

CICS for MVS/ESA.



Application Programming Reference

Version 4 Release 1

CICS for MVS/ESA.



Application Programming Reference

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 vii.

Second edition (April 1997)

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

This is the second edition of the Application Programming Reference for CICS/ESA 4.1. It is based on the first edition, SC33-1170-00, 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	vii	CONVERSE (3650-3680)	47
Programming interface information	vii	CONVERSE (3767)	47
Trademarks and service marks	vii	CONVERSE (3770)	48
		CONVERSE (3790 full-function or inquiry)	48
Preface	ix	CONVERSE (3790 3270-display)	49
Determining if a publication is current	ix	CONVERSE (non-VTAM default)	53
Bibliography	x	CONVERSE (MRO)	53
CICS/ESA 4.1 library	x	CONVERSE (System/3)	54
Other CICS books	xi	CONVERSE (System/7)	54
Books from related libraries	xi	CONVERSE (2260)	55
		CONVERSE (2741)	56
Summary of changes	xiii	CONVERSE (2770)	57
Changes for the first edition of the CICS/ESA 4.1 book	xiii	CONVERSE (2780)	57
Changes for the second edition of the CICS/ESA 4.1		CONVERSE (3270 display)	58
book	xiv	CONVERSE (3600 BTAM)	58
		CONVERSE (3735)	60
Chapter 1. Commands	1	CONVERSE (3740)	60
Command format	1	DELAY	65
CICS syntax notation used in this book	1	DELETE	67
# Possible ABEND AICA timeout	2	DELETEQ TD	70
Argument values	2	DELETEQ TS	71
MVS/ESA restrictions	4	DEQ	72
CICS-value data areas (cvdas)	5	DUMP TRANSACTION	74
LENGTH options	5	ENDBR	77
NOHANDLE option	5	ENQ	78
RESP and RESP2 options	5	ENTER TRACENUM	80
CICS/ESA interface to JES	6	EXTRACT ATTACH (LUTYPE6.1)	82
Using the CICS interface to JES	8	EXTRACT ATTACH (MRO)	84
Commands by function	9	EXTRACT ATTRIBUTES (APPC)	86
ABEND	12	EXTRACT ATTRIBUTES (MRO)	87
ADDRESS	13	EXTRACT LOGONMSG	88
ADDRESS SET	15	EXTRACT PROCESS	89
ALLOCATE (APPC)	16	EXTRACT TCT	91
ALLOCATE (LUTYPE6.1)	18	FORMATTIME	92
ALLOCATE (MRO)	20	FREE	95
ASKTIME	21	FREE (APPC)	96
ASSIGN	22	FREE (LUTYPE6.1)	97
BIF DEEDIT	30	FREE (MRO)	98
BUILD ATTACH (LUTYPE6.1)	31	FREEMAIN	99
BUILD ATTACH (MRO)	33	GDS ALLOCATE	102
CANCEL	35	GDS ASSIGN	104
CHANGE PASSWORD	37	GDS CONNECT PROCESS	105
CHANGE TASK	38	GDS EXTRACT ATTRIBUTES	107
CONNECT PROCESS	39	GDS EXTRACT PROCESS	108
CONVERSE (VTAM default)	41	GDS FREE	109
CONVERSE (APPC)	41	GDS ISSUE ABEND	110
CONVERSE (LUTYPE2/LUTYPE3)	42	GDS ISSUE CONFIRMATION	111
CONVERSE (LUTYPE4)	42	GDS ISSUE ERROR	112
CONVERSE (LUTYPE6.1)	43	GDS ISSUE PREPARE	113
CONVERSE (SCS)	43	GDS ISSUE SIGNAL	114
CONVERSE (3270 logical)	44	GDS RECEIVE	115
CONVERSE (3600-3601)	44	GDS SEND	117
CONVERSE (3600-3614)	45	GDS WAIT	119
CONVERSE (3650 interpreter)	45	GETMAIN	120
CONVERSE (3650-3270)	46	HANDLE ABEND	123
CONVERSE (3650-3653)	46	HANDLE AID	125

HANDLE CONDITION	127	RECEIVE (System/7)	211
IGNORE CONDITION	129	RECEIVE (2260)	212
ISSUE ABEND	130	RECEIVE (2741)	212
ISSUE ABORT	131	RECEIVE (2980)	213
ISSUE ADD	133	RECEIVE (3270 display)	215
ISSUE CONFIRMATION	135	RECEIVE (3600 BTAM)	215
ISSUE COPY (3270 display)	137	RECEIVE (3735)	216
ISSUE COPY (3270 logical)	138	RECEIVE (3740)	216
ISSUE DISCONNECT (default)	139	RECEIVE (3790 3270-display)	217
ISSUE DISCONNECT (LUTYPE6.1)	140	RECEIVE MAP	220
ISSUE END	141	# RECEIVE MAP MAPPINGDEV	223
ISSUE ENDFILE	143	RECEIVE PARTN	225
ISSUE ENDOUTPUT	144	RELEASE	227
ISSUE EODS	145	RESETBR	229
ISSUE ERASE	146	RETRIEVE	232
ISSUE ERASEAUP	148	RETURN	235
ISSUE ERROR	149	REWRITE	237
ISSUE LOAD	150	ROUTE	239
ISSUE NOTE	151	SEND (VTAM default)	242
ISSUE PASS	152	SEND (APPC)	242
ISSUE PREPARE	154	SEND (LUTYPE2/LUTYPE3)	243
ISSUE PRINT	155	SEND (LUTYPE4)	243
ISSUE QUERY	156	SEND (LUTYPE6.1)	244
ISSUE RECEIVE	157	SEND (SCS)	244
ISSUE REPLACE	159	SEND (3270 logical)	245
ISSUE RESET	161	SEND (3600 pipeline)	245
ISSUE SEND	162	SEND (3600-3601)	246
ISSUE SIGNAL (APPC)	164	SEND (3600-3614)	246
ISSUE SIGNAL (LUTYPE6.1)	165	SEND (3650 interpreter)	247
ISSUE WAIT	166	SEND (3650-3270)	247
LINK	168	SEND (3650-3653)	248
LOAD	172	SEND (3650-3680)	248
MONITOR	174	SEND (3767)	249
POINT	176	SEND (3770)	249
POP HANDLE	177	SEND (3790 full-function or inquiry)	250
POST	178	SEND (3790 SCS)	250
PURGE MESSAGE	180	SEND (3790 3270-display)	251
PUSH HANDLE	181	SEND (3790 3270-printer)	251
QUERY SECURITY	182	SEND (non-VTAM default)	255
READ	185	SEND (MRO)	255
READNEXT	189	SEND (System/3)	256
READPREV	193	SEND (System/7)	256
READQ TD	196	SEND (2260)	257
READQ TS	198	SEND (2741)	257
RECEIVE (VTAM default)	201	SEND (2980)	258
RECEIVE (APPC)	201	SEND (3270 display)	258
RECEIVE (LUTYPE2/LUTYPE3)	202	SEND (3600 BTAM)	259
RECEIVE (LUTYPE4)	202	SEND (3735)	259
RECEIVE (LUTYPE6.1)	203	SEND (3740)	260
RECEIVE (3270 logical)	203	SEND CONTROL	264
RECEIVE (3600 pipeline)	204	SEND MAP	268
RECEIVE (3600-3601)	204	# SEND MAP MAPPINGDEV	274
RECEIVE (3600-3614)	205	SEND PAGE	276
RECEIVE (3650)	205	SEND PARTNSET	279
RECEIVE (3767)	206	SEND TEXT	280
RECEIVE (3770)	206	SEND TEXT MAPPED	284
RECEIVE (3790 full-function or inquiry)	207	SEND TEXT NOEDIT	286
RECEIVE (non-VTAM default)	210	SIGNOFF	289
RECEIVE (MRO)	210	SIGNON	290
RECEIVE (System/3)	211	SPOOLCLOSE	293

SPOOLOPEN INPUT	294	Appendix J. BMS-related constants	375
SPOOLOPEN OUTPUT	296	Magnetic slot reader (MSR) control value constants,	
SPOOLREAD	299	DFHMSRCA	377
SPOOLWRITE	301	MSR control byte values	377
START	303	Attention identifier constants, DFHAID	379
STARTBR	309		
SUSPEND	313	Appendix K. BMS macro summary	381
SYNCPOINT	314	Mapset, map, and field definition	381
SYNCPOINT ROLLBACK	315	Partition set definition	381
UNLOCK	316	DFHMDF	383
VERIFY PASSWORD	318	DFHMDI	390
WAIT CONVID (APPC)	320	DFHMDS	396
WAIT EVENT	321	DFHPDI	403
WAIT EXTERNAL	322	DFHPSD	404
WAIT JOURNALNUM	324		
WAIT SIGNAL	325	Index	405
WAIT TERMINAL	326		
WAITCICS	327		
WRITE	329		
WRITE JOURNALNUM	332		
WRITE OPERATOR	334		
WRITEQ TD	336		
WRITEQ TS	338		
XCTL	341		
Appendix A. EXEC interface block	343		
EIB fields	343		
Appendix B. Codes returned by ASSIGN	351		
ASSIGN TERMCODE	351		
ASSIGN FCI	351		
Appendix C. Translated code for CICS commands	353		
COBOL	353		
C	353		
PL/I	353		
Assembler language	353		
Appendix D. Terminal control	357		
Commands and options for terminals and logical units	357		
TCAM-supported terminals and logical units	358		
BTAM programmable terminals	358		
Teletypewriter programming	359		
Display device operations	360		
Appendix E. SAA Resource Recovery	363		
SRRCMT	363		
SRRBACK	363		
Appendix F. Common Programming Interface Communications (CPI Communications)	365		
Appendix G. API restrictions for distributed program link	367		
Summary of the restricted API commands	367		
List of API commands	367		
Appendix H. CVDA numeric values	371		
Appendix I. National language codes	373		

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..

Programming interface information

This book is intended to help you write application programs using EXEC CICS commands. This book documents General-use Programming Interface and Associated Guidance Information provided by CICS.

General-use programming interfaces allow the customer to write programs that obtain the services of CICS.

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	AFP	AT
BookManager	C/370	CICS
CICS OS/2	CICS/ESA	CICS/MVS
CICS/VSE	DATABASE 2	DB2
ESA/370	IBM	IBMLink
IMS/ESA	MVS/ESA	MVS/SP
RACF	System/370	Systems Application Architecture
SAA	SQL/DS	VTAM

Preface

What this book is about: This book describes the CICS/ESA application programming interface; it contains *reference* information needed to prepare COBOL, C, PL/I, and assembler-language application programs, using CICS commands, to be executed under CICS. *Guidance* information is in the &dfhpe001.. For information about debugging CICS applications, see the *CICS/ESA Problem Determination Guide*.

Who this book is for: The book is intended primarily for use by application programmers, but will also be useful to system programmers and systems analysts.

What you need to know to understand this book: We assume that you have some experience in writing programs in COBOL, C, PL/I, or assembler language. You should also have a knowledge of the CICS concepts and terminology introduced in the *CICS/ESA 3.3 Facilities and Planning Guide* and you should have read the *CICS Application Programming Primer (VS COBOL II)* and the *CICS/ESA Application Programming Guide*.

How to use this book: This book is for reference. Each of the commands has a standard format, as follows:

- The syntax of the command
- A description of what the command does
- An alphabetical list of the options and their functions
- An alphabetical list of conditions, and their causes, that can occur during execution of a command.

Notes on terminology

- **ASM** is used sometimes as the abbreviation for assembler language.

| **What this book does not cover:** The EXEC CICS commands for system programming; that is COLLECT, DISABLE, | ENABLE, INQUIRE, PERFORM, RESYNC, and SET are not covered in this book but you will find them in the *CICS/ESA | System Programming Reference*.

The EXEC CICS FEPI commands available for use with the CICS/ESA Front End Programming Interface feature are not discussed in this book, but in the *CICS/ESA Front End Programming Interface Feature User's Guide*.

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 Transaction Server 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 ix. The softcopy books are regularly updated to include the latest information.

Other CICS books

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

CICS Clients

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

Books from related libraries

MVS: see the following books:

MVS/ESA System Programming Library: Application Development 31-Bit Addressing, GC28-1820
MVS/ESA System Programming Library: Application Development Guide, GC28-1821
MVS/ESA System Programming Library: System Generation, GC28-1825
MVS/ESA System Programming Library: Initialization and Tuning, GC28-1828
MVS/ESA System Programming Library: Application Development Guide, GC28-1852
MVS/ESA System Programming Library: Application Development Guide - Extended Addressability, GC28-1854
MVS/ESA System Programming Library: Application Development Macro Reference, GC28-1857
MVS/ESA JCL User's Guide, GC28-1473
MVS/ESA Operations: Systems Commands, GC28-1826
ESA/370 Principles of Operation, SA22-7200.

Data facility product:

MVS/ESA Data Administration: Macro Instruction Reference, SC26-4506
MVS/ESA VSAM Administration: Macro Instruction Reference, SC26-4517
MVS/ESA VSAM Administration Guide, SC26-4518.

IMS: For information about IMS, see the following books. If you use the CICS-DL/I interface, see one of the following IMS books:

IMS/VS Version 2 Application Programming for CICS/OS/VS Users, SC26-4177
IMS Application Programming: Design Guide, SC26-4279
IMS Application Programming: EXEC DLI Commands manual, SC26-4280
IMS Application Programming: DL/I Calls manual, SC26-4274.

These books contain information about IMS application programming for CICS users, and tell you how to write online, batch, or CICS shared database programs that use the EXEC DLI interface or the DL/I CALL interface with the DL/I interface block (DIB). If you use the DL/I CALL interface with the TCA, you should see the *CICS/OS/VS Version 1 Release 7 Application Programmer's Reference Manual* (Macro Level).

Note: The EXEC DLI interface is no longer documented in the *CICS/ESA Application Programming Reference*.

Systems Network Architecture: See the following Systems Network Architecture (SNA) book for further information about SNA:

Sessions between Logical Units, GC20-1868.

Systems Application Architecture: See the following Systems Application Architecture (SAA) for further information about SAA:

SAA Common Programming Interface Communications Reference, SC26-4399.

SQL: For information about executing SQL in a CICS application program see the following books:

IBM DATABASE 2 Application Programming Guide for CICS Users, SC26-4080
IBM DATABASE 2 Reference, SC26-4078.

Other related books: You may also want to refer to the following IBM books:

Distributed Processing Programming Executive/Distributed Presentation Services (DPPX/DPS):

DPPX/Distributed Presentation Services Version 2 System Programming Guide, SC33-0117

OS/VS COBOL Compiler and Library Programmer's Guide, SC28-6483

VS COBOL II Application Programming Guide, SC26-4045

OS PL/I Version 2 Programming Guide, SC26-4307

An Introduction to the IBM 3270 Information Display System,
GA27-2739

3274 Control Unit Reference Summary, GX20-1878.

Component Description: IBM 2721 Portable Audio Terminal,
GA27-3029.

IBM 2780 Data Transmission Terminal Component
Description, GA27-3035

CICS/OS/VS IBM 3270 Data Stream Device Guide,
SC33-0232

IBM 3270 Data Stream Programmer's Reference, GA23-0059

CICS/OS/VS IBM 4700/3600/3630 Guide, SC33-0233

Summary of changes

This book includes information about the new or enhanced facilities introduced by CICS/ESA 4.1. Technical changes in CICS/ESA 4.1 are shown by vertical bars in the left hand margin.

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

| There are new commands for:

| CHANGE PASSWORD
| VERIFY PASSWORD

| The spool interface commands have been moved into this book from the *CICS/ESA System Programming Reference*:

| SPOOLCLOSE
| SPOOLOPEN INPUT
| SPOOLOPEN OUTPUT
| SPOOLREAD
| SPOOLWRITE

| The UMT commands are no longer shown separately but are included in the full command:

| DELETE
| ENDBR
| READ
| READNEXT
| READPREV
| RESETBR
| REWRITE
| STARTBR
| UNLOCK
| WRITE

| There are new options for ASSIGN:

| ASRAKEY
| ASRASPC
| ASRASTG
| INVOKINGPROG
| RETURNPROG

| There are new options for SIGNON:

| ESMREASON
| GROUPID
| LANGUAGECODE
| LANGINUSE

| There is a new option and a new condition for START:

| USERID
| USERIDERR

| TOKEN option added to

| DELETE
| READ
| REWRITE
| SPOOLCLOSE
| SPOOLOPEN

| SPOOLREAD
| SPOOLWRITE
| UNLOCK

| DEFAULT and ALTERNATE options added to

| CONVERSE
| SEND
| SEND CONTROL
| SEND MAP
| SEND TEXT
| SEND TEXT NOEDIT

| 4-digit years options added to FORMATTIME

| YYYYDDD
| YYYYMMDD
| YYYYDDMM
| DDDMMYYYY
| MMDDYYYY

| PARTNER option added to

| ALLOCATE (APPC)
| CONNECT PROCESS
| GDS ALLOCATE
| GDS CONNECT PROCESS

| NAME option added to

| WAIT EVENT
| WAIT EXTERNAL
| WAITCICS

| LOGMODE option added to

| ISSUE PASS

| FMH option added to

| SEND (SCS)

| FMHPARM option added to

| SEND TEXT

| LOADING option added to

| DELETE
| READ
| STARTBR
| WRITE

| STATE option removed from

| GDS EXTRACT PROCESS

| CONVDATA option removed from

| GDS EXTRACT ATTRIBUTES
| GDS FREE

| PARTNERIDERR condition added to

| ALLOCATE (APPC)
| CONNECT PROCESS

| INVREQ condition added to

| ISSUE ENDFILE
| ISSUE ENDOUTPUT

| ISSUE EODS
| ISSUE ERASEAUP
| ISSUE PRINT
| ISSUE RESET
| UNLOCK
| INVREQ condition removed from
| WAIT JOURNALNUM

| **Changes for the second edition of the
| CICS/ESA 4.1 book**

| 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.

Chapter 1. Commands

This book shows the syntax of each command, describes the purpose and format of each command and its options, and gives a list of the conditions that can arise during the execution of a command.

Note: The INQUIRE and SET commands of the application programming interface (API) are primarily for the use of the system programmer; they are not described in this book. For details of the commands, refer to the *CICS/ESA System Programming Reference*.

For information about translating the commands, see the *CICS/ESA Application Programming Guide* for translator options, and the *CICS/ESA System Definition Guide* for the JCL.

Command format

The general format of a CICS command is EXECUTE CICS (or EXEC CICS) followed by the name of the required **command**, and possibly by one or more **options**, as follows:

EXEC CICS command option(arg)....

where:

command describes the operation required (for example, READ).

option describes any of the many optional facilities available with each function. Some options are followed by an argument in parentheses. You can write options (including those that require arguments) in any order.

arg (**short for argument**) is a value such as "data-value" or "name". A "data-value" can be a constant, this means that an argument that sends data to CICS is generally a "data-value". An argument that receives data from CICS must be a "data-area".

Some arguments described as "data-area" can both send and receive data (such as LENGTH). In these cases, you must ensure that the "data-area" is not in protected storage.

An example of a CICS command is as follows:

```
EXEC CICS READ
      FILE('FILEA')
      INTO(FILEA)
      RIDFLD(KEYNUM)
      UPDATE
```

You must add the appropriate end-of-command delimiter; see "CICS syntax notation used in this book."

CICS syntax notation used in this book

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

The "EXEC CICS" that always precedes each command's keyword is not included; nor is the "END-EXEC" statement used in COBOL or the semicolon (;) used in PL/I and C that you must code at the end of each CICS command. In the C language, a null character can be used as an end-of-string marker, but CICS does not recognize this; you must therefore never have a comma or period followed by a space (X'40') in the middle of a coding line.

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 may code.
	A set of alternatives—any number of which you may code once.
	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 READ FILE(filename) you must code READ FILE and () unchanged, but are free to code any valid text string to mean the name of the file.

```
# _____ Apar 80057 _____
# Documentation for Apar 80057 added 29 Feb 1996
# (TUCKER)
```

Possible ABEND AICA timeout

A task can be abended by CICS if it has been running for
longer than the runaway time, ICVR, specified in the system
initialization table (SIT). This abend can be prevented by
coding an EXEC CICS SUSPEND command in the
application. This causes the dispatcher to suspend the task
that issued the request and allow any task of higher priority
to run. If there is no task ready to run, the program that
issued the suspend is resumed. For further information
about abend AICA, see the Problem Determination Guide.

Argument values

The parenthesized argument values that follow options in a CICS command are specified as follows:

- data-value
- data-area
- cvda (CICS-value data area)
- ptr-value
- ptr-ref
- name
- label
- hhmss
- filename
- systemname

COBOL argument values

The argument values can be replaced as follows:

- “data-value” can be replaced by any COBOL data name of the correct data type for the argument, or by a constant that can be converted to the correct type for the argument. The data type can be specified as one of the following:
 - Halfword binary — PIC S9(4) COMP
 - Fullword binary — PIC S9(8) COMP
 - Character string — PIC X(n) where “n” is the number of bytes.
- “data-area” can be replaced by any COBOL data name of the correct data type for the argument. The data type can be specified as one of the following:
 - Halfword binary — PIC S9(4) COMP
 - Fullword binary — PIC S9(8) COMP
 - Character string — PIC X(n) where “n” is the number of bytes.

Where the data type is unspecified, “data-area” can refer to an elementary or group item.

- “cvda” is described in “CICS-value data areas (cvdas)” on page 5.
- “ptr-value” can be replaced by the name of any base locator for linkage (BLL) cell, or by any COBOL data name that contains a copy of such a pointer in a BLL cell.

- “ptr-ref” can be replaced by the name of any BLL cell. In VS COBOL II, you can replace “ptr-ref” with a pointer variable or an ADDRESS special register.
- “name” can be replaced by either of the following:
 - A character string in single quotation marks (that is, a nonnumeric literal). If this is shorter than the required length, it is padded with blanks.
 - A COBOL data area with a length equal to the length required for the name. The value in “data-area” is the name to be used by the argument. If “data-area” is shorter than the required length, the excess characters are undefined.

