATI(YES) TTI(YES)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(YES)
LOGONMSG(NO)		CREATESESS(NO)

Figure 64. DFH3270P TYPETERM definition - locally attached (non-SNA). This definition is for a 3284 Model 2 printer. It is suitable for the following devices: 3262, 3268, 3284, 3287, 3288, 3289, and 5550.

TYPETERM(DFHLU62T)	GROUP(DFHTYPE)	
DEVICE(APPC)		
	PAGESIZE(1,40)	AUTOPAGE(YES)
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(NONE)
SENDSIZE(2048)	RECEIVESIZE(2048)	IOAREALEN(0,0)
		ATI(YES) TTI(YES)
LOGONMSG(NO)		CREATESESS(NO)

Figure 65. DFHLU62T TYPETERM definition - APPC (LU6.2) terminal. This definition is for an APPC single session terminal and is also suitable for the following devices: DISPLAYWRITER, SCANMASTER, and SYSTEM/38.

TYPETERM(DFHLU0E2)	GROUP(DFHTYPE)	
DEVICE(3270)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(0,0)	ALTPAGE(0,0)	
BRACKET(YES)	BUILDCHAIN(NO)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(YES)	UCTRAN(YES)
RECEIVESIZE(0)	SENDSIZE(0)	IOAREALEN(512,0)
ERRASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(ALL)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 66. DFHLU0E2 TYPETERM definition. Non-SNA model 2 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE NSX32702

TYPETERM(DFHLU0M2)	GROUP(DFHTYPE)	
DEVICE(3270)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(0,0)	ALTPAGE(0,0)	
BRACKET(YES)	BUILDCHAIN(NO)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(NO)	UCTRAN(YES)
RECEIVESIZE(0)	SENDSIZE(0)	IOAREALEN(512,0)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(NO)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 67. DFHLU0M2 TYPETERM definition. Non-SNA model 2 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE D4B32782

TYPETERM(DFHLU0M3)	GROUP(DFHTYPE)	
DEVICE(3270)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(32,80)	ALTPAGE(32,80)	
BRACKET(YES)	BUILDCHAIN(NO)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(NO)	UCTRAN(YES)
RECEIVESIZE(0)	SENDSIZE(0)	IOAREALEN(512,0)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(NO)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 68. DFHLU0M3 TYPETERM definition - Non-SNA model 3. This definition matches the VTAM supplied LOGMODE D4B32783.

Supplied TYPETERMs

TYPETERM(DFHU0M4)	GROUP (DFHTYPE)	
DEVICE (3270)	TERMMODEL (2)	SHIPPABLE (YES)
DEFSCREEN (24,80)	PAGESIZE (24,80)	AUTOPAGE (NO)
ALTSSCREEN (43,80)	ALTPAGE (43,80)	
BRACKET (YES)	BUILDCHAIN (NO)	ROUTEDMSGS (ALL)
AUDIBLEALARM (YES)	EXTENDEDDES (NO)	UCTRAN (YES)
RECEIVESIZE (0)	SENDSIZE (0)	IOAREALEN (512,0)
ERRLASTLINE (YES)	ERRINTENSIFY (YES)	QUERY (NO)
DISCREQ (YES)	RELREQ (YES)	AUTOCONNECT (NO)
LOGONMSG (YES)	CREATESESS (NO)	
ATI (YES)	TTI (YES)	

Figure 69. DFHLU0M4 TYPETERM definition - Non-SNA model 4. This definition matches the VTAM supplied LOGMODE D4B32784.

TYPETERM(DFHU0M5)	GROUP (DFHTYPE)	
DEVICE (3270)	TERMMODEL (2)	SHIPPABLE (YES)
DEFSCREEN (24,80)	PAGESIZE (24,80)	AUTOPAGE (NO)
ALTSSCREEN (27,132)	ALTPAGE (27,132)	
BRACKET (YES)	BUILDCHAIN (NO)	ROUTEDMSGS (ALL)
AUDIBLEALARM (YES)	EXTENDEDDES (NO)	UCTRAN (YES)
RECEIVESIZE (0)	SENDSIZE (0)	IOAREALEN (512,0)
ERRLASTLINE (YES)	ERRINTENSIFY (YES)	QUERY (NO)
DISCREQ (YES)	RELREQ (YES)	AUTOCONNECT (NO)
LOGONMSG (YES)	CREATESESS (NO)	
ATI (YES)	TTI (YES)	

Figure 70. DFHLU0M5 TYPETERM definition - Non-SNA model 5. This definition matches the VTAM supplied LOGMODE D4B32785.

TYPETERM(DFHU2E2)	GROUP (DFHTYPE)	
DEVICE (LUTYPE2)	TERMMODEL (2)	SHIPPABLE (YES)
DEFSCREEN (24,80)	PAGESIZE (24,80)	AUTOPAGE (NO)
ALTSSCREEN (0,0)	ALTPAGE (0,0)	
BRACKET (YES)	BUILDCHAIN (YES)	ROUTEDMSGS (ALL)
AUDIBLEALARM (YES)	EXTENDEDDES (YES)	UCTRAN (YES)
RECEIVESIZE (1024)	SENDSIZE (3840)	IOAREALEN (256,4000)
ERRLASTLINE (YES)	ERRINTENSIFY (YES)	QUERY (ALL)
DISCREQ (YES)	RELREQ (YES)	AUTOCONNECT (NO)
LOGONMSG (YES)	CREATESESS (NO)	
ATI (YES)	TTI (YES)	

Figure 71. DFHLU2E2 TYPETERM definition. SNA LU type 2 model 2 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE SNX32702.

TYPETERM(DFHLU2M2)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(0,0)	ALTPAGE(0,0)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(NO)	UCTRAN(YES)
RECEIVESIZE(1024)	SENDSIZE(1536)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(NO)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 72. DFHLU2M2 TYPETERM definition - SNA LU type 2 model 2. This definition matches the VTAM supplied LOGMODE D4A32782.

TYPETERM(DFHLU2E3)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(32,80)	ALTPAGE(32,80)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(YES)	UCTRAN(YES)
RECEIVESIZE(1024)	SENDSIZE(3840)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(ALL)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 73. DFHLU2E3 TYPETERM definition. SNA LU type 2 model 3 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE SNX32703.

TYPETERM(DFHLU2M3)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(32,80)	ALTPAGE(32,80)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(NO)	UCTRAN(YES)
RECEIVESIZE(1024)	SENDSIZE(1536)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(NO)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 74. DFHLU2M3 TYPETERM definition - SNA LU type 2 model 3. This definition matches the VTAM supplied LOGMODE D4A32783.

Supplied TYPETERMs

TYPETERM(DFHLU2E4)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(43,80)	ALTPAGE(43,80)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(YES)	UCTRAN(YES)
RECEIVESIZE(1024)	SENDSIZE(3840)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(ALL)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 75. DFHLU2E4 TYPETERM definition. SNA LU type 2 model 4 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE SNX32704.

TYPETERM(DFHLU2M4)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(43,80)	ALTPAGE(43,80)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(NO)	UCTRAN(YES)
RECEIVESIZE(1024)	SENDSIZE(1536)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(NO)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 76. DFHLU2M4 TYPETERM definition - SNA LU type 2 model 4. This definition matches the VTAM supplied LOGMODE D4A32784.

TYPETERM(DFHLU2E5)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(27,132)	ALTPAGE(27,132)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDES(NO)	UCTRAN(YES)
RECEIVESIZE(1024)	SENDSIZE(3480)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(ALL)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 77. DFHLU2E5 TYPETERM definition - SNA LU type 2 model 5. SNA LU type 2 model 5 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE LSX32705.

TYPETERM(DFHLU2M5)	GROUP(DFHTYPE)	
DEVICE(LUTYPE2)	TERMMODEL(2)	SHIPPABLE(YES)
DEFSCREEN(24,80)	PAGESIZE(24,80)	AUTOPAGE(NO)
ALTSCREEN(27,132)	ALTPAGE(27,132)	
BRACKET(YES)	BUILDCHAIN(YES)	ROUTEDMSGS(ALL)
AUDIBLEALARM(YES)	EXTENDEDDS(NO)	UCTRAN(YES)
RECEIVESIZE(1024)	SENDSIZE(1536)	IOAREALEN(256,4000)
ERRLASTLINE(YES)	ERRINTENSIFY(YES)	QUERY(NO)
DISCREQ(YES)	RELREQ(YES)	AUTOCONNECT(NO)
LOGONMSG(YES)	CREATESESS(NO)	
ATI(YES)	TTI(YES)	

Figure 78. DFHLU2M5 TYPETERM definition - SNA LU type 2 model 5. This definition matches the VTAM supplied LOGMODE D4A32785.

Model TERMINAL definitions in group DFHTERM

The CICS-supplied CSD group DFHTERM contains the following model TERMINAL definitions for automatic installation:

LU2	SNA 3270 Model 2 display using TYPETERM DFHLU2
LU3	SNA 3270 Model 2 printer using TYPETERM DFHLU3
SCSP	SNA 3278 Model 2 printer using TYPETERM DFHSCSP
3270	Non-SNA 3270 Model 2 display using TYPETERM DFH3270
3284	Non-SNA 3270 Model 2 printer using TYPETERM DFH3270P
LU62	APPC (LU6.2) single session terminal using TYPETERM DFHLU62T
L0E2	Non-SNA Model 2 display using TYPETERM DFHLU0E2
L0M2	Non-SNA Model 2 display using TYPETERM DFHLU0M2
L0M3	Non-SNA Model 3 display using TYPETERM DFHLU0M3
L0M4	Non-SNA Model 4 display using TYPETERM DFHLU0M4
L0M5	Non-SNA Model 5 display using TYPETERM DFHLU0M5
L2E2	SNA LU2 Model 2 display using TYPETERM DFHLU2E2
L2M2	SNA LU2 Model 2 display using TYPETERM DFHLU2M2
L2E3	SNA LU2 Model 3 display using TYPETERM DFHLU2E3
L2M3	SNA LU2 Model 3 display using TYPETERM DFHLU2M3
L2E4	SNA LU2 Model 4 display using TYPETERM DFHLU2E4
L2M4	SNA LU2 Model 4 display using TYPETERM DFHLU2M4
L2E5	SNA LU2 Model 5 display using TYPETERM DFHLU2E5
L2M5	SNA LU2 Model 5 display using TYPETERM DFHLU2M5

The following figures show the contents of each definition.

TERMINAL(LU2)	GROUP(DFHTERM)	TYPETERM(DFHLU2)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2)	

Figure 79. LU2 model TERMINAL definition - SNA LU type 2. This definition is for a 3278 Model 2 display. It is suitable for the following devices: 3178, 3179, 3277, 3278, 3279, 3290, 3270PC, 3270PC/G, 3270PC/GX, 8775, and 5550.

TERMINAL(LU3)	GROUP(DFHTERM)	TYPETERM(DFHLU3)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU3)	

Figure 80. LU3 model TERMINAL definition - SNA LU type 3. This definition is for a 3287 printer. It is suitable for the following devices: 3262, 3268, 3284, 3286, 3287, 3288, 3289, and 5550.

TERMINAL(SCSP)	GROUP(DFHTERM)	TYPETERM(DFHSCSP)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHSCSP)	

Figure 81. SCSP model TERMINAL definition - SNA LU type 1. This definition is for a 3287 printer. It is suitable for the following devices: 3262, 3268, 3287, 3289, and 5550.

TERMINAL(3270)	GROUP(DFHTERM)	TYPETERM(DFH3270)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270)	

Figure 82. 3270 model TERMINAL definition - locally attached (non-SNA). This definition is for a 3278 Model 2 display. It is suitable for the following devices: 3178, 3179, 3277, 3278, 3279, and 3290.

TERMINAL(3284)	GROUP(DFHTERM)	TYPETERM(DFH3270P)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270P)	

Figure 83. 3284 model TERMINAL definition - locally attached (non-SNA). This definition is for a 3284 Model 2 printer. It is suitable for the following devices: 3262, 3268, 3284, 3287, 3288, 3289, and 5550.

TERMINAL(LU62)	GROUP(DFHTERM)	TYPETERM(DFH3270P)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270P)	

Figure 84. LU62 model TERMINAL definition - APPC (LU6.2) terminal. This definition is for an APPC single session terminal and is also suitable for the following devices: DISPLAYWRITER, SCANMASTER, and SYSTEM/38.

TERMINAL(L0E2)	GROUP(DFHTERM)	TYPETERM(DFH3270P)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270P)	

Figure 85. L0E2 model TERMINAL definition. Non-SNA model 2 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE NSX32702.

TERMINAL(L0M2)	GROUP(DFHTERM)	TYPETERM(DFH3270P)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270P)	

Figure 86. L0M2 model TERMINAL definition - Non-SNA model 2. This definition matches the VTAM supplied LOGMODE D4B32782.

TERMINAL(L0M3)	GROUP(DFHTERM)	TYPETERM(DFH3270P)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270P)	

Figure 87. L0M3 model TERMINAL definition - Non-SNA model 3. This definition matches the VTAM supplied LOGMODE D4B32783.

TERMINAL(L0M4)	GROUP(DFHTERM)	TYPETERM(DFH3270P)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270P)	

Figure 88. L0M4 model TERMINAL definition - Non-SNA model 4. This definition matches the VTAM supplied LOGMODE D4B32784.

TERMINAL(L0M5)	GROUP(DFHTERM)	TYPETERM(DFH3270P)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFH3270P)	

Figure 89. L0M5 model TERMINAL definition - Non-SNA model 5. This definition matches the VTAM supplied LOGMODE D4B32785.

Supplied model TERMINALS

TERMINAL(L2E2)	GROUP(DFHTERM)	TYPETERM(DFHLU2E2)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2E2)	

Figure 90. L2E2 model TERMINAL definition. SNA LU type 2 model 2 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE SNX32702.

TERMINAL(L2M2)	GROUP(DFHTERM)	TYPETERM(DFHLU2M2)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2M2)	

Figure 91. L2M2 model TERMINAL definition - SNA LU type 2 model 2. This definition matches the VTAM supplied LOGMODE D4A32782.

TERMINAL(L2E3)	GROUP(DFHTERM)	TYPETERM(DFHLU2E3)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2E3)	

Figure 92. L2E3 model TERMINAL definition. SNA LU type 2 model 3 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE SNX32703.

TERMINAL(L2M3)	GROUP(DFHTERM)	TYPETERM(DFHLU2M3)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2M3)	

Figure 93. L2M3 model TERMINAL definition - SNA LU type 2 model 3. This definition matches the VTAM supplied LOGMODE D4A32783.

TERMINAL(L2E4)	GROUP(DFHTERM)	TYPETERM(DFHLU2E4)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2E4)	

Figure 94. L2E4 model TERMINAL definition.. SNA LU type 2 model 4 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE SNX32704.

TERMINAL(L2M4)	GROUP(DFHTERM)	TYPETERM(DFHLU2M4)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2M4)	

Figure 95. L2M4 model TERMINAL definition - SNA LU type 2 model 4. This definition matches the VTAM supplied LOGMODE D4A32784.

TERMINAL(L2E5)	GROUP(DFHTERM)	TYPETERM(DFHLU2E5)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2E5)	

Figure 96. L2E5 model TERMINAL definition.. SNA LU type 2 model 5 with extended data stream (Query). This definition matches the VTAM supplied LOGMODE LSX32705.

TERMINAL(L2M5)	GROUP(DFHTERM)	TYPETERM(DFHLU2M5)
AUTINSTMODEL(ONLY)	AUTINSTNAME(DFHLU2M5)	

Figure 97. L2M5 model TERMINAL definition - SNA LU type 2 model 5. This definition matches the VTAM supplied LOGMODE D4A32785.

PROFILE definitions in group DFHISC

The CICS-supplied CSD group DFHISC contains the following PROFILE definitions for intersystem communication sessions.

DFHCICSF

CICS uses this profile for the session to the remote system or region when a CICS application program issues a function shipping request.

The definition is as follows:

PROFILE(DFHCICSF)			
GROUP(DFHISC)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(ALL)	INBFMH(ALL)	RAQ(NO)
	LOGREC(NO)		
RECOVERY	NEPCCLASS(0)		

Figure 98. DFHCICSF profile definition

DFHCICSR

CICS uses this profile in transaction routing for communication between the user transaction (running in the application-owning region) and the interregion link or APPC link.

The definition is as follows:

PROFILE(DFHCICSR)			
GROUP(DFHISC)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(ALL)	INBFMH(ALL)	RAQ(NO)
	LOGREC(NO)		
RECOVERY	NEPCCLASS(0)		

Figure 99. DFHCICSR profile definition

DFHCICSS

CICS uses this profile in transaction routing for communication between the relay transaction (running in the terminal-owning region) and the interregion link or APPC link. You can specify a different profile by means of the TRPROF option on the TRANSACTION definition.

The definition is as follows:

PROFILE(DFHCICSS)			
GROUP(DFHISC)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(ALL)	INBFMH(ALL)	RAQ(NO)
	LOGREC(NO)		
RECOVERY	NEPCCLASS(0)		

Figure 100. DFHCICSS profile definition

PROFILE definitions in group DFHSTAND

The CICS-supplied CSD group DFHSTAND contains the following PROFILE definitions.

DFHCICSA

This is the default profile for alternate facilities acquired by the application program ALLOCATE command. A different profile can be named explicitly on the ALLOCATE command.

The definition is as follows:

PROFILE(DFHCICSA)			
GROUP(DFHSTAND)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(ALL)	INBFMH(ALL)	RAQ(NO)
	LOGREC(NO)		
RECOVERY	NEPCCLASS(0)		

Figure 101. DFHCICSA profile definition

DFHCICSE

This is the error profile for principal facilities. CICS uses this profile to pass an error message to the principal facility when the required profile cannot be found.

The definition is as follows:

PROFILE(DFHCICSE)			
GROUP(DFHSTAND)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(ALL)	INBFMH(NO)	RAQ(NO)
	LOGREC(NO)		
RECOVERY	NEPCCLASS(0)		

Figure 102. DFHCICSE profile definition

DFHCICSP

This is the default profile for the page retrieval transaction CSPG. You can specify a different profile for a particular transaction by means of the PROFILE option on the TRANSACTION definition.

The definition is as follows:

PROFILE(DFHCICSP)			
GROUP(DFHSTAND)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
UCTRAN(YES)			
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(ALL)	INBFMH(NO)	RAQ(NO)
LOGREC(NO)	RECOVERY	NEPCCLASS(0)	

Figure 103. DFHCICSP profile definition

DFHCICST

This is the default profile for principal facilities. You can specify a different profile for a particular transaction by means of the PROFILE option on the TRANSACTION definition.

The definition is as follows:

PROFILE(DFHCICST)			
GROUP(DFHSTAND)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(ALL)	INBFMH(NO)	RAQ(NO)
	LOGREC(NO)		
RECOVERY	NEPCCLASS(0)		

Figure 104. DFHCICST profile definition

DFHCICSV

This is the profile for principal facilities, when the transaction will support only VTAM devices.