“filename”, as used in FILE(filename), specifies the name of the file. It has 1–8 characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

“systemname”, as used in SYSID(systemname), specifies the name of the system the request is directed to. It has 1–4 characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

- “label” can be replaced by any COBOL paragraph name or a section name.
- “hhmss” can be replaced by a decimal constant or by a data name of the form PIC S9(7) COMP-3. The value must be of the form 0HHMMSS+ where:

HH represents hours from 00 through 99

MM represents minutes from 00 through 59

SS represents seconds from 00 through 59.

In OS/VS COBOL, you must code the LENGTH options. In VS COBOL II there is no need to code the LENGTH option unless you want the program to read or write data of a length different from that of the referenced variable.

C argument values

The argument values can be replaced as follows:

- “data-value” can be replaced by any C expression that can be converted to the correct data type for the argument. The data type can be specified as one of the following:
 - Halfword binary — short int
 - Fullword binary — long int
 - Character string — char[n] where “n” is the number of bytes.

“data-value” includes “data-area” as a subset.

- “data-area” can be replaced by any C data reference that has the correct data type for the argument. The data type can be specified as one of the following:
 - Halfword binary — short int
 - Fullword binary — long int
 - Character string — char[n] where “n” is the number of bytes.

If the data type is unspecified, “data-area” can refer to a scalar data type, array, or structure. The reference must be to contiguous storage.

- “cvda” is described in “CICS-value data areas (cvdas)” on page 5.
- “ptr-value” (which includes “ptr-ref” as a subset) can be replaced by any C expression that can be converted to an address.
- “ptr-ref” can be replaced by any C pointer type reference.
- “name” can be replaced by either of the following:
 - A character string in double quotation marks (that is, a literal constant).
 - A C expression or reference whose value can be converted to a character array with a length equal to the maximum length allowed for the name. The value of the character array is the name to be used by the argument.

“filename”, as used in FILE(filename), specifies the name of the file. It has 1–8 characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

“systemname”, as used in SYSID(systemname), specifies the name of the system the request is directed to. It has 1–4 characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

- “hhmss” is not supported in the C language.
- “label” is not supported in the C language.
- “hhmss” can be replaced by an integer constant; otherwise the application is responsible for ensuring that the value passed to CICS is in packed decimal format. The language does not provide a packed decimal type.

HH represents hours from 00 through 99
MM represents minutes from 00 through 59
SS represents seconds from 00 through 59.

Many commands involve the transfer of data between the application program and CICS.

In most cases, the LENGTH option must be specified if SET is used; the syntax of each command and its associated options show whether or not this rule applies.

PL/I argument values

The argument values can be replaced as follows:

- “data-value” can be replaced by any PL/I expression that can be converted to the correct data type for the argument. The data type can be specified as one of the following:
 - Halfword binary — FIXED BIN(15)
 - Fullword binary — FIXED BIN(31)
 - Character string — CHAR(n) where “n” is the number of bytes.

“data-value” includes “data-area” as a subset.

- “data-area” can be replaced by any PL/I data reference that has the correct data type for the argument. The data type can be specified as one of the following:
 - Halfword binary — FIXED BIN(15)
 - Fullword binary — FIXED BIN(31)
 - Character string — CHAR(n) where “n” is the number of bytes.

If the data type is unspecified, “data-area” can refer to an element, array, or structure; for example, FROM(P→STRUCTURE) LENGTH(LNG). The reference must be to connected storage.

The data area must also have the correct PL/I alignment attribute: ALIGNED for binary items, and UNALIGNED for strings.

If you use a varying data string without an explicit length, the data passed begins with two length bytes, and its length is the maximum length declared for the string. If you explicitly specify a length in the command, the data passed has this length; that is, the two length bytes followed by data up to the length you specified.

- “cvda” is described in “CICS-value data areas (cvdas)” on page 5.
- “ptr-value” (which includes “ptr-ref” as a subset) can be replaced by any PL/I expression that can be converted to POINTER.
- “ptr-ref” can be replaced by any PL/I reference of type POINTER ALIGNED.
- “name” can be replaced by either of the following:
 - A character string in single quotation marks (that is, a literal constant).
 - A PL/I expression or reference whose value can be converted to a character string with a length equal to the maximum length allowed for the name. The value of the character string is the name to be used by the argument.

“filename”, as used in FILE(filename), specifies the name of the file. It has 1–8 characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

“systemname”, as used in SYSID(systemname), specifies the name of the system the request is directed to. characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

- “label” can be replaced by any PL/I expression whose value is a label.
- “hhmss” can be replaced by a decimal constant or an expression that can be converted to a FIXED DECIMAL(7,0). The value must be of the form 0HHMMSS+ where:

HH represents hours from 00 through 99
MM represents minutes from 00 through 59
SS represents seconds from 00 through 59.

If the UNALIGNED attribute is added to the ENTRY declarations generated by the CICS translator by a DEFAULT DESCRIPTORS statement, data-area or pointer-reference arguments to CICS commands must also be UNALIGNED. Similarly for the ALIGNED attribute, data-area or pointer-reference arguments must be ALIGNED.

Many commands involve the transfer of data between the application program and CICS.

In most cases, the length of the data to be transferred must be provided by the application program. However, if a data area is specified as the source or target, it is not necessary to provide the length explicitly, because the command-language translator generates a default length value of either STG(data-area) or CSTG(data-area), as appropriate.

Assembler-language argument values

In general, an argument may be either the address of the data or the data itself (in assembler-language terms, either a relocatable expression or an absolute expression).

A relocatable expression must not contain unmatched brackets (outside quotation marks) or unmatched quotation marks (apart from length-attribute references). If this rule is obeyed, any expression can be used, including literal constants, such as =AL2(100), forms such as 20(0,R11), and forms that use the macro-replacement facilities.

An absolute expression must be a single term that is either a length-attribute reference, or a self-defining constant.

Care must be taken with equated symbols, which should be used only when referring to registers (pointer references). If an equated symbol is used for a length, for example, it is treated as the address of the length and an unpredictable error occurs.

The argument values can be replaced as follows:

- “data-value” can be replaced by a relocatable expression that is an assembler-language reference to data of the correct type for the argument, or by a constant of the correct type for the argument.
- “data-area” can be replaced by a relocatable expression that is an assembler-language reference to data of the correct type for the argument.
- “cvda” is described in “CICS-value data areas (cvdas)” on page 5.
- “ptr-value” can be replaced by an absolute expression that is an assembler-language reference to a register.
- “ptr-ref” can be replaced by an absolute expression that is an assembler-language language reference to a register.
- “name” can be replaced **either** by a character string in single quotation marks, **or** by an assembler-language language relocatable expression reference to a character

string. The length is equal to the maximum length allowed for the name. The value of the character string is the name to be used by the argument.

“filename”, as used in FILE(filename), specifies the name of the file. It has 1–8 characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

“systemname”, as used in SYSID(systemname), specifies the name of the system the request is directed to. It has 1–4 characters from A–Z, 0–9, \$, @, and #, (lowercase characters are converted to uppercase).

- “label” refers to a destination address to which control is transferred. It can be replaced by the label of the destination instruction or by the label of an address constant for the destination. This constant must not specify a length.

You can also use the expression =A(dest) where “dest” is a relocatable expression denoting the destination.

For example, the following commands are equivalent:

```
HANDLE CONDITION ERROR(DEST)
HANDLE CONDITION ERROR(ADCON)
HANDLE CONDITION ERROR(=A(DEST))
:
DEST BR 14
ADCON DC A(DEST)
```

- “hhmmss” can be replaced by a self-defining decimal constant, or an assembler-language reference to a field defined as PL4. The value must be of the form OHHMMSS+ where:

HH represents hours from 00 through 99
MM represents minutes from 00 through 59
SS represents seconds from 00 through 59.

Many commands involve the transfer of data between the application program and CICS.

In most cases, the length of the data to be transferred must be provided by the application program. However, if a data area is specified as the source or target, it is not necessary to provide the length explicitly, because the command-language translator generates a default length.

For example:

```
xxx DC CL8
:
EXEC CICS ... LENGTH(L'xxx)
```

MVS/ESA restrictions

| The following general restrictions apply to all CICS commands:

| The restrictions that apply to CICS commands that access user

- The program must be in primary addressing mode when

invoking any CICS service. The primary address space must be the home address space. All parameters passed to CICS must reside in the primary address space.

- CICS does not always preserve access registers across CICS commands or macro invocations. If your program uses access registers, it should save them before invoking a CICS service, and restore them before reusing them.

CICS-value data areas (cvdas)

There are options on a number of commands that describe or define a resource. CICS supplies, in CICS-value data areas, the values associated with these options. The options are shown in the syntax of the commands with the term “cvda” in parentheses.

You pass a cvda value in two different ways:

- You can assign a cvda value with the translator routine DFHVALUE. This allows you to change a cvda value in the program as the result of other run-time factors.

For example:

```
MOVE DFHVALUE(NOTPURGEABLE) TO AREA-A.  
EXEC CICS WAIT EXTERNAL ECBLIST() NUMEVENTS()  
          PURGEABILITY(AREA-A).
```

- If the required action is always the same, you can declare the value directly.

For example:

```
EXEC CICS WAITCICS ECBLIST() NUMEVENTS() PURGEABLE.
```

You receive a cvda value by defining a fullword binary data area and then testing the returned value with the translator routine DFHVALUE. For example:

```
EXEC CICS CONNECT PROCESS .... STATE(AREA-A)  
IF AREA-A = DFHVALUE(ALLOCATED) ....  
IF AREA-A = DFHVALUE(CONFFREE) ....
```

LENGTH options

In VS COBOL II, PL/I, and assembler language, the translator defaults certain lengths,

```
# 

|                     |
|---------------------|
| <b>Apar PQ06624</b> |
|---------------------|

  
# Documentation for Apar PQ06624 added 20/08/97
```

if the NOLENGTH translator option is not specified. This means they are optional in programs that specify data areas.

In OS/VS COBOL and C, all LENGTH options must be specified.

When a CICS command offers the LENGTH option, it is generally expressed as a signed halfword binary value. This puts a theoretical upper limit of 32 763 bytes on LENGTH. In

practice (depending on issues of recoverability, function shipping, and so on), the achievable upper limit varies from command to command, but is somewhat less than this theoretical maximum.

To be safe, do not let the value assigned to the length option # for any CICS command exceed 24KB.

For **journalled items**, the length may be further restricted by the buffer size of the journal.

For **journal** commands, the restrictions apply to the sum of the LENGTH and PFLENGTH values.

For **transient data**, and **file control** commands, the data set definitions may themselves impose further restrictions.

NOHANDLE option

You can use the NOHANDLE option with any command to specify that you want no action to be taken for any condition or AID resulting from the execution of that command. For further information about the NOHANDLE option, see the *CICS/ESA Application Programming Guide*.

| Note that using the C language implies NOHANDLE on all | commands.

RESP and RESP2 options

You can use the RESP option with any command to test whether a condition was raised during its execution. With some commands, when a condition can be raised for more than one reason, you can, if you have already specified RESP, use the RESP2 option to determine exactly why a condition occurred.

RESP(xxx)

“xxx” is a user-defined fullword binary data area. On return from the command, it contains a value corresponding to the condition that may have been raised, or to a normal return, that is, xxx=DFHRESP(NORMAL). You can test this value by means of DFHRESP, as follows:

```
# EXEC CICS WRITEQ TS FROM(abc)  
#                               QUEUE(qname)  
#                               NOSUSPEND  
#                               RESP(xxx)  
#                               RESP2(yyy)  
#                               .  
#                               .  
# IF xxx=DFHRESP(NOSPACE) THEN ...
```

The above form of DFHRESP applies to both COBOL and PL/I.

An example of a similar test in C:

```

switch (xxx) {
  case DFHRESP(NORMAL) : break;
  case DFHRESP(INVREQ) : Invreq_Cond();
                        break;
  default                : Errors();
}

```

An example of a similar test in assembler language:

```
CLC   xxx,DFHRESP(NOSPACE)
```

which the translator changes to:

```
CLC   xxx,=F'18'
```

As the use of RESP implies NOHANDLE, you must be careful when using RESP with the RECEIVE command, because NOHANDLE overrides the HANDLE AID command as well as the HANDLE CONDITION command, with the result that PF key responses are ignored.

RESP2(yyy)

“yyy” is a user-defined fullword binary data area. On return from the command, it contains a value that further qualifies the response to certain commands. Unlike the RESP values, RESP2 values have no associated symbolic names and there is no translator built-in function corresponding to DFHRESP, so you must test the fullword binary value itself. RESP2 values are given in the description of each command that returns them.

CICS/ESA interface to JES

The CICS interface to JES (the Job Entry Subsystem component of MVS) provides specialist programmer EXEC commands for accessing the system spool files maintained by JES2 and JES3. You can support the requirements of other products to exchange files with other systems connected through a JES remote spooling communications subsystem (RSCS) network. The term JES is used to refer to both JES2 and JES3.

The CICS interface to JES enables you to:

- Retrieve data for a specific user from the local JES spool. See Figure 1 on page 7.
- Create a file and write records directly to the local JES spool. See Figure 2 on page 7.
- Send a JES spool file to a specific remote destination. See Figure 3 on page 7.

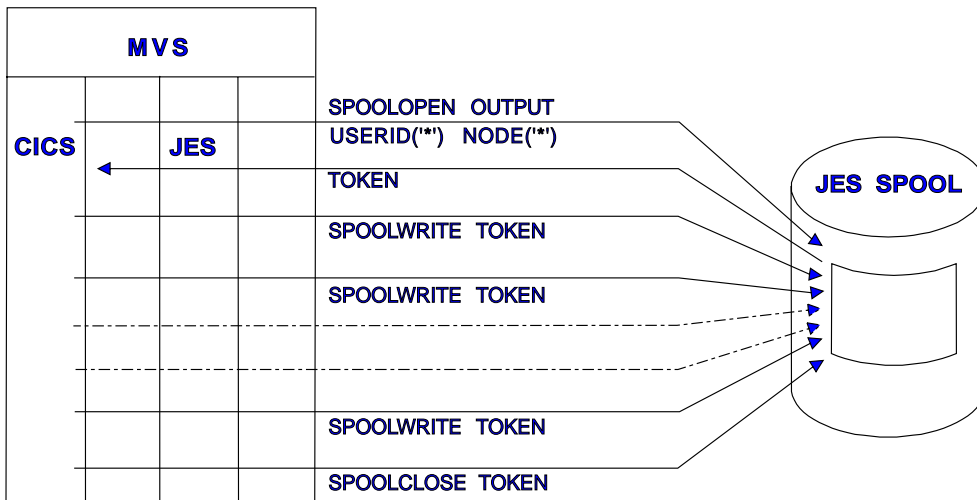
There are internal limits in JES2 and JES3 that you should consider when you are designing applications to use this interface. Some of these internal limits can depend on which release of JES you are using. You should therefore read the following in conjunction with the appropriate JES documentation.

JES2

- Number of SYSOUT data sets
There is an upper limit to the number of data sets that can be created by a single job. If this limit is exceeded during a CICS run, subsequent SPOOLOPEN OUTPUT requests return the ‘ALLOCERR’ condition.
- Output queue and job queue sizes
The number of Job Output Elements and Job Queue Elements may need to be increased to accommodate the additional output processing. Timely processing of the data sets created using this interface minimizes this requirement.
- Spool space
Although the spool space for a data set created using this interface is reused after it has been processed, some control information is retained for the life of the job. You may have to increase the spool file allocation to allow for this.

JES3

- Job queue size
The number of Job Queue Elements may need to be increased to accommodate the additional output processing. Also, more JSAM buffers may be required. Timely processing of the data sets created using this interface minimizes this requirement.
- Spool space
Although the spool space for a data set created using this interface is reused after it has been processed, some control information is retained for the life of the job. You may have to increase the spool file allocation to allow for this.



| Figure 1. Retrieve data from the JES spool

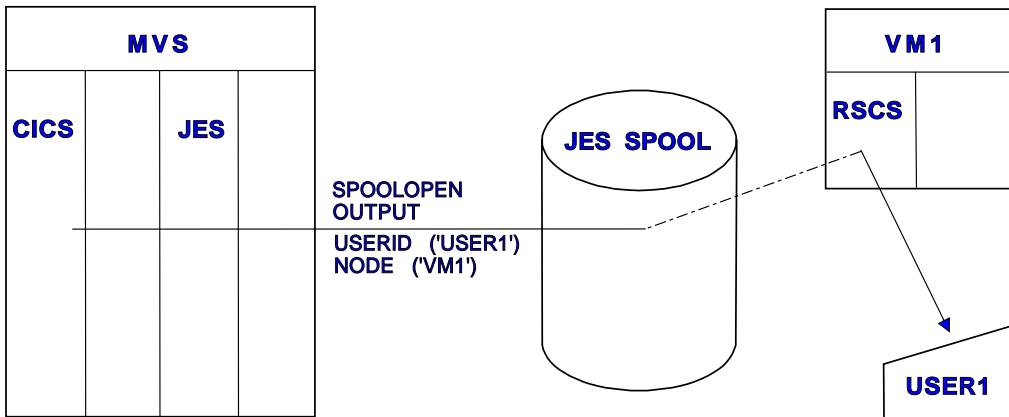


Figure 2. Create a file and write directly to the JES spool

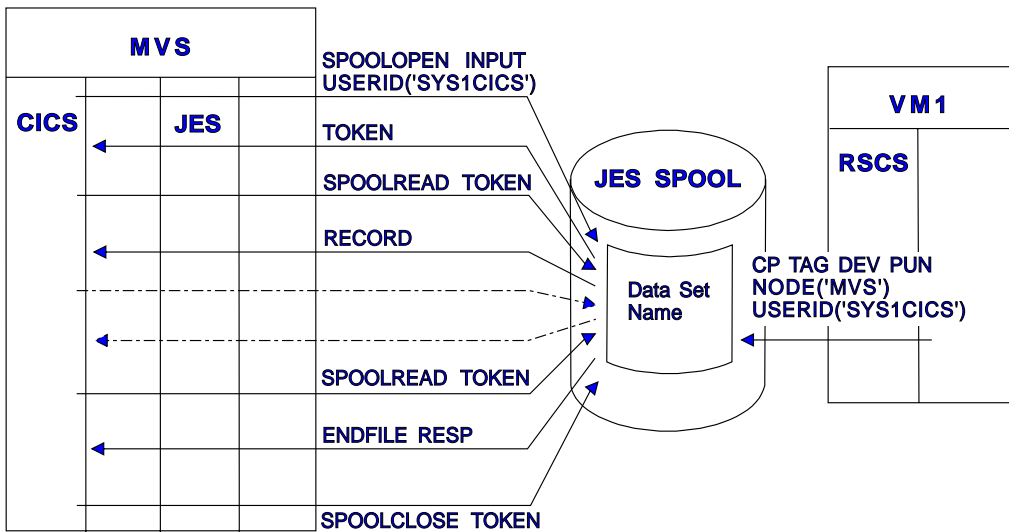


Figure 3. Send the written file to a remote destination

| For both JES2 and JES3, some performance degradation
| can be experienced if a backlog of CICS-created data sets is
| allowed to accumulate. You should ensure that procedures
| exist to detect and remedy such situations.

Using the CICS interface to JES

To use the CICS interface to JES, you must code DFHSIT SPOOL=YES.

You must specify RESP or NOHANDLE on the EXEC CICS SPOOLCLOSE, SPOOLOPEN, SPOOLREAD, and SPOOLWRITE commands. RESP bears a one-to-one correspondence with HANDLE CONDITION. If you do not code RESP, your program abends. You can also code the RESP2 option.

NOHANDLE, RESP, and RESP2 are not shown in the syntax boxes or in the option lists for the EXEC CICS SPOOL commands described in this manual. At the end of the description of each spool command, there is a list of RESP values and the RESP2 values that are specific to the CICS-JES interface.

Transactions that process SYSOUT data sets larger than 1000 records, either for INPUT or for OUTPUT, are likely to have a performance impact on the rest of CICS. When you cannot avoid such a transaction, you should carefully evaluate general system performance. You should introduce a pacing mechanism if the effects on the rest of CICS are unacceptable.

All access to a JES spool file must be completed within one logical unit of work. Issuing an EXEC CICS SYNCPOINT command implicitly issues a SPOOLCLOSE command for any open report.

Input

A remote application must route any files intended for a CICS transaction to a specific user name at the system where CICS resides. See Figure 1 on page 7 for an example of a CP command used by a VM system to do this. The figure also shows the EXEC CICS SPOOL commands you use to retrieve the data.

The CICS transaction issues the SPOOLOPEN command, specifying the writer name on the USERID parameter and optionally the class of output within the writer name. The normal response is:

1. No input for this external writer.
2. The single-thread is busy (see below).
3. The file is allocated to you for retrieval, and is identified by the "token" returned by CICS. The token must be included on every SPOOL command for retrieving the data set.

In cases (1) and (2), the transaction should retry the SPOOLOPEN after a suitable interval, by restarting itself.

In case (3), the transaction should then retrieve the file with SPOOLREAD commands, and proceed to SPOOLCLOSE as rapidly as possible to release the path for other users. This is especially important for **input** from JES because the input path is **single-threaded**. When there is more than one

transaction using the interface, their files can be differentiated by using different writer names or different classes within a single writer name. Furthermore, you should ensure that the transactions either terminate or wait for a short period between SPOOLCLOSE and a subsequent SPOOLOPEN. If you do not do this, one transaction can prevent others from using the interface.

JES exits

Both JES2 and JES3 provide a way of screening incoming files. For JES2, the TSO/E Interactive Data Transmission Facility Screening and Notification exit is used. The JES3 equivalent is the Validate Incoming Netdata File exit.

You should review any use your installation makes of these exits to ensure that files that are to be read using the CICS interface to JES are correctly processed.

Output

The transaction program issues SPOOLOPEN to allocate an output data set, specifying a remote NODE and USERID. SPOOLOPEN returns a unique token to the transaction, which must be used in all subsequent SPOOLWRITE and SPOOLCLOSE commands to identify the file being written to. Finally, the transaction issues SPOOLCLOSE to close and deallocate the report, and permit its immediate printing or onward routing by the system spooler. The normal response received from the SPOOLOPEN OUTPUT command is:

The file is allocated to you and identified by the token returned by CICS.
Data can now be written to it.