The definition is as follows:

PROFILE(DFHCICSV)			
GROUP(DFHSTAND)	SCRNSIZE(DEFAULT)	PRINTERCOMP(NO)	
JOURNALLING	MSGJRNL(NO)		
PROTECTION	MSGINTEG(NO)	ONEWTE(NO)	PROTECT(NO)
PROTOCOLS	DVSUPRT(VTAM)	INBFMH(NO)	RAQ(NO)
	LOGREC(NO)		
RECOVERY	NEPCCLASS(0)		

Figure 105. DFHCICSV profile definition

Supplied PROFILES

DFHPPF01 and DFHPPF02

Profiles DFHPPF01 and DFHPPF02 are used during CICS initialization, for tasks that are attached before the CSD definitions have been installed.

The definitions are as follows:

PROFILE(DFHPPF01)	GROUP(DFHSTAND)	DESCRIPTION(VTAM-ONLY PROFILE)
SCRNSIZE(DEFAULT)	MSGJRNL(NO)	MSGINTEG(NO)
ONEWTE(NO)	PROTECT(NO)	DVSUPRT(VTAM)
INBFMH(NO)	RAQ(NO)	LOGREC(NO)
NEPCLASS(000)		
PROFILE(DFHPPF02)	GROUP(DFHSTAND)	DESCRIPTION(ALL-NULLS PROFILE)
SCRNSIZE(DEFAULT)	MSGJRNL(NO)	MSGINTEG(NO)
ONEWTE(NO)	PROTECT(NO)	DVSUPRT(ALL)
INBFMH(NO)	RAQ(NO)	LOGREC(NO)
NEPCLASS(000)		

Figure 106. DFHPPF01 and DFHPPF02 profile definitions

Model definitions in group DFHPGAIP

The CICS-supplied CSD group DFHPGAIP contains the following model definitions for use with autoinstall for programs.

DFHPGAPG

This is the default PROGRAM definition for program autoinstall.

The definition is as follows;

```
PROGRAM(DFHPGAPG)  GROUP(DFHPGAIP)
DESCRIPTION(default program for program autoinstall)
RELOAD(NO)        RESIDENT(NO)
USAGE(NORMAL)     USELPACOPY(NO)    STATUS(ENABLED)
RSL(00)           CEDF(YES)         DATALOCATION(BELOW)
EXECKEY(USER)    EXECUTIONSET(FULLAPI)
```

APAR PN76605

deleted LANGUAGE(COBOL) from this figure.

Figure 107. DFHPGAPG program definition

DFHPGAMP

This is the default MAPSET definition for program autoinstall.

The definition is as follows;

```
MAPSET(DFHPGAMP)  GROUP(DFHPGAIP)
DESCRIPTION(default mapset for program autoinstall)
RESIDENT(NO)     USAGE(NORMAL)
USELPACOPY(NO)   STATUS(ENABLED)
RSL(00)
```

Figure 108. DFHPGAMP mapset definition

DFHPGAPT

This is the default PARTITION definition for program autoinstall.

The definition is as follows:

```
PARTITIONSET(DFHPGAPT)  GROUP(DFHPGAIP)
DESCRIPTION(default partitionset for program autoinstall)
RESIDENT(NO)           USAGE(NORMAL)
USELPACOPY(NO)         STATUS(ENABLED)
RSL(00)
```

Figure 109. DFHPGAPT partitionset definition

Supplied model definitions for program autoinstall

Appendix C. Keyword cross-reference tables

This chapter contains two cross-reference tables relating macro operands (past and present) to RDO keywords.

Macro operand to RDO keyword

This list is in alphabetic order of macro operands, giving the equivalent RDO keyword. "RDO keyword to macro operand" on page 382 gives the same list in order of RDO keyword.

MACRO OPERAND	RESOURCE TYPE	RDO ATTRIBUTE
ACCMETH=INDIRECT	Connection	ACCESSMETHOD(INDIRECT)
ACCMETH=IRC	Connection	ACCESSMETHOD(IRC)
ACCMETH=VTAM	Connection	ACCESSMETHOD(VTAM)
ACCMETH=(IRC,XM)	Connection	ACCESSMETHOD(XM)
ALTPGE	Typeterm	ALTPAGE
ALTPRT(.COPY)	Terminal	ALTPRINTCOPY
ALTPRT(label,)	Terminal	ALTPRINTER
ALTSCRN	Typeterm	ALTSCREEN
ALTSFX	Typeterm	ALTSUFFIX
ANTICPG	See note 2.	
BASE	File	NSRGROUP
BMSFEAT (see note 5)	Typeterm	ROUTEDMSGS(ALL)
BMSFEAT=FMHPARM	Typeterm	FMHPARM(YES)
BMSFEAT=NOROUTE	Typeterm	ROUTEDMSGS(NONE)
BMSFEAT=NOROUTEALL	Typeterm	ROUTEDMSGS(SPECIFIC)
BMSFEAT=OBFMT	Typeterm	OBFORMAT(YES)
BMSFEAT=OBOPID	Typeterm	OBOPERID(YES)
BRACKET	Typeterm	BRACKET
BUFFER	Typeterm	SENDSIZE
BUFFER	Sessions	SENDSIZE
BUFFERS=(1K(count))	Lsrpool	DATA1K
BUFFERS=(2K(count))	Lsrpool	DATA2K
BUFFERS=(4K(count))	Lsrpool	DATA4K
BUFFERS=(8K(count))	Lsrpool	DATA8K
BUFFERS=(12K(count))	Lsrpool	DATA12K
BUFFERS=(16K(count))	Lsrpool	DATA16K
BUFFERS=(20K(count))	Lsrpool	DATA20K
BUFFERS=(24K(count))	Lsrpool	DATA24K
BUFFERS=(28K(count))	Lsrpool	DATA28K
BUFFERS=(32K(count))	Lsrpool	DATA32K
BUFFERS=(512(count))	Lsrpool	DATA512
BUFFERS=(4K(,count))	Lsrpool	HSDATA4K
BUFFERS=(8K(,count))	Lsrpool	HSDATA8K
BUFFERS=(12K(,count))	Lsrpool	HSDATA12K
BUFFERS=(16K(,count))	Lsrpool	HSDATA16K
BUFFERS=(20K(,count))	Lsrpool	HSDATA20K
BUFFERS=(24K(,count))	Lsrpool	HSDATA24K
BUFFERS=(28K(,count))	Lsrpool	HSDATA28K
BUFFERS=(32K(,count))	Lsrpool	HSDATA32K
BUFND	File	DATABUFFERS
BUFNI	File	INDEXBUFFERS
CHNASSY	Sessions	BUILDCHAIN
CHNASSY	Typeterm	BUILDCHAIN
CONNECT	Typeterm	AUTOCONNECT
CONNECT	Connection	AUTOCONNECT

Macro to RDO

MACRO OPERAND

CONNECT
CONSLID
DATASTR
DEFSCRN
DISP

DSNAME
DSNSHR
DTB=NO
DTB=YES
DTB=YES,NO

DTB=YES,WAIT
DTIMOUT
DUMP
DVSUPRT
ERRATT=BLINK

ERRATT=color
ERRATT=INTENSIFY
ERRATT=LASTLINE
ERRATT=REVERSE
ERRATT=UNDERLINE

EXTSEC
FDUMP
FEATURE=APLKYBD
FEATURE=APLTEXT
FEATURE=ASCII-7

FEATURE=ASCII-8
FEATURE=AUDALARM
FEATURE=BTRANS
FEATURE=COLOR
FEATURE=COPY

FEATURE=DCKYBD
FEATURE=EXTDS
FEATURE=HILIGHT
FEATURE=KATAKANA
FEATURE=MSRCNTRL

FEATURE=OUTLINE
FEATURE=PARALLEL
FEATURE=PARTNS
FEATURE=PTRADAPT
FEATURE=PS

FEATURE=QUERYALL
FEATURE=QUERYCOLD
FEATURE=SELCTPEN
FEATURE=SINGLE
FEATURE=SOSI

FEATURE=TEXTKYBD
FEATURE=TEXTPRINT
FEATURE=UCTRAN
FEATURE=VALIDATION
FF

FILE
FILSTAT=ENABLED|DISABLED
FILSTAT=OPENED|CLOSED
GMMMSG
HF

INBFMH
INDSYS
JFILEID
JID
JID=SYSTEM

RESOURCE TYPE

Sessions
Terminal
Connection
Typeterm
File

File
File
See note 4.
Transaction
Transaction

Transaction
Transaction
Transaction
Profile
Typeterm

Typeterm
Typeterm
Typeterm
Typeterm
Typeterm

See note 9.
See note 3.
Typeterm
Typeterm
Typeterm

Typeterm
Typeterm
Typeterm
Typeterm
Typeterm

Typeterm
Connection
Typeterm
Typeterm
Typeterm

Typeterm
Typeterm
Typeterm
Connection
Typeterm

Typeterm
Typeterm
Typeterm
Typeterm
Typeterm

File
File
File
Typeterm
Typeterm

Profile
Connection
Profile
File
File

RDO ATTRIBUTE

AUTOCONNECT
CONSOLE
DATASTREAM
DEFSCREEN
DISPOSITION

DSNAME
DSNSHARING

INDOUBT(BACKOUT)
INDOUBT(COMMIT)

INDOUBT(WAIT)
DTIMOUT
DUMP
DVSUPRT
ERRHILIGHT(BLINK)

ERRCOLOR
ERRINTENSIFY
ERRLASTLINE
ERRHILIGHT(REVERSE)
ERRHILIGHT(UNDERLINE)

APLKYBD
APLTEXT
ASCII(7)

ASCII(8)
AUDIBLEALARM
BACKTRANS
COLOR
COPY

DUALCASEKYBD
EXTENDEDDS
HILIGHT
KATAKANA
MSRCONTROL

OUTLINE
SINGLESESS(NO)
PARTITIONS
PRINTADAPTER
PROGSYMBOLS

QUERY(ALL)
QUERY(COLD)
LIGHTPEN
SINGLESESS(YES)
SOSI

TEXTKYBD
TEXTPRINT
UCTRAN
VALIDATION
FORMFEED

FILE
STATUS
OPENTIME
LOGONMSG
HORIZFORM

INBFMH
INDSYS
JOURNAL
FWDRECOVLOG
JOURNAL(1)

MACRO OPERAND	RESOURCE TYPE	RDO ATTRIBUTE
JREQ=WN	File	JNLADD
JREQ=RU	File	JNLREAD(UPDATEONLY)
JREQ=RO	File	JNLREAD(READONLY)
JREQ=SYN	File	JNLSYNCREAD
JREQ=ASY	File	JNLSYNCWRITE(NO)
JREQ=WU	File	JNLUPDATE
KEYLEN	File	KEYLENGTH
KEYLEN	Lsrpool	MAXKEYLENGTH
LDC	Typeterm	LDCLIST
LOCALQ	Transaction	LOCALQ
LOG=YES	File	RECOVERY
LOGMODE	Typeterm	LOGMODE
LOGREC	Profile	LOGREC
LRECL	File	RECORDSIZE
LSRPOOL	File	See note 8
MAPSET	Mapset	MAPSET
MAXSESS	Sessions	MAXIMUM
MODENAM	Profile	MODENAME
MODENAM	Sessions	MODENAME
MODENAM	Terminal	MODENAME
MSGJRNL	Profile	MSGJRNL
MSGPOPT=(,CCONTRL)	See note 1.	
MSGPOPT=(,MSGINTEG)	See note 1.	
MSGPOPT=(,ONEWTE)	See note 1.	
MSGPOPT=(,PROTECT)	See note 1.	
MSGPREQ=(,CCONTRL)	Profile	CHAINCONTROL
MSGPREQ=(,MSGINTEG)	Profile	MSGINTEG
MSGPREQ=(,ONEWTE)	Profile	ONEWTE
MSGPREQ=(,PROTECT)	Profile	PROTECT
NATLANG	Terminal	NATLANG
NEPCLAS	Profile	NEPCLASS
NEPCLAS	Typeterm	NEPCLASS
NETNAME	Connection	NETNAME
NETNAME	Terminal	NETNAME
NETNAMQ	Sessions	NETNAMEQ
PARTSET	Partitionset	PARTITIONSET
PARTSET	Transaction	PARTITIONSET
PASSWD	File	PASSWORD
PGESIZE	Typeterm	PAGESIZE
PGESTAT=AUTOPAGE	Typeterm	AUTOPAGE(YES)
PGESTAT=PAGE	Typeterm	AUTOPAGE(NO)
PGMLANG	Program	LANGUAGE
PGMSTAT	Mapset	STATUS
PGMSTAT	Partitionset	STATUS
PGMSTAT	Program	STATUS
PIPELN	Terminal	POOL
PRINTTO(.COPY)	Terminal	PRINTERCOPY
PRINTTO(label,)	Terminal	PRINTER
PRMSIZE	See note 1	
PROFILE	Profile	PROFILE
PROGRAM	Program	PROGRAM
PROGRAM	Transaction	PROGRAM
PTRCOMP	Profile	PRINTERCOMP
RAQ	Profile	RAQ
RECEIVE=(,n)	Sessions	RECEIVECOUNT
RECEIVE=(x,)	Sessions	RECEIVEPFX
RECFM	Connection	RECORDFORMAT
RECFORM=U V F)	File	RECORDFORMAT
RELOAD	Program	RELOAD
RELREQ=(,NO YES)	Sessions	DISCREQ

Macro to RDO

MACRO OPERAND	RESOURCE TYPE	RDO ATTRIBUTE
RELREQ=(NO YES,)	Sessions	RELREQ
RELREQ=(,NO YES)	Typeterm	DISCREQ
RELREQ=(NO YES,)	Typeterm	RELREQ
RES=ALIGN	See note 1.	
RES=FIX	See note 1.	
RES=NO	Program	RESIDENT(NO)
RES=PAGEOUT	See note 1.	
RES=YES	Mapset	RESIDENT(YES)
RES=YES	Partitionset	RESIDENT(YES)
RES=YES	Program	RESIDENT(YES)
RESTART	Transaction	RESTART
RMTNAME	File	REMOTENAME
RMTNAME	Terminal	REMOTENAME
RMTNAME	Transaction	REMOTENAME
RNOTIFY	Typeterm	RECOVNOTIFY
ROPTION	Typeterm	RECOVOPTION
RSCLMT	Lsrpool	SHARELIMIT
RTIMOUT	Profile	RTIMOUT
RUSIZE	Sessions	RECEIVESIZE
RUSIZE	Typeterm	RECEIVESIZE
SCRNSZE	Profile	SCRNSIZE
SEND=(,n)	Sessions	SENDCOUNT
SEND=(x,)	Sessions	SENDPFX
SERVREQ=ADD	File	ADD
SERVREQ=BROWSE	File	BROWSE
SERVREQ=DELETE	File	DELETE
SERVREQ=READ	File	READ
SERVREQ=UPDATE	File	UPDATE
SESTYPE	Typeterm	SESSIONTYPE ⁷
SIGNOFF	Typeterm	SIGNOFF
SIZE=number	File	MAXNUMRECS
SPURGE	Transaction	SPURGE
STRNO	File	STRINGS
STRNO	Lsrpool	STRINGS
SYSIDNT	Connection	CONNECTION
SYSIDNT	File	REMOTESYSTEM
SYSIDNT	Sessions	CONNECTION
SYSIDNT	Terminal	REMOTESYSTEM
SYSIDNT	Transaction	REMOTESYSTEM
TASKNO	Terminal	TASKLIMIT
TASKREQ	Transaction	TASKREQ
TCLASS	Transaction	TCLASS
TCTUAL	Sessions	USERAREALEN
TCTUAL	Typeterm	USERAREALEN
TIOAL	Sessions	IOAREALEN
TIOAL	Typeterm	IOAREALEN
TPURGE	Transaction	TPURGE
TRACE	Transaction	TRACE
TRANSID	Terminal	TRANSACTION
TRANSID	Transaction	TRANSACTION
TRMIDNT	Sessions	SESSNAME
TRMIDNT	Terminal	TERMINAL
TRMMODL	Typeterm	TERMMODEL
TRMPRTY	Sessions	SESSPRIORITY
TRMPRTY	Terminal	TERMPRIORITY

⁷ Not relevant for LU6.1

MACRO OPERAND	RESOURCE TYPE	RDO ATTRIBUTE
TRMSTAT (see note 6)	Typeterm	ATI(NO) and TTI(NO)
TRMSTAT (see note 7)	Typeterm	CREATESESS(YES)
TRMSTAT=INPUT	See note 1.	
TRMSTAT=INTLOG	Typeterm	CREATESESS(YES)
TRMSTAT=NOINTLOG	Typeterm	CREATESESS(NO)
TRMSTAT='OUT OF SERVICE'	Connection	INSERVICE(No)
TRMSTAT='OUT OF SERVICE'	Sessions	INSERVICE(No)
TRMSTAT='OUT OF SERVICE'	Terminal	INSERVICE(No)
TRMSTAT=RECEIVE	Typeterm	ATI(YES) and TTI(NO)
TRMSTAT=TRANSACTION	Typeterm	ATI(NO) and TTI(YES)
TRMSTAT=TRANSCIVE	Typeterm	ATI(YES) and TTI(YES)
TRMTYPE	Connection	PROTOCOL
TRMTYPE	Sessions	PROTOCOL
TRMTYPE	Typeterm	DEVICE
TRNPRTY	Transaction	PRIORITY
TRNSTAT	Transaction	STATUS
TRPROF	Transaction	TRPROF
TWASIZE	Transaction	TWASIZE
TYPE=CICSTABLE USERTABLE	File	TABLE
USAGE=MAP	Mapset	USAGE
USAGE=MAP	Partitionset	USAGE
USAGE=MAP	Program	USAGE
USERID	Sessions	USERID
USERID	Terminal	USERID
USERSEC	Connection	ATTACHSEC
USERSEC	Terminal	ATTACHSEC
VF	Typeterm	VERTICALFORM
XSNAME	Connection	SECURITYNAME
XSNAME	Terminal	SECURITYNAME
XTRANID	Transaction	XTRANID

Notes:

1. No RDO equivalent is provided.
2. There is no equivalent attribute in RDO because anticipatory paging is no longer supported.
3. There is no RDO equivalent of FDUMP. Control over the production of dumps is available by manipulating the system dump table.
4. The equivalent of DTB=NO is not supported by RDO; the equivalent of DTB=YES is forced for those transactions migrated from the PCT to RDO.
5. The RDO equivalent of not specifying either NOROUTE or NOROUTEALL for BMSFEAT is ROUTEDMSGS(ALL).
6. The RDO equivalent of not specifying any of TRANCEIVE, RECEIVE, or TRANSACTION is ATI(NO), TTI(NO).
7. The RDO equivalent of not specifying either INTLOG or NOINTLOG for TRMSTAT is CREATESESS(YES).
8. The RDO equivalent of the macro keyword LSRPOOL is LSRPOOLID on CEDA DEFINE LSRPOOL or on CEDA DEFINE FILE.
9. Security checking is now mandatory for all transactions.

RDO keyword to macro operand

This list is in alphabetic order of RDO keyword, giving the equivalent macro operand. "Macro operand to RDO keyword" on page 377 gives the same list in order of macro operand.

RDO ATTRIBUTE	RESOURCE TYPE	MACRO OPERAND
ACCESSMETHOD	Connection	ACCMETH
ADD	File	SERVREQ=ADD
ALIAS	Transaction	See note 1.
ALTPAGE	Typeterm	ALTPGE
ALTPRINTCOPY	Terminal	ALTPRT(,COPY)
ALTPRINTER	Terminal	ALTPRT(label,)
ALTSCREEN	Typeterm	ALTSCRN
ALTSUFFIX	Typeterm	ALTSFX
APLKYBD	Typeterm	FEATURE=APLKYBD
APLTEXT	Typeterm	FEATURE=APLTEXT
ASCII(7)	Typeterm	FEATURE=ASCII-7
ASCII(8)	Typeterm	FEATURE=ASCII-8
ATI(NO) and TTI(YES)	Typeterm	TRMSTAT=TRANSACTION
ATI(YES) and TTI(NO)	Typeterm	TRMSTAT=RECEIVE
ATI(YES) and TTI(YES)	Typeterm	TRMSTAT=TRANSCEIVE
ATTACHSEC	Connection	USERSEC=IDENTIFY
ATTACHSEC	Connection	USERSEC=LOCAL
ATTACHSEC	Connection	USERSEC=MIXIDPE
ATTACHSEC	Connection	USERSEC=PERSISTENT
ATTACHSEC	Connection	USERSEC=VERIFY
ATTACHSEC	Terminal	USERSEC=IDENTIFY
ATTACHSEC	Terminal	USERSEC=LOCAL
ATTACHSEC	Terminal	USERSEC=MIXIDPE
ATTACHSEC	Terminal	USERSEC=PERSISTENT
ATTACHSEC	Terminal	USERSEC=VERIFY
AUDIBLEALARM	Typeterm	FEATURE=AUDALARM
AUTINSTMODEL	Terminal	See note 1.
AUTINSTNAME	Terminal	See note 1.
AUTOCONNECT	Connection	CONNECT
AUTOCONNECT	Sessions	CONNECT
AUTOCONNECT	Typeterm	CONNECT
AUTOPAGE(NO)	Typeterm	PGESTAT=PAGE
AUTOPAGE(YES)	Typeterm	PGESTAT=AUTOPAGE
BACKTRANS	Typeterm	FEATURE=BTRANS
BACKUPTYPE	File	See note 1.
BINDPASSWORD	Connection	BINDPWD
BINDPASSWORD	Terminal	BINDPWD
BRACKET	Typeterm	BRACKET
BROWSE	File	SERVREQ=BROWSE
BUILDCHAIN	Sessions	CHNASSY=YES
BUILDCHAIN	Typeterm	CHNASSY
CEDF	Program	See note 1.
CGCSGID	Typeterm	See note 1.
CHAINCONTROL	Profile	MSGPREQ=CCTRL
CMDSEC	Transaction	See note 1
COLOR	Typeterm	FEATURE=COLOR
CONFDATA	Transaction	See note 1.
CONNECTION	Connection	SYSIDNT
CONNECTION	Sessions	SYSIDNT
CONNTYPE	Connection	See note 1.
CONSOLE	Terminal	CONSLID
COPY	Typeterm	FEATURE=COPY
CREATESESS(NO)	Typeterm	TRMSTAT=NOINTLOG
CREATESESS(YES)	Typeterm	See note 2.
DATABUFFERS	File	BUFND

RDO ATTRIBUTE	RESOURCE TYPE	MACRO OPERAND
DATASTREAM	Connection	DATASTR
DEFSCREEN	Typeterm	DEFSCRN
DELETE	File	SERVREQ=DELETE
DEVICE	Typeterm	TRMTYPE
DISCREQ	Sessions	RELREQ=(,NO YES)
DISCREQ	Typeterm	RELREQ=(,NO YES)
DISPOSITION	File	DISP
DSNAME	File	DSNAME
DSNSHARING	File	DSNSHR
DTIMOUT	Transaction	DTIMOUT
DUALCASEKYBD	Typeterm	FEATURE=DCKYBD
DUMP	Transaction	DUMP
DVSUPRT	Profile	DVSUPRT
DYNAMIC	Transaction	See note 1
ERRCOLOR	Typeterm	ERRATT=color
ERRHIGHLIGHT(BLINK)	Typeterm	ERRATT=BLINK
ERRHIGHLIGHT(REVERSE)	Typeterm	ERRATT=REVERSE
ERRHIGHLIGHT(UNDERLINE)	Typeterm	ERRATT=UNDERLINE
ERRINTENSIFY	Typeterm	ERRATT=INTENSIFY
ERRLASTLINE	Typeterm	ERRATT=LASTLINE
EXTENDEDDES	Typeterm	FEATURE=EXTDS
FILE	File	FILE
FMHPARM(YES)	Typeterm	BMSFEAT=FMHPARM
FORMFEED	Typeterm	FF
FWDRECOVLOG	File	JID
GROUP	Terminal	See note 1.
GROUP	Tranclass	See note 1.
GROUP	Typeterm	See note 1.
HILIGHT	Typeterm	FEATURE=HILIGHT
HORIZFORM	Typeterm	HF
HSDATA4K	Lsrpool	BUFFERS=(4K(,count))
HSDATA8K	Lsrpool	BUFFERS=(8K(,count))
HSDATA12K	Lsrpool	BUFFERS=(12K(,count))
HSDATA16K	Lsrpool	BUFFERS=(16K(,count))
HSDATA20K	Lsrpool	BUFFERS=(20K(,count))
HSDATA24K	Lsrpool	BUFFERS=(24K(,count))
HSDATA28K	Lsrpool	BUFFERS=(28K(,count))
HSDATA32K	Lsrpool	BUFFERS=(32K(,count))
HSINDEX4K	Lsrpool	See note 1.
HSINDEX8K	Lsrpool	See note 1.
HSINDEX12K	Lsrpool	See note 1.
HSINDEX16K	Lsrpool	See note 1.
HSINDEX20K	Lsrpool	See note 1.
HSINDEX24K	Lsrpool	See note 1.
HSINDEX28K	Lsrpool	See note 1.
HSINDEX32K	Lsrpool	See note 1.
INBFMH	Profile	INBFMH
INDEXBUFFERS	File	BUFNI
INDOUBT(BACKOUT)	Transaction	DTB=YES
INDOUBT(COMMIT)	Transaction	DTB=YES,NO
INDOUBT(WAIT)	Transaction	DTB=YES,WAIT
INDSYS	Connection	INDSYS
INSERVICE (No)	Connection	TRMSTAT='OUT OF SERVICE'
INSERVICE (No)	Sessions	TRMSTAT='OUT OF SERVICE'
INSERVICE (No)	Terminal	TRMSTAT='OUT OF SERVICE'
IOAREALEN	Sessions	TIOAL
IOAREALEN	Typeterm	TIOAL
ISOLATE	Transaction	See note 1.
JNLADD(BEFORE)	File	JREQ=WN
JNLREAD(UPDATEONLY)	File	JREQ=RU

RDO to Macro

RDO ATTRIBUTE	RESOURCE TYPE	MACRO OPERAND
JNLREAD(READONLY)	File	JREQ=RO
JNLSYNCREAD	File	JREQ=SYN
JNLSYNWRITE(NO)	File	JREQ=ASY
JNLUPDATE	File	JREQ=WU
JOURNAL	File	JID
JOURNAL	Profile	JFILEID
KATAKANA	Typeterm	FEATURE=KATAKANA
KEYLENGTH	File	KEYLEN
LANGUAGE	Program	PGMLANG
LDCLIST	Typeterm	LDC
LIGHTPEN	Typeterm	FEATURE=SELCTPEN
LOCALQ	Transaction	LOCALQ
LOGMODE	Typeterm	LOGMODE
LOGONMSG	Typeterm	GMMMSG
LOGREC	Profile	LOGREC
LSRPOOL	Lsrpool	See note 1
LSRPOOLID	File	LSRPOOL
LSRPOOLID	Lsrpool	See note 4
MAPSET	Mapset	MAPSET
MAXACTIVE	Tranclass	See note 1.
MAXIMUM	Sessions	MAXSESS
MAXKEYLENGTH	Lsrpool	KEYLEN
MAXNUMRECS	File	SIZE
MAXQTIME	Connection	See note 1.
MODENAME	Profile	MODENAM
MODENAME	Sessions	MODENAM
MODENAME	Terminal	MODENAM
MSGINTEG	Profile	MSGPREQ=(,MSGINTEG)
MSGJRNL	Profile	MSGJRNL
MSRCONTROL	Typeterm	FEATURE=MSRCNTRL
NATLANG	Terminal	NATLANG
NEPCLASS	Profile	NEPCLAS
NEPCLASS	Sessions	See note 1
NEPCLASS	Typeterm	NEPCLAS
NETNAME	Connection	NETNAME
NETNAME	Terminal	NETNAME
NETNAMEQ	Sessions	NETNAMQ
NSRGROUP	File	BASE
OBFORMAT(YES)	Typeterm	BMSFEAT=OBFMT
OBOPERID(YES)	Typeterm	BMSFEAT=OBOPID
ONEWTE	Profile	MSGPREQ=(,ONEWTE)
OPENTIME	File	FILSTAT=OPEN CLOSED
PASSWORD	File	PASSWD
POOL	Terminal	PIPELN
PRIMEDSIZE	Transaction	PRMSIZE
PRINTADAPTER	Typeterm	FEATURE=PTRADAPT
PRINTER	Terminal	PRINTTO(label,)
PRINTERCOMP	Profile	PTRCOMP
PRINTERCOPY	Terminal	PRINTTO(,COPY)
PRIORITY	Transaction	TRNPRTY
PROFILE	Profile	PROFILE
PROGRAM	Program	PROGRAM
PROGRAM	Transaction	PROGRAM
PROGSYMBOLS	Typeterm	FEATURE=PS
PROTECT	Profile	MSGPREQ=(,PROTECT)
PROTOCOL	Connection	TRMTYPE
PROTOCOL	Sessions	TRMTYPE
PSRECOVERY	Connection	See note 1.
PURGETHRESH	Tranclass	See note 1.
QUERY(ALL)	Typeterm	FEATURE=QUERYALL

RDO ATTRIBUTE	RESOURCE TYPE	MACRO OPERAND
QUERY(COLD)	Typeterm	FEATURE=QUERYCOLD
QUEUELIMIT	Connection	See note 1.
RAQ	Profile	RAQ
READ	File	SERVREQ=READ
RECEIVECOUNT	Sessions	RECEIVE=(,n)
RECEIVEPFX	Sessions	RECEIVE=(x,)
RECEIVESIZE	Sessions	RUSIZE
RECEIVESIZE	Typeterm	RUSIZE
RECORDFORMAT	Connection	RECFM
RECORDFORMAT	File	RECFORM=(U V F)
RECORDSIZE	File	LRECL
RECOVERY	File	LOG=YES
RECOVNOTIFY	Typeterm	RNOTIFY
RECOVOPTION	Sessions	ROPTION
RECOVOPTION	Typeterm	ROPTION
RELOAD	Program	RELOAD
RELREQ	Sessions	RELREQ=(NO YES,)
RELREQ	Typeterm	RELREQ=(NO YES,)
REMOTENAME	Connection	RMTNAME
REMOTENAME	File	RMTNAME
REMOTENAME	Terminal	RMTNAME
REMOTENAME	Transaction	RMTNAME
REMOTESYSNET	Connection	See note 1.
REMOTESYSTEM	Connection	SYSIDNT
REMOTESYSTEM	File	SYSIDNT
REMOTESYSTEM	Terminal	SYSIDNT
REMOTESYSTEM	Transaction	SYSIDNT
RESIDENT	Mapset	RES
RESIDENT	Partitionset	RES
RESIDENT	Program	RES
RESSEC	Transaction	RSLC
RESTART	Transaction	RESTART
ROUTEDMSGS(NONE)	Typeterm	BMSFEAT=NOROUTE
ROUTEDMSGS(SPECIFIC)	Typeterm	BMSFEAT=NOROUTEALL
ROUTEDMSGS(ALL)	Typeterm	See note 3.
RTIMOUT	Profile	RTIMOUT
RUNAWAY	Transaction	See note 1.
SCRNSIZE	Profile	SCRNSZE
SECURITYNAME	Connection	XSNAME
SECURITYNAME	Terminal	XSNAME
SENDCOUNT	Sessions	SEND=(,n)
SENDPFX	Sessions	SEND=(x,)
SENDSIZE	Sessions	BUFFER
SENDSIZE	Typeterm	BUFFER
SESSIONS	Sessions	See note 1.
SESSIONTYPE	Typeterm	SESTYPE
SESSNAME	Sessions	TRMIDNT
SESSPRIORITY	Sessions	TRMPRTY
SHARELIMIT	Lsrpool	RSCLMT
SHIPPABLE	Typeterm	See note 1.
SHUTDOWN	Transaction	See note 1.
SIGNOFF	Typeterm	SIGNOFF
SINGLESESS	Connection	FEATURE=SINGLE
DATA1K	Lsrpool	BUFFERS=(1K(count))
DATA2K	Lsrpool	BUFFERS=(2K(count))
DATA4K	Lsrpool	BUFFERS=(4K(count))
DATA8K	Lsrpool	BUFFERS=(8K(count))
DATA12K	Lsrpool	BUFFERS=(12K(count))
DATA16K	Lsrpool	BUFFERS=(16K(count))
DATA20K	Lsrpool	BUFFERS=(20K(count))

RDO to Macro

RDO ATTRIBUTE	RESOURCE TYPE	MACRO OPERAND
DATA24K	Lsrpool	BUFFERS=(24K(count))
DATA28K	Lsrpool	BUFFERS=(28K(count))
DATA32K	Lsrpool	BUFFERS=(32K(count))
DATA512	Lsrpool	BUFFERS=(512(count))
SOSI	Typeterm	FEATURE=SOSI
SPURGE	Transaction	SPURGE
STATUS	File	FILSTAT=ENABLED DISABLED
STATUS	Mapset	PGMSTAT
STATUS	Partitionset	PGMSTAT
STATUS	Program	PGMSTAT
STATUS	Transaction	TRNSTAT
STORAGECLEAR	Transaction	See note 1.
STRINGS	File	STRNO
STRINGS	Lsrpool	STRNO
TABLE	File	TYPE=CICSTABLE USERTABLE
TASKDATAKEY	Connection	See note 1.
TASKLIMIT	Terminal	TASKNO
TASKREQ	Transaction	TASKREQ
TERMINAL	Terminal	TRMIDNT
TERMMODEL	Typeterm	TRMMODL
TERMPRIORITY	Terminal	TRMPRTY
TEXTKYBD	Typeterm	FEATURE=TEXTKYBD
TEXTPRINT	Typeterm	FEATURE=TEXTPRINT
TPURGE	Transaction	TPURGE
TRACE	Transaction	TRACE
TRANCLASS	Tranclass	See note 1.
TRANCLASS	Transaction	See note 1.
TRANSACTION	Terminal	TRANSID
TRANSACTION	Transaction	TRANSID
TRPROF	Transaction	TRPROF
TTI(NO) and ATI(YES)	Typeterm	TRMSTAT=RECEIVE
TTI(YES) and ATI(NO)	Typeterm	TRMSTAT=TRANSACTION
TTI(YES) and ATI(YES)	Typeterm	TRMSTAT=TRANCEIVE
TYPETERM	Terminal	See note 1.
TYPETERM	Typeterm	See note 1.
TWASIZE	Transaction	TWASIZE
UCTRAN	Typeterm	FEATURE=UCTRAN
UPDATE	File	SERVREQ=UPDATE
USAGE	Mapset	USAGE=MAP
USAGE	Partitionset	USAGE=MAP
USAGE	Program	USAGE=MAP
USELPACOPY	Mapset	SHR=YES
USELPACOPY	Partitionset	SHR=YES
USELPACOPY	Program	SHR=YES
USERAREALEN	Sessions	TCTUAL
USERAREALEN	Typeterm	TCTUAL
USERID	Sessions	USERID
USERID	Terminal	USERID
VALIDATION	Typeterm	FEATURE=VALIDATION
VERTICALFORM	Typeterm	VF
XRFSIGNOFF	Typeterm	See note 1
XTRANID	Transaction	XTRANID

Notes:

1. There is no macro equivalent.
2. CREATESESS(YES) is equivalent to not specifying either INTLOG or NOINTLOG for TRMSTAT.
3. ROUTEDMSGs(ALL) is equivalent to not specifying either NOROUTE or NOROUTEALL for BMSFEAT.
4. The macro equivalent of the LSRPOOLID keyword on CEDA DEFINE LSRPOOL is LSRPOOL on the DFHFCT macro.

Appendix D. Migrating the TCT to the CSD

This section describes the process of converting terminal and system definition macros for VTAM devices into their RDO equivalents. Before you read further, note that you **must** use RDO for the TCT if you use VTAM as your telecommunication access method for at least some of your resources, as VTAM TCTs are no longer supported online (other than LDCs).

Note also that there is a special facility, called **autoinstall**, that can remove the need for some migration. You may decide that you don't need to migrate all of your DFHTCT macros, because you are going to use autoinstall. It would be wise, nevertheless, to read the following section and maybe perform the migration process anyway. This will allow you to study the RDO definitions that the utility produces, and could help you decide how many different **autoinstall models** you will need. For more information, see Chapter 9, "Autoinstall for VTAM terminals" on page 105.

This chapter is divided into three sections:

1. The steps involved in the migration process
2. The steps involved in migrating remote terminal definitions, depending on the method of defining remote terminals you choose
3. Some notes on migrating different types of macro, including links and sessions with other systems.

The needs of your installation may introduce various complications into the migration process:

- You may wish to introduce autoinstall in stages, or for some terminals but not all
- If you have remote terminals for transaction routing, there are three approaches you could adopt toward migrating your terminal definitions, as described in "Migrating remote terminal definitions" on page 394.

Whatever your choices, you will finish up with a TCT built from one or more of the following:

- Entries built automatically through autoinstall, which are dynamic and will not be present in the TCT until autoinstalled
- Entries defined and installed using RDO and the GRPLIST operand of DFHSIT
- Entries for non-VTAM terminals defined using DFHTCT macros.

Autoinstalled TCT entries never override any others. If a table entry already exists for a given terminal, CICS will not try to autoinstall it. (For more information see Chapter 9, "Autoinstall for VTAM terminals" on page 105.)

Summary of implementation

To migrate to RDO, you have to do the following:

1. Edit your existing DFHTCT source.
2. Assemble and link-edit the edited source.
3. Process the link-edited table using the MIGRATE command of the DFHCSDUP utility.
4. Check the output produced by the MIGRATE command.
5. Use the DFHCSDUP ADD command to create a list of groups on your CSD. Name this list in your SIT for initialization.
6. Modify your macro source so that it no longer generates table entries that have been migrated, then reassemble it to produce the non-VTAM part of your TCT (if any).
7. Start up your system with the non-VTAM TCT (if any) and the list you have created. Use one of the terminals whose definition was included in the list to further modify your list and perhaps to regroup and rename migrated definitions.