If the node is a remote MVS system, the data set is queued on the JES spool against the destination userid. The ID of this destination user was specified on the SPOOLOPEN OUTPUT USERID parameter. If the node is a remote VM system, the data is queued in the VM RDR queue for the ID that was specified on the same USERID parameter.

Commands by function

The following is a list of EXEC CICS commands categorized according to the function they perform.

Abend support

ABEND
HANDLE ABEND

APPC basic conversation

GDS ALLOCATE
GDS ASSIGN
GDS CONNECT PROCESS
GDS EXTRACT ATTRIBUTES
GDS EXTRACT PROCESS
GDS FREE
GDS ISSUE ABEND
GDS ISSUE CONFIRMATION
GDS ISSUE ERROR
GDS ISSUE PREPARE
GDS ISSUE SIGNAL
GDS RECEIVE
GDS SEND
GDS WAIT

APPC mapped conversation

ALLOCATE (APPC)
CONNECT PROCESS
CONVERSE (APPC)
EXTRACT ATTRIBUTES (APPC)
EXTRACT PROCESS
FREE (APPC)
ISSUE ABEND
ISSUE CONFIRMATION
ISSUE ERROR
ISSUE PREPARE
ISSUE SIGNAL (APPC)
RECEIVE (APPC)
SEND (APPC)
WAIT CONVID

Authentication

| CHANGE PASSWORD
| SIGNOFF
| SIGNON
| VERIFY PASSWORD

Batch data interchange

ISSUE ABORT
ISSUE ADD
ISSUE END
ISSUE ERASE
ISSUE NOTE
ISSUE QUERY
ISSUE RECEIVE
ISSUE REPLACE
ISSUE SEND
ISSUE WAIT

BMS

PURGE MESSAGE
RECEIVE MAP
RECEIVE PARTN
ROUTE
SEND CONTROL
SEND MAP
SEND PAGE
SEND PARTNSET
SEND TEXT
SEND TEXT MAPPED
SEND TEXT NOEDIT

Built-in functions

BIF DEEDIT

Console support

WRITE OPERATOR

Diagnostic services

DUMP TRANSACTION
ENTER TRACENUM

Environment services

ADDRESS
ADDRESS SET
ASSIGN

Exception support

HANDLE CONDITION
IGNORE CONDITION
POP HANDLE
PUSH HANDLE

File control

DELETE
ENDBR
READ
READNEXT
READPREV
RESETBR
REWRITE
STARTBR
UNLOCK
WRITE

Interval control

ASKTIME
CANCEL
DELAY
FORMATTIME
POST
RETRIEVE
START
WAIT EVENT

Journaling

WAIT JOURNALNUM
WRITE JOURNALNUM

Monitoring

MONITOR

Program control

LINK
LOAD
RELEASE
RETURN
XCTL

Security

QUERY SECURITY

Spool Interface (JES)

SPOOLCLOSE
SPOOLOPEN INPUT
SPOOLOPEN OUTPUT
SPOOLREAD
SPOOLWRITE

Storage control

FREEMAIN
GETMAIN

Syncpoint

SYNCPOINT
SYNCPOINT ROLLBACK

Task control

CHANGE TASK
DEQ
ENQ
SUSPEND
WAIT EXTERNAL
WAITCICS

Temporary storage control

DELETEQ TS
READQ TS
WRITEQ TS

Terminal control

ALLOCATE (LUTYPE6.1)
ALLOCATE (MRO)
BUILD ATTACH (LUTYPE6.1)
BUILD ATTACH (MRO)
CONVERSE (default)
CONVERSE (LUTYPE2/LUTYPE3)
CONVERSE (LUTYPE4)
CONVERSE (LUTYPE6.1)
CONVERSE (MRO)
CONVERSE (SCS)
CONVERSE (System/3)
CONVERSE (System/7)
CONVERSE (2260)
CONVERSE (2741)
CONVERSE (2770)
CONVERSE (2780)
CONVERSE (3270 display)
CONVERSE (3270 logical)
CONVERSE (3600 BTAM)
CONVERSE (3600-3601)
CONVERSE (3600-3614)
CONVERSE (3650 interpreter)
CONVERSE (3650-3270)
CONVERSE (3650-3653)
CONVERSE (3650-3680)
CONVERSE (3735)
CONVERSE (3740)
CONVERSE (3767)
CONVERSE (3770)
CONVERSE (3790 full-function or inquiry)
CONVERSE (3790 3270-display)
EXTRACT ATTACH (LUTYPE6.1)
EXTRACT ATTACH (MRO)
EXTRACT ATTRIBUTES (MRO)
EXTRACT LOGONMSG
EXTRACT TCT
FREE (LUTYPE6.1)
FREE

FREE (MRO)
 HANDLE AID
 ISSUE COPY (3270 display)
 ISSUE COPY (3270 logical)
 ISSUE DISCONNECT
 ISSUE ENDFILE
 ISSUE ENDOUTPUT
 ISSUE EODS
 ISSUE ERASEAUP
 ISSUE LOAD
 ISSUE PASS
 ISSUE PRINT
 ISSUE RESET
 ISSUE SIGNAL (LUTYPE6.1)
 POINT
 RECEIVE (default)
 RECEIVE (LUTYPE2/LUTYPE3)
 RECEIVE (LUTYPE4)
 RECEIVE (LUTYPE6.1)
 RECEIVE (MRO)
 RECEIVE (System/3)
 RECEIVE (System/7)
 RECEIVE (2260)
 RECEIVE (2741)
 RECEIVE (2770)
 RECEIVE (2780)
 RECEIVE (2980)
 RECEIVE (3270 display)
 RECEIVE (3270 logical)
 RECEIVE (3600 BTAM)
 RECEIVE (3600-3601)
 RECEIVE (3600-3614)
 RECEIVE (3650)
 RECEIVE (3735)
 RECEIVE (3740)
 RECEIVE (3767)
 RECEIVE (3770)
 RECEIVE (3790 full-function or inquiry)
 RECEIVE (3790 3270-display)
 SEND (default)
 SEND (LUTYPE2/LUTYPE3)
 SEND (LUTYPE4)
 SEND (LUTYPE6.1)
 SEND (MRO)
 SEND (SCS)
 SEND (System/3)
 SEND (System/7)
 SEND (2260)
 SEND (2741)
 SEND (2770)
 SEND (2880)
 SEND (2980)
 SEND (3270 display)
 SEND (3270 logical)
 SEND (3600 BTAM)
 SEND (3600 pipeline)
 SEND (3600-3601)
 SEND (3600-3614)
 SEND (3650 interpreter)
 SEND (3650-3270)

SEND (3650-3653)
 SEND (3650-3680)
 SEND (3735)
 SEND (3740)
 SEND (3767)
 SEND (3770)
 SEND (3790 full-function or inquiry)
 SEND (3790 SCS)
 SEND (3790 3270-display)
 SEND (3790 3270-printer)
 WAIT SIGNAL
 WAIT TERMINAL

Transient data

DELETEQ TD
 READQ TD
 WRITEQ TD

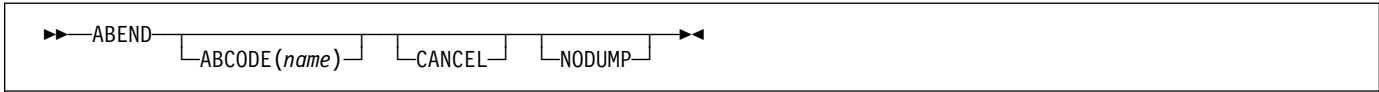
ABEND

ABEND

Function

Terminate task abnormally.

Command syntax



ABEND terminates a task abnormally.

The main storage associated with the terminated task is released; optionally, a transaction dump of this storage can be obtained.

The following example shows how to terminate a task abnormally:

```
EXEC CICS ABEND ABCODE('BCDE')
```

ABEND options

ABCODE(name)

specifies that main storage related to the task that is being terminated is to be dumped. The ABCODE is used as a transaction dumpcode to identify the dump. The name should have four characters and should not contain any leading or imbedded blanks. If ABCODE is not coded, the dump is identified by ????.

Do not start the name with the letter A, because this is reserved for CICS itself.

CANCEL

specifies that exits established by HANDLE ABEND commands are to be ignored. An ABEND CANCEL command cancels all exits at any level in the task (and terminates the task abnormally). If the PL/I STAE execution-time option has been specified, an abnormal termination exit is established by PL/I. This exit is revoked by the CANCEL option.

NODUMP

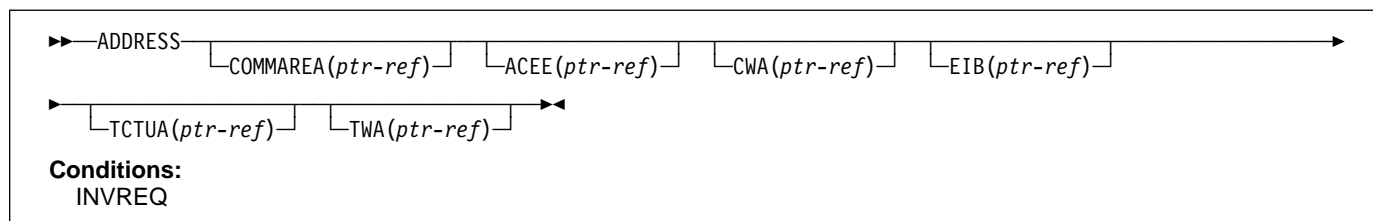
specifies an abend without causing a dump to be taken.

ADDRESS

Function

Get access to CICS storage areas.

Command syntax



Note for dynamic transaction routing

Using ADDRESS with CWA could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

ADDRESS accesses the following areas:

- the communication area available to the invoked program (COMMAREA)
- the access control environment element (ACEE)
- the common work area (CWA)
- the EXEC interface block (EIB)
- the terminal control table user area (TCTUA)
- the transaction work area (TWA).

In assembler language, no more than four options may be specified in one ADDRESS command.

Note: In OS/VS COBOL only, if an ADDRESS command is included in a COBOL program that is to be compiled using the optimization feature, it must be followed by SERVICE RELOAD statements to reload the BLL cell being used.

ADDRESS options

ACEE(ptr-ref)

is a pointer to the access control environment element, the control block that is generated by an external security manager (ESM) when the user signs on. If the user is not signed on, the address of the CICS DFLTUSER's ACEE is returned. If an ACEE does not exist, CICS sets the pointer reference to the null value, X'FF000000'.

Note: Take care when addressing an ACEE in a server program invoked by a distributed program link. The ACEE address returned depends on the link security and may not be the same as the address of the user signed on at the local system.

COMMAREA(ptr-ref)

is the address of the currently defined communication area for the currently executing program. COMMAREA

is used to pass information between application programs. The pointer reference is set to the address of the current COMMAREA. If the COMMAREA does not exist, the pointer reference is set to the null value, X'FF000000'.

In C, you must use ADDRESS COMMAREA to get the address of the communication area, because this is not passed as an argument to a C main function.

CWA(ptr-ref)

provides information available to applications running in a single CICS system. CICS sets the pointer reference to the address of the CWA. If a CWA does not exist, CICS sets the pointer reference to the null value, X'FF000000'.

EIB(ptr-ref)

gets addressability to the EXEC interface block in application routines other than the first invoked by CICS (where addressability to the EIB is provided automatically). If the application program is translated with SYSEIB in the XOPTS parameter list, this option returns the address of the system EIB.

If TASKDATALOC(ANY) is defined on the transaction definition, the address of the data may be above or below the 16MB line.

If TASKDATALOC(BELOW) is defined on the transaction definition, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

C functions must use ADDRESS EIB to get the address of the EXEC interface block, because this address is not passed as an argument to a C main function. You must code an ADDRESS EIB statement at the beginning of each application if you want access to the EIB, or if you

ADDRESS

| are using a command that includes the RESP or RESP2
| option.

TCTUA(ptr-ref)

passes information between application programs, but only if the same terminal is associated with the application programs involved (which can be in different tasks). CICS sets the pointer reference to the address of the TCTUA. If a TCTUA does not exist, CICS sets the pointer reference to the null value, X'FF000000'. The data area contains the address of the TCTUA of the principal facility, not that for any alternate facility that may have been allocated.

TWA(ptr-ref)

passes information between application programs but only if they are in the same task. The pointer reference is set to the address of the TWA. If a TWA does not exist, the pointer reference is set to the null value, X'FF000000'.

If TASKDATALOC(ANY) is defined on the transaction definition, the address of the data may be above or below the 16MB line.

If TASKDATALOC(BELOW) is defined on the transaction definition, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

ADDRESS condition

INVREQ

indicates that the TCTUA option is not allowed in a server program invoked by a distributed program link (RESP2=200).

Default action: terminate the task abnormally.

ADDRESS SET

Function

Set the address of a structure or pointer.

Command syntax

```

ADDRESS SET (data-area) USING (ptr-ref)
ADDRESS SET (ptr-ref) USING (data-area)

```

| The value from the USING option is used to set the
| reference in the SET option.

| COBOL example of ADDRESS SET

To set the address of a structure to a known pointer value:

```

EXEC CICS ADDRESS SET(address of struc)
        USING(ptr)

```

| To set a pointer variable to the address of a structure:

```

EXEC CICS ADDRESS SET(ptr)
        USING(address of struc01)

```

ADDRESS SET options

SET(data-area/ptr-ref)

| sets a pointer reference.

USING(ptr-ref/data-area)

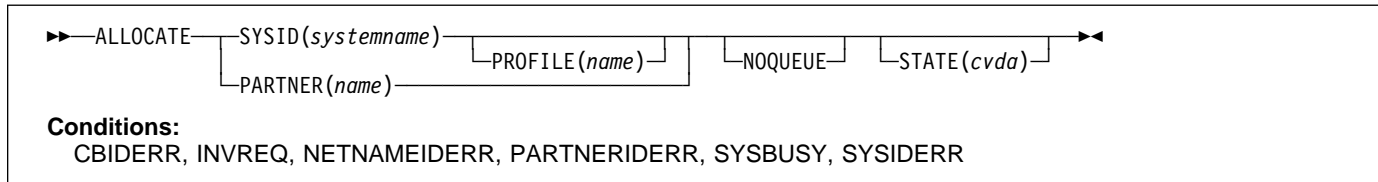
| supplies a pointer value.

ALLOCATE (APPC)

Function

Acquire a session to a remote APPC logical unit for use by an APPC mapped conversation.

Command syntax



ALLOCATE acquires the session and optionally selects a set of session-processing options. CICS makes one of the sessions associated with the named system available to the application program.

CICS returns, in EIBRSRCE in the EIB, the 4-byte CONVID (conversion identifier) that the application program uses in all subsequent commands that relate to the conversation.

If the session requested is not available, the application is suspended until the session does become available. In such a case, the suspension of the application can be prevented by specifying either the NOQUEUE or the NOSUSPEND option. NOSUSPEND is still supported as an equivalent for NOQUEUE, but NOQUEUE is the preferred keyword.

A session is available for allocation only if it is **all** of the following:

- A contention winner
- Already bound
- Not already allocated.

The action taken by CICS if a session is not immediately available depends on whether you specify NOQUEUE (or the equivalent NOSUSPEND option) and also on whether your application has executed a HANDLE command for the SYSBUSY condition. The possible combinations are shown below:

HANDLE for SYSBUSY condition issued

The command is not queued and control is returned immediately to the label specified in the HANDLE command, whether or not you have specified NOQUEUE. pt.No HANDLE issued for SYSBUSY condition pd.If you have specified NOQUEUE, the request is not queued and control is returned immediately to your application program. The SYSBUSY code (X'D3') is set in the EIBRCODE field of the EXEC interface block. You should test this field immediately after issuing the ALLOCATE command.

If you have omitted the NOQUEUE option, CICS queues the request (and your application waits) until a session is available.

ALLOCATE (APPC) options

NOQUEUE

overrides the default action when a SESSBUSY or SYSBUSY condition arises. These conditions indicate that the session requested is not immediately available. The default action is to suspend application execution until the session is available. NOQUEUE inhibits this waiting; control returns immediately to the application program instruction following the command.

PARTNER(name)

specifies the name (8 characters) of a set of definitions that include the names of a remote LU (NETNAME) and a communication profile to be used on the allocated session. You can use this option as an alternative to specifying SYSID and PROFILE explicitly.

PROFILE(name)

specifies the name (1–8 characters) of a set of session-processing options that are to be used during the execution of mapped commands for the session specified in the SYSID option. If you specify SYSID and omit PROFILE, a default profile (DFHCICSA) is selected.

STATE(cvda)

gets the state of the current conversation. The cvda value returned by CICS is ALLOCATED.

SYSID(systemname)

specifies the name (1–4 characters) by which the remote APPC LU is known to this CICS. This option requests that one of the sessions to the named system is to be allocated.

ALLOCATE (APPC) conditions

CBIDERR

occurs if the requested PROFILE cannot be found.
 Default action: terminate the task abnormally.

INVREQ

occurs if the ALLOCATE command is not valid for the device to which it is directed.

Default action: terminate the task abnormally.

| **NETNAMEIDERR**

| occurs if the name specified in the NETNAME parameter
| of the RDO definition for the PARTNER specified on the
| allocate command is invalid.

| Default action: terminate the task abnormally.

| **PARTNERIDERR**

| occurs if the name specified in the PARTNER option is
| not recognized by CICS.

| Default action: terminate the task abnormally.

SYSBUSY

| occurs for one of the following reasons:

- | • The request for a session cannot be serviced
| immediately. This is only possible if the NOQUEUE
| option is set.
- | • The ALLOCATE command is issued when
| persistent session recovery is still in process and
| the sessions needed to satisfy the command are not
| yet recovered.

| Default action: ignore the condition.

SYSIDERR

occurs if CICS is unable to provide the application
program with a suitable session, for one of the following
reasons:

- The name specified in the SYSID option is not
recognized by CICS.
- The mode name derived from the PROFILE option
is not one of the mode names defined for the APPC
system entry.
- All the sessions in the group specified by SYSID
and mode name are out of service, or all sessions
are out of service.
- | • The AID (automatic initiate descriptor) representing
| your ALLOCATE has been canceled.

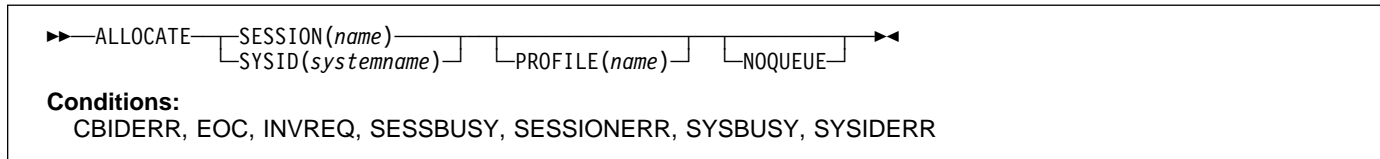
Default action: terminate the task abnormally.

ALLOCATE (LUTYPE6.1)

Function

Acquire a session to a remote LUTYPE6.1 logical unit.

Command syntax



ALLOCATE acquires an alternate facility and optionally selects a set of session-processing options. If SYSID is specified, CICS makes available to the application program one of the sessions associated with the named system. The name of this session can be obtained from EIBRSRCE in the EIB. If SESSION is specified, CICS makes the named session available.

| If the session requested is not available, the application is
 | suspended until the session does become available. In such
 | a case, the suspension of the application can be prevented
 | by specifying either the NOQUEUE or the NOSUSPEND
 | option. NOSUSPEND is still supported as an equivalent for
 | NOQUEUE, but NOQUEUE is the preferred keyword.

ALLOCATE (LUTYPE6.1) options

NOQUEUE

overrides the default action when a SESSBUSY or SYSBUSY condition arises. These conditions indicate that the session requested is not immediately available. The default action is to suspend application execution until the session is available. NOQUEUE inhibits this waiting; control returns immediately to the application program instruction following the command.

PROFILE(name)

specifies the name (1–8 characters) of a set of session-processing options that are to be used during execution of terminal control commands for the session specified in the SYSID or SESSION options. If the PROFILE option is omitted, a default profile (DFHCICSA) is selected.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

SYSID(systemname)

specifies the name (1–4 characters) of a system TCTSE. This option specifies that one of the sessions to the named system is to be allocated.

ALLOCATE (LUTYPE6.1) conditions

CBIDERR

occurs if the requested PROFILE cannot be found.
 Default action: terminate the task abnormally.

EOC

occurs when a request/response unit (RU) is received with the end-of-chain indicator set. Field EIBEOC also contains this indicator.
 Default action: ignore the condition.

INVREQ

occurs when the specified session is already allocated to this task, or the session is an APPC session.
 Default action: terminate the task abnormally.

SESSBUSY

occurs if the request for the specified session cannot be serviced immediately. This is only possible if NOQUEUE is set.
 Default action: queue the request until a session is available.

SESSIONERR

occurs if the name specified in the SESSION option is not that of an LUTYPE6.1 session TCTTE, or if the session cannot be allocated because it is out of service.
 Default action: terminate the task abnormally.

SYSBUSY

| occurs for one of the following reasons:
 |
 | • The request for a session cannot be serviced
 | immediately. This is only possible if NOQUEUE is
 | set.
 |
 | • The ALLOCATE command is issued when
 | persistent session recovery is still in process and
 | the sessions needed to satisfy the command are not
 | yet recovered.
 |
 | Default action: ignore the condition.

SYSIDERR

occurs if CICS is unable to provide the application program with a suitable session, for one of the following reasons:

- The name specified in the SYSID option is not recognized by CICS.
- All sessions are out of service.
- The AID (automatic initiate descriptor) representing your ALLOCATE has been canceled.

Default action: terminate the task abnormally.

ALLOCATE (MRO)

ALLOCATE (MRO)

Function

Acquire an MRO session.

Command syntax

```
▶▶—ALLOCATE—SYSID(systemname)—┬─NOQUEUE┬─STATE(cvda)—▶▶
```

Conditions:

INVREQ, SYSBUSY, SYSIDERR

ALLOCATE acquires an alternate facility. CICS makes available to the application program one of the sessions associated with the system named in the SYSID option. The name of this session can be obtained from EIBRSRCE in the EIB.

(If the PROFILE option is specified, it is ignored on an MRO session. This is useful if you want to code a command that can be used on either an LUTYPE6.1 or MRO session.)

| If the session requested is not available, the application is
| suspended until the session does become available. In such
| a case, the suspension of the application can be prevented
| by specifying the NOQUEUE option.