The rest of this section discusses these steps in more detail.

Editing your existing DFHTCT source

The first step in migrating your DFHTCT macros to the CSD is to assemble the source. If your TCT is large, you might find it most convenient to assemble small segments of the source individually.

Each resource definition on the CSD must belong to a CSD group. DFHCSDUP MIGRATE places all definitions in groups as it processes them.

You probably already have good reasons for grouping resources. For example, you might like to group TERMINAL definitions according to the physical location of the terminals, or according to the department or function of the people using them.

You can specify the names to be given to groups of definitions generated from your DFHTCT macros. You do this by adding special macro instructions to the table source. The rules for group names are given on page 199. The form of each macro instruction is:

```
DFHTCT TYPE=GROUP,GROUP=xxxxxxx      (up to 8 characters)
```

All definitions following a particular TYPE=GROUP macro statement will be migrated into the named group in the CSD file. A new TYPE=GROUP statement overrides all previous ones.

If you don't insert such macros into your deck, the whole table will be migrated into the same group on the CSD. That group will, by default, be named TCTxx, a substring of your table's name, DFHTCTxx.

It is better to create many small groups for your TERMINALS than one vast group. You should aim to have no more than 100 resource definitions in any one group. (See "How many resource definitions should a group contain?" on page 18.)

You must include ACCMETH=VTAM in your DFHTCT TYPE=INITIAL macro. This is to enable the necessary VTAM control blocks, for example the ACB, to be built as part of the TCT. Note that the default for the ACCMETH= operand is ACCMETH=NONVTAM.

Some information from the TYPE=INITIAL macro is not migrated, because it is now specified using the DFHSIT macro. For further guidance on this, see the *CICS/ESA System Definition Guide*.

Assembling and link-editing

When you have edited your terminal control table source, assemble it using the CICS/ESA 4.1 macro library specifying MIGRATE=YES on the DFHTCT TYPE=INITIAL macro, and then link-edit the assembler output.

The assembly and link-edit of a TCT will lead to the creation of two separate load modules. Assembly of a suffixed TCT (source name DFHTCTxx) produces a single text file. However, when this is link-edited into a load library, two members are created:

- DFHTCTxx, which contains the non-RDO-eligible definitions in control block format
- DFHRDTxx, which contains the RDO-eligible definitions in command format

You need to be aware of the existence of these two tables if you have to copy or move assembled TCT tables between load libraries.

If you get a return code greater than 4, remove the cause of the error and reassemble. (For example, internally duplicated entries are not acceptable.)

You must assemble and link-edit your tables into your CICS load library, using procedure DFHAUPLE.

Using the MIGRATE command

After assembling it and link-editing it, you migrate the revised part of your table using the DFHCSDUP MIGRATE command. The form of the command is:

```
MIGRATE TABLE(DFHTCTxx) [TYPESGROUP(tgrpname)]
```

For guidance on the JCL for DFHCSDUP, see the *CICS/ESA Operations and Utilities Guide*.

The migration process triggered off by this MIGRATE command results in the creation of CSD records for the following:

- TYPETERM definitions
- TERMINAL definitions
- SESSIONS definitions
- CONNECTION definitions.

For the RDO equivalents of macro operands, see “Macro operand to RDO keyword” on page 377.

For the macro equivalents of RDO attribute keywords, see “RDO keyword to macro operand” on page 382.

The most important points about these definitions, for the purposes of migration, are described here.

TYPETERM definitions

These are derived from attributes of TYPE=TERMINAL macros which are often identical for many terminals.

They are put into the CSD group named in the TYPESGROUP parameter. If no TYPESGROUP is specified, they are put in the group currently being created, with the TERMINAL definitions.

The “typeterm” attributes of each TYPE=TERMINAL table macro are checked with existing TYPETERM definitions and if they don’t match with any of these, a new TYPETERM is added to the CSD.

The existing TYPETERMs checked are:

- TYPETERMs in the group currently being created
- TYPETERMs in the group specified in the TYPESGROUP parameter of the MIGRATE command

However, the scope of the checking is never extended to include any other TYPETERMs in other groups already on the CSD. Such groups may have been created using RDO or by a previous MIGRATE.

For this reason, it is a good idea to use the TYPESGROUP parameter to avoid creating duplicate TYPETERMs in different groups. It is convenient to keep the TYPETERMs in a separate group anyway.

TYPETERMs created on the CSD during the migration are named systematically, in a way related to the TRMTYPE parameter of the original terminal definition. The name will consist of a prefix (of 3-5 characters) with a 3-character suffix.

For example, a TYPETERM defining attributes for a 3270 printer will be named 3270P001. Variants with the same TRMTYPE will be named 3270P002, and so on. The migration process will ensure that this name is used as the TYPETERM parameter of every TERMINAL definition that references it.

Migration may produce some TYPETERM names that would be incomprehensible to those who will have to use them when defining terminals. You can later rename such

TCT migration

TYPETERMs using the RDO command RENAME, and ALTER the TERMINAL definitions that refer to them. The naming rules for TYPETERM identifiers are given on page 239.

TERMINAL definitions

The operands of a TYPE=TERMINAL macro that are not accounted for by the TYPETERM named on the new TERMINAL definition, become the other attributes of the TERMINAL definition. The TERMINAL name comes from the old TRMIDNT. The naming rules for TERMINAL identifiers in RDO are given on page 201.

This means that if the existing TRMIDNT contains characters not belonging to this set, the definition will not be migrated from the TCT to the CSD, and you must define these resources with CEDA, using new names acceptable to RDO.

All the TERMINAL definitions created by the migration process have AUTINSTMODEL(NO) and they all point to a TYPETERM definition with SHIPPABLE(NO).

SESSIONS definitions

SESSIONS definitions created by migration are different for different types of links and sessions. They are described in “Migrating different macro types” on page 396. SESSIONS identifiers in RDO are subject to the same character set restrictions as TERMINAL identifiers.

CONNECTION definitions

CONNECTION definitions created by migration are different for different types of links and sessions. They are described in “Migrating different macro types” on page 396. CONNECTION identifiers in RDO are subject to the same character set restrictions as TYPETERM identifiers. This restriction has always applied to the SYSIDNT name in the DFHTCT macro.

Checking the output from MIGRATE

When you have migrated some table entries you should check that the process has worked satisfactorily.

The output listing from the MIGRATE utility tabulates all definitions successfully migrated to the CSD. The listing contains diagnostic messages for resources that failed to migrate for some reason. For example, you may have terminals whose TRMIDNTs contain characters not acceptable to RDO.

You can use RDO to DEFINE resources that have failed to migrate. You should make sure that you define these resources on the CSD, and make any changes to application programs that depended on TERMINAL names that you have had to change, **before** you reassemble the TCT with MIGRATE=COMPLETE.

Using DFHCSDUP to add migrated groups to a list

You must install some migrated definitions in the system when you initialize it. You do this by means of a list named in the DFHSIT GRPLIST operand. The MIGRATE command created groups of resource definitions. You must create a list by using the ADD command to add some of your groups to it. For more information about this, see “ADD a group to a list” on page 44. You can choose a name for the list: you will specify this name in the GRPLIST operand.

To start off with, you need to include a group containing a definition of a terminal that you can use for RDO: to create other lists and groups, and to install other groups of definitions in the active CICS system. To enable you to use RDO, you also need to include in your list the CICS-supplied definitions for the resources RDO itself uses, and for the resources used

by other CICS-supplied transactions, including CEMT, that you will want to use. The easiest way to do this is to use the APPEND command to append the list called DFHLIST to your own list. For more information about this, see “APPEND a list to another list” on page 85.

Operations after migration

When you are happy that you have no problems using RDO, how you continue depends on whether you can completely abandon the use of the TCT.

You must continue to use a TCT containing resource definition macros, if you have any of the following resources:

- BTAM terminals in a remote system
- Sequential devices
- Logical device codes (LDCs).

If you cannot abandon your TCT, follow the advice in “Final modification of the macro source.” If you can abandon your TCT, follow the advice in “Abandoning the use of DFHTCT macros (if possible).”

Final modification of the macro source

If you need to retain a TCT to manage resources that you cannot define using RDO, you should eventually remove from your TCT source all the definitions that have been successfully migrated to the CSD, or that are now being created by autoinstall. This will save time on table assemblies.

After reducing the TCT source to a bare minimum, you should reassemble it to ensure that you have not made any mistakes.

You continue to suffix the TCT, and code TCT=xx, where xx is the table name suffix, in DFHSIT or as an initialization override. If you have VTAM resources **and** DFHTCT macros, you must code ACCMETH=(VTAM, NONVTAM) in the TCT.

Abandoning the use of DFHTCT macros (if possible)

If you are able to use RDO, with or without autoinstall, to manage **all** the resources formerly defined using the DFHTCT macroinstructions, you can now abandon your DFHTCT macro source altogether. You should adopt the following procedure:

1. First check that the assembly confirms that all the definitions in the TCT were eligible for RDO.
2. Check again that the CSD now contains all the definitions needed for operation with RDO. If you are using autoinstall, make sure that the CSD contains all necessary model TERMINAL definitions.
3. If the CSD now contains all the necessary definitions, you should code TCT=NO as an initialization parameter in the SIT. This will ensure that a “dummy” TCT (DFHTCTDY) is used.

DFHTCTDY is supplied by IBM. It contains predefined values for the DFHTCT TYPE=INITIAL parameters. If you wish to use different DFHTCT TYPE=INITIAL parameters, you may code and assemble your own version of DFHTCTDY. Note that some of the old DFHTCT TYPE=INITIAL options are now coded as SIT operands. For further guidance on this, see the *CICS/ESA System Definition Guide*.

Where next?

If you have only straightforward terminal definitions to migrate, you can go ahead and migrate them now.

If you have remote terminal definitions for transaction routing, you will find more guidance about migrating them in the next section.

The chapter ends with a summary of the different types of TCT macro, showing what the migration process does with each type. You will find this particularly helpful in planning your migration if you have intercommunication resources.

Migrating remote terminal definitions

“Terminals for transaction routing” on page 188 describes three methods of defining terminals so that they can be used for transaction routing. If you use transaction routing using MRO or APPC ISC, you should read that section before migrating your TCT.

When you have decided which method you will use to define transaction routing terminals, you should follow the appropriate migration procedure, described below. What you actually have to do to perform the migration depends on the type of macros you have used to define your remote terminals. If you have TYPE=REMOTE entries, you may have to migrate them to become remote definitions, just as you migrate the ordinary TYPE=TERMINAL entries to become local definitions. If you have TYPE=REGION macros, you probably use copy book definitions for both local and remote entries, as shown in this example. We refer to the migration procedure for this example in the procedures below.

Table DFHTCTLA used for local system (ACIC)

```
DFHTCT TYPE=INITIAL,SUFFIX=LA,SYSDNT=ACIC
.
.
.
COPY COMTERMS
.
.
.
DFHTCT TYPE=FINAL
```

Table DFHTCTRB used for remote system (BCIC)

```
DFHTCT TYPE=INITIAL,SUFFIX=RB,SYSDNT=BCIC
.
.
.
DFHTCT TYPE=REGION,SYSDNT=ACIC
COPY COMTERMS
.
.
.
DFHTCT TYPE=FINAL
```

Method 1. Maintaining local and remote definitions separately

To migrate your TCT definitions to one or more CSD files you should do the following:

1. Assemble the TCT for the terminal-owning system.
2. Assemble the TCT for the application-owning system. If you have more than one application-owning system, they may be able to share remote definitions, so you may not need to assemble more than one application-owning TCT.
3. Allocate definitions for different systems to different groups, if sharing a CSD between systems.

In the copy book example, if you use TYPE=GROUP macros to delimit groups within COMTERMS, you will have to edit COMTERMS after assembling DFHTCTLA and before assembling DFHTCTRB to change the TYPE=GROUP macros to name a different set of groups. (This will not apply if the tables are to be migrated to different CSD files.)

4. Use DFHCSDUP to migrate all the TCTs. The commands look like this:

```
MIGRATE TABLE(DFHTCTLA) TYPESGROUP(TTS)
MIGRATE TABLE(DFHTCTRB) TYPESGROUP(TTS)
```

Migration of the remote terminals will normally create definitions that use the same TYPETERMs created for the corresponding local definitions.

5. Include the groups containing the local definitions in the GRPLIST for the terminal-owning system.
6. Include the groups containing the remote definitions in the GRPLIST for each application-owning system.

If there are only a small number of terminals that need more than one definition, it is probably best to migrate all the definitions in the terminal-owning TCT and then to use RDO to create the corresponding remote definitions with the required REMOTESYSTEM attribute. You can use the COPY and ALTER commands to do this.

Method 2. Sharing terminal definitions

To migrate your TCT definitions to the shared CSD file, you should do the following:

1. Suppress the assembly of remote definitions in all the TCTs to be migrated, by removing (or commenting-out) the TCT source for:
 - TYPE=REMOTE entries
 - TYPE=REGION and subsequent TYPE=TERMINAL entries.
2. Assemble the TCT for the terminal-owning system, suppressing any macros for remote definitions that you may have if there is more than one terminal-owning system.
3. Assemble the TCT for the application-owning system, if it contains any definitions for local terminals. Again, you should suppress any macros for remote definitions, if you do need to assemble this TCT.
4. Use DFHCSDUP to migrate all the TCTs. The commands look like this:

```
MIGRATE TABLE(DFHTCTLA) TYPESGROUP(TTS)
MIGRATE TABLE(DFHTCTRB) TYPESGROUP(TTS)
```

Migration of the remote terminals will normally create definitions that use the same TYPETERMs created for the corresponding local definitions.

5. Initialize the CICS terminal-owning system, installing at least one TERMINAL definition and other resource definitions necessary for using RDO.
6. Use the ALTER command to name the REMOTESYSTEM as the SYSIDNT of the terminal-owning system, on all the TERMINAL definitions that might be shared. You can probably use a generic name to do this in one command. For example:


```
CEDA ALTER TERMINAL(*) GROUP(TTS) REMOTESYSTEM(ACIC)
```
7. Include the groups containing the definitions in the GRPLIST for each system, terminal-owning and application-owning.

Method 3. Making terminal definitions shippable

To migrate your TCT definitions to one or more CSD files you should do the following:

1. Suppress the assembly of remote definitions in all the TCTs to be migrated, by removing (or commenting-out) the TCT source for:
 - TYPE=REMOTE entries
 - TYPE=REGION and subsequent TYPE=TERMINAL entries.

TCT migration

2. Assemble the TCT for the terminal-owning system, suppressing any macros for remote definitions that you may have if there is more than one terminal-owning system.
3. Assemble the TCT for the application-owning system, if it contains any definitions for local terminals. Again, you should suppress any macros for remote definitions, if you do need to assemble this TCT.
4. Use DFHCSDUP to migrate the TCT. The command looks like this:

```
MIGRATE TABLE(DFHTCTLA) TYPESGROUP(tgrpname)
```
5. Initialize the CICS terminal-owning system, installing at least one TERMINAL definition and other resource definitions necessary for using RDO.
6. Use the ALTER command to change the SHIPPABLE attribute to YES, on all the TYPETERM definitions that might be used for shipping. You can probably use a generic name to do this in one command. For example:

```
CEDA ALTER TYPETERM(*) GROUP(grpname) SHIPPABLE(YES)
```
7. Include the groups containing the definitions in the GRPLIST for the terminal-owning system.

Migrating different macro types

The following is a summary of the different types of TCT macro that you might have, and what the results of the RDO migration will be. Migration of macros for the following devices and systems is described:

- Remote terminals for transaction routing
- MVS Consoles
- Pipeline terminals for VTAM pooled sessions
- Devices with LDC lists
- Terminals referencing printers
- MRO links and sessions
- LUTYPE6.1 CICS-CICS ISC links and sessions
- LUTYPE6.1 CICS-IMS links and sessions
- APPC (LUTYPE6.2) links and parallel sessions
- APPC (LUTYPE6.2) single session terminal
- Shared DL/I database links
- INDIRECT connections.

Remote terminals for transaction routing

You may have coded your macros in one of two ways:

- Individual terminals naming remote system:

```
DFHTCT TYPE=REMOTE,SYSIDNT=ssss,TRMIDNT=tttt,TRMTYPE=.....
```
- Or a series of TYPE=TERMINAL macros not naming the SYSIDNT, but following a TYPE=REGION that names the SYSIDNT:

```
DFHTCT TYPE=REGION,SYSIDNT=ssss
DFHTCT TYPE=TERMINAL,TRMIDNT=ttt1,TRMTYPE=yyyy...
DFHTCT TYPE=TERMINAL,TRMIDNT=ttt2,TRMTYPE=yyyy...
... and so on.
```

DFHCSDUP MIGRATE always produces a TERMINAL-TYPETERM pair of definitions:

```
DEFINE TERMINAL(tttt) GROUP(g) TYPETERM(xxxxxxxx)
      REMOTESYSTEM(ssss)
DEFINE TYPETERM(xxxxxxxx) GROUP(g) DEVICE(dddddddd)
```

Matching TYPETERMs are eliminated by DFHCSDUP MIGRATE, as described for ordinary terminals.

MVS consoles

The definition of an MVS console is essentially a special case of a local terminal definition. The TYPETERM in RDO identifies the device as a console, with the CONSOLE value for the DEVICE attribute. The console identifier for a particular console is specified on the TERMINAL definition as the CONSOLE attribute, which is the equivalent of the CONSLID= value on the macro. It is also possible to use the CONSNAME attribute of the TERMINAL definition (see page 199).

Pipeline terminals for VTAM pooled sessions

These terminals represent a special case of the definition of VTAM terminals.

A sequence of TYPE=TERMINAL macros is coded, the last one being tagged with PIPELN=LAST to indicate that the pool is complete:

```
DFHTCT TYPE=GROUP, GROUP=poolg
DFHTCT TYPE=TERMINAL, TRMIDNT=ttt1, TRMTYPE=(3600|3650),
      NETNAME=nnnnnnn1, SESTYPE=PIPELN, PIPELN=POOL
DFHTCT TYPE=TERMINAL, TRMIDNT=ttt2, TRMTYPE=(3600|3650),
      NETNAME=nnnnnnn2, SESTYPE=PIPELN, PIPELN=POOL
DFHTCT TYPE=TERMINAL, TRMIDNT=ttt3, TRMTYPE=(3600|3650),
      NETNAME=nnnnnnn3, SESTYPE=PIPELN, PIPELN=POOL
DFHTCT TYPE=TERMINAL, TRMIDNT=ttt4, TRMTYPE=(3600|3650),
      NETNAME=nnnnnnn4, SESTYPE=PIPELN, PIPELN=LAST, TASKNO=nn
```

When migrated using:

```
MIGRATE TABLE(DFHTCTxx) TYPESGROUP(typeg)
```

these macros result in the following definitions:

```
DEFINE TERMINAL(ttt1) GROUP(poolg) POOL(nnnnnnn1) TYPETERM(xxxxxxx1)
      NETNAME(nnnnnnn1)
DEFINE TERMINAL(ttt2) GROUP(poolg) POOL(nnnnnnn1) TYPETERM(xxxxxxx1)
      NETNAME(nnnnnnn2)
DEFINE TERMINAL(ttt3) GROUP(poolg) POOL(nnnnnnn1) TYPETERM(xxxxxxx1)
      NETNAME(nnnnnnn3)
DEFINE TERMINAL(ttt4) GROUP(poolg) POOL(nnnnnnn1) TYPETERM(xxxxxxx1)
      NETNAME(nnnnnnn4) TASKLIMIT(nn)
DEFINE TYPETERM(xxxxxxx1) GROUP(typeg) DEVICE(3600|3650)
      SESSIONTYPE(PIPELINE)
```

The POOL name is automatically generated using the NETNAME of the first TERMINAL in the POOL.