For more information about MRO and IRC, see the *CICS/ESA Intercommunication Guide*.

ALLOCATE (MRO) options

NOQUEUE

overrides the default action when a SESSBUSY or SYSBUSY condition arises. These conditions indicate that the session requested is not immediately available. The default action is to suspend application execution until the session is available. NOQUEUE inhibits this waiting; control returns immediately to the application program instruction following the command.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* value returned by CICS is ALLOCATED.

SYSID(*systemname*)

specifies the name (1–4 characters) of a system TCTSE. This option specifies that one of the sessions to the named system is to be allocated.

ALLOCATE (MRO) conditions

INVREQ

occurs if an incorrect command has been issued for the LU or terminal in use.

Default action: terminate the task abnormally.

SYSBUSY

occurs if the request for a session cannot be serviced immediately. This is only possible if NOQUEUE is set.

Default action: ignore the condition.

SYSIDERR

occurs if CICS is unable to provide the application program with a suitable session, for one of the following reasons:

- The name specified in the SYSID option is not recognized by CICS.
- All sessions are out of service.
- The AID (automatic initiate descriptor) representing your ALLOCATE has been canceled.

Default action: terminate the task abnormally.

ASKTIME

Function

Request current date and time of day.

Command syntax



ASKTIME updates the date (EIBDATE) and CICS time-of-day clock (EIBTIME) fields in the EIB. These two fields initially contain the date and time when the task started. Refer to Appendix A, “EXEC interface block” on page 343 for details of the EIB.

ASKTIME option

ABSTIME(data-area)

specifies the data area for the time, in packed decimal, since 00:00 on 1 January 1900 (in milliseconds rounded to the nearest hundredth of a second).

| You can use FORMATTIME to change the data into
| other familiar formats.

| For example, after execution of:

| EXEC CICS ASKTIME ABSTIME(utime)

| “utime” contains a value similar in format to
| 002837962864820.

The format of “data-area” is:

```

COBOL: PIC S9(15) COMP-3
C      char data_area[8];
PL/I:  FIXED DEC(15)
ASM:   PL8
  
```

ASSIGN

Function

Request values from outside the application program's local environment.

Command syntax

<p>▶▶—ASSIGN</p> <p>—ABCODE (<i>data-area</i>)</p> <p>—ABDUMP (<i>data-area</i>)</p> <p>—ABPROGRAM (<i>data-area</i>)</p> <p>—ALTSCRNHT (<i>data-area</i>)</p> <p>—ALTSCRNWD (<i>data-area</i>)</p> <p>—APLKYBD (<i>data-area</i>)</p> <p>—APLTEXT (<i>data-area</i>)</p> <p>—APPLID (<i>data-area</i>)</p> <p>—ASRAINTRPT (<i>data-area</i>)</p> <p>—ASRAKEY (<i>cvda</i>)</p> <p>—ASRAPSW (<i>data-area</i>)</p> <p>—ASRAREGS (<i>data-area</i>)</p> <p>—ASRASPC (<i>cvda</i>)</p> <p>—ASRASTG (<i>cvda</i>)</p> <p>—BTRANS (<i>data-area</i>)</p> <p>—CMDSEC (<i>data-area</i>)</p> <p>—COLOR (<i>data-area</i>)</p> <p>—CWALENG (<i>data-area</i>)</p> <p>—DEFSCRNHT (<i>data-area</i>)</p> <p>—DEFSCRNWD (<i>data-area</i>)</p> <p>—DELIMITER (<i>data-area</i>)</p> <p>—DESTCOUNT (<i>data-area</i>)</p> <p>—DESTID (<i>data-area</i>)</p> <p>—DESTIDLENG (<i>data-area</i>)</p> <p>—DSSCS (<i>data-area</i>)</p> <p>—DS3270 (<i>data-area</i>)</p> <p>—EWASUPP (<i>data-area</i>)</p> <p>—EXTDS (<i>data-area</i>)</p> <p>—FACILITY (<i>data-area</i>)</p> <p>—FCI (<i>data-area</i>)</p> <p>—GCHARS (<i>data-area</i>)</p> <p>—GCODES (<i>data-area</i>)</p> <p>—GMMI (<i>data-area</i>)</p> <p>—HIGHLIGHT (<i>data-area</i>)</p> <p>—INITPARM (<i>data-area</i>)</p> <p>—INITPARMLEN (<i>data-area</i>)</p> <p>—INPARTN (<i>data-area</i>)</p> <p>—INVOKINGPROG (<i>data-area</i>)</p> <p>—KATAKANA (<i>data-area</i>)</p> <p>—LANGINUSE (<i>data-area</i>)</p> <p>—LDCMNEM (<i>data-area</i>)</p> <p>—LDCNUM (<i>data-area</i>)</p> <p>—MAPCOLUMN (<i>data-area</i>)</p> <p>—MAPHEIGHT (<i>data-area</i>)</p> <p>—MAPLINE (<i>data-area</i>)</p>	<p>—MAPWIDTH (<i>data-area</i>)</p> <p>—MSRCONTROL (<i>data-area</i>)</p> <p>—NATLANGINUSE (<i>data-area</i>)</p> <p>—NETNAME (<i>data-area</i>)</p> <p>—NEXTTRANSID (<i>data-area</i>)</p> <p>—NUMTAB (<i>data-area</i>)</p> <p>—OPCLASS (<i>data-area</i>)</p> <p>—OPERKEYS (<i>data-area</i>)</p> <p>—OPID (<i>data-area</i>)</p> <p>—OPSECURITY (<i>data-area</i>)</p> <p>—ORGABCODE (<i>data-area</i>)</p> <p>—OUTLINE (<i>data-area</i>)</p> <p>—PAGENUM (<i>data-area</i>)</p> <p>—PARTNPAGE (<i>data-area</i>)</p> <p>—PARTNS (<i>data-area</i>)</p> <p>—PARTNSET (<i>data-area</i>)</p> <p>—PRINSYSID (<i>data-area</i>)</p> <p>—PROGRAM (<i>data-area</i>)</p> <p>—PS (<i>data-area</i>)</p> <p>—QNAME (<i>data-area</i>)</p> <p>—RESSEC (<i>data-area</i>)</p> <p>—RESTART (<i>data-area</i>)</p> <p>—RETURNPROG (<i>data-area</i>)</p> <p>—SCRNHT (<i>data-area</i>)</p> <p>—SCRNWD (<i>data-area</i>)</p> <p>—SIGDATA (<i>data-area</i>)</p> <p>—SOSI (<i>data-area</i>)</p> <p>—STARTCODE (<i>data-area</i>)</p> <p>—STATIONID (<i>data-area</i>)</p> <p>—SYSID (<i>data-area</i>)</p> <p>—TASKPRIORITY (<i>data-area</i>)</p> <p>—TCTUALENG (<i>data-area</i>)</p> <p>—TELLERID (<i>data-area</i>)</p> <p>—TERMCODE (<i>data-area</i>)</p> <p>—TERMPRIORITY (<i>data-area</i>)</p> <p>—TEXTKYBD (<i>data-area</i>)</p> <p>—TEXTPRINT (<i>data-area</i>)</p> <p>—TRANPRIORITY (<i>data-area</i>)</p> <p>—TWALENG (<i>data-area</i>)</p> <p>—UNATTEND (<i>data-area</i>)</p> <p>—USERID (<i>data-area</i>)</p> <p>—USERNAME (<i>data-area</i>)</p> <p>—USERPRIORITY (<i>data-area</i>)</p> <p>—VALIDATION (<i>data-area</i>)</p>
--	---

Conditions:
INVREQ

ASSIGN gets values from outside the local environment of the application program. The data obtained depends on the specified options. Up to sixteen options can be specified in one ASSIGN command.

Where any of the following options apply to terminals or terminal-related data, the reference is always to the principal facility.

If the principal facility is a remote terminal, the data returned is obtained from the local copy of the information; the request

is not routed to the system to which the remote terminal is attached.

Transaction routing is as far as possible transparent to the ASSIGN command. In general, the values returned are the same whether the transaction is local or remote.

For more details on these options, see the *CICS/ESA Intercommunication Guide*.

ASSIGN options

ABCODE(data-area)

returns a 4-character current abend code (abend codes are documented in the *CICS/ESA Messages and Codes* manual). If an abend has not occurred, the variable is set to blanks.

ABDUMP(data-area)

returns a 1-byte value. X'FF' indicates that a dump has been produced and that ABCODE contains an abend code. X'00' indicates either that no dump has been produced, or that ABCODE contains blanks.

ABPROGRAM(data-area)

returns an 8-character name of the failing program for the latest abend.

If the abend originally occurred in a DPL server program running in a remote system, ABPROGRAM returns the DPL server program name.

This field is set to binary zeros if it is not possible to determine the failing program at the time of the abend.

ALTSCRNHT(data-area)

returns the alternate screen height defined for the terminal as a halfword binary variable. If the task is not initiated from a terminal, INVREQ occurs.

ALTSCRNWD(data-area)

returns the alternate screen width defined for the terminal as a halfword binary variable. If the task is not initiated from a terminal, INVREQ occurs.

APLKYBD(data-area)

returns a 1-byte indicator showing whether the terminal keyboard has the APL keyboard feature. X'FF' indicates "yes". X'00' indicates "no". If the task is not initiated from a terminal, INVREQ occurs.

APLTEXT(data-area)

returns a 1-byte indicator showing whether the terminal keyboard has the APL text feature. X'FF' indicates "yes". X'00' indicates "no". If the task is not initiated from a terminal, INVREQ occurs.

APPLID(data-area)

returns an 8-character applid of the CICS system owning the transaction.

If your system is using XRF (that is, XRF=YES has been specified in the system initialization parameters), the value returned is the **generic** applid (that is, the applid

that identifies the active and alternate CICS systems). An application program is unaffected by a takeover from the active to the alternate.

ASRAINTRPT(data-area)

returns an 8-character program status word (PSW) containing interrupt information at the point when the latest abend with a code of ASRA, ASRB, ASRD, or AICA occurred.

The field contains binary zeros if no ASRA, ASRB, ASRD, or AICA abend occurred during the execution of the issuing transaction, or if the abend originally occurred in a remote DPL server program.

ASRAKEY(cvda)

returns the execution key at the time of the last ASRA, ASRB, AICA, or AEYD, abend, if any. The CVDA values on the ASRAKEY option are as follows:

CICSEXECKEY is returned if the task was executing in CICS-key at the time of the last ASRA, ASRB, AICA, or AEYD abend. Note that all programs execute in CICS key if CICS subsystem storage protection is not active.

USEREXECKEY

is returned if the task was executing in user-key at the time of the last ASRA, ASRB, AICA, or AEYD abend.

NONCICS

is returned if the execution key at the time of the last abend was not one of the CICS keys; that is, not key 8 or key 9.

NOTAPPLIC

is returned if there has not been an ASRA, ASRB, AICA, or AEYD abend.

ASRAPSW(data-area)

returns an 8-character program status word (PSW) at the point when the latest abend with a code of ASRA, ASRB, ASRD, or AICA occurred.

The field contains binary zeros if no ASRA, ASRB, ASRD, or AICA abend occurred during the execution of the issuing transaction, or if the abend originally occurred in a remote DPL server program.

ASRAREGS(data-area)

returns the contents of general registers 0–15 at the point when the latest ASRA, ASRB, ASRD, or AICA abend occurred.

The contents of the registers are returned in the data area (64 bytes long) in the order 0, 1, ..., 14, 15.

Note that the data area is set to binary zeros if no ASRA, ASRB, ASRD, or AICA abend occurred during the execution of the issuing transaction or the abend originally occurred in a remote DPL server program.

ASSIGN

ASRASPC(cvda)

returns the type of space in control at the time of the last ASRA, ASRB, AICA, or AEYD, abend, if any. The CVDA values on the ASRASPC option are:

- | | |
|------------------|--|
| SUBSPACE | is returned if the task was executing in either its own subspace or the common subspace at the time of the last ASRA, ASRB, AICA, or AEYD abend. |
| BASESPACE | is returned if the task was executing in the base space at the time of the last ASRA, ASRB, AICA, or AEYD abend. Note that all tasks execute in base space if transaction isolation is not active. |
| NOTAPPLIC | is returned if there has not been an ASRA, ASRB, AICA, or AEYD abend. |

ASRASTG(cvda)

returns the type of storage being addressed at the time of the last ASRA or AEYD, abend, if any. The CVDA values on the ASRASTG option are:

- | | |
|------------------|---|
| CICS | is returned if the storage being addressed is CICS-key storage. This can be in one of the CICS dynamic storage areas (CDSA or ECDSA), or, in one of the read-only dynamic storage areas (RDSA or ERDSA) when CICS is running with the NOPROTECT option on the RENTPGM system initialization parameter or when storage protection is not active. |
| USER | is returned if the storage being addressed is user-key storage in one of the user dynamic storage areas (UDSA or EUDSA). |
| READONLY | is returned if the storage being addressed is read-only storage in one of the read-only dynamic storage areas (RDSA or ERDSA) when CICS is running with the PROTECT option on the RENTPGM system initialization parameter. |
| NOTAPPLIC | is returned if: <ul style="list-style-type: none">• There is no ASRA or AEYD abend found for this task• The affected storage in an abend is not managed by CICS• The ASRA abend is not caused by an 0C4 abend• An ASRB or AICA abend has occurred since the last ASRA or AEYD abend. |

BTRANS(data-area)

returns a 1-byte indicator showing whether the terminal is defined as having the background transparency

capability (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

CMDSEC(data-area)

returns a 1-byte indicator showing whether command security checking has been defined for the current task. (X for "yes", blank for "no".)

COLOR(data-area)

returns a 1-byte indicator showing whether the terminal is defined as having the extended color capability (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

CWALENG(data-area)

returns a halfword binary field indicating the length of the CWA. If no CWA exists, a zero length is returned.

DEFSCRNHT(data-area)

returns a halfword binary variable that contains the default screen height defined for the terminal. If the task is not initiated from a terminal, INVREQ occurs.

DEFSCRNWD(data-area)

returns a halfword binary variable that contains the default screen width defined for the terminal. If the task is not initiated from a terminal, INVREQ occurs.

DELIMITER(data-area)

returns a 1-byte data-link control character for a 3600, and can be:

- X'80'** Input ended with end-of-text (ETX)
- X'40'** Input ended with end-of-block (ETB)
- X'20'** Input ended with inter-record separator (IRS)
- X'10'** Input ended with start of header (SOH)
- X'08'** Transparent input.

If the task is not initiated from a terminal, INVREQ occurs.

DESTCOUNT(data-area)

returns a halfword binary field. This option has two uses:

1. Following a BMS ROUTE command, it shows that the value required is the number of different terminal types in the route list, and hence the number of overflow control areas that may be required.
2. Within BMS overflow processing, it shows that the value required is the relative overflow control number of the destination that has encountered overflow. If this option is specified when overflow processing is not in effect, the value obtained is meaningless. If no BMS commands have been issued, INVREQ occurs.

DESTID(data-area)

returns an 8-byte identifier of the outboard destination, padded with blanks on the right to eight characters. If this option is specified before a batch data interchange command has been issued in the task, INVREQ occurs.

DESTIDLENG(data-area)

returns a halfword binary length of the destination identifier obtained by DESTID. If this option is specified before a batch data interchange command has been issued in the task, INVREQ occurs.

DSSCS(data-area)

returns a 1-byte indicator showing whether the principal facility is a basic SCS data stream device. (X'FF' for "yes", or X'00' for "no".)

DS3270(data-area)

returns a 1-byte indicator showing whether the principal facility is a 3270 data stream device. (X'FF' for "yes", or X'00' for "no".)

EWASUPP(data-area)

returns a 1-byte indicator showing whether Erase Write Alternative is supported. (X'FF' for "yes", X'00' for "no".)

EXTDS(data-area)

returns a 1-byte indicator showing whether the terminal accepts the 3270 extended data stream. Extended data stream capability is required for a terminal that supports the query feature, color, extended highlighting, programmed symbols or validation. A terminal that accepts the query structured field command also has this indicator set. If extended data stream is on, the device supports the write structured field COMMAND and Outbound Query Structured field.

(For guidance information about query structured fields, see the *CICS/ESA 3270 Data Stream Device Guide*.)

If the task is not initiated from a terminal, INVREQ occurs.

FACILITY(data-area)

returns a 4-byte identifier of the facility that initiated the transaction; that is the facility control address as defined using the RDO option TYPETERM. If this option is specified, and there is no allocated facility, INVREQ occurs.

Note: You should always use the QNAME option (described on page 27) to get the name of the transient data intrapartition queue whose trigger level caused the transaction to be initiated.

FCI(data-area)

returns a 1-byte facility control indicator, see "ASSIGN FCI" on page 351. This indicates the type of facility associated with the transaction; for example, X'01' indicates a terminal or logical unit. The obtained value is always returned.

GCHARS(data-area)

returns a halfword binary graphic character set global identifier (the GCSGID). The value is a number in the range 1 through 65534 representing the set of graphic characters that can be input or output at the terminal. If the task is not initiated from a terminal, INVREQ occurs.

GCODES(data-area)

returns a halfword binary code page global identifier (the CPGID). The value is a number in the range 1 through 65534 representing the EBCDIC code page defining the code points for the characters that can be input or output at the terminal. If the task is not initiated from a terminal, INVREQ occurs.

GMMI(data-area)

returns a 1-byte indicator showing whether a "good morning" message applies to the terminal associated with the running transaction. (X'FF' for "yes", or X'00' for "no".) If this option is specified and the current task is not associated with a terminal, the INVREQ condition occurs.

HIGHLIGHT(data-area)

returns a 1-byte indicator showing whether the terminal is defined as having the extended highlight capability (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

INITPARM(data-area)

returns the 60-character data-area containing any initialization parameters specified for the program in the INITPARM system initialization parameter. If there are no parameters for the program, the area is filled with binary zeros. (See the *CICS/ESA System Definition Guide* for further information about the INITPARM option.)

INITPARMLEN(data-area)

returns a halfword binary length of the INITPARM. If there is no parameter for it, INITPARMLEN contains binary zeros.

INPARTN(data-area)

returns the 1- or 2-character name of the most recent input partition. If no map has yet been positioned, or if BMS routing is in effect, or if the task is not initiated from a terminal, INVREQ occurs.

INVOKINGPROG(data-area)

returns the 8-character name of the application program that used the LINK or XCTL command to link or transfer control to the current program.

If you issue the ASSIGN INVOKINGPROG command in a remote program that was invoked by a distributed program link (DPL) command, CICS returns the name of the program that issued the DPL command.

If you issue the ASSIGN INVOKINGPROG command in an application program at the highest level, CICS returns eight blanks.

If you issue the ASSIGN INVOKINGPROG command in a

Apar 62320

Documentation for Apar 62320 added 15 Nov 1994 (TUCKER)

ASSIGN

| user-replaceable module or a program list table program,
| CICS returns eight blanks.

— **Apar 67276** —

Documentation for Apar 67276 added 2 Mar 1995
(TUCKER)

If you issue the ASSIGN INVOKINGPROG command
from a global user exit, task-related exit, or application
program linked to from such an exit, CICS returns the
name of the most recent invoking program that was not
a global user exit or task-related user exit.

KATAKANA(data-area)

returns a 1-byte indicator showing whether the principal facility supports katakana (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

— **Apar 89178** —

Documentation for Apar 89178 added 16 Oct 1996
(DONOGHUE)

LANGINUSE(data-area)

returns a 3-byte mnemonic code showing the language
in use. 3-byte mnemonic has a 1:1 correspondence
with the 1-byte NATLANGINUSE option. See
Appendix I, "National language codes" on page 373 for
possible values of the code.

LDCMNEM(data-area)

returns a 1-byte logical device code (LDC) mnemonic of the destination that has encountered overflow. If this option is specified when overflow processing is not in effect, the value obtained not significant. If no BMS commands have been issued, INVREQ occurs.

LDCNUM(data-area)

returns a 1-byte LDC numeric value of the destination that has encountered overflow. This indicates the type of the LDC, such as printer or console. If this option is specified when overflow processing is not in effect, the value obtained is not significant.

MAPCOLUMN(data-area)

returns a halfword binary number of the column on the display containing the origin of the most recently positioned map. If no map has yet been positioned, or if BMS routing is in effect, or if the task is not initiated from a terminal, INVREQ occurs.

MAPHEIGHT(data-area)

returns a halfword binary height of the most recently positioned map. If no map has yet been positioned, or if BMS routing is in effect, or if the task is not initiated from a terminal, INVREQ occurs.

MAPLINE(data-area)

returns a halfword binary number of the line on the display containing the origin of the most recently positioned map. If no map has yet been positioned, or if BMS routing is in effect, or if the task is not initiated from a terminal, INVREQ occurs.

MAPWIDTH(data-area)

returns a halfword binary width of the most recently positioned map. If no map has yet been positioned, or if BMS routing is in effect, or if the task is not initiated from a terminal, INVREQ occurs.

MSRCONTROL(data-area)

returns a 1-byte indicator showing whether the terminal supports magnetic slot reader (MSR) control (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

NATLANGINUSE(data-area)

returns a 1-byte mnemonic code showing the national
language associated with the USERID for the current
task (which could be the default USERID). Refer to the
SIGNON command for an explanation of how this value
is derived. See Appendix I, "National language codes"
on page 373 for possible values of the code.
| (NATLANGINUSE does not show the system default
| language as specified on the NATLANG system
| initialization parameter.)

NETNAME(data-area)

returns the 8-character name of the logical unit in the VTAM network. If the task is not initiated from a terminal, INVREQ occurs. If the principal facility is not a local terminal, the value returned is a null string.

NEXTTRANSID(data-area)

returns the 4-character next transaction identifier as set by SET NEXTTRANSID or RETURN TRANSID. It returns blanks if there are no more transactions.

NUMTAB(data-area)

returns a 1-byte number of the tabs required to position the print element in the correct passbook area of the 2980. If the task is not initiated from a terminal, INVREQ occurs.

OPCLASS(data-area)

returns, in a 24-bit string, the operator class used by BMS for routing terminal messages, as defined in the CICS segment.

OPERKEYS(data-area)

is accepted for compatibility with previous releases.