When the pool is installed, the terminal IDs are sorted in ascending alphabetic order. The first terminal to be installed will become the pool header.

Matching TYPETERMs are eliminated by DFHCSDUP MIGRATE, as described for ordinary terminals.

Devices with LDC lists

For 3600, 3770 batch, 3770 and 3790 batch data interchange, and LUTYPE4 logical units, you can specify the name of an LDC list (Logical Device Code list). In RDO this information is held on the TYPETERM definition.

You cannot define the LDC list itself using RDO. The LDC list and its contents must still be defined by the macro method. The TERMINAL and TYPETERM using the LDC list can be created in RDO, and the LDC list defined by using the macro is named by the LDCLIST attribute on the TYPETERM definition.

TCT migration

You cannot define a list of LDC codes explicitly on the TYPETERM definition.⁸ This simplifies the interface to these facilities and allows tables to be migrated to the CSD. If you have a DFHTCT TYPE=TERMINAL macro with an LDC= specification of the form:

```
LDC=(aa=nnn,bb=nnn,...)
```

then the assembly will produce a level 8 MNOTE. This will tell you to recode the list either as a **local LDC list** or as an **extended local LDC list**. (See page 231.)

You should then recode the terminal entry in this form:

```
DFHTCT TYPE=TERMINAL,TRMIDNT=tttt,TRMTYPE=uuuuu,  
      ACCMETH=VTAM,LDC=nnnnnnnn
```

where nnnnnnnn is the name you gave your LDC list when you defined it.

DFHCSDUP MIGRATE always produces a TERMINAL-TYPETERM pair of definitions:

```
DEFINE TERMINAL(tttt) TYPETERM(xxxxxxxxx)  
DEFINE TYPETERM(xxxxxxxxx) DEVICE(dddddddd) LDCLIST(nnnnnnnn)
```

Terminals referencing printers

A pair of TCT entries could be related, when a terminal references a printer, by using the PRINTTO or ALTPRT operands in the TCT macro, which referred to the printer TCTTE by means of the assembler label of the printer entry.

In RDO, the terminal refers to its associated printer by the 4-character TERMINAL name (old TRMIDNT) of the definition for the printer. The corresponding keywords in RDO are PRINTER and ALTPRINTER.

```
      DFHTCT TYPE=TERMINAL,TRMIDNT=tttt,  
            TRMTYPE=3270,PRINTTO=label1  
label1 DFHTCT TYPE=TERMINAL,TRMIDNT=pppp,TRMTYPE=3270P
```

When migrated, this becomes:

```
DEFINE TERMINAL(tttt) TYPETERM(xxxxxxxxx) PRINTER(pppp)  
DEFINE TERMINAL(pppp) TYPETERM(xxxxxxxxx)
```

Links and sessions - method 1

This method applies both to MRO and to LUTYPE6.1 CICS-CICS ISC links and sessions.

MRO links and sessions

An MRO link and a set of parallel sessions are defined by:

```
DFHTCT TYPE=SYSTEM,SYSDNT=ssss,ACCMETH=IRC|IRC,XM),
```

DFHCSDUP MIGRATE always produces a CONNECTION-SESSIONS pair of definitions:

```
DEFINE CONNECTION(ssss) ACCESSMETHOD(IRC|XM)  
DEFINE SESSIONS(sssssprp) CONNECTION(ssss) PROTOCOL(LU61)  
      SENDPFX(sp)      SENDCOUNT(m)  
      RECEIVEPFX(rp)   RECEIVECOUNT(n)
```

The SESSIONS name sssssprp is synthesized by concatenating the names of the SYSDNT, SEND(sp) and RECEIVE(rp), for example:

```
SYSDNT=BCIC,SEND=(SA,5),RECEIVE=(RA,3)  
=====> SESSIONS(BCICSARA)
```

⁸ This also applies to the DFHTCT TYPE=TERMINAL macro, although earlier CICS releases allowed explicit definition of a list of LDC codes.

LUTYPE6.1 CICS-CICS ISC links and sessions

These are as for MRO, but with ACCESSMETHOD(VTAM).

LUTYPE6.1 CICS-CICS ISC links and sessions may also be defined and migrated as for method 2.

Links and sessions - method 2

This method applies to both LUTYPE6.1 CICS-CICS ISC and LUTYPE6.1 CICS-IMS links and sessions.

LUTYPE6.1 CICS-IMS links and sessions

The ISC link is defined by:

```
DFHTCT TYPE=SYSTEM, SYSIDNT=ssss, ACCMETH=VTAM
```

The parallel sessions for the link are defined individually by:

```
DFHTCT TYPE=TERMINAL, TRMIDNT=tttt, SYSIDNT=tttt, TRMTYPE=LUTYPE6,
      SESTYPE(SEND|RECEIVE), NETNAMQ=nnnnnnnn
```

DFHCSDUP MIGRATE produces a CONNECTION definition from the TYPE=SYSTEM macro:

```
DEFINE CONNECTION(ssss) ACCESSMETHOD(VTAM) PROTOCOL(LU61)
```

and a SESSIONS definition from each subsequent TYPE=TERMINAL macro.

For a SEND session:

```
DEFINE SESSIONS(sssstttt) CONNECTION(ssss) PROTOCOL(LU61)
      SESSNAME(tttt) NETNAMEQ(nnnnnnnn)
      SENDCOUNT(1)
```

For a RECEIVE session:

```
DEFINE SESSIONS(sssstttt) CONNECTION(ssss) PROTOCOL(LU61)
      SESSNAME(tttt) NETNAMEQ(nnnnnnnn)
      RECEIVECOUNT(1)
```

The SESSIONS name sssstttt is synthesized by concatenating the old SYSIDNT and TRMIDNT values.

The SESSNAME name tttt is the macro TRMIDNT value.

The NETNAMEQ name nnnnnnnn is the macro NETNAMQ value.

LUTYPE6.1 CICS-CICS ISC links and sessions

These are as for CICS-IMS but without NETNAMEQ.

LUTYPE6.1 CICS-CICS ISC links and sessions may also be defined and migrated as for method 1.

Shared DL/I database links

A link between a CICS system and a batch system that uses a DL/I database with CICS is defined by:

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

The migration always produces a CONNECTION-SESSIONS pair of definitions:

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

You can use this method to define DL/I shared database links to CICS, but we recommend that you use IMS/ESA data sharing instead. This does not use CICS IRC, nor does it require any CICS resource definition.

APPC (LUTYPE6.2) links and parallel sessions

For APPC, the sessions are grouped into modesets. Each modeset is defined in the macro method with a TYPE=MODESET macro, and the equivalent in RDO is one SESSIONS definition on the CSD.

The ISC link is defined by:

```
DFHTCT TYPE=SYSTEM,SYSIDNT=ssss,ACCMETH=VTAM,TRMTYPE=LUTYPE62
```

The parallel sessions for the link are defined collectively by:

```
DFHTCT TYPE=MODESET,MODENAM=mmmmmmm,SYSIDNT=ssss,
      MAXSESS=(m1,m2)
```

DFHCSDUP MIGRATE produces a CONNECTION definition from the TYPE=SYSTEM macro:

```
DEFINE CONNECTION(ssss) ACCESSMETHOD(VTAM) PROTOCOL(APPC)
```

and a SESSIONS definition from the TYPE=MODESET macro:

```
DEFINE SESSIONS(xxxxxxx) CONNECTION(ssss) PROTOCOL(APPC)
      MAXIMUM(m1,m2) MODENAME(mmmmmmm)
```

The SESSIONS name xxxxxxxx is derived from the old SYSIDNT value concatenated with a 3-character identifier generated using the same algorithm as the old macro, for example for SYSIDNT=SYS1:

```
MODESET1: MAXSESS=4 gives SESSIONS(SYS1AAC) (starting count)
MODESET2: MAXSESS=2 gives SESSIONS(SYS1AAG) (MODESET1 + 4)
MODESET3: MAXSESS=3 gives SESSIONS(SYS1AAI) (MODESET2 + 2)
```

The MODENAME name mmmmmmmm is the macro MODENAM value.

APPC (LUTYPE6.2) single session terminal

A single TYPE=SYSTEM macro is used:

```
DFHTCT TYPE=SYSTEM,SYSIDNT=ssss,ACCMETH=VTAM,TRMTYPE=LUTYPE62,
      FEATURE=SINGLE,MODENAM=mmmmmmm
```

DFHCSDUP MIGRATE produces a CONNECTION definition and a SESSIONS definition:

```
DEFINE CONNECTION(ssss) ACCESSMETHOD(VTAM) PROTOCOL(APPC)
      SINGLESESS(YES)
```

```
DEFINE SESSIONS(xxxxxxx) CONNECTION(ssss) PROTOCOL(APPC)
      MODENAME(mmmmmmm) MAXIMUM(1,0)
```

The SESSIONS name xxxxxxxx is derived from the old SYSIDNT value concatenated with a 3-character identifier generated using a similar algorithm to the old macro, for example, for SYSIDNT=SYS1:

```
1st definition: gives SESSIONS(SYS1AAC) (starting count)
2nd definition: gives SESSIONS(SYS1AAF) ...
```

The MODENAME name mmmmmmmm is the macro MODENAM value.

You will not be able to autoinstall these CONNECTION and SESSIONS definitions. If you want to use autoinstall for your APPC single session terminals, you must redefine them as TERMINALS referencing a TYPETERM with DEVICE(APPC) (see page 188).

INDIRECT connections

The intermediate system is defined as:

```
DFHTCT TYPE=SYSTEM,SYSIDNT=yyyy,ACCMETH=IRC|(IRC,XM)|VTAM
```

This is migrated as any other TYPE=SYSTEM macro would be.

The indirect link is defined as:

```
DFHTCT TYPE=SYSTEM,SYSIDNT=xxxx,ACCMETH=INDIRECT,INDSYS=yyyy,  
NETNAME=nnnnnnnn
```

The migration of the macro for the indirect link produces an RDO definition of the form:

```
DEFINE CONNECTION(xxxx) INDSYS(yyyy) ACCESSMETHOD(INDIRECT)  
NETNAME(nnnnnnnn)
```

Index

Special Characters

@BCH, shared DL/I batch links 399

Numerics

3270 terminals (non-SNA)

eligible for autoinstall 105

3270E option

FEATURE operand 315

3290 terminal 223

3600 option

LDC operand 304

3614 and 3624 devices

ineligible for autoinstall 105

3770 Data Communication System

LDCs for batch LU

TCT example 306

6670 Information Distributor

TCT example 306

A

ABCODE operand

DFHSRT TYPE=SYSTEM|USER 297

abend codes 297

DFHSRT TYPE=SYSTEM 297

DFHSRT TYPE=USER 297

above the line, loading 245

ACCESS operand

DFHDLDBD TYPE=ENTRY 267

ACCESSMETHOD attribute 134

ACCMETH operand

DFHFCT TYPE=FILE 270

DFHTCT TYPE=INITIAL 300

DFHTCT TYPE=LINE 308, 311

DFHTCT TYPE=REMOTE 322

DFHTCT TYPE=TERMINAL 312, 322

remote terminals 322

sequential devices 308

TCAM devices 311, 312

ADD command

CEDA 44

DFHCSDUP 82

ADD option

SERVREQ operand 274

ADDCNT option

PERFORM operand 287

ALIAS attribute

TRANSACTION definition 208

ALTER (resource definition) command

PROFILE attribute keywords 163

ALTER command

CEDA 45

DFHCSDUP 83

ALTPAGE attribute

TYPETERM 226

ALTPGE operand

DFHTCT TYPE=TERMINAL 312

TCAM devices 312

ALTSCREEN attribute

TYPETERM 226

ALTSCRN operand

DFHTCT TYPE=TERMINAL 312

TCAM devices 312

ALTSFX operand

DFHTCT TYPE=TERMINAL 313

TCAM devices 313

ALTSUFFIX attribute

TYPETERM 227

AMT (autoinstall model table) 110

APF (authorized program facility) 253, 295

APLKYBD attribute

TYPETERM 227

APLKYBD option

FEATURE operand 315

APLTEXT attribute

TYPETERM 227

APLTEXT option

FEATURE operand 315

APPEND command

CEDA 47

DFHCSDUP 85, 86

examples 86

APPLID field on CEDA panels 29

APPLID of remote system 137

APPLID operand

DFHSIT

for controlling access to groups and lists 25, 61, 67

DFHSIT macro 132

ARCHJCL operand

DFHJCT TYPE=ENTRY 280

ASCII attribute

TYPETERM 227

ASIS option

SYSWAIT operand 283

ATI attribute

TYPETERM 228

ATTACHSEC attribute

CONNECTION 135

AUDALARM option

FEATURE operand 314

AUDIBLEALARM attribute

TYPETERM 228

authorized program facility (APF) 253, 295
 AUTINSTMODEL definition
 installed singly 20, 59
 AUTINSTMODEL TERMINAL definition
 keeping in a separate group 18
 AUTOARCH option
 JOUROPT operand 281
 AUTOCONNECT attribute
 CONNECTION 135
 SESSIONS definition 177
 TYPETERM 108, 113, 115, 228
 autoinstall
 for APPC connections 119
 for programs, mapsets, and partitionsets 123
 models 109, 116
 Autoinstall and VTAM 109
 autoinstall control program
 CICS-supplied (DFHZATDX) 116
 user-written 116
 automatic logging 281
 automatic teller machines (3614 and 3624)
 ineligible for autoinstall 105
 AUTOPAGE attribute
 TYPETERM 228

B

BACKTRANS attribute
 TYPETERM 228
 batch links with shared DL/I, TYPE=IRCBCH 399
 BCHLU option
 LDC operand 304
 BINDPASSWORD attribute
 CONNECTION 341
 BINDSECURITY attribute
 CONNECTION 135
 BLKKEYL operand
 DFHFCT TYPE=FILE 270
 BLKSIZE operand
 DFHDCT TYPE=SDSCI 257
 DFHFCT TYPE=FILE 270
 DFHTCT TYPE=SDSCI 307, 310
 sequential devices 307
 TCAM devices 310
 BLOCKED option
 RECFORM operand 273
 BMS ROUTE requests
 autoinstalled terminals 107
 BMSFEAT operand
 TCAM devices 313
 TYPE=TERMINAL 313
 BRACKET attribute
 TYPETERM 228
 BROWSE option
 SERVREQ operand 274

BTAM 221, 393
 BTRANS option
 FEATURE operand 314
 BUFFER operand
 DFHTCT TYPE=TERMINAL 313
 TCAM devices 313
 BUFNO operand
 DFHDCT TYPE=SDSCI 258
 BUFSIZE operand
 DFHJCT TYPE=ENTRY 280
 BUILDCHAIN attribute
 SESSIONS definition 177
 TYPETERM 228

C

CADL transient data destination 264
 CAIL transient data destination 264
 CCPI transient data destination 264
 CCSE transient data destination 264
 CCSI transient data destination 264
 CCSO transient data destination 264
 CDBC transient data destination 264
 CDUL transient data destination 264
 CEDF attribute
 PROGRAM definition 170
 CEMT INQUIRE TERMINAL
 lock on TCT entry 112
 CGCSGID attribute
 TYPETERM 228
 CHAINCONTROL attribute
 PROFILE 164
 CHECK command
 to check for consistency 24
 CHECK command (CEDA) 48
 checking resource definitions 24
 CICS system definition file (CSD)
 See CSD (CICS system definition file)
 CICS-supplied autoinstall control program
 (DFHZATDX) 116
 CICS-supplied autoinstall models 116
 CICS-supplied resource definitions 15
 CICS-supplied TYPETERM definitions 116
 CLASS operand
 DFHMCT TYPE=EMP 285
 DFHMCT TYPE=RECORD 288
 CLEARCONV attribute
 TYPETERM definition 180, 237
 CLOCK operand
 DFHMCT TYPE=EMP 286
 CLSDST, issued by CICS at logon 112
 CLT (command list table) 253—255
 CLT (command list table) 246, 253
 DFHCLT macros 253
 DFHCLT TYPE=COMMAND 254
 DFHCLT TYPE=INITIAL 253

CLT (command list table) *(continued)*
 DFHCLT TYPE=LISTEND 255
 DFHCLT TYPE=LISTSTART 254
 DFHCLT TYPE=WTO 254
 CMDSEC attribute
 TRANSACTION definition 208
 cold start
 nonrecovery of autoinstalled TCT entries 113
 use of GRPLIST to recreate tables 20
 using to remove table entries 20
 colon symbol on CEDA panels 30
 COLOR attribute
 TYPETERM 229
 COLOR option
 FEATURE operand 314
 command line on RDO panels
 entering values for attributes
 dependent defaults 26
 forced values 26
 command list table (CLT) 253—255
 CLT (command list table) 246, 253
 DFHCLT macros 253
 DFHCLT TYPE=COMMAND 254
 DFHCLT TYPE=INITIAL 253
 DFHCLT TYPE=LISTEND 255
 DFHCLT TYPE=LISTSTART 254
 DFHCLT TYPE=WTO 254
 COMMAND operand
 DFHCLT TYPE=COMMAND 254
 commands
 valid for locked group 25, 61, 67
 COMMIT attribute
 TRANSACTION definition 210
 commit process 20, 59
 communication lines, DFHTCT TYPE=LINE
 TCAM SNA devices 311
 compatibility mode 19
 access from CEDA panels 31
 COMPLETE option
 MIGRATE operand 269, 301
 CONNECTION attribute
 CONNECTION 136
 SESSIONS definition 177
 CONNECTION definition
 ACCESSMETHOD attribute 132, 134
 ATTACHSEC attribute 135
 AUTOCONNECT attribute 135
 BINDPASSWORD attribute 341
 BINDSECURITY attribute 135
 CONNECTION attribute 136
 CONNTYPE attribute 136
 DATASTREAM attribute 136
 DESCRIPTION attribute 136
 INDSYS attribute 132, 136
 INSERVICE attribute 136
 installed by group 21, 59
 CONNECTION definition *(continued)*
 keeping in a group with related SESSIONS 18
 MAXQTIME attribute 136
 NETNAME attribute 132, 137
 PROTOCOL attribute 132, 137
 PSRECOVERY attribute 137
 QUEUELIMIT attribute 138
 RECORDFORMAT attribute 138
 REMOTENAME attribute 138
 REMOTESYSNET attribute 138
 REMOTESYSTEM attribute 138
 SECURITYNAME attribute 139
 SINGLESESS attribute 139
 USEDFLTUSER attribute 139
 connections, INDIRECT
 resource definitions 132
 CONNTYPE attribute
 CONNECTION definition 136
 consoles
 ineligible for autoinstall 105
 preset security 203
 consumer transaction facility (3614 and 3624)
 ineligible for autoinstall 105
 continuous printing (autopaging) 305
 COPY attribute
 TYPETERM 229
 COPY command
 CEDA 50
 DFHCSDUP 87
 examples 87
 FROMCSD option 87
 MERGE option 87
 REPLACE option 87
 COPY option
 FEATURE operand 314
 COUNT operand
 DFHMCT TYPE=EMP 286
 CPLD transient data destination 265
 CPLI transient data destination 265
 CRDI transient data destination 265
 CREATESESS attribute
 TYPETERM 229
 creating a group 16
 creating a list 16
 CRUCIAL option
 JOUROPT operand 281
 CSCS transient data destination 265
 CSD (CICS system definition file)
 contents after initialization 347
 creating a CSD 15
 group i
 recommended maximum size 18
 recommended minimum size 18
 groups and lists 15
 not used at restart 20
 offline utility program, DFHCSDUP 73

CSD (CICS system definition file) *(continued)*
organization 15
sharing a CSD 15
sharing between MRO systems 24
use of multiple files for non-MRO CICS systems 25
use of read-only file for production system 25

CSD sharing 19
CSDL transient data destination 265
CSFL transient data destination 265
CSFU transaction 271
CSKL transient data destination 265
CSML transient data destination 265
CSNE transient data destination 265
CSPL transient data destination 265
CSRL transient data destination 265
CSSL transient data destination 265
CSTL transient data destination 262, 265
CSZL transient data destination 265
CSZX transient data destination 265
CTLAPPLS operand
DFHRST TYPE=RSE 295
CYCLIC option
LAYOUT operand 282

D

data management block directory list
See database directory, DL/I (DDIR)
data sets, journal 279
database control (DBCTL) 295
database description block (DBD) 267
database directory, DL/I (DDIR) 248
database description block (DBD) 267
DFHDLDBD macros 267
DFHDLDBD TYPE=ENTRY 267
DFHDLDBD TYPE=INITIAL 267
DATAID operand
DFHTST TYPE=LOCAL 333
DFHTST TYPE=RECOVERY 332
DFHTST TYPE=REMOTE 333
DFHTST TYPE=SECURITY 334
DATALOCATION attribute
PROGRAM 170
DATASTREAM attribute
CONNECTION 136
DBCTL (database control) 295
DBD (database description block) 267
DBD operand
DFHDLDBD TYPE=ENTRY 267
DCKYBD option
FEATURE operand 315
DCT (destination control table) 246, 257—265
CICS-required entries 264
coding examples 263
data set control information 257
DFHDCT macros 257

DCT (destination control table) *(continued)*
DFHDCT TYPE=EXTRA 258
DFHDCT TYPE=INDIRECT 259
DFHDCT TYPE=INITIAL 257
DFHDCT TYPE=INTRA 260
DFHDCT TYPE=REMOTE 262
DFHDCT TYPE=SDSCI 257
extrapartition destinations 258
indirect destinations 259
intrapartition destinations 260
remote destinations 262
DDIR (DL/I database directory) 248
database description block (DBD) 267
DFHDLDBD macros 267
DFHDLDBD TYPE=ENTRY 267
DFHDLDBD TYPE=INITIAL 267
DDNAME operand
DFHTCT TYPE=SDSCI 307, 310
sequential devices 307
TCAM devices 310
default values
dependent on other values 26
DEFERRED option
OPEN operand 259, 282
DEFINE command
CEDA 52
CEDA tutorial 29
DFHCSDUP 89
examples 89
PROFILE attribute keywords 163
DEFSCREEN attribute
TYPETERM 229
DEFSCRN operand
DFHTCT TYPE=TERMINAL 313
TCAM devices 313
DELETE command
CEDA 53
DFHCSDUP 91
examples 91
DELETE option
SERVREQ operand 274
deleting resource definitions from system tables
at autoinstall log off 112
deletion of existing entry at installation 21, 59
deletion of TCT entry at log off (autoinstall) 112
DELIVER option
PERFORM operand 287
dependent default values 26
DESC operand
DFHCLT TYPE=WTO 255
DESCRIPTION attribute
CONNECTION 136
PROFILE 164
PROGRAM 170
SESSIONS 177
TRANSACTION definition 209

DESCRIPTION attribute (*continued*)
 TYPETERM 229

DESTFAC operand
 DFHDCT TYPE=INTRA 260

DESTID operand
 DFHDCT TYPE=EXTRA 259
 DFHDCT TYPE=INDIRECT 259
 DFHDCT TYPE=INTRA 260
 DFHDCT TYPE=REMOTE 262

destination control table (DCT) 246, 257—265
 CICS-required entries 264
 coding examples 263
 data set control information 257
 DFHDCT macros 257
 DFHDCT TYPE=EXTRA 258
 DFHDCT TYPE=INDIRECT 259
 DFHDCT TYPE=INITIAL 257
 DFHDCT TYPE=INTRA 260
 DFHDCT TYPE=REMOTE 262
 DFHDCT TYPE=SDSCI 257
 extrapartition destinations 258
 indirect destinations 259
 intrapartition destinations 260
 remote destinations 262

destinations, transient data 257

DESTRCV operand
 DFHDCT TYPE=INTRA 261

DEVICE attribute
 TYPETERM 229

DEVICE operand
 DFHTCT TYPE=SCSCI 307
 DFHTCT TYPE=SDSCI 310
 sequential devices 307
 TCAM devices 310

DFHCLT operands
 TYPE=LISTEND 255
 TYPE=LISTSTART 254
 TYPE=WTO 254

DFHCMAC, online messages and codes 350

DFHCNV macro 248

DFHCOMP1, compatibility group 351

DFHCOMP2, compatibility group 351

DFHCOMP3, compatibility group 351

DFHCOMP4, compatibility group 352

DFHCSD CICS system definition file
 See CSD (CICS system definition file)

DFHCSDUP system definition utility program 73
 command processing considerations 79
 invocation from a user program 77
 invoking as a batch program 75
 processing system definition file 73

DFHDCT operands
 TYPE=EXTRA 258
 TYPE=INDIRECT 259
 TYPE=INITIAL 257
 TYPE=INTRA 260

DFHDCT operands (*continued*)
 TYPE=REMOTE 262
 TYPE=SDSCI 257

DFHDLDBD operands
 TYPE=ENTRY 267
 TYPE=INITIAL 267

DFHDLPSB operands
 TYPE=ENTRY 268
 TYPE=INITIAL 268

DFHFCT macro
 SERVREQ operand 25

DFHFCT operands
 TYPE=DATASET 269
 TYPE=FILE 269
 TYPE=GROUP 275
 TYPE=INITIAL 269
 TYPE=REMOTE 276

DFHJCT operands
 TYPE=ENTRY 280
 TYPE=INITIAL 280

DFHMCT operands
 TYPE=EMP 285
 TYPE=INITIAL 285
 TYPE=RECORD 288

DFHPLT operands
 TYPE=ENTRY 292
 TYPE=INITIAL 291

DFHRDT, RDO-eligible TCT entries 300

DFHRST operands
 TYPE=INITIAL 295
 TYPE=RSE 295
 TYPE=SUBSYS 295

DFHSIT macro
 APPLID operand 25, 61, 67, 132
 GRPLIST operand 20
 OPIDENT operand 61, 67
 START operand 20, 23
 TCTUALOC operand 182, 240

DFHSNT macro
 OPIDENT operand 25, 61, 67

DFHSRT operands
 TYPE=INITIAL 297
 TYPE=SYSTEM 297
 TYPE=USER 297

DFHTCT operands
 remote terminals 321, 322
 sequential devices 307, 308
 TYPE=GROUP 301
 TYPE=INITIAL 300
 TYPE=LINE 308
 TYPE=REGION 321
 TYPE=REMOTE 322
 TYPE=SDSCI 307
 TYPE=TERMINAL 308, 322

DFHTLT operands
 TYPE=ENTRY 329

DFHTLT operands (*continued*)
 TYPE=INITIAL 329

DFHTST operands 332
 TYPE=ENTRY 332
 TYPE=INITIAL 331
 TYPE=RECOVERY 332, 333, 334

DFHXLTL operands
 TYPE=INITIAL 337

DFHZATDX autoinstall control program 116

directories, DDIR and PDIR 267—268

DISCREQ attribute
 SESSIONS definition 177
 TYPETERM 230

DISK1 option
 JTYPE operand 282

DISK2 option
 JTYPE operand 282

DISP operand
 DFHFCT TYPE=FILE 270

DISPLAY command
 CEDA tutorial 33

DISPLAY command (CEDA) 54

DL/I, shared, batch links, TYPE=IRCBCH 399

DLI (DL1) operand
 DFHDLDBD TYPE=INITIAL 267
 DFHDLPSB TYPE=INITIAL 268

DMB directory
 See database directory, DL/I (DDIR)

DSCNAME operand
 DFHDCT TYPE=EXTRA 259
 DFHDCT TYPE=SDSCI 258
 DFHTCT TYPE=LINE 311
 DFHTCT TYPE=SDSCI 307, 310
 sequential devices 307
 TCAM devices 310, 311

DSN operand
 DFHTCT TYPE=LDC 304

DSNAME operand
 DFHFCT TYPE=FILE 270

DTIMOUT attribute
 TRANSACTION definition 209

dual-purpose resource definition 21

DUALCASEKYBD attribute
 TYPETERM 230

DUMP attribute
 TRANSACTION definition 209

DVC operand
 DFHTCT TYPE=LDC 304

DVSUPRT attribute
 PROFILE 164

DYNAMIC attribute
 TRANSACTION definition 209

E

emergency restart
 recreation of tables 20
 temporary recovery of autoinstalled entries 113

ending CEDA 29
 exiting from CEDA 29
 quitting from CEDA 29

ERRATT operand
 DFHTCT TYPE=INITIAL 300
 DFHTCT TYPE=TERMINAL 314
 TCAM devices 314

ERRCOLOR attribute
 TYPETERM 230

ERRHIGHLIGHT attribute
 TYPETERM 230

ERRINTENSIFY attribute
 TYPETERM 230

ERRLASTLINE attribute
 TYPETERM 230

ERROPT operand
 DFHDCT TYPE=SDSCI 258

error messages
 produced by CEDA 31

EXCLUDE operand
 DFHMCT TYPE=RECORD 288

EXCNT option
 PERFORM operand 287

EXEC CICS START command
 autoinstalled terminals 107

EXECKEY attribute
 PROGRAM definition 171

EXECUTIONSET attribute
 PROGRAM definition 171

EXPAND command (CEDA) 56

EXTA option
 DFHJCT JOUROPT operand 281

EXTDS option
 FEATURE operand 314

extended local LDC list and system LDC table 303
 DFHTCT TYPE=LDC 303

EXTENDEDDES attribute
 TYPETERM 230

EXTRACT command (DFHCSDUP)
 examples 93
 OBJECTS option 93
 USERPROGRAM option 93

EXTSEC attribute
 TRANSACTION 341

F

FCT (file control table) 246, 269
 coding example 277
 configurator 275
 DFHFCT macros 269

FCT (file control table) *(continued)*
 DFHFCT TYPE=DATASET 269
 DFHFCT TYPE=FILE 269
 DFHFCT TYPE=GROUP 275
 DFHFCT TYPE=INITIAL 269
 DFHFCT TYPE=REMOTE 276

FEATURE operand
 DFHTCT TYPE=TERMINAL 314
 TCAM devices 314

FF operand
 DFHTCT TYPE=TERMINAL 315
 TCAM devices 315

FIELD operand
 DFHMCT TYPE=EMP 286

file control table (FCT) 246, 269
 coding example 277
 configurator 275
 DFHFCT macros 269
 DFHFCT TYPE=DATASET 269
 DFHFCT TYPE=FILE 269
 DFHFCT TYPE=GROUP 275
 DFHFCT TYPE=INITIAL 269
 DFHFCT TYPE=REMOTE 276

FILE definition
 installed singly 20, 59

file identifier
 entries in JCT 280
 journal identifier (JFILEID) 281
 type (JTYPE) 282

FILE operand
 DFHFCT TYPE=FILE 271
 DFHFCT TYPE=REMOTE 276

FILE option
 DESTFAC operand 260

FILE, DFHFCT TYPE= macro 269

FILSTAT operand 271
 DFHFCT TYPE=FILE 271

FINAL option
 LOCAL operand 305

FINAL, TYPE= macro 249

FIXBLK option
 RECFORM operand 258

FIXBLKA option
 RECFORM operand 258

FIXBLKM option
 RECFORM operand 258

FIXED option
 RECFORM operand 273

FIXUNB option
 RECFORM operand 258

FIXUNBA option
 RECFORM operand 258

FIXUNBM option
 RECFORM operand 258

FMHPARM attribute
 TYPETERM 230

forced values 26
 format of macros 250
 FORMAT operand
 DFHJCT TYPE=ENTRY 281
 FORMFEED attribute
 TYPETERM 230

G

generic names
 accepted by which commands 40

Generic naming in CEDA 40

GROUP operand
 DFHTCT TYPE=GROUP 301

groups of resource definitions
 cross-checking (CHECK) 24
 installing
 while CICS is running 21, 59
 keeping resource definitions together 18
 locked, valid commands 25, 61, 67
 of application resource definitions 18
 of AUTINSTMODEL TERMINAL definitions 18
 of CONNECTION and SESSIONS definitions 18
 of PARTITIONSET definitions 18
 of PROFILE definitions 18
 of TERMINAL definitions 18
 of TYPETERM definitions 18
 recommended maximum size 18
 recommended minimum size 18

GRPLIST
 DFHSIT operand 20
 not used at restart 20
 SIT override 23

H

HF operand
 TCAM devices 315
 TYPE=TERMINAL 315

HIGHLIGHT attribute
 TYPETERM 231

HIGHLIGHT option
 FEATURE operand 314

HORIZFORM attribute
 TYPETERM 231

I

ID operand
 DFHMCT TYPE=EMP 285

IGNORE option
 ERROPT operand 258

IMS
 defining CICS-IMS links and sessions 132

INAREAL operand
 DFHTCT TYPE=LINE 308, 311

INAREAL operand (*continued*)
 sequential devices 308
 TCAM devices 311

INBFMH attribute
 PROFILE 164

INCLUDE operand
 DFHMCT TYPE=RECORD 289

INDDST operand
 DFHDCT TYPE=INDIRECT 260

INDIRECT connections
 resource definitions 132

INDOUBT attribute
 TRANSACTION definition 210

INDSYS attribute
 CONNECTION 136
 for INDIRECT connections 132

INITIAL option
 LOCAL operand 305
 OPEN operand 259, 282

INITIAL, TYPE= macro 248

INITIALIZE command (DFHCSDUP) 94

initializing CICS
 cold start 20
 emergency restart 20
 GRPLIST operand of DFHSIT 22
 setting up a list 22
 warm start 20
 with one of several lists 23

INPUT option
 TRMSTAT operand 309, 317
 TYPEFLE operand 258

INSERVICE attribute
 CONNECTION 136
 SESSIONS definition 178, 341

INSTALL command
 to install a group of resource definitions 21, 59

INSTALL command (CEDA) 58

installation of resource definitions
 and deletion of existing entry 21, 59
 cold start 20
 emergency restart 20
 quiescing resources first 21, 59
 warm start 20

INTENSIFY option
 ERRATT operand 301

IOAREALEN attribute
 SESSIONS definition 178
 TYPETERM 231

IRCBCH, shared DL/I batch links 399

ISADSCN operand
 DFHTCT TYPE=LINE 308
 sequential devices 308

ISOLATE attribute
 TRANSACTION definition 210

J

JCT (journal control table) 246, 279—283
 coding example 283
 defining to CICS 279
 DFHJCT macros 279
 DFHJCT TYPE=ENTRY 280
 DFHJCT TYPE=INITIAL 280
 journal buffers 279

JFILEID operand
 DFHJCT TYPE=ENTRY 281

JID operand
 DFHFCT TYPE=FILE 271

JOBNAME operand
 DFHRST TYPE=SUBSYS 295

JOURNAL attribute
 PROFILE 165

journal buffer
 overview 279

journal control table (JCT) 246, 279—283
 coding example 283
 defining to CICS 279
 DFHJCT macros 279
 DFHJCT TYPE=ENTRY 280
 DFHJCT TYPE=INITIAL 280
 journal buffers 279

journal data sets 279

journal devices 279

journal management statistics 280

journal records 279

JOUROPT operand
 DFHJCT TYPE=ENTRY 281

JREQ operand
 DFHFCT TYPE=FILE 272

JTYPE operand
 DFHJCT TYPE=ENTRY 282

K

KATAKANA attribute
 TYPETERM 231

KATAKANA option
 FEATURE operand 314

KEY option
 SERVREQ operand 274

KEYLEN operand
 DFHFCT TYPE=FILE 272
 DFHFCT TYPE=REMOTE 276

L

LABEL operand
 DFHJCT TYPE=ENTRY 282

LANGUAGE attribute
 PROGRAM 171

LASTLINE option
 ERRATT operand 301
 LASTTRM operand
 DFHTCT TYPE=TERMINAL 315
 LAYOUT operand
 DFHJCT TYPE=ENTRY 282
 LDC (logical device code) 302—306
 LDC list 303, 305
 extended local, and system LDC table, DFHTCT
 TYPE=LDC 303
 local, DFHTCT TYPE=LDCLIST 305
 LDC operand
 DFHTCT TYPE=LDC 304
 DFHTCT TYPE=LDCLIST 305
 DFHTLT TYPE=INITIAL 329
 TCAM devices 315
 TYPE=TERMINAL 315
 LDCLIST attribute
 TYPETERM 231
 LDCs for 3770 batch logical unit
 TCT example 306
 LENGTH operand
 DFHDCT TYPE=EXTRA 259
 DFHDCT TYPE=REMOTE 262
 LG option
 DESTRCV operand 261
 LIGHTPEN attribute
 TYPETERM 232
 LINE option
 FEATURE operand 315
 LINEAR option
 LAYOUT operand 282
 LINSTAT='OUT OF SERVICE' operand
 DFHTCT TYPE=LINE 308
 sequential devices 308
 LIST command (DFHCSDUP)
 OBJECTS option 95
 list of group names
 cross-checking (CHECK) 24
 different GRPLIST for different MRO systems 24
 for initialization, setting up 22
 using one of several for initialization 23
 loading, above the line 245
 local LDC list 305
 DFHTCT TYPE=LDCLIST 305
 LOCAL operand
 DFHTCT TYPE=LDC 305
 local temporary storage DATAIDs 332
 DFHTST TYPE=LOCAL 332
 LOCALQ attribute
 TRANSACTION definition 211
 LOCK command (CEDA) 61, 67
 LOCK GROUP command
 controlling access to groups 25, 61, 67
 LOCK LIST command
 controlling access to lists 25, 61, 67
 locked group, valid commands 25, 61, 67
 locks, internal
 put on by CEMT INQUIRE TERMINAL 112
 LOG operand
 DFHFCT TYPE=FILE 272
 log-on process, with autoinstall 109
 log, system 281
 logical device code (LDC) 302—306
 LOGMODE attribute
 TYPETERM 232
 LOGNODECOM attribute
 TYPETERM 232
 logoff (autoinstall), deletion of TCT entry 112
 LOGONMSG attribute
 TYPETERM 233
 LOGREC attribute
 PROFILE 165
 LPLEN operand
 DFHTCT TYPE=TERMINAL 308
 sequential devices 308
 LRECL operand
 DFHFCT TYPE=FILE 273
 DFHFCT TYPE=REMOTE 276
 LRU option
 JOUROPT operand 281
 LSRPOOL definition
 installed singly 20, 59
 LUTYPE 0 terminals
 eligible for autoinstall 105
 LUTYPE 1 terminals
 eligible for autoinstall 105
 LUTYPE 2 terminals
 eligible for autoinstall 105
 LUTYPE 3 terminals
 eligible for autoinstall 105
 LUTYPE 4 terminals
 eligible for autoinstall 105
 TCT example 306
 LUTYPE4 option
 LDC operand 304
 LUTYPE6.1
 CICS-CICS ISC links and sessions
 ineligible for autoinstall 105
 resource definitions 131
 CICS-IMS links and sessions
 ineligible for autoinstall 105
 resource definitions 132
 MRO sessions
 ineligible for autoinstall 105
 resource definitions 130