OPID(data-area)

returns, in a 24-bit string, the operator identification used by BMS for routing terminal messages, as defined in the CICS segment.

— **Apar 87704** —

Documentation for Apar 87704 added 19/12/96

If the task is initiated from a remote terminal, the OPID
returned by this command is not necessarily that
associated with the user that is signed on at the remote
terminal. If you wish to know the OPID of the
signed-on user, you should use the INQUIRE

```
# TERMINAL system programming command, which is
# described in the System Programming Reference.
#
#   Apar 92639
#   - Following line removed, no longer true. If the
#   task is not initiated from a terminal, INVREQ occurs.
#
#   Apar PQ01572
#   The following paragraph was added by Apar
#   PQ01572 24/03/97
#
# The OPID may also be different from that of the user
# currently signed on, if it has been changed with the SET
# TERMINAL command.
```

OPSECURITY(data-area)

is accepted for compatibility with previous releases.

ORGABCODE(data-area)

returns as a 4-byte original abend code in cases of repeated abends.

OUTLINE(data-area)

returns a 1-byte indicator showing whether the terminal is defined as having the field outlining capability (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

PAGENUM(data-area)

returns a halfword binary current page number for the destination that has encountered an overflow. If this option is specified when overflow processing is not in effect, the value obtained is meaningless. If no BMS commands have been issued, INVREQ occurs.

PARTNPAGE(data-area)

returns a 2-byte name of the partition that most recently caused page overflow. If no BMS commands have been issued, INVREQ occurs.

PARTNS(data-area)

returns a 1-byte indicator showing whether the terminal supports partitions (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

PARTNSET(data-area)

returns the name (1–6 characters) of the application partition set. A blank value is returned if there is no application partition set. If the task is not initiated from a terminal, INVREQ occurs.

PRINSYSID(data-area)

returns the 4-character name by which the other system is known in the local system; that is, the CONNECTION definition that defines the other system. For a single-session APPC device defined by a terminal definition, the returned value is the terminal identifier.

This only applies when the principal facility is one of the following:

- An MRO session to another CICS system

- An LU6.1 session to another CICS or IMS/VS system
- An APPC session to another CICS system, or to another APPC system or device.

If the principal facility is not an MRO, LU6.1, or APPC session, or if the task has no principal facility, INVREQ occurs.

Note: Special considerations apply generally when transaction routing. In particular an ASSIGN PRINSYSID command cannot be used in a routed transaction to find the name of the terminal-owning region. (See the *CICS/ESA Intercommunication Guide* for more information about transaction routing.)

PROGRAM(data-area)

returns an 8-character name of the currently running program.

PS(data-area)

returns a 1-byte indicator showing whether the terminal is defined as having the programmed symbols capability (X'FF') or not (X'00'). If the task is not initiated from a terminal, INVREQ occurs.

QNAME(data-area)

returns a 4-character name of the transient data intrapartition queue that caused this task to be initiated by reaching its trigger level. If the task is not initiated by automatic transaction initiation (ATI), INVREQ occurs.

RESSEC(data-area)

returns a 1-byte indicator showing whether resource security checking has been defined for the transaction running. (X for "yes", blank for "no".)

RESTART(data-area)

returns a 1-byte indicator showing whether a restart of the task (X'FF'), or a normal start of the task (X'00'), has occurred.

RETURNPROG(data-area)

returns the 8-character name of the program to which control is to be returned when the current program has finished executing. The values returned depend on how the current program was given control, as follows:

- If the current program was invoked by a LINK command, including a distributed program link, RETURNPROG returns the same name as INVOKINGPROG.
- If the current program was invoked by an XCTL command, RETURNPROG returns the name of the application program in the chain that last issued a LINK command. For example:

```
Program A links to program B
Program B links to program C
Program C transfers control to program D
Program D issues an ASSIGN RETURNPROG command,
and CICS returns the name of Program B.
```

ASSIGN

If the program that invoked the current program with an XCTL command is at the highest level, CICS returns eight blanks.

- If the ASSIGN RETURNPROG command is issued in the program at the top level, CICS returns eight blanks.
- If the ASSIGN RETURNPROG command is issued in a

Apar 62320

Documentation for Apar 62320 added 15 Nov 1994 (TUCKER)

user-replaceable module, or a program list table program, CICS returns eight blanks.

Apar 67276

Documentation for Apar 67276 added 2 Mar 1995 (TUCKER)

- If the ASSIGN RETURNPROG is issued in a global user exit, task-related exit, or application program linked to from such an exit, CICS returns the name of the program that control is returned to when all intermediate global user exit and task-related user exit programs have completed.

SCRNHT(data-area)

returns a halfword binary variable that contains the height of the 3270 screen defined for the current task. If the task is not initiated from a terminal, INVREQ occurs.

SCRNWD(data-area)

returns a halfword binary variable that contains the width of the 3270 screen defined for the current task. If the task is not initiated from a terminal, INVREQ occurs.

SIGDATA(data-area)

returns a 4-byte character string containing the inbound signal data received from a logical unit. If the task is not initiated from a terminal, INVREQ occurs.

SOSI(data-area)

returns a 1-byte indicator showing whether the terminal is defined as having the mixed EBCDIC/DBCS fields capability (X'FF') or not (X'00'). The DBCS subfields within an EBCDIC field are delimited by SO (shift-out) and SI (shift-in) characters. If the task is not initiated from a terminal, INVREQ occurs.

STARTCODE(data-area)

returns a 2-byte indicator showing how the transaction issuing the request was started. It can have the following values:

Code Transaction started by

- D** A distributed program link (DPL) request. The program cannot issue I/O requests against its principal facility, nor can it issue any syncpoint requests.

DS A distributed program link (DPL) request, as in code D, with the exception that the program can issue syncpoint requests.

QD Transient data trigger level.

S START command without data.

SD START command with data.

SZ FEPI START command.

TD Terminal input.

U User-attached task.

STATIONID(data-area)

returns a 1-byte station identifier of a 2980. If the task is not initiated from a terminal, INVREQ occurs.

SYSID(data-area)

returns the 4-character name given to the local CICS system. This value may be specified in the SYSID option of a file control, interval control, temporary storage, or transient data command, in which case the resource to be accessed is assumed to be on the local system.

TASKPRIORITY(data-area)

returns a halfword binary current priority of the issuing task (0–255). When the task is first attached, this is the sum of the user, terminal, and transaction priorities, but this value can be changed during execution by a CHANGE TASK command.

TCTUALENG(data-area)

returns a halfword binary length of the terminal control table user area (TCTUA). If no TCTUA exists, a zero length is returned.

TELLERID(data-area)

returns a 1-byte teller identifier of a 2980. If the task is not initiated from a terminal, INVREQ occurs.

TERMCODE(data-area)

returns a 2-byte code giving the type and model number of the terminal associated with the task.

The first byte is a code identifying the terminal type, derived from the DEVICE attribute of the TYPETERM RDO attribute (described in the *CICS/ESA Resource Definition Guide*). The second byte is a single-character model number as specified in the TERMMODEL attribute.

The meanings of the type codes are given in Appendix B, “Codes returned by ASSIGN” on page 351.

TERMPRIORITY(data-area)

returns a halfword binary terminal priority (0–255).

TEXTKYBD(data-area)

returns a 1-byte indicator showing whether the principal facility supports TEXTKYBD. (X'FF' for “yes”, or X'00' for “no”.) If the task is not initiated from a terminal, INVREQ occurs.

TEXTPRINT(data-area)

returns a 1-byte indicator showing whether the principal facility supports TEXTPRINT. (X'FF' for "yes", or X'00' for "no".) If the task is not initiated from a terminal, INVREQ occurs.

TRANPRIORITY(data-area)

returns a halfword binary transaction priority (0–255).

TWALENG(data-area)

returns a halfword binary length of the transaction work area (TWA). If no TWA exists, a zero length is returned.

UNATTEND(data-area)

returns a 1-byte indicator showing whether the mode of operation of the terminal is unattended, that is to say no person is actually attending the terminal. These indicators are X'FF' for unattended and X'00' for attended. If the task is not initiated from a terminal, INVREQ occurs.

USERID(data-area)

returns an 8-byte userid of the signed-on user. If no user is explicitly signed on, CICS returns the default userid. Special considerations apply if you are using an intercommunication environment. See the *CICS/ESA Intercommunication Guide* for more information about the ASSIGN command for LUTYPE6.1, APPC, and MRO.

Note: If you want to test if the user is explicitly signed on, use the UNATTEND option.

USERNAME(data-area)

returns a 20-character name of the user obtained from the external security manager (ESM).

USERPRIORITY(data-area)

returns a halfword binary operator priority (0–255).

VALIDATION(data-area)

returns a 1-byte indicator showing whether the terminal is defined as having the validation capability (X'FF') or not (X'00'). Validation capability consists of the mandatory fill, mandatory enter, and trigger attributes. If the task is not initiated from a terminal, INVREQ occurs.

ASSIGN condition**INVREQ**

occurs in any of the following situations:

- The task does not have a signed-on user (RESP2=1).
- No BMS command has yet been issued, BMS routing is in effect, or no map has yet been positioned (RESP2=2).
- No batch data interchange (BDI) command has yet been issued (RESP2=3).
- The task is not initiated by automatic transaction initiation (ATI) (RESP2=4).
- The task is not associated with a terminal; or the task has no principal facility; or the principal facility is not an MRO, LU6.1, or APPC session (RESP2=5).
- Command syntax options are not allowed in a server program invoked by a distributed program link (RESP2=200).

Default action: terminate the task abnormally.

BIF DEEDIT

Function

Deediting (built-in function).

Command syntax

```
▶—BIF DEEDIT—FIELD(data-area)—LENGTH(data-value)—▶
```

Conditions:
LENGERR

BIF DEEDIT provides the built-in function DEEDIT. It specifies that alphabetic and special characters are removed from an EBCDIC data field, and the remaining digits right-aligned and padded to the left with zeros as necessary.

If the field ends with a minus sign or a carriage-return (CR), a negative zone (X'D') is placed in the rightmost (low-order) byte.

If the zone portion of the rightmost byte contains one of the characters X'A' through X'F', and the numeric portion contains one of the hexadecimal digits X'0' through X'9', the rightmost byte is returned unaltered (see the example). This permits the application program to operate on a zoned numeric field. The returned value is in the field that initially contained the unedited data.

Note that a 1-byte field is returned unaltered, no matter what the field contains.

Example

```
EXEC CICS BIF DEEDIT
      FIELD(CONTG)
      LENGTH(9)
```

This removes all characters other than digits from CONTG, a 9-byte field, and returns the edited result in that field to the application program. Two examples of the contents of CONTG before and after execution of the command are:

Original value	Returned value
14-6704/B	00146704B
\$25.68	000002568

Note that a decimal point is an EBCDIC special character and as such is removed.

BIF DEEDIT options

FIELD(*data-area*)

specifies the field to be edited.

LENGTH(*data-value*)

specifies the field length in bytes.

BIF DEEDIT conditions

LENGERR

occurs if the LENGTH value is less than 1.

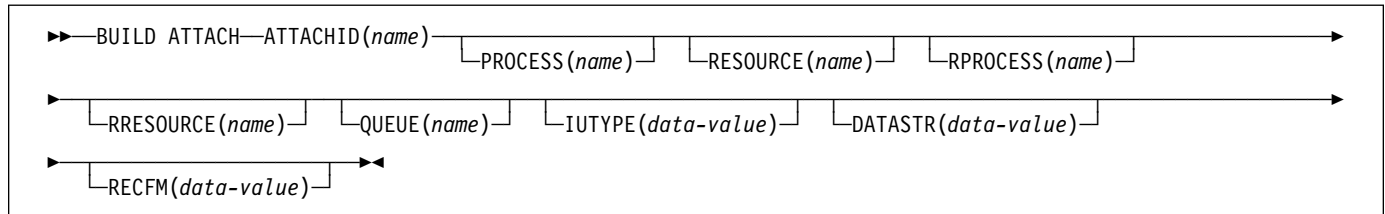
Default action: terminate the task abnormally.

BUILD ATTACH (LUTYPE6.1)

Function

Specify values for an LUTYPE6.1 attach header.

Command syntax



BUILD ATTACH (LUTYPE6.1) specifies a set of values to be placed in the named attach header control block. This control block contains values that are to be sent in an LUTYPE6.1 attach FMH (Function Management Header) that is constructed by CICS, and is sent only when a SEND ATTACHID or CONVERSE ATTACHID command is executed. The specified values override existing values in the control block; unspecified values are set to default values.

BUILD ATTACH (LUTYPE6.1) options

ATTACHID(name)

specifies that the set of values is to be placed in an attach header control block identified by the specified name (1–8 characters).

DATASTR(data-value)

corresponds to the data stream profile field, ATTDSP, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the data stream profile field in an attach FMH. For most CICS applications, the option can be omitted.

Information about CICS to IMS communication is given in the *CICS/ESA Intercommunication Guide*. For details of communication between a CICS system and any other subsystem, including details of structured fields and logical record management, refer to documentation supplied by the subsystem about how to use the data stream profile field in an attach FMH.

The “data-value” is a halfword binary. Only the low-order byte is used. The SNA-defined meanings of the bits are as follows:

- 0–7 reserved - must be set to zero
- 8–11 0000 - user-defined
 - 1111 - SCS data stream
 - 1110 - 3270 data stream
 - 1101 - structured field
 - 1100 - logical record management
- 12–15 defined by the user if bits 8–11 are set to 0000; otherwise reserved (must be set to zero)

A value of “structured field” indicates that chains begin with four bytes of data that are used to interpret the following data: overall length (2 bytes), class identifier (1 byte), and subclass identifier (1 byte). A value of “logical record management” indicates that chains can be split into separate fields by the data receiver.

If the option is omitted, a value of “user-defined” is assumed.

IUTYPE(data-value)

corresponds to the interchange unit field, ATTIU, in an LUTYPE6.1 attach FMH.

The “data-value” is a halfword binary. Only the low-order 7 bits are used. The SNA-defined meanings of the bits are as follows:

- 0–10 reserved - must be set to zero
- 11 0 - not end of multichain interchange unit
 - 1 - end of multichain interchange unit
- 12,13 reserved - must be set to zero
- 14,15 00 - multichain interchange unit
 - 01 - single-chain interchange unit
 - 10 - reserved
 - 11 - reserved

If the option is omitted, values of “not end of multichain interchange unit” and “multichain interchange unit” are assumed.

BUILD ATTACH (LUTYPE6.1)

PROCESS(name)

corresponds to the process name, ATTDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, a transaction running in one system can acquire a session to the second system and can identify the transaction to be attached; in the second system, the identification is carried in the first chain of data sent across the session.

In general, the first four bytes of data identify the transaction to be attached. However an attach FMH, identifying the transaction to be attached, can be built and sent; the PROCESS option is used to specify the transaction name. (Note that the receiving CICS system uses just the first four bytes of the process name as a transaction name.)

No significance is attached by CICS to process names in attach FMHs sent in chains of data other than the first.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the process name field in an attach FMH.

QUEUE(name)

corresponds to the queue name, ATTDQN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the queue name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the queue name field in an attach FMH.

RECFM(data-value)

corresponds to the deblocking algorithm field, ATTDDBA, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the deblocking algorithm field in an attach FMH. For most CICS applications, the option can be omitted.

The "data-value" is a halfword binary value. Only the low-order byte is used. The SNA-defined meanings of the bits are as follows:

0-7	reserved - must be set to zero
8-15	X'00' - reserved
	X'01' - variable-length variable-blocked
	X'02' - reserved
	X'03' - reserved
	X'04' - chain of RUs
	X'05' through X'FF' - reserved

If the option is omitted, a value of "chain of RUs" is assumed.

RESOURCE(name)

corresponds to the resource name, ATTPRN, in an LUTYPE6.1 attach FMH.

RPROCESS(name)

corresponds to the return-process name, ATTRDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return-process name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return-process name field in an attach FMH.

RRESOURCE(name)

corresponds to the return-resource name, ATTRPRN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return-resource name in an attach FMH.

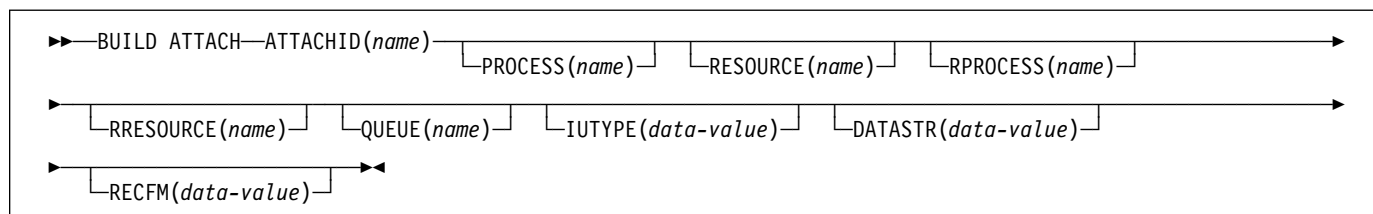
For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return-resource name field in an attach FMH.

BUILD ATTACH (MRO)

Function

Specify values for an MRO attach header.

Command syntax



BUILD ATTACH (MRO) specifies a set of values to be placed in the named attach header control block. This control block contains values that are to be sent in an MRO attach FMH (Function Management Header) that is constructed by CICS, and is sent only when a SEND ATTACHID or CONVERSE ATTACHID command is executed. The specified values override existing values in the control block; unspecified values are set to default values.

For more information about MRO and IRC, see the *CICS/ESA Intercommunication Guide*.

BUILD ATTACH (MRO) options

ATTACHID(name)

specifies that the set of values is to be placed in an attach header control block identified by the specified name (1–8 characters).

DATASTR(data-value)

corresponds to the data stream profile field, ATTDSP, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the data stream profile field in an attach FMH. For most CICS applications, the option can be omitted.

The “data-value” is a binary halfword. Only the low-order byte is used. The SNA-defined meanings of the bits are as follows:

- | | |
|-------|--|
| 0–7 | reserved - must be set to zero |
| 8–11 | 0000 - user-defined |
| | 1111 - SCS data stream |
| | 1110 - 3270 data stream |
| | 1101 - structured field |
| | 1100 - logical record management |
| 12–15 | defined by the user if bits 8–11 are set to 0000; otherwise reserved (must be set to zero) |

A value of “structured field” indicates that chains begin with four bytes of data that are used to interpret the

following data; overall length (2 bytes), class identifier (1 byte), and subclass identifier (1 byte). A value of “logical record management” indicates that chains can be split into separate fields by the data receiver.

If the option is omitted, a value of “user-defined” is assumed.

IUTYPE(data-value)

corresponds to the interchange unit field, ATTIU, in an LUTYPE6.1 attach FMH.

The “data-value” is a halfword binary. Only the low-order 7 bits are used. The SNA-defined meanings of the bits are as follows:

- | | |
|-------|--|
| 0–10 | reserved - must be set to zero |
| 11 | 0 - not end of multichain interchange unit |
| | 1 - end of multichain interchange unit |
| 12,13 | reserved - must be set to zero |
| 14,15 | 00 - multichain interchange unit |
| | 01 - single chain interchange unit |
| | 10 - reserved |
| | 11 - reserved |

If the option is omitted, values of “not end of multichain interchange unit” and “multichain interchange unit” are assumed.

PROCESS(name)

corresponds to the process name, ATTDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, a transaction running in one system can acquire a session to the second system and can identify the transaction to be attached; in the second system the identification is carried in the first chain of data sent across the session. In general, the first four bytes of data identify the transaction to be attached. However an attach FMH, identifying the transaction to be attached, can be built and sent; the PROCESS option is used to specify the transaction name. (Note that the receiving CICS system uses just the first four bytes of the process name as a transaction name.)

BUILD ATTACH (MRO)

No significance is attached by CICS to process names in attach FMHs sent in chains of data other than the first.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the process name field in an attach FMH.

QUEUE(name)

corresponds to the queue name, ATTDQN, in an attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the queue name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the queue name field in an attach FMH.

RECFM(data-value)

corresponds to the deblocking algorithm field, ATTDDBA, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the deblocking algorithm field in an attach FMH. For most CICS applications, the option can be omitted.

The “data-value” is a halfword binary value. Only the low-order 8 bits are used. The SNA-defined meanings of the bits are as follows:

0-7	reserved - must be set to zero
8-15	X'00' - reserved
	X'01' - variable-length variable-blocked
	X'02' - reserved
	X'03' - reserved
	X'04' - chain of RUs
	X'05' to X'FF' - reserved

If the option is omitted, a value of “chain of RUs” is assumed.

RESOURCE(name)

corresponds to the resource-name, ATTPRN, in an LUTYPE6.1 attach FMH.

RPROCESS(name)

corresponds to the return-process name, ATTRDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return-process name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return-process name field in an attach FMH.

RRESOURCE(name)

corresponds to the return-resource name, ATTRPRN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return-resource name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return-resource name field in an attach FMH.

CANCEL

Function

Cancel interval control requests.

Command syntax

```

▶ CANCEL [REQID(name)] [TRANSID(name)] [SYSID(systemname)] ▶▶

```

Conditions:

ISCINVREQ, NOTAUTH, NOTFND, SYSIDERR

Note for dynamic transaction routing

Using CANCEL with REQID (of a POST, DELAY, or START) could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

CANCEL cancels a previously issued DELAY, POST, or START command. If you include the SYSID option, the command is shipped to a remote system. If you omit SYSID, the TRANSID option, if present, indicates where the command is to be executed. The effect of the cancelation varies depending on the type of command being canceled, as follows:

- A DELAY command can be canceled only before it has expired, and only by a task other than the task that issued the DELAY command (which is suspended for the duration of the request). The REQID used by the suspended task must be specified. The effect of the cancelation is the same as an early expiration of the original DELAY. That is, the suspended task becomes dispatchable as though the original expiration time had been reached.
- When a POST command issued by the same task is to be canceled, no REQID need be specified. Cancelation can be requested either before or after the original request has expired. The effect of the cancelation is as if the original request had never been made.
- When a POST command issued by another task is to be canceled, the REQID of that command must be specified. The effect of the cancelation is the same as an early expiration of the original POST request. That is, the timer event control area for the other task is posted as though the original expiration time had been reached.
- When a START command is to be canceled, the REQID of the original command must be specified. The effect of the cancelation is as if the original command had never been made. The cancelation is effective only before the original command has expired.