M

MACRF operand
 DFHTCT TYPE=SDSCI 307, 310
 sequential devices 307

MACRF operand (*continued*)
 TCAM devices 310

MAPSET definition
 installed singly 20, 59

MAXACTIVE attribute
 TRANCLASS definition 205

MAXIMUM attribute
 for APPC parallel sessions 130
 SESSIONS
 for APPC single session terminals 131
 SESSIONS definition 178

maximum recommended number of resource definitions in
 group 18

MAXQTIME attribute
 CONNECTION definition 136

MCT (monitoring control table) 247, 285—290
 coding example 290
 control data recording 288
 DFHMCT macros 285
 DFHMCT TYPE=EMP 285
 DFHMCT TYPE=INITIAL 285
 DFHMCT TYPE=RECORD 288
 user event monitoring points 285

MESSAGE attribute
 TYPETERM definition 236

messages
 produced by CEDA 31

messages and codes online (DFHCMAC) 350

MF operand
 DFHCLT TYPE=WTO 255

MIGRATE command
 DFHCSDUP 97
 TABLE option 97

MIGRATE command (DFHCSDUP)
 examples 98
 FCT migration 97
 TCT migration 97, 98
 TYPESGROUP option 97
 TYPETERM creation 98

MIGRATE operand
 DFHFCT TYPE=INITIAL 269
 DFHTCT TYPE=INITIAL 301

migrating definitions 275
 DFHFCT TYPE=GROUP 275

migrating TCT entries to RDO 301

minimum recommended number of resource definitions in
 group 18

MLTCNT option
 PERFORM operand 287

MODENAME attribute
 PROFILE 165
 SESSIONS
 for APPC parallel sessions 130
 for APPC single session terminals 131
 SESSIONS definition 178

monitoring control table (MCT) 247, 285—290
 coding example 290
 control data recording 288
 DFHMCT macros 285
 DFHMCT TYPE=EMP 285
 DFHMCT TYPE=INITIAL 285
 DFHMCT TYPE=RECORD 288
 user event monitoring points 285

MOVE command (CEDA) 63

MSGINTEG attribute
 PROFILE 165

MSGJRNL attribute
 PROFILE 165

MSRCNTRL option, FEATURE operand 314

MSRCONTROL attribute
 TYPETERM 233

MXSSASZ operand
 DFHDLPSB TYPE=ENTRY 268

N

NACNT option
 PERFORM operand 287

naming control tables 249

NEPCLASS attribute
 PROFILE 165
 SESSIONS definition 178
 TYPETERM 233

NETNAME attribute 113
 CONNECTION 137
 for INDIRECT connections 132
 CONNECTION definition 137
 no entry in TCT for 109
 relationship with TERMINAL name 116
 role in logging on with autoinstall 109

NETNAME operand
 DFHTCT TYPE=TERMINAL 316
 TCAM devices 316

NETNAMEQ attribute
 SESSIONS
 for LUTYPE6.1 CICS-IMS sessions 132
 SESSIONS definition 178

NOEXCTL option
 SERVREQ operand 274

NONE attribute
 TYPETERM definition 180, 237

NONVTAM option
 specifying ACCMETH=NONVTAM 300

notation, syntax xii

NPDELAY operand
 DFHTCT TYPE=LINE 311
 TCAM devices 311

NTO (network terminal option) 105

O

OBFORMAT attribute
 TYPETERM 233

OBJECTS option
 LIST command (DFHCSDUP) 95

OBOPERID attribute
 TYPETERM 234

ONEWTE attribute
 PROFILE 165

online messages and codes (DFHCMAC) 350

OPEN operand
 DFHDCT TYPE=EXTRA 259
 DFHJCT TYPE=ENTRY 282

OPERID attribute
 SESSIONS definition 341

OPERPRIORITY attribute
 SESSIONS definition 341

OPERRSL attribute
 SESSIONS definition 341

OPERSECURITY attribute
 SESSIONS definition 342

OPIIDENT operand
 DFHSNT
 for controlling access to groups and lists 25, 61, 67

OPNDST, issued by CICS at logon 109

OPTCD operand
 DFHTCT TYPE=SDSCI 310
 TCAM devices 310

ORCNT option
 PERFORM operand 287

OSADSCN operand
 DFHTCT TYPE=LINE 308
 sequential devices 308

OUT OF SERVICE option
 LINSTAT operand 308
 TRMSTAT operand 309, 317

OUTLINE attribute
 TYPETERM 234

OUTLINE option
 FEATURE operand 314

OUTPUT option
 TYPEFLE operand 258

output process queue, TCAM 311

OUTQ operand
 DFHTCT TYPE=LINE 311
 TCAM devices 311

P

PAGESIZE attribute
 TYPETERM 234

parallel APPC sessions 130

PARTITIONS attribute
 TYPETERM 234

PARTITIONSET attribute
 TRANSACTION definition 211

PARTITIONSET definition
 installed singly 20, 59
 keeping in a separate group 18

PARTNER definition
 installed singly 20, 59

PARTNS option
 FEATURE operand 314

PAUSE option
 JOUROPT operand 281

PCI (process control information) field 311

PCLOCK option
 PERFORM operand 287

PCPUCLK option
 PERFORM operand 287

PDIR (program specification block directory) 248
 DFHDLPSB macros 268
 DFHDLPSB TYPE=ENTRY 268
 DFHDLPSB TYPE=INITIAL 268

PERFORM operand
 DFHMCT TYPE=EMP 286

PERFORM option
 CLASS operand 288

performance
 See MCT (monitoring control table)

PF keys on CEDA panels 29

PGESIZE operand
 DFHTCT TYPE=LDC 305
 DFHTCT TYPE=TERMINAL 308, 316
 sequential devices 308
 TCAM devices 316

PGESTAT operand
 DFHTCT TYPE=LDC 305
 DFHTCT TYPE=TERMINAL 316
 TCAM devices 316

PH option
 DESTRCV operand 261

pipeline terminals 224
 ineligible for autoinstall 105

PLT (program list table) 247, 291—293
 coding example 293
 control section 291
 DFHPLT TYPE=ENTRY 292
 DFHPLT TYPE=INITIAL 291

POOL operand
 DFHTCT TYPE=LINE 311
 TCAM devices 311

POOL option
 FEATURE operand 315

preparing control tables 245

preset security 120, 309, 341

preset terminal security
 consoles 203

PRIMEDSIZE attribute
 TRANSACTION 342

PRINTADAPTER attribute
 TYPETERM 234
 PRINTERCOMP attribute
 PROFILE 165
 printers 305
 eligible for autoinstall 105
 PGESTAT operand 305
 printing, continuous (autopaging) 305
 PRIORITY attribute
 TRANSACTION definition 212
 process control information (PCI) field 311
 PROFILE attribute
 PROFILE 166
 PROFILE definition 163
 CHAINCONTROL attribute 164
 DESCRIPTION attribute 164
 DVSUPRT attribute 164
 INBFMH attribute 164
 installed singly 20, 59
 JOURNAL attribute 165
 keeping in a separate group 18
 LOGREC attribute 165
 MODENAME attribute 165
 MSGINTEG attribute 165
 MSGJRNL attribute 165
 NEPCLASS attribute 165
 ONEWTE attribute 165
 PRINTERCOMP attribute 165
 PROFILE attribute 166
 PROTECT attribute 166
 RAQ attribute 166
 RTIMOUT attribute 166
 SCRNSIZE attribute 166
 UCTRAN attribute 167
 PROGRAM attribute
 PROGRAM 172
 TRANSACTION definition 212
 PROGRAM definition
 CEDF attribute 170
 DATALOCATION attribute 170
 DESCRIPTION attribute 170
 EXECKEY attribute 171
 EXECUTIONSET attribute 171
 installed singly 20, 59
 LANGUAGE attribute 171
 PROGRAM attribute 172
 RELOAD attribute 172
 REMOTENAME attribute 172
 REMOTESYSTEM attribute 172
 RESIDENT attribute 172
 STATUS attribute 172
 TRANSID attribute 173
 USAGE attribute 173
 USELPACOPY attribute 173
 program list table (PLT) 247, 291—293
 coding example 293

program list table (PLT) (*continued*)
 control section 291
 DFHPLT TYPE=ENTRY 292
 DFHPLT TYPE=INITIAL 291
 PROGRAM operand
 DFHPLT TYPE=ENTRY 292
 program specification block (PSB) 268
 PROGSYMBOLS attribute
 TYPETERM 235
 PROTECT attribute
 PROFILE 166
 PROTOCOL attribute
 CONNECTION
 for APPC links 130
 for APPC single session terminals 131
 for LUTYPE6.1 CICS-IMS links 132
 for LUTYPE6.1 links 131
 for MRO links 130
 CONNECTION definition 137
 SESSIONS
 for APPC parallel sessions 130
 for APPC single session terminals 131
 for LUTYPE6.1 CICS-IMS sessions 132
 for LUTYPE6.1 sessions 131
 for MRO sessions 130
 SESSIONS definition 179
 PS option
 FEATURE operand 314
 PSB (program specification block) 268
 PSB directory (PDIR) 248
 DFHDLPSB macros 268
 DFHDLPSB TYPE=ENTRY 268
 DFHDLPSB TYPE=INITIAL 268
 PSB operand
 DFHDLPSB TYPE=ENTRY 268
 PSRECOVERY attribute
 CONNECTION definition 137
 PTRADAPT option
 FEATURE operand 314
 PURGETHRESH attribute
 TRANCLASS definition 205

Q

QUERY attribute
 TYPETERM 235
 QUERY function
 with autoinstall 109
 QUERYALL option
 FEATURE operand 314
 QUERYCOLD option
 FEATURE operand 315
 QUEUEID operand
 DFHTCT TYPE=LINE 311
 TCAM devices 311

QUEUELIMIT attribute
 CONNECTION definition 138
queues, transient data
 See transient data destinations

R

R option
 MACRF operand 310
RAQ attribute
 PROFILE 166
RCT (resource control table) 248
RDBACK option
 TYPEFLE operand 258
RDO command
 ALTER PROFILE 163
 DEFINE PROFILE 163
RDO transaction
 entering values for attributes
 dependent defaults 26
 forced values 26
RDO-eligible TCT entries, DFHRDT 300
READ option
 SERVREQ operand 274
read-only CSD, for production system 25
RECEIVE option
 TRMSTAT operand 309, 317
RECEIVECOUNT attribute
 SESSIONS
 for LUTYPE6.1 CICS-IMS sessions 132
 for LUTYPE6.1 sessions 131
 for MRO sessions 130
RECEIVECOUNT attribute, SESSIONS definition 179
RECEIVEPFX attribute
 SESSIONS
 for LUTYPE6.1 sessions 131
 for MRO sessions 130
RECEIVEPFX attribute, SESSIONS definition 179
RECEIVESIZE attribute
 SESSIONS definition 179
 TYPETERM definition 235
RECFM operand
 DFHTCT TYPE=SDSCI 307, 310
 sequential devices 307
 TCAM devices 310
RECFORM operand
 DFHDCT TYPE=SDSCI 258
 DFHFCT TYPE=FILE 273
RECORDFORMAT attribute
 CONNECTION 138
RECOVER operand
 DFHSRT TYPE=SYSTEM|USER 297
recoverable service table (RST) 247, 295—296
 coding example 296
 DFHRST macro types 295
 DFHRST TYPE=INITIAL 295

recoverable service table (RST) (*continued*)
 DFHRST TYPE=RSE 295
 DFHRST TYPE=SUBSYS 295
recoverable temporary storage 332
 DFHTST TYPE=RECOVERY 332
recovery of destinations
 DESTRCV operand 261
RECOVNOTIFY attribute
 TYPETERM definition 236
RECOVOPTION attribute
 SESSIONS definition 180
 TYPETERM definition 180, 236
RECSIZE operand
 DFHDCT TYPE=SDSCI 258
RELEASESESS attribute
 TYPETERM definition 180, 237
RELOAD attribute
 PROGRAM 172
RELREQ attribute
 SESSIONS definition 180
 TYPETERM 237
RELTYPE operand
 DFHFCT TYPE=FILE 274
remote destinations 262
remote files 276
remote temporary storage DATAIDs 333
 DFHTST TYPE=REMOTE 333
remote terminals for transaction routing
 DFHTCT macros 321
 DFHTCT TYPE=REGION operands 321
 DFHTCT TYPE=REMOTE operands 322
 DFHTCT TYPE=TERMINAL operands 322
REMOTENAME attribute
 CONNECTION 138
 PROGRAM 172
 TRANSACTION definition 212
REMOTESYSNET attribute
 CONNECTION 138
REMOTESYSTEM attribute
 CONNECTION 138
 TRANSACTION definition 212
REMOVE command
 CEDA 65
 DFHCSDUP 99
 examples 99
removing resource definitions from system tables
 at a cold start 20
 at autoinstall log off 112
RENAME command (CEDA) 66
required entries in destination control table 264
RESIDENT attribute
 PROGRAM 172
resource control table (RCT) 248
resource definitions
 at system initialization
 cold start 20
 emergency restart 20

resource definitions (*continued*)

- at system initialization (*continued*)
 - warm start 20
- checking 24
- dual-purpose 21
- for APPC links and parallel sessions 130
- for APPC single session terminals
 - advantages of TERMINAL-TYPETERM method 131
 - CONNECTION and SESSIONS method 131
- for INDIRECT connections 132
- for LUTYPE6.1 CICS-CICS ISC links and sessions 130
- for LUTYPE6.1 CICS-IMS links and sessions 132
- for LUTYPE6.1 links and sessions 131
- for MRO links and sessions 130
- installing in CICS system
 - using the INSTALL command 21, 59
- removing from system tables
 - at a cold start 20
 - at autoinstall log off 112
- resource security checking (RESSEC) 24
- RESSEC (resource security checking) 24
- RESSEC attribute
 - TRANSACTION 24
 - TRANSACTION definition 212
- restart
 - emergency
 - recreation of tables 20
 - temporary recovery of autoinstalled entries 113
 - warm
 - nonrecovery of autoinstalled entries 113
 - recreation of tables 20
- RESTART attribute
 - TRANSACTION definition 212
- RETRY option
 - JOUROPT operand 282
- REUSE option
 - SERVREQ operand 274
- REWIND operand
 - DFHDCT TYPE=SDSCI 258
- RKP operand
 - DFHFCT TYPE=FILE 274
- RMTNAME operand
 - DFHDCT TYPE=EXTRA 259
 - DFHDCT TYPE=INTRA 261
 - DFHDCT TYPE=REMOTE 262
 - DFHDLPSB TYPE=ENTRY 268
 - DFHFCT TYPE=REMOTE 277
 - DFHTCT TYPE=REMOTE 322
 - DFHTCT TYPE=TERMINAL 322
 - DFHTST TYPE=REMOTE 334
 - remote terminals 322
- ROUTCDE operand
 - DFHCLT TYPE=WTO 255
- ROUTEDMSGS attribute
 - TYPETERM 237
- RST (recoverable service table) 247, 295—296
 - coding example 296
 - DFHRST macro types 295
 - DFHRST TYPE=INITIAL 295
 - DFHRST TYPE=RSE 295
 - DFHRST TYPE=SUBSYS 295
- RTIMOUT attribute
 - PROFILE 166

S

- SCLOCK option
 - PERFORM operand 287
- SCPUCLK option
 - PERFORM operand 287
- SCRNSIZE attribute
 - PROFILE 166
- SCS printers
 - eligible for autoinstall 105
- SDLC 3767 devices
 - eligible for autoinstall 105
- security
 - different CSD files for non-MRO CICS systems 25
 - LOCK and UNLOCK commands
 - controlling access to groups 25, 61, 67
 - controlling access to lists 25, 61, 67
 - preset 120, 309, 341
 - read-only CSD file for production system 25
 - resource security checking (RESSEC) 24
 - using a special list for restricted resources 23
- security checking
 - DFHTST TYPE=SECURITY 334
 - temporary storage 334
- SECURITYNAME attribute
 - CONNECTION 139
- segment search argument, PSB 268
- SELCTPEN option
 - FEATURE operand 315
- SENDCOUNT attribute
 - SESSIONS
 - for LUTYPE6.1 CICS-IMS sessions 132
 - for LUTYPE6.1 sessions 131
 - for MRO sessions 130
- SENDCOUNT attribute, SESSIONS definition 181
- SENDPFX attribute
 - SESSIONS
 - for LUTYPE6.1 sessions 131
 - for MRO sessions 130
- SENDPFX attribute, SESSIONS definition 181
- SENDSIZE attribute
 - SESSIONS definition 181
 - TYPETERM 237
- sequential devices 393
- SERVICE command (DFHCSDUP)
 - FROMCSD operand 100
 - LEVEL operand 100