CANCEL options

REQID(name)

specifies a name (1–8 characters), which should be unique, to identify a command. This name is used as a temporary storage identifier. The temporary storage queue thus identified must be defined as a local queue on the CICS system where the CANCEL command is processed.

This option cannot be used to cancel a POST command issued by the same task (for which, the REQID option is ignored if it is specified).

SYSID(systemname) — remote systems only

specifies the name (1–4 characters) of the system for the CANCEL command.

TRANSID(name)

```
# specifies the symbolic identifier (1–4 characters) of a
# transaction to be used to determine where the CANCEL
# command is to be executed, if SYSID is not specified. If
# the TRANSID is defined as REMOTE, the CANCEL
# request is function-shipped to the remote system. If
# SYSID is specified, the transaction is assumed to be on
# a remote system.
```

CANCEL

CANCEL conditions

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

NOTAUTH

| occurs when a resource security check has failed on the
| specified TRANSID or on the TRANSID of the START
| command that corresponds to the request identification.

Default action: terminate the task abnormally.

NOTFND

occurs if the request identifier specified fails to match an unexpired interval control command.

Default action: terminate the task abnormally.

SYSIDERR

| occurs when the SYSID option specifies a name that is
| neither the local system nor a remote system (made
| known to CICS by defining a CONNECTION). It also
| occurs when the link to the remote system is closed.

Default action: terminate the task abnormally.

CHANGE PASSWORD

Function

Change the password recorded by an external security manager (ESM) for a specified userid.

Command syntax

```

▶ CHANGE PASSWORD(data-value)—NEWPASSWORD(data-value)—USERID(data-value)—ESMREASON(data-area)
ESMRESP(data-area)

```

Conditions:

INVREQ, NOTAUTH, USERIDERR

Unlike the SIGNON command, CHANGE PASSWORD does not depend upon the principal facility, so it can be issued when the facility is an APPC session.

Warning: Clear password fields after use

You should clear the password fields on the EXEC CICS commands that have a password option as soon as possible after use. This is to ensure that passwords are not revealed in system or transaction dumps.

CHANGE PASSWORD options

Options ESMRESP and ESMREASON return the response and reason codes, if any, from the external security manager.

ESMREASON(*data-area*)

returns the reason code, in a fullword binary field, that CICS receives from the external security manager.

If the ESM is RACF, this field is the RACF reason code.

ESMRESP(*data-area*)

returns the response code, in a fullword binary field, that CICS receives from the external security manager.

If the ESM is RACF, this field is the RACF return code.

NEWPASSWORD(*data-value*)

specifies the new password, 8 characters, for the specified userid. The password is changed only if the current password is correctly specified.

PASSWORD(*data-value*)

specifies the current password, 8 characters, for the specified userid.

USERID(*data-value*)

specifies the userid, 8 characters, of the user whose password is being changed.

CHANGE PASSWORD conditions

INVREQ

occurs in any of the following situations:

- There is an unknown return code in ESMRESP from the external security manager (RESP2=13).
- The CICS external security manager interface is not initialized (RESP2=18).
- The external security manager is not responding (RESP2=29).

Default action: terminate the task abnormally.

NOTAUTH

occurs in any of the following situations:

- The supplied password is wrong (RESP2=2). If the external security manager is RACF, the revoke count maintained by RACF is incremented.
- The new password is not acceptable (RESP2=4).
- The USERID is revoked (RESP2=19).
- The change password request failed during SECLABEL processing (RESP2=22).
- The user is revoked in the connection to the default group (RESP2=31).

Default action: terminate the task abnormally.

USERIDERR

occurs if the USERID is not known to the external security manager (RESP2=8).

Default action: terminate the task abnormally.

CHANGE TASK

CHANGE TASK

Function

Change priority of a task.

Command syntax

```
▶ CHANGE TASK [ PRIORITY(data-value) ] ◀
```

Conditions:
INVREQ

CHANGE TASK changes the priority of the issuing task. It has immediate effect (unlike SET TASK), because control is relinquished during execution of the command so that the current task has to be redispached. The redispach does not happen until tasks that are of higher or equal priority, and that are also ready to run, are dispatched.

RCF 11362/11353

Documentation for RCFs 11362 and 11353 added
22/11/96

If you omit the PRIORITY option, the task does not lose control and the priority remains the same. This is effectively a no-op.

CHANGE TASK option

PRIORITY(*data-value*)

specifies a fullword binary value in the range 0–255, defining the priority of the task. You can also have a value of –1 but this does not change the priority or cause a redispach.

CHANGE TASK condition

INVREQ

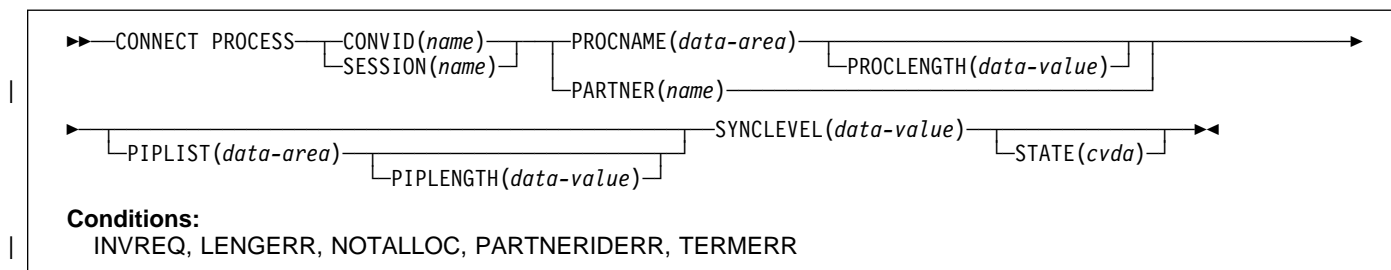
means that your PRIORITY value is outside the range 0–255 (RESP2=1).

CONNECT PROCESS

Function

Initiate APPC mapped conversation.

Command syntax



CONNECT PROCESS allows an application to specify a process name and synchronization level to be passed to CICS and used when the remote partner is attached.

CONNECT PROCESS options

CONVID(name)

identifies, as a 4-character name, the conversation which the command relates to. The name specifies the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB.

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

PARTNER(name)

specifies the name (8 characters) of a set of definitions that includes the name (or extended name) of a remote partner transaction (TPNAME or XTPNAME). You can use this option as an alternative to PROCNAME and PROCLENGTH.

PIPELENGTH(data-value)

specifies the total length (halfword binary value) of the specified process initialization parameter (PIP) list.

PIPLIST(data-area)

specifies the PIP data to be sent to the remote system. The PIP list consists of variable-length records, each containing a single PIP. A PIP starts with a 2-byte inclusive length field (LL), followed by a 2-byte reserved field, and then the parameter data.

PROCLENGTH(data-value)

specifies the length (as a halfword binary value in the range 1–64) of the name specified by the PROCNAME option.

PROCNAME(data-area)

specifies the partner process (that is, the transaction) to be attached in the remote system.

One byte is sufficient to identify a CICS transaction. The APPC architecture allows a range of 1–64 bytes but leaves each product free to set its own maximum. CICS complies by allowing a range of 1–64 bytes. If the remote system is CICS, this option can specify the 4-byte transaction identifier or the TPNAME value given in the relevant TRANSACTION definition. Alternatively, you can examine the full identifier by coding the user exit XZCATT.

No character checking is performed on the TPN by CICS.

For programming information about the user exit XZCATT, see the *CICS/ESA Customization Guide*.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

CONNECT PROCESS

SYNCLEVEL(data-value)

specifies the synchronization level (halfword binary value) for the current conversation. The possible values are:

- 0 None
- 1 Confirm
- 2 Syncpoint.

A CANCEL TASK request by a user node error program (NEP) can cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

CONNECT PROCESS conditions

INVREQ

occurs in any of the following situations:

- A synchronization level other than 0, 1, or 2, has been requested in the SYNCLEVEL option.
- The command is not valid for the terminal or LU in use.
- The command has been used on a conversation that is in use by CPI-Communications or that is an APPC basic conversation. In the latter case, GDS CONNECT PROCESS should have been used.
- A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- An out-of-range value is supplied in the PROCLENGTH option.
- The value specified in the PIPELENGTH option is less than 0.
- The value specified in the PIPELENGTH option exceeds the CICS implementation limit of 32 763.
- A PIPLIST length element (LL) has a value less than 4.
- The sum of the length elements (LLs) in the PIPLIST does not equal the value specified by PIPELENGTH.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the specified CONVID value does not relate to a conversation owned by the application.

Default action: terminate the task abnormally.

| PARTNERIDERR

| occurs if the name specified in the PARTNER option is
| not recognized by CICS.

| Default action: terminate the task abnormally.

TERMERR

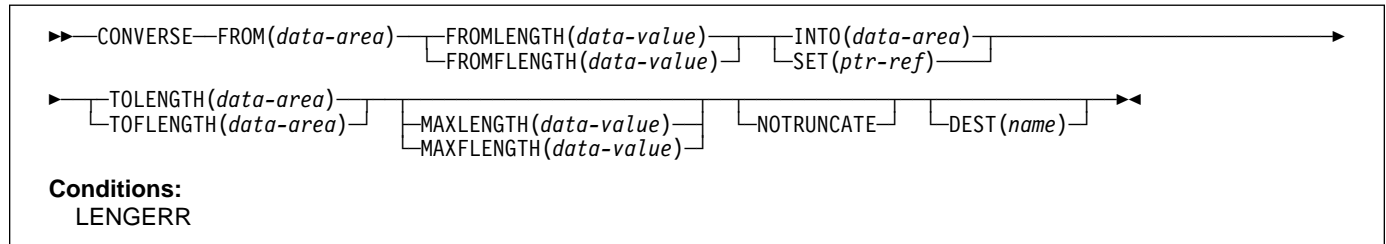
occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

CONVERSE (VTAM default)

Function

Communicate on standard CICS terminal support.

Command syntax



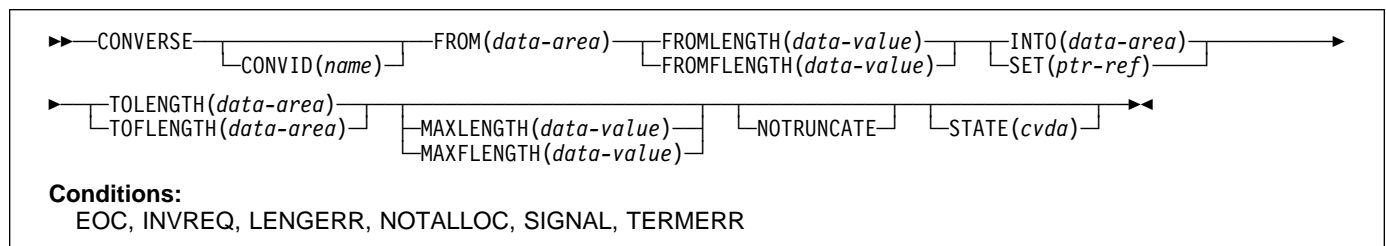
This form of the CONVERSE command is used by all CICS-supported VTAM terminals for which the other CONVERSE descriptions are not appropriate.

CONVERSE (APPC)

Function

Communicate on an APPC mapped conversation.

Command syntax



CONVERSE sends, then receives, data on an APPC mapped conversation.

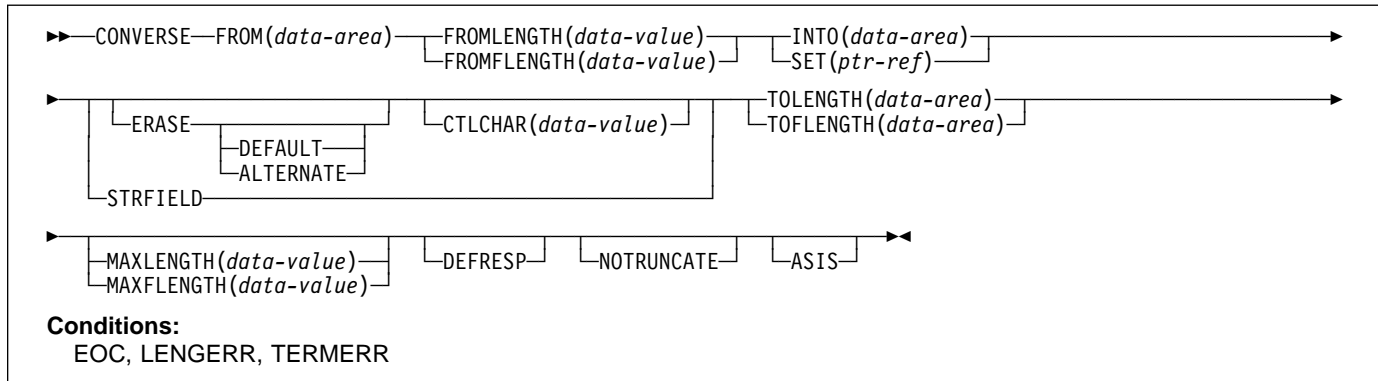
CONVERSE (VTAM)

CONVERSE (LUTYPE2/LUTYPE3)

Function

Communicate on a 3270-display logical unit (LUTYPE2) or 3270-printer logical unit (LUTYPE3).

Command syntax



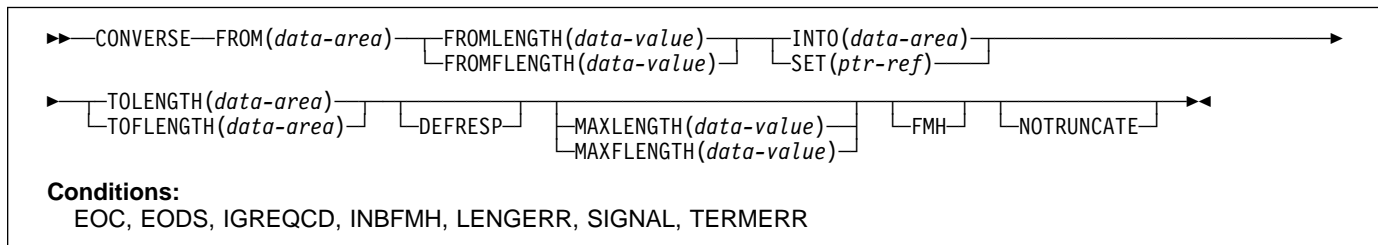
CONVERSE communicates on a 3270-display logical or 3270-printer logical unit.

CONVERSE (LUTYPE4)

Function

Communicate on an LUTYPE4 logical unit.

Command syntax



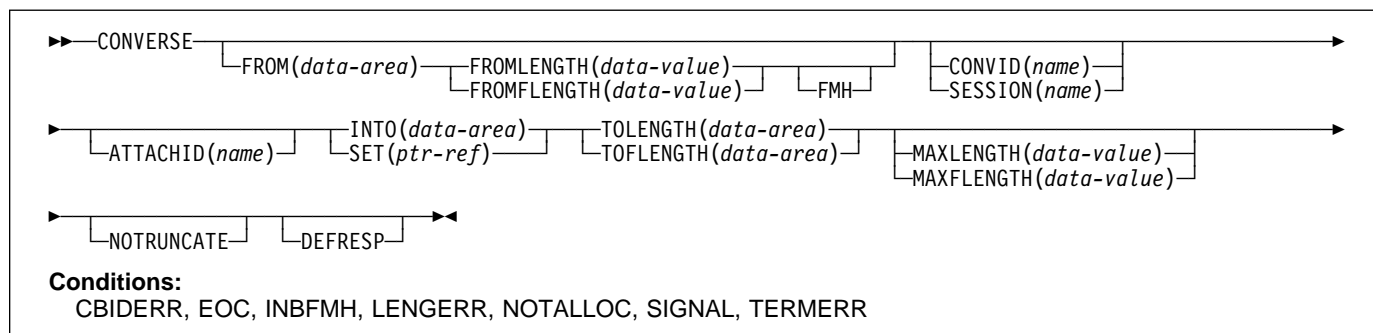
CONVERSE communicates on an LUTYPE4 logical unit.

CONVERSE (LUTYPE6.1)

Function

Communicate on an LUTYPE6.1 logical unit.

Command syntax



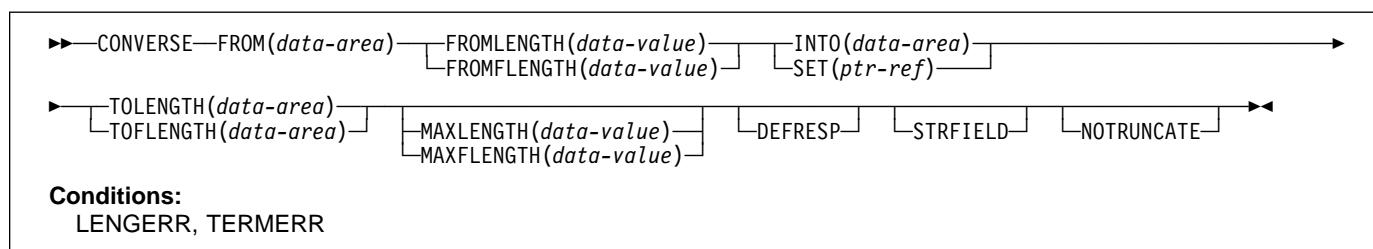
CONVERSE communicates on an LUTYPE6.1 logical unit.

CONVERSE (SCS)

Function

Communicate on a 3270 SCS printer logical unit.

Command syntax



CONVERSE communicates on a 3270 SNA character string (SCS) printer logical unit. The SCS printer logical unit accepts a character string as defined by Systems Network Architecture (SNA). Some devices connected under SNA can send a signal that can be detected by the HANDLE CONDITION SIGNAL command, which in turn can invoke an appropriate handling routine. If necessary, a WAIT SIGNAL command can be used to make the application program wait for the signal. The PA keys on a 3287 can be used in this way, or with a RECEIVE command.

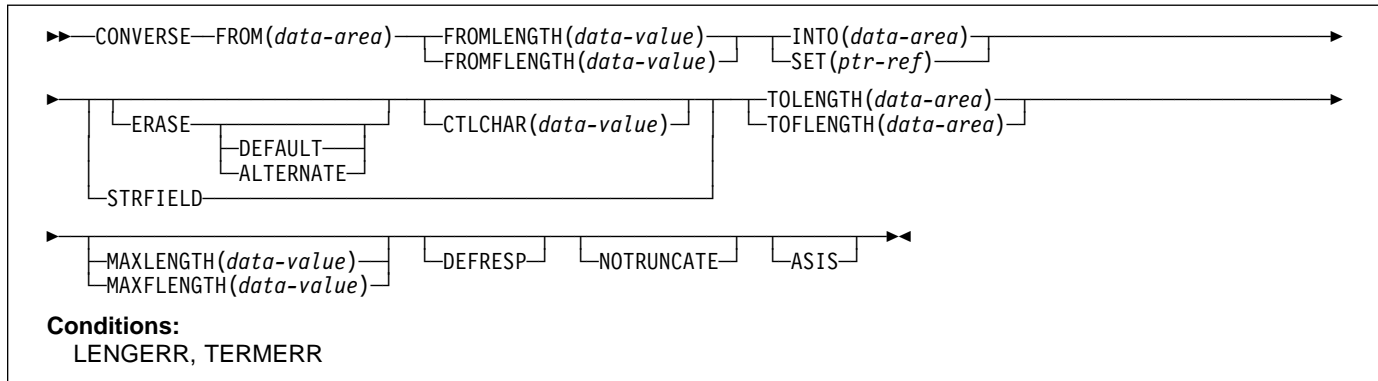
CONVERSE (VTAM)

CONVERSE (3270 logical)

Function

Communicate on a 3270 logical unit.

Command syntax



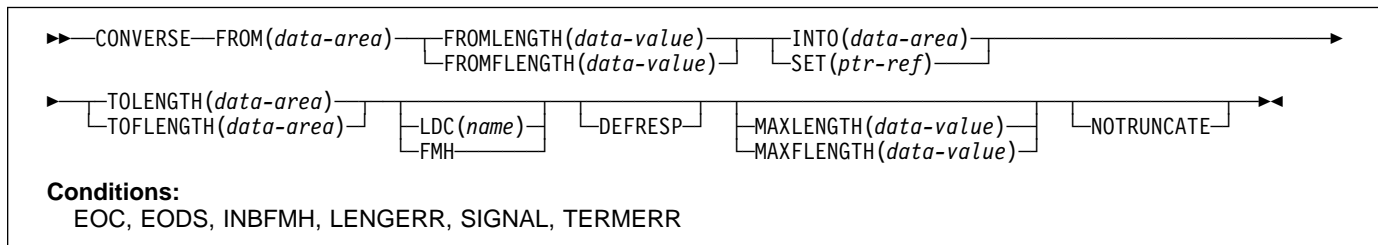
CONVERSE communicates on a 3270 logical unit.

CONVERSE (3600-3601)

Function

Communicate on a 3600 (3601) logical unit.

Command syntax



CONVERSE communicates on a 3600 logical unit. This form of the CONVERSE command also applies to the 4700 and the 3630 plant communication system.

A logical device code (LDC) is a code that can be included in an outbound FMH (Function Management Header) to specify the disposition of the data (for example, to which subsystem terminal it should be sent). Each code can be represented by a unique LDC mnemonic.

The installation can specify up to 256 2-character mnemonics for each TCTTE, and two or more TCTTEs can share a list of these mnemonics. A numeric value (0 through 255) corresponds to each LDC mnemonic for each TCTTE.

A 3600 device and a logical page size are also associated with an LDC. "LDC" or "LDC value" is used in this book to refer to the code specified by the user; "LDC mnemonic" refers to the 2-character symbol that represents the LDC numeric value.

When the LDC option is specified in the CONVERSE command, the numeric value associated with the mnemonic for the particular TCTTE is inserted in the FMH. This value is chosen by the installation, and is interpreted by the 3601 application program.

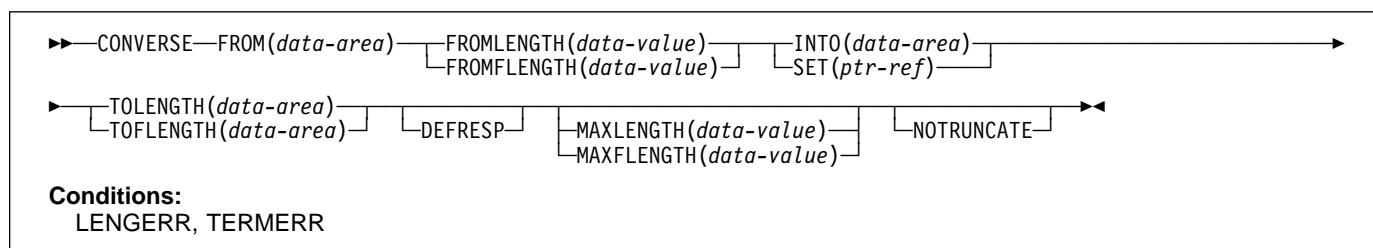
On output, the FMH can be built by the application program or by CICS. If your program supplies the FMH, you place it at the # front of your output data and specify the FMH option on your CONVERSE command. If you omit the FMH option, CICS will # provide an FMH but you must reserve the first three bytes of the message for CICS to fill in.

CONVERSE (3600-3614)

Function

Communicate on a 3600 (3614) logical unit.

Command syntax



CONVERSE communicates on a 3600 logical unit.

The data stream and communication format used between a CICS application program and a 3614 is determined by the 3614. The application program is therefore device dependent when handling 3614 communication.

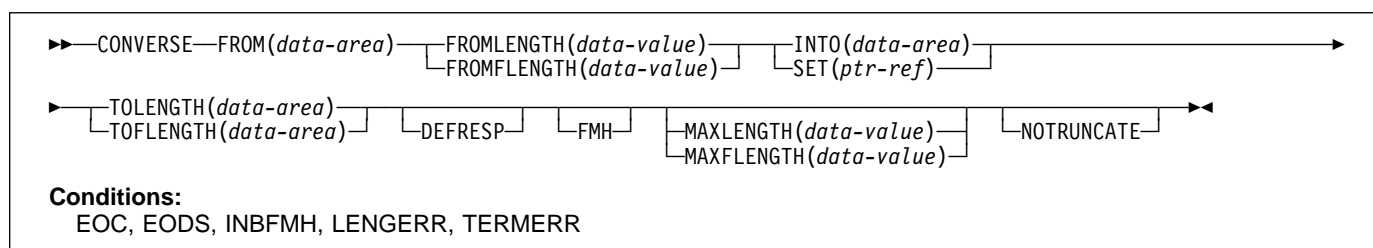
For further information about designing 3614 application programs for CICS, refer to the *CICS/OS/VS IBM 4700/3600/3630 Guide*.

CONVERSE (3650 interpreter)

Function

Communicate on a 3650 interpreter logical unit.

Command syntax



CONVERSE communicates on a 3650 interpreter logical unit.

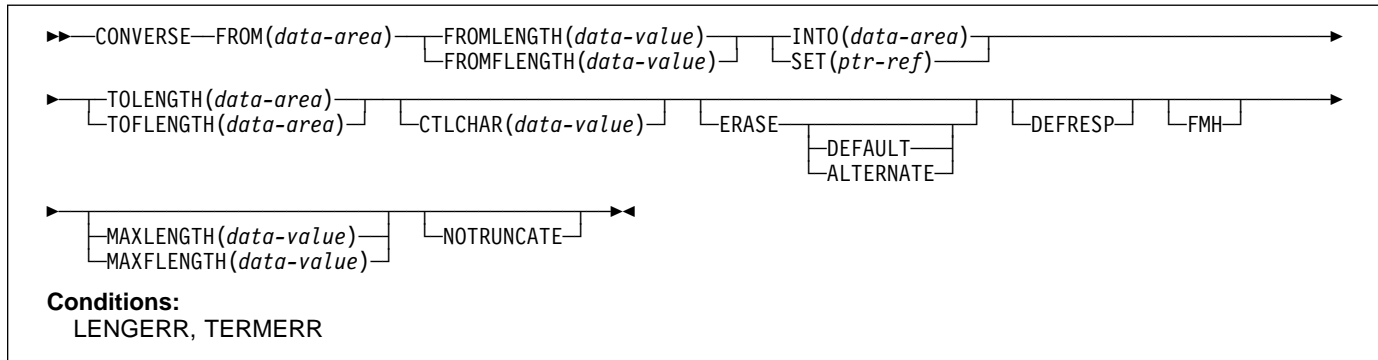
CONVERSE (VTAM)

CONVERSE (3650-3270)

Function

Communicate on a 3650 host conversational (3270) logical unit.

Command syntax



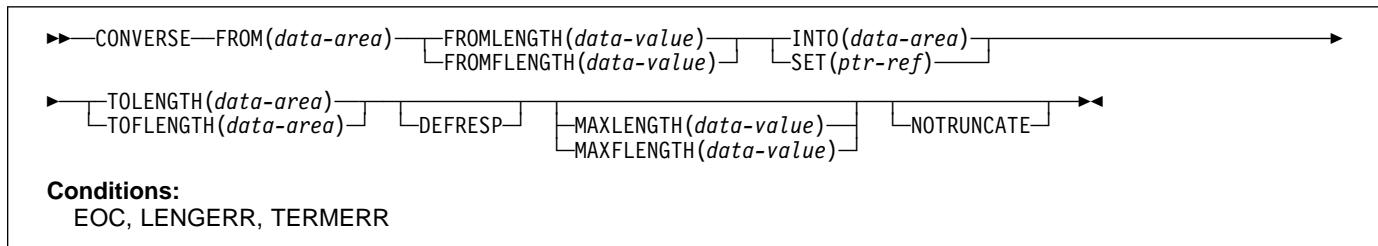
CONVERSE communicates on a 3650 host conversational logical unit.

CONVERSE (3650-3653)

Function

Communicate on a 3650 host conversational (3653) logical unit.

Command syntax



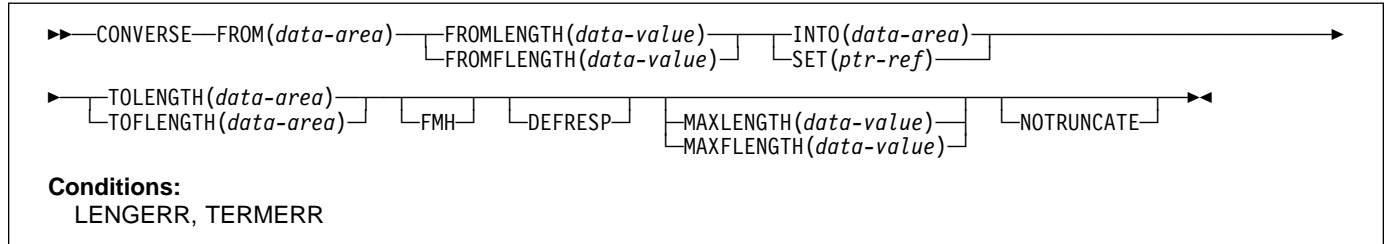
CONVERSE communicates on a 3650 host conversational logical unit.

CONVERSE (3650-3680)

Function

Communicate on a 3650 host command processor (3680) logical unit.

Command syntax



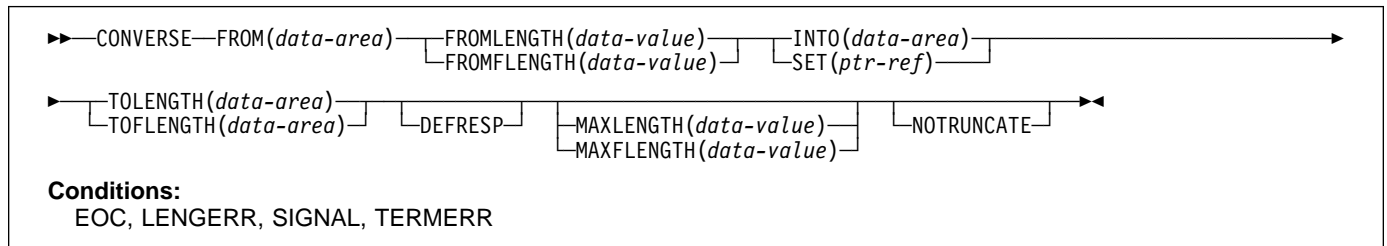
CONVERSE communicates on a 3650 host command processor logical unit.

CONVERSE (3767)

Function

Communicate on a 3767 interactive logical unit.

Command syntax



CONVERSE communicates on a 3767 interactive logical unit. This command also applies to the 3770 interactive logical unit.

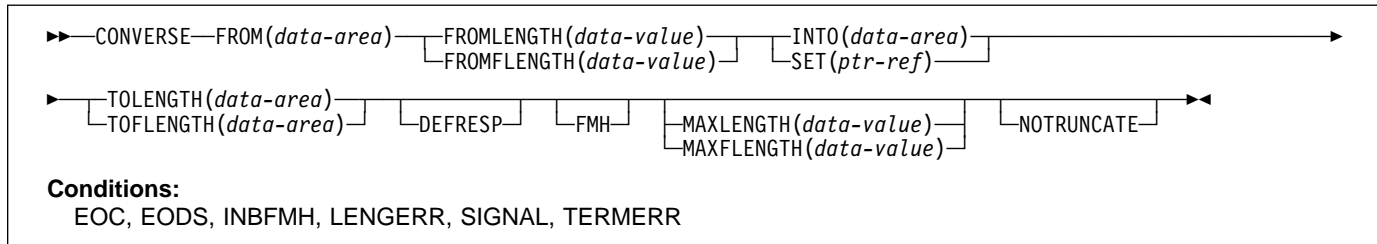
CONVERSE (VTAM)

CONVERSE (3770)

Function

Communicate on a 3770 batch logical unit.

Command syntax



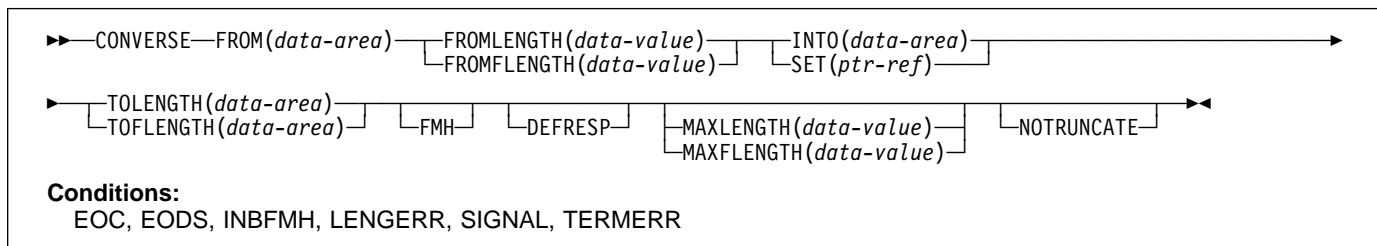
CONVERSE communicates on a 3770 batch logical unit.

CONVERSE (3790 full-function or inquiry)

Function

Communicate on a 3790 full-function or inquiry logical unit.

Command syntax



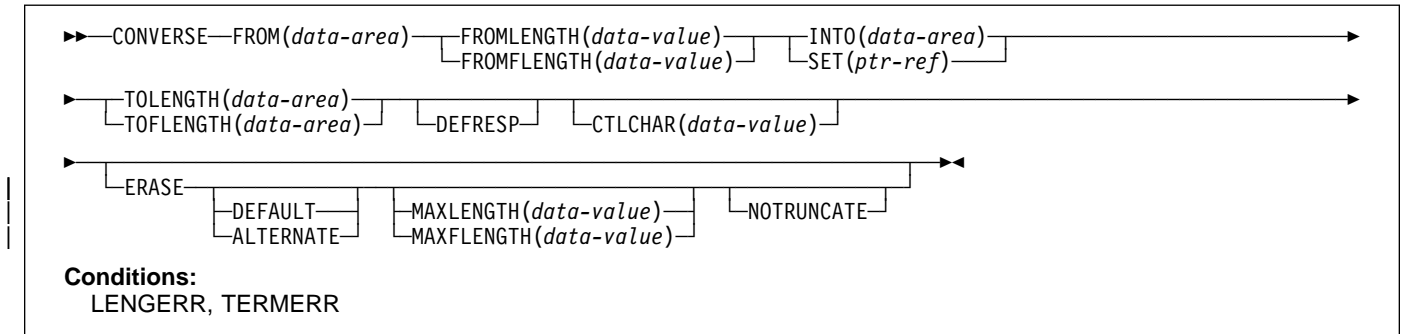
CONVERSE communicates on a 3790 full-function or inquiry logical unit.

CONVERSE (3790 3270-display)

Function

Communicate on a 3790 (3270-display) logical unit.

Command syntax



CONVERSE communicates on a 3790 logical unit.

CONVERSE (VTAM)

CONVERSE (VTAM) options

ALTERNATE

sets the terminal to use the ALTERNATE screen size.

ASIS

indicates that output is to be sent in transparent mode (with no recognition of control characters and accepting any of the 256 possible combinations of eight bits as valid transmittable data).

Note: If you are using a katakana terminal, you might see some messages containing mixed English and katakana characters. That is because katakana terminals cannot display mixed-case output. Uppercase characters in the data stream appear as uppercase English characters, but lowercase characters appear as katakana characters. If this happens, ask your system programmer to specify MSGCASE=UPPER in the system initialization parameters so that messages contain uppercase characters only. This note applies to any command that is used to receive katakana characters, not just to CONVERSE commands.

ATTACHID(name)

specifies that an attach header (created by a BUILD ATTACH command) is to precede, and be concatenated with, the user data supplied in the FROM option. "name" (1–8 characters) identifies the attach header control block to be used in the local task.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If the option is omitted, the principal facility for the task is used by default.

CTLCHAR(data-value)

specifies a 1-byte write control character (WCC) that controls the CONVERSE command. A COBOL user must specify a data area containing this character.

If the option is omitted, all modified data tags are reset to zero, and the keyboard is restored.

DEFAULT

sets the terminal to use the DEFAULT screen size.

DEFRESP

indicates that a definite response is required when the output operation has been completed.

DEST(name)

specifies the 4-byte symbolic name of the TCAM destination to which the message is to be sent. This

option is meaningful only for terminals defined using DFHTCT TYPE=SDSCI with DEVICE=TCAM.

If you use the DEST option, you must be aware of any restrictions placed on device-dependent data streams by the message control facility in use. See the section on CICS-TCAM interface in the *CICS/ESA Customization Guide* for programming information.

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSZE option in the profile definition.

FMH

specifies that a function management header has been included in the data to be written. If the ATTACHID option is specified as well, the concatenated FMH flag is set in the attach FMH. The use of FMH is optional and is not supported for all terminal types. If not supplied, CICS takes no action, except for 3600/4700 terminals, where an FMH is mandatory. In this case, if FMH is not specified, CICS supplies one and places it in the first 3 bytes of the message, which you must reserve for this purpose.

FROM(data-area)

specifies the data to be written to the terminal or logical unit, or sent to the partner transaction. This option may, when relevant, be omitted if ATTACHID is specified.

FROMLENGTH(data-value)

is a fullword alternative to FROMLENGTH.

FROMLENGTH(data-value)

specifies the length, as a halfword binary value, of the data. For a description of a safe upper limit, see "LENGTH options" on page 5.

INTO(data-area)

specifies the receiving field for the data read from the terminal or logical unit, or the application target data area into which data is to be received from the application program connected to the other end of the current conversation.

LDC(name)

specifies the 2-character mnemonic used to determine the appropriate logical device code (LDC) numeric value. The mnemonic identifies an LDC entry in the terminal control table TYPE=LDC.

MAXLENGTH(data-value)

is a fullword alternative to MAXLENGTH.

MAXLENGTH(data-value)

specifies the maximum amount (halfword binary value) of data that CICS is to recover in response to a CONVERSE (default) command. If INTO is specified, MAXLENGTH overrides the use of TOLENGTH as an input to CICS. If SET is specified, MAXLENGTH provides a way for the program to limit the amount of data it receives at one time.

If the value specified is less than zero, zero is assumed.

If the length of data exceeds the value specified and the NOTRUNCATE option is not present, the data is truncated to that value and the LENGERR condition occurs. The data area specified in the TOLENGTH option is set to the original length of data.

If the length of data exceeds the value specified and the NOTRUNCATE option is present, CICS retains the remaining data and uses it to satisfy subsequent RECEIVE commands. The data area specified in the TOLENGTH option is set to the length of data returned.

If no argument is coded for MAXLENGTH, CICS defaults to TOLENGTH.

NOTRUNCATE

specifies that, when the data available exceeds the length requested, the remaining data is not to be discarded but is to be retained for retrieval by subsequent RECEIVE commands.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If both this option and CONVID are omitted, the principal facility for the task is used.

SET(ptr-ref)

specifies the pointer reference to be set to the address of the data read from the terminal. pointer reference, unless changed by other commands or statements, is valid until the next CONVERSE (default) command or the end of task.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

STRFIELD

specifies that the data area specified in the FROM option contains structured fields. If this option is specified, the contents of all structured fields must be handled by the application program. The CONVERSE command must be used if the data area contains a read partition structured field. (Structured fields are described in the *CICS/ESA 3270 Data Stream Device Guide*.)

CTLCHAR and ERASE are mutually exclusive with STRFIELD, and their use with STRFIELD generates an error message.

TOFLENGTH(data-area)

is a fullword alternative to TOLENGTH.

TOLENGTH(data-area)

specifies the length (halfword binary value) of the data to be received. If you specify INTO, but omit MAXLENGTH, "data-area" specifies the maximum length that the program accepts. If the value is less than zero, zero is assumed.

If the length of the data exceeds the value specified, but NOTRUNCATE is omitted, the data is truncated to that value, and the LENGERR condition occurs. When the data is received, the data area is set to the length of the data.

For a description of a safe upper limit, see "LENGTH options" on page 5.

CONVERSE (VTAM) conditions

Some of the following conditions can occur in combination with others. CICS checks for these conditions in the following order:

1. EODS
2. INBFMH
3. EOC.

If more than one occurs, only the first is passed to the application program. EIBRCODE, however, is set to indicate all the conditions that occurred.

CBIDERR

occurs if the requested attach header control block named in ATTACHID cannot be found.

CONVERSE (VTAM)

Default action: terminate the task abnormally.

EOC

occurs when a request/response unit (RU) is received with the end-of-chain indicator set. Field EIBEOC also contains this indicator.

Default action: ignore the condition.

EODS

occurs when an end-of-data-set indicator is received.

Default action: terminate the task abnormally.

IGREQCD

occurs when an attempt is made to execute a CONVERSE command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

INBFMH

occurs if a request/response unit (RU) contains a function management header (FMH). Field EIBFMH contains this indicator and it should be used in preference to INBFMH. The IGNORE CONDITION command can be used to ignore the condition.

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- The command is used on a conversation that is in use by CPI Communications, or that is an APPC basic conversation. In the latter case, the application should have issued a GDS SEND INVITE followed by a GDS RECEIVE.
- A distributed program link server application specified the function shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- Data received is discarded by CICS because its length exceeds the maximum that the program accepts (see TOLENGTH and MAXLENGTH options), and the NOTRUNCATE option is not specified.
- An out-of-range value is supplied in one of the options, FROMLENGTH, FROMFLENGTH, MAXLENGTH, MAXFLENGTH, TOLENGTH, or TOFLENGTH.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application, or does not relate to a conversation owned by the application.

Default action: terminate the task abnormally.

SIGNAL

occurs when an inbound SIGNAL data-flow control command is received from a logical unit or session, or the partner transaction. EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

TERMERR

occurs for a terminal or session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

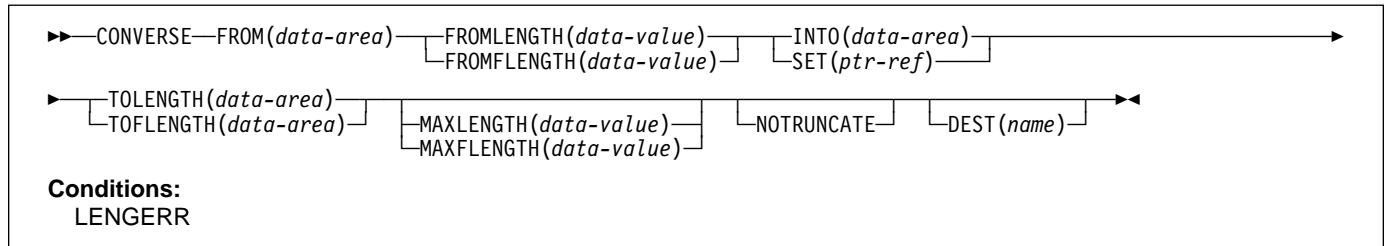
Default action: terminate the task abnormally with abend code ATNI.

CONVERSE (non-VTAM default)

Function

Communicate on standard CICS terminal support.

Command syntax



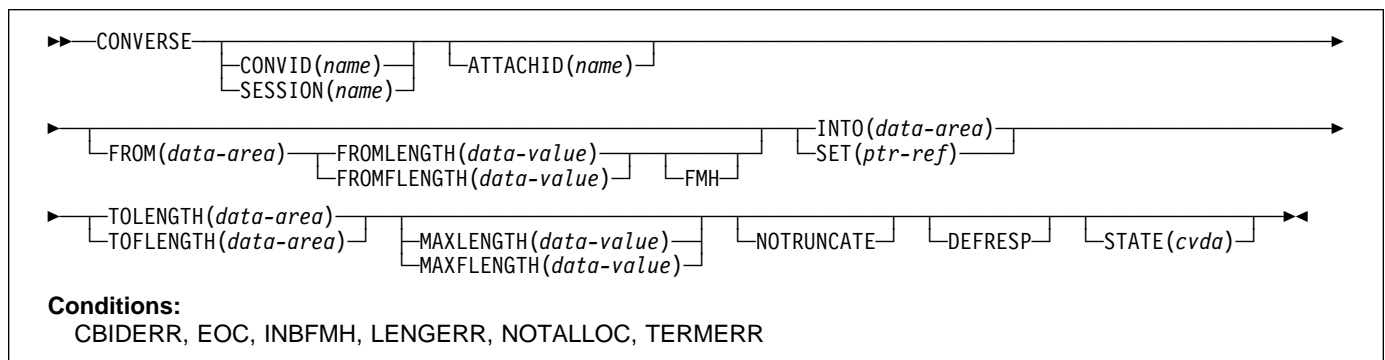
This form of the CONVERSE command is used by all CICS-supported terminals for which the other CONVERSE descriptions are not appropriate.