SERVREQ operand
 DFHFCT TYPE=FILE 274

SESSIONS attribute
 SESSIONS definition 181

SESSIONS definition
 AUTOCONNECT attribute 177
 BUILDCHAIN attribute 177
 CONNECTION attribute 177
 description 175
 DESCRIPTION attribute 177
 DISCREQ attribute 177
 INSERVICE attribute 178, 341
 installation 175
 installed by group 21, 59
 IOAREALEN attribute 178
 keeping in a group with related CONNECTION 18
 MAXIMUM attribute 178
 MODENAME attribute 178
 NEPCLASS attribute 178
 NETNAMEQ attribute 178
 OPERID attribute 341
 OPERPRIORITY attribute 341
 OPERRSL attribute 341
 OPERSECURITY attribute 342
 PROTOCOL attribute 179
 RECEIVECOUNT attribute 179
 RECEIVEPFX attribute 179
 RECEIVESIZE attribute 179
 RECOVOPTION attribute 180
 RELREQ attribute 180
 SENDCOUNT attribute 181
 SENDPFX attribute 181
 SENDSIZE attribute 181
 SESSIONS attribute 181
 SESSNAME attribute 181
 SESSPRIORITY attribute 181
 TRANSACTION attribute 182, 344

SESSIONTYPE attribute
 TYPETERM 238

SESSNAME attribute
 SESSIONS
 for LUTYPE6.1 CICS-IMS sessions 132
 SESSIONS definition 181

SESSPRIORITY attribute
 SESSIONS definition 181

SESTYPE operand
 DFHTCT TYPE=TERMINAL 316, 317
 TCAM devices 316, 317

shared DL/I batch links, TYPE=IRCBCH 399

SHIPPABLE attribute
 TYPETERM 238

sign-on table (SNT)
 OPIDENT operand 25

SIGNOFF attribute
 TYPETERM 238

SIGNOFF operand
 DFHTCT TYPE=LINE 311
 DFHTCT TYPE=TERMINAL 316
 TCAM devices 311, 316

single session terminals
 APPC
 advantages of TERMINAL-TYPETERM method 131
 eligible for autoinstall 105
 resource definitions, CONNECTION and SESSIONS
 method 131

SINGLESESS attribute
 CONNECTION 139
 for APPC single session terminals 131

SIT (system initialization table) 248
 APPLID (CICS system name) 25, 61, 67
 size of group, maximum recommended 18
 size of group, minimum recommended 18

SKIP option
 ERROPT operand 258

SMF (System Management Facility) 281, 282
 journals 280

SMF option
 FORMAT operand 281
 JTYPE operand 282

SNA TCAM devices 311

SNT (sign-on table)
 OPIDENT operand 25

SOSI attribute
 TYPETERM 239

SOSI option
 FEATURE operand 315

SPURGE attribute
 TRANSACTION definition 213

SRCHM operand
 DFHFCT TYPE=FILE 275

SRT (system recovery table) 297—298
 coding example 298
 DFHSRT TYPE=INITIAL 297

STANDARD option
 LABEL operand 282

START operand, DFHSIT 20, 23

start, cold
 nonrecovery of autoinstalled TCT entries 113

start, emergency
 temporary recovery of autoinstalled TCT entries 113

start, warm
 nonrecovery of autoinstalled TCT entries 113

STARTIO option
 SYSWAIT operand 283

statistics, journal management 280

STATUS attribute
 PROGRAM 172
 TRANSACTION definition 213

storage queues, DFHTST TYPE=LOCAL
 local temporary 332

- storage queues, DFHTST TYPE=REMOTE
 - remote temporary 333
- STORAGECLEAR attribute
 - TRANSACTION definition 213
- SUBCNT option
 - PERFORM operand 287
- SUBSYSID operand
 - DFHRST TYPE=SUBSYS 295
- SUFFIX operand
 - DFHDCT TYPE=INITIAL 257
 - DFHTLT TYPE=INITIAL 329
- suffixing control tables 249
- SYNAD operand
 - DFHTCT TYPE=SDSCI 311
 - TCAM devices 311
- syntax notation xii, 251
- SYSDEFAULT attribute
 - TYPETERM definition 180, 236
- SYSID field on CEDA panels 29
- SYSIDNT operand
 - DFHDCT TYPE=EXTRA 259
 - DFHDCT TYPE=INTRA 261
 - DFHDCT TYPE=REMOTE 262
 - DFHDLPSB TYPE=ENTRY 268
 - DFHFCT TYPE=REMOTE 277
 - DFHSIT TYPE=INITIAL 301
 - DFHTCT TYPE=INITIAL 301
 - DFHTCT TYPE=REGION 321
 - DFHTCT TYPE=REMOTE 322
 - DFHTCT TYPE=TERMINAL 322
 - DFHTST TYPE=REMOTE 333
 - lowercase in TCT 300
 - remote terminals 321, 322
- system definition file (CSD)
 - See CSD (CICS system definition file)
- system initialization parameters
 - PGAICTLG 124
- system initialization table (SIT) 248
- system LDC table and extended local LDC list 303
 - DFHTCT TYPE=LDC 303
- system log 281
- System Management Facility (SMF) 281, 282
 - journals 280
- SYSTEM option
 - DESTFAC operand 260
 - JFILEID operand 281
 - LDC operand 304
- system recovery table (SRT) 297—298
 - coding example 298
 - DFHSRT TYPE=INITIAL 297
- system tables, CICS
 - deletion of existing entry (reinstalling resource definitions) 21, 59
 - installing
 - at CICS initialization 19

- SYSWAIT operand
 - DFHJCT TYPE=ENTRY 282

T

- table entries
- TAPE1 option
 - JTYPE operand 282
- TAPE2 option
 - JTYPE operand 282
- TASKDATAKEY attribute
 - TRANSACTION definition 213
- TASKDATALOC attribute
 - TRANSACTION definition 214
- TASKREQ attribute
 - TRANSACTION definition 214
- TASKREQ operand
 - DFHXLT TYPE=ENTRY 337
- TCAM devices
 - DFHTCT macros 309
 - output process queue 311
 - POOL feature 311
 - process queue identifier 311
 - SNA 311
- TCAMFET operand
 - DFHTCT TYPE=LINE 311
 - TCAM devices 311
- TCLASS attribute
 - TRANSACTION definition 214, 343
- TCT (terminal control table) 116, 247, 299—327
 - CONNECTION
 - for APPC links 130
 - for APPC single session terminals 131
 - for INDIRECT connections 132
 - for LUTYPE6.1 CICS-IMS links 132
 - for LUTYPE6.1 links 131
 - for MRO links 130
 - DFHTCT macros, summary 299
 - DFHTCT TYPE=GROUP 301
 - introduction 299
 - list of CICS terminals 324
 - migrating entries to RDO 301
 - sequential devices 306
 - TCAM examples 319
 - TCT entry, autoinstalled 108, 109, 110, 116
 - TCT entry, XRF tracking 113
- TCT entry
 - automatic deletion, with autoinstall 115
 - deleted at logoff, with autoinstall 112
 - XRF tracking 113
- TCT migration
 - summary of implementation 389
- TCTUA (terminal control table user area)
 - location in virtual storage 182, 240
 - size 182, 240

TCTUAL operand
 DFHTCT TYPE=LINE 308, 311
 DFHTCT TYPE=TERMINAL 308, 316
 sequential devices 308
 TCAM devices 311, 316

TCTUALOC operand
 DFHSIT macro 182, 240

temporary storage
 recoverable, DFHTST TYPE=RECOVERY 332
 security checking, DFHTST TYPE=SECURITY 334

temporary storage security checking 334
 DFHTST TYPE=SECURITY 334

temporary storage table (TST) 247, 331—335
 control section 331
 DFHTST TYPE=INITIAL 331
 DFHTST TYPE=RECOVERY 332, 333, 334
 local temporary storage DATAIDs, DFHTST
 TYPE=LOCAL 332
 recoverable temporary storage 332, 333, 334
 recoverable temporary storage, DFHTST
 TYPE=RECOVERY 332
 remote temporary storage DATAIDs, DFHTST
 TYPE=REMOTE 333

TCT
 TCAM devices 319
 temporary storage security checking, DFHTST
 TYPE=SECURITY 334

TERMINID, see TERMINAL attribute

TERMINAL attribute
 TERMINAL 116

terminal control table (TCT) 116, 247, 299—327
 CONNECTION
 for APPC links 130
 for APPC single session terminals 131
 for INDIRECT connections 132
 for LUTYPE6.1 CICS-IMS links 132
 for LUTYPE6.1 links 131
 for MRO links 130
 DFHTCT macros, summary 299
 DFHTCT TYPE=GROUP 301
 introduction 299
 list of CICS terminals 324
 migrating entries to RDO 301
 sequential devices 306
 TCAM examples 319
 TCT entry, autoinstalled 108, 109, 110, 116
 TCT entry, XRF tracking 113

TERMINAL definition
 AUTINSTMODEL 18
 keeping in a separate group 18
 installation when out of service 21, 59
 installed by group 21, 59
 keeping in a separate group 18
 TERMINAL 105
 TERMINAL attribute 105, 116
 USEDFTUSER attribute 202

terminal identifier, see TERMINAL attribute

terminal list table (TLT) 247, 329—330
 coding examples 330
 control section 329
 DFHTLT TYPE=ENTRY 329
 DFHTLT TYPE=INITIAL 329
 effect of autoinstall 109

TERMINAL option
 DESTFAC operand 260

terminals
 supported by RDO 221

terminals, CICS, list of 324

TERMMODEL attribute
 TYPETERM 239

TEXTKYBD attribute
 TYPETERM 239

TEXTKYBD option
 FEATURE operand 315

TEXTPRINT attribute
 TYPETERM 239

TEXTPRINT option
 FEATURE operand 315

time sharing option (TSO)
 the DFHCSDUP program 77

TIOAL operand
 DFHTCT TYPE=TERMINAL 316
 TCAM devices 316

TLT (terminal list table) 247, 329—330
 coding examples 330
 control section 329
 DFHTLT TYPE=ENTRY 329
 DFHTLT TYPE=INITIAL 329
 effect of autoinstall 109

TLX terminals using NTO
 eligible for autoinstall 105

TPNAME attribute
 TRANSACTION definition 214

TPURGE attribute
 TRANSACTION definition 214

TRACE attribute
 TRANSACTION definition 215

TRANCLASS attribute
 TRANSACTION definition 206, 215

TRANCLASS definition
 installed singly 20, 59
 MAXACTIVE attribute 205
 PURGETHRESH attribute 205

TRANSACTION attribute
 SESSIONS definition 182, 344
 TRANSACTION definition 216
 TYPETERM definition 236

TRANSACTION definition
 ALIAS attribute 208
 CMDSEC attribute 208
 COMMIT attribute 210
 CONFDATA attribute 208

TRANSACTION definition (*continued*)

- DESCRIPTION attribute 209
- DTIMOUT attribute 209
- DUMP attribute 209
- DYNAMIC attribute 209
- EXTSEC attribute 341
- INDOUBT attribute 210
- installed singly 20, 59
- ISOLATE attribute 210
- LOCALQ attribute 211
- PARTITIONSET attribute 211
- PRIMEDSIZE attribute 342
- PRIORITY attribute 212
- PROGRAM attribute 212
- REMOTENAME attribute 212
- REMOTESYSTEM attribute 212
- RESSEC attribute 24, 212
- RESTART attribute 212
- SPURGE attribute 213
- STATUS attribute 213
- STORAGECLEAR attribute 213
- TASKDATAKEY attribute 213
- TASKDATALOC attribute 214
- TASKREQ attribute 214
- TCLASS attribute 214, 343
- TPNAME attribute 214
- TPURGE attribute 214
- TRACE attribute 215
- TRANCLASS attribute 206, 215
- TRANSACTION attribute 216
- TRANSEC attribute 344
- TRPROF attribute 216
- TWASIZE attribute 216
- XTPNAME attribute 216
- XTRANID attribute 216

transaction list table (XLT) 247, 337—338

- coding example 338
- control section 337
- DFHXLT TYPE=INITIAL 337

TRANSACTION option

- TRMSTAT operand 309, 317

transaction routing 321

- autoinstall restriction 109
- DFHTCT macros for remote terminals 321
- using INDIRECT connections for 132

TRANSCIVE option

- TRMSTAT operand 309, 317

TRANSEC attribute

- TRANSACTION definition 344

TRANSID operand

- DFHDCT TYPE=INTRA 261
- DFHTCT TYPE=TERMINAL 308, 316
- DFHXLT TYPE=ENTRY 337
- sequential devices 308
- TCAM devices 316

- transient data destinations 257
- TRANSPARENCY option
 - FEATURE operand 315
- trigger level, for intrapartition transient data 261
- TRIGLEV operand
 - DFHDCT TYPE=INTRA 261
- TRMIDNT operand
 - DFHTCT TYPE=REMOTE 322
 - DFHTCT TYPE=TERMINAL 309, 316, 322
 - DFHTLT TYPE=ENTRY 329
 - lowercase in TCT 300
 - remote terminals 322
 - sequential devices 309
 - TCAM devices 316
- TRMIDNT, see TERMINAL attribute
- TRMMODL operand
 - DFHTCT TYPE=TERMINAL 316
 - TCAM devices 316
- TRMPRTY operand
 - DFHTCT TYPE=TERMINAL 309, 317
 - sequential devices 309
 - TCAM devices 317
- TRMSTAT operand
 - DFHTCT TYPE=TERMINAL 309, 317
 - sequential devices 309
- TRMTYPE operand
 - DFHTCT TYPE=LINE 308, 312
 - DFHTCT TYPE=REMOTE 323
 - DFHTCT TYPE=TERMINAL 317, 322
 - remote terminals 322, 323
 - sequential devices 308
 - TCAM devices 312, 317
- TRPROF attribute
 - TRANSACTION definition 216
- TSAGE operand
 - DFHTST TYPE=INITIAL 332
- TSO (time sharing option)
 - the DFHCSDUP program 77
- TST (temporary storage table) 247, 331—335
 - control section 331
 - DFHTST TYPE=INITIAL 331
 - DFHTST TYPE=RECOVERY 332, 333, 334
 - local temporary storage DATAIDs, DFHTST TYPE=LOCAL 332
 - recoverable temporary storage 332, 333, 334
 - recoverable temporary storage, DFHTST TYPE=RECOVERY 332
 - remote temporary storage DATAIDs, DFHTST TYPE=REMOTE 333
- TCT
 - TCAM devices 319
 - temporary storage security checking, DFHTST TYPE=SECURITY 334
- TTI attribute
 - TYPETERM 239

TWASIZE attribute
 TRANSACTION definition 216
 TWX terminals using NTO
 eligible for autoinstall 105
 TYPE=FINAL macro 249
 TYPE=INITIAL macro 248
 TYPE=LOCAL, DFHTST macro
 local temporary storage DATAIDs, DFHTST 332
 TYPE=REMOTE, DFHTST macro
 remote temporary storage DATAIDs, DFHTST 333
 TYPE=SECURITY macro
 temporary storage security checking, DFHTST 334
 TYPEFLE operand
 DFHDCT TYPE=SDSCI 258
 TYPETERM attribute
 TYPETERM 239
 TYPETERM definition 110
 CLEARCONV attribute 180, 237
 DESCRIPTION attribute 229
 installed by group 21, 59
 keeping in a separate group 18
 MESSAGE attribute 236
 NONE attribute 180, 237
 RECEIVESIZE attribute 235
 RECOVNOTIFY attribute 236
 RECOVOPTION attribute 180, 236
 RELEASESESS attribute 180, 237
 SYSDEFAULT attribute 180, 236
 TRANSACTION attribute 236
 UNCONDREL attribute 180, 237

U

UCTRAN attribute
 PROFILE 167
 TYPETERM 240
 UCTRAN option
 FEATURE operand 315
 UNBLOCKED option
 RECFORM operand 273
 UNCONDREL attribute
 TYPETERM definition 180, 237
 UNDEFINED option
 RECFORM operand 273
 UNLOCK GROUP command
 controlling access to groups 25, 61, 67
 UNLOCK LIST command
 controlling access to lists 25, 61, 67
 UPDATE option
 SERVREQ operand 274, 275
 UPGRADE command (DFHCSDUP)
 USING operand 101
 USAGE attribute
 PROGRAM 173
 USEDFTUSER attribute
 CONNECTION definition 139

USEDFTUSER attribute (*continued*)
 TERMINAL definition 202
 USELPACOPY attribute
 PROGRAM 173
 user-written program for autoinstall 116
 USERAREALEN attribute
 SESSIONS 182
 TYPETERM 240
 USERDEFINE command (CEDA) 68
 USERID attribute
 SESSIONS 182
 USERID operand
 DFHDCT TYPE=INITIAL 257
 DFHDCT TYPE=INTRA 260
 DFHTCT TYPE=TERMINAL 309, 317
 sequential devices 309
 TCAM devices 317
 userid, for intrapartition transient data 260
 USERVAR of remote system 137
 utility programs, offline
 initializing CICS system definition file, DFHCSDUP 73
 running under TSO 77

V

VALIDATION attribute
 TYPETERM 240
 VALIDATION option
 FEATURE operand 315
 values for attributes
 defaults dependent on other values 26
 forced, dependent on other values 26
 VARBLK option
 RECFORM operand 258
 VARBLKA option
 RECFORM operand 258
 VARBLKM option
 RECFORM operand 258
 VARIABLE option
 RECFORM operand 274
 VARUNB option
 RECFORM operand 258
 VARUNBA option
 RECFORM operand 258
 VARUNBM option
 RECFORM operand 258
 VERIFY command (DFHCSDUP) 102
 VERIFY operand
 DFHFCT TYPE=FILE 275
 VERTICALFORM attribute
 TYPETERM 240
 VF operand
 TCAM devices 318
 TYPE=TERMINAL 318
 VIEW command (CEDA) 71

VOLCNT operand
 DFHJCT TYPE=ENTRY 283
VTAM
 application name of remote system 137
VTAM and autoinstall 109
VTAM option
 ACCMETH operand 300

W

W option
 MACRF operand 310
warm restart
 nonrecovery of autoinstalled entries 113
 recreation of tables 20
WTOL operand
 DFHCLT TYPE=WTO 255

X

XLT (transaction list table) 247, 337—338
 coding example 338
 control section 337
 DFHXLT TYPE=INITIAL 337
XRFSIGNOFF attribute
 TYPETERM 240
XTPNAME attribute
 TRANSACTION definition 216
XTRANID attribute
 TRANSACTION definition 216

Sending your comments to IBM

CICS for MVS/ESA

Resource Definition Guide

SC33-1166-02

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 1962 870229
 - From within the U.K., use 01962 870229
- Electronically, use the appropriate network ID:
 - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
 - IBMLink: WINVMD(IDRCF)
 - Internet: idrcf@winvmd.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

Resource Definition Guide

SC33-1166-02

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	

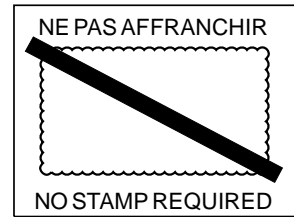
If your country is not listed here, your local IBM representative will be pleased to forward your comments to us. Or you can pay the postage and send the form direct to IBM (this includes mailing in the U.K.).

1 Cut along this line

2 Fold along this line

By air mail
Par avion

IBRS/CCR NUMBER: PHQ-D/1348/SO



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.

SC33-1166-02