CONVERSE (MRO)

Function

Communicate on an MRO session.

Command syntax



CONVERSE communicates on an MRO session. For more information about MRO and IRC, see the *CICS/ESA Intercommunication Guide*.

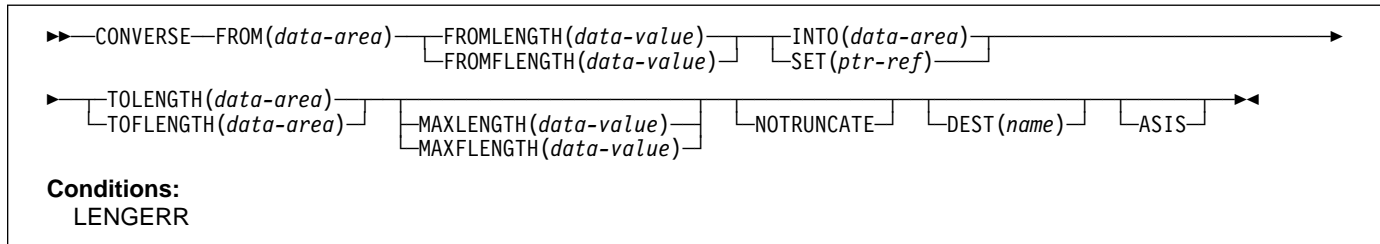
CONVERSE (non-VTAM)

CONVERSE (System/3)

Function

Communicate on a System/3 terminal.

Command syntax



CONVERSE communicates on a System/3 terminal. CONVERSE in this form also applies to the following devices:

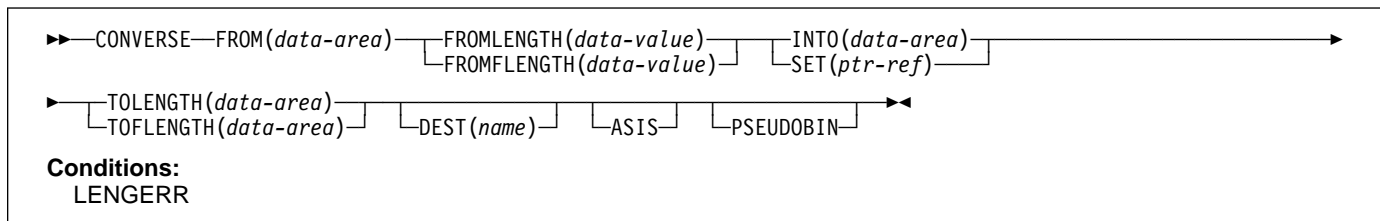
- 2770 data communication system
- 2780 data transmission terminal
- 3660 supermarket scanning system
- 3780 communication terminal.

CONVERSE (System/7)

Function

Communicate on a System/7 terminal.

Command syntax



CONVERSE communicates on a System/7 terminal.

Transactions are normally initiated from the System/7 by issuing a 4-character transaction code that is transmitted in BCD mode. Pseudobinary mode can be used only while communicating with an active CICS transaction; it cannot be used to initiate the transaction. The message length is given as the number of words to be transmitted (not as the number of characters).

When a transaction is initiated on a System/7, CICS services that System/7 only for the duration of the transaction; that is, to ensure efficient use of the line, any other System/7s on the same line are locked out for the duration of the transaction. CICS application programs for the multipoint System/7 should be designed with the shortest possible execution time.

The first word (two characters) of every message sent to the System/7 must be an identification word, except words beginning with "@", which are reserved by CICS.

When the PSEUDOBIN option is specified on RECEIVE or SEND, the length of the data area provided by the application program must be at least twice that of the data to be read.

In the case of a System/7 on a dial-up (switched) line, the System/7 application program must, initially, transmit a four-character terminal identification. (This terminal identification is generated during preparation of the TCT through use of the DFHTCT TYPE=TERMINAL, TRMIDNT=parameter specification.) CICS responds with either a 'ready' message, indicating that the terminal identifier is valid and that the System/7 may proceed as if it were on a leased line, or an 'INVALID TERMINAL IDENTIFICATION' message, indicating that the terminal identifier sent by the System/7 did not match the TRMIDNT= parameter specification.

Whenever CICS initiates the connection to a dial-up System/7, CICS writes a null message, consisting of three idle characters, prior to starting the transaction. If there is no program resident in the System/7 capable of supporting the Asynchronous Communication Control Adapter (ACCA), BTAM error routines cause a data check message to be recorded on the CICS (host) system console. This is normal if the task initiated by CICS is to IPL the System/7. Although the data check message is printed, CICS ignores the error and continues normal processing. If a program capable of supporting the ACCA is resident in the System/7 at the time this message is transmitted, no data check occurs.

When a disconnect is issued to a dial-up System/7, the "busy" bit is sometimes left on in the interrupt status word of the ACCA. If the line connection is reestablished by dialing from the System/7 end, the 'busy' condition of the ACCA prevents message transmission from the System/7. To overcome this problem, the System/7 program must reset the ACCA after each disconnect and before message transmission is attempted. This can be done by issuing the following instruction:

```
PWRI  0,8,3,0  RESET ACCA
```

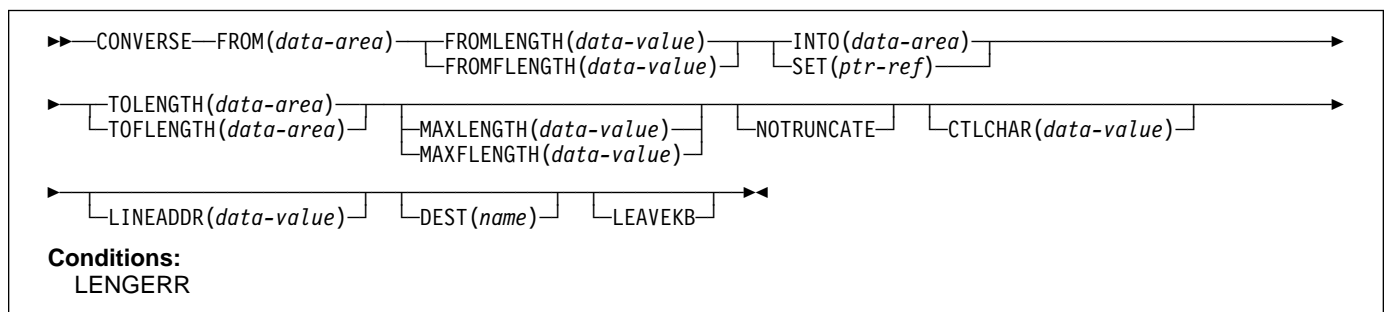
This procedure is not necessary when the line is reconnected by CICS (that is, by an automatically initiated transaction).

CONVERSE (2260)

Function

Communicate on a 2260 or 2265 display station.

Command syntax



CONVERSE communicates on a 2260 or 2265 display station.

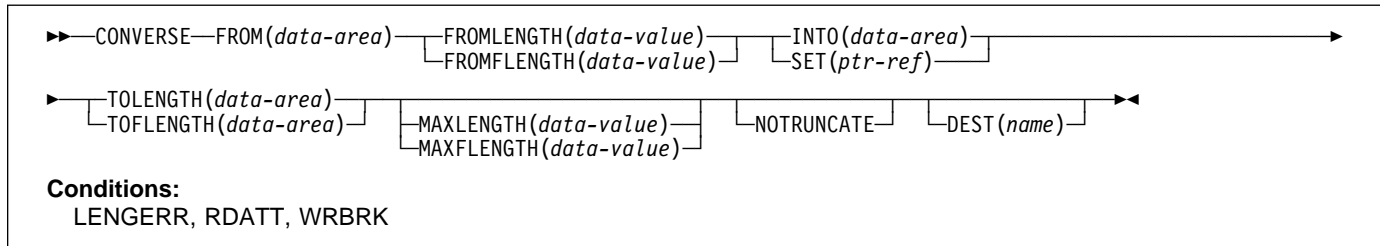
CONVERSE (non-VTAM)

CONVERSE (2741)

Function

Communicate on a 2741 communication terminal.

Command syntax



CONVERSE communicates on a 2741 terminal.

Read attention

If the terminal operator presses the attention key on the 2741 after typing a message, it is recognized as a read attention if:

- Read attention is supported by the system
- The message is read by a RECEIVE command.

When this occurs, control is transferred to a CICS read-attention exit routine, if it has been generated into the system. This routine is a skeleton program that can be tailored by the system programmer to carry out actions such as the following:

- Perform data analysis or modification on a read attention.
- Return a common response to the terminal operator following a read attention.
- Return a response and request additional input that can be read into the initial input area or into a new area.
- Request new I/O without requiring a return to the task to request additional input.

When the read-attention exit routine is completed, control is returned to the application program at the address specified in the HANDLE CONDITION RDATT command. The return is made when either of the following occurs:

- The exit routine issues no more requests for input.
- The exit routine issues a RECEIVE request and the operator terminates the input with a carriage return. (If the operator terminates the input with an attention, the exit routine is reentered and is free to issue another RECEIVE request.)

If a HANDLE CONDITION RDATT command is not included in the application program or read attention is not supported, the attention is treated as if the return key had been pressed.

Write break

If the terminal operator presses the attention key on the 2741 while a message is being received, it is recognized as a write break if:

- Write break is supported by the system.
- A HANDLE CONDITION WRBRK command is active in the application program.

When this occurs, the remaining portion of the message is not sent to the terminal. The write is terminated as though it were successful, and a new-line character (X'15') is sent to cause a carrier return. Control is returned to the application program at the address specified for the WRBRK condition.

If a HANDLE CONDITION WRBRK command is not included in the application program or if write break is not supported, the break is treated as an I/O error.

CONVERSE (2770)

The support and command syntax for 2770 is the same as for System/3, see “CONVERSE (System/3)” on page 54.

The 2770 recognizes a read interrupt and responds by transmitting the contents of the I/O buffer. After the contents of the buffer have been transmitted, the 2770 responds to the next read continue with an EOT. If the I/O buffer is empty, the 2770 transmits an EOT. CICS issues a read interrupt and read continue to relinquish use of the line and to enable the application program to write to the 2770.

Input from a 2770 consists of one or more logical records. CICS provides one logical record for each read request to the application program. The size of a logical record cannot exceed the size of the I/O buffer. If the input spans multiple buffers, multiple reads must be issued by the application program.

The 2265 component of the 2770 Data Communication System is controlled by data stream characters, not by BTAM macros; appropriate screen control characters should be included in the output area.

For 2770 input, data is deblocked to ETX, ETB, RS, and US characters. These characters are moved with the data to the input area but are not included in the data length; characters such as NL, CR, and LF are passed in the input area as data.

CONVERSE (2780)

The support and command syntax for 2780 is the same as for System/3, see “CONVERSE (System/3)” on page 54.

Output to a 2780 requires that the application program contains an appropriate “escape sequence” for component selection associated with the output message. For details of the 2780, see the publication *IBM 2780 Data Transmission Terminal: Component Description*.

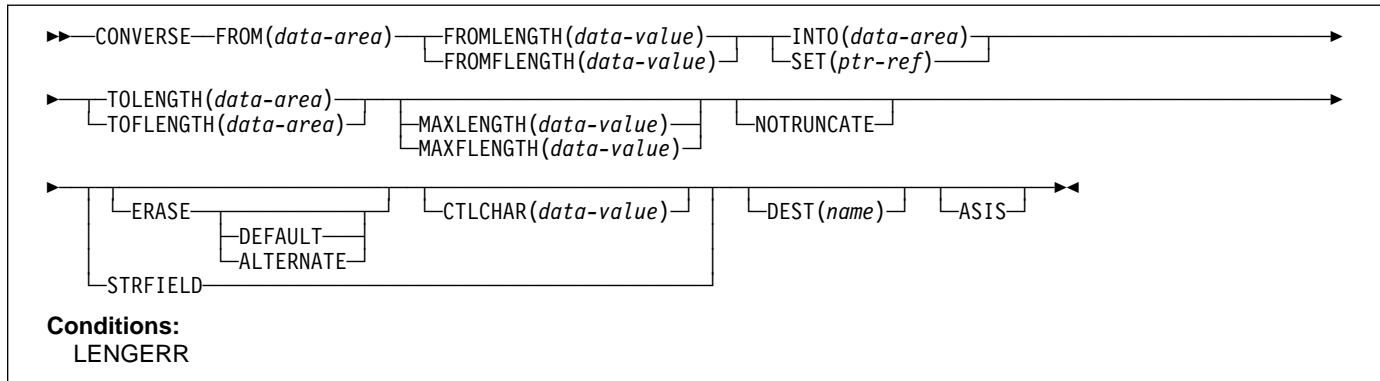
CONVERSE (non-VTAM)

CONVERSE (3270 display)

Function

Communicate on a 3270 information display system (TCAM).

Command syntax



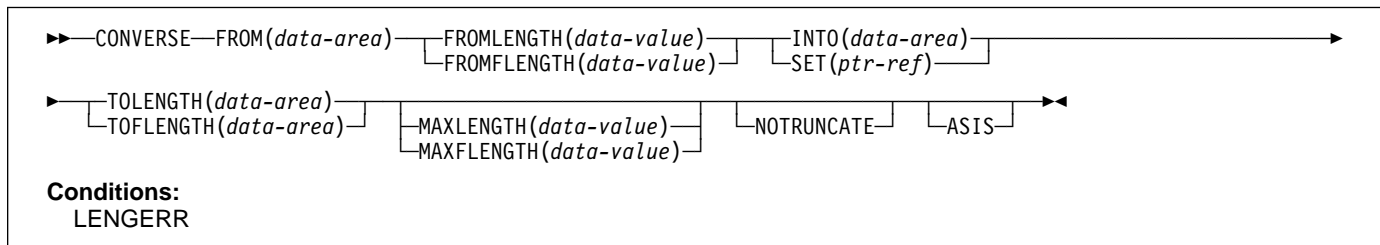
CONVERSE communicates on a 3270 information display system.

CONVERSE (3600 BTAM)

Function

| Communicate on a 3600 finance communication system (BTAM) - only valid for remote terminals.

Command syntax



CONVERSE communicates on a 3600 finance communication system. This form of the CONVERSE command also applies to the 4700 finance communication system. See "BTAM programmable terminals" on page 358.

Input

The unit of transmission from a 3601 Finance Communication Controller to CICS is a segment consisting of the start-of-text data-link control character (STX), the 1-byte identification of the 3600 logical workstation that issued the processor write, the data, and either an end-of-block (ETB) or an end-of-text (ETX) control character.

A logical workstation sends a message either in one segment, in which case the segment ends with ETX, or in more than one segment, in which case only the last segment ends with ETX, all others ending with ETB.

The input area passed to the user-written application program consists of the data only. The 1-byte field TCTTEDLM, which can be obtained using an ASSIGN DELIMITER command, contains flags describing the data-link control character (ETB, ETX, or IRS) that ended the segment. The application program can issue terminal control commands to read the data until it receives a segment ending with ETX. If blocked data is transmitted, it is received by CICS as blocks of segments. Only the first segment in a block starts with the STX control character, and all segments are separated by IRS characters. None of the segments contains ETB or ETX characters except the last, which has the ETX character.

For blocked input, the flags in TCTTEDLM only indicate end of segment, not end of message. The CICS application program still receives only the data, but user-defined conventions may be required to determine the end of the message.

The field TCTTEDLM also indicates the mode of the input, either transparent or nontransparent. Blocked input is nontransparent.

The terminal control facility does not pass input containing a start-of-header (SOH) data-link control character to a user-written application program. If it receives an SOH, it sets an indicator in TCTTEDLM, passes the input to the user exit in the terminal control facility, and then discards it.

Output

When an application program issues a SEND command, the terminal control facility determines, from the value specified in the BUFFER parameter of the DFHTCT TYPE=TERMINAL system macro, the number of segments to be built for the message. It sends the message to the 3600 logical unit either in one segment consisting of a start-of-text character (STX), the data, and an end-of-text character (ETX); or in more than one segment, in which case only the last ends with ETX, all others ending with ETB.

The host input buffer of the 3600 controller and the input segment of the receiving logical unit must be large enough to accommodate the data sent by CICS. However, space for the data-link control characters need not be included. The 3600 application program reads the data from the host, by means of an LREAD, until it has received the entire message.

CICS system output messages begin with DFH followed by a 4-byte message number and the message text. These messages are sent in nontransparent mode. CICS user-written application programs should not send messages starting with DFH to the 3601.

Resend message

When a logical unit sends a message to the host and a 'short on storage' condition exists or the input is unsolicited (the active task associated with the terminal has not issued a read), the terminal control facility sends a "resend" message to the logical unit. No message is sent to the destinations CSMT or CSTL.

The first eight bytes of data sent to CICS can be used by the 3600 application program to define a convention to associate responses received from CICS with transactions sent to the host; for example, sequence numbers could be used.

If a CICS user-written application program has already issued a SEND command when a resend situation occurs, the resend message is not sent to the 3601 until the user-written application program message has been sent. A 3600 logical unit cannot receive a resend message while receiving a segmented message.

Only one resend message at a time can be queued for a logical unit. If a second resend situation occurs before CICS has written the first, a resend message is sent containing the eight bytes of data that accompanied the second input transaction from the 3600 logical unit.

The resend message is sent in transparent mode if the input data from the 3601 to be retransmitted is received by CICS in transparent mode. Otherwise it is sent in nontransparent mode.

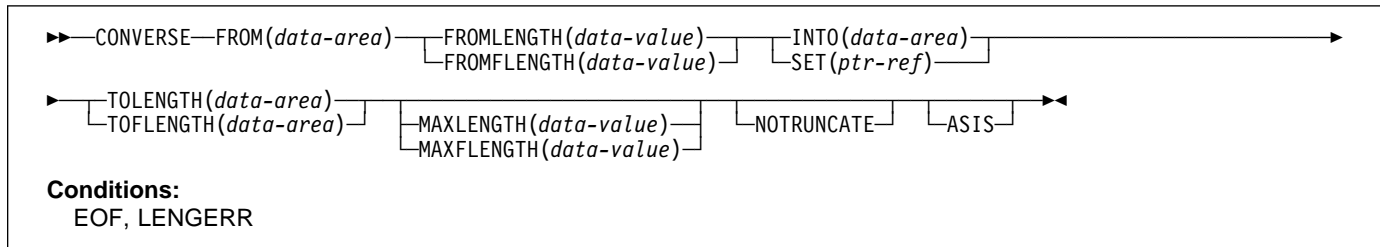
CONVERSE (non-VTAM)

CONVERSE (3735)

Function

Communicate on a 3735 programmable buffered terminal.

Command syntax



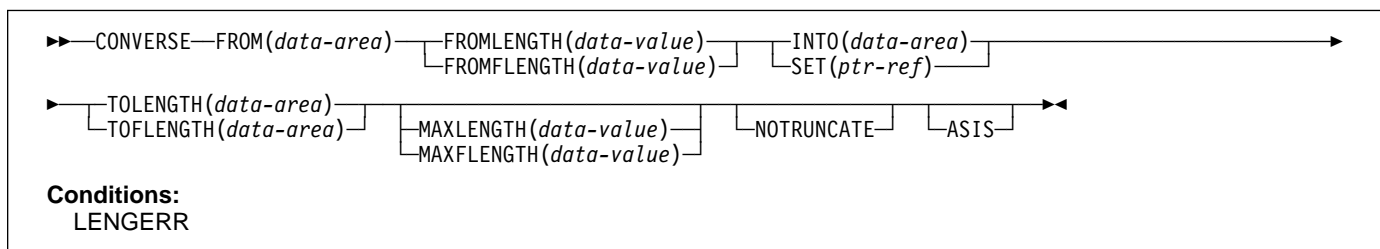
CONVERSE communicates on a 3735 programmable buffered terminal. The 3735 programmable buffered terminal may be serviced by CICS in response to terminal-initiated input (automatic answering), or as a result of an automatic calling or time-initiated transaction.

CONVERSE (3740)

Function

Communicate on a 3740 data entry system.

Command syntax



CONVERSE communicates on a 3740 data entry system.

In batch mode, many files are exchanged between the 3740 and CICS in a single transmission. The transmission of an input batch must be complete before an output transmission can be started.

CONVERSE (non-VTAM) options**ALTERNATE**

set the terminal to use the ALTERNATE screen size.

ASIS

indicates that output is to be sent in transparent mode (with no recognition of control characters and accepting any of the 256 possible combinations of eight bits as valid transmittable data).

Note: If you are using a katakana terminal, you might see some messages containing mixed English and katakana characters. That is because katakana terminals cannot display mixed-case output. Uppercase characters in the data stream appear as uppercase English characters, but lowercase characters appear as katakana characters. If this happens, ask your system programmer to specify MSGCASE=UPPER in the system initialization parameters so that messages contain uppercase characters only.

ATTACHID(name)

specifies that an attach header (created by a BUILD ATTACH command) is to precede, and be concatenated with, the user data supplied in the FROM option. "name" (1–8 characters) identifies the attach header control block to be used in the local task.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

CTLCHAR(data-value)

specifies a 1-byte write control character (WCC¹) that controls the CONVERSE command. A COBOL user must specify a data area containing this character. If the option is omitted, all modified data tags are reset to zero and the keyboard is restored.

DEFAULT

set the terminal to use the DEFAULT screen size.

DEFRESP

indicates that a definite response is required when the output operation has been completed.

DEST(name)

specifies the 4-byte symbolic name of the TCAM destination to which the message is to be sent. This option is meaningful only for terminals defined using DFHTCT TYPE=SDSCI with DEVICE=TCAM.

If you use the DEST option, you must be aware of any restrictions placed on device-dependent data streams by the message control facility in use. See the section on the CICS-TCAM interface in the *CICS/ESA Customization Guide* for programming information.

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSZE option in the profile definition.

¹ Documented in the *IBM 3270 Data Stream Programmer's Reference* manual.

FMH

specifies that a function management header has been included in the data to be written. If the ATTACHID option is specified as well, the concatenated FMH flag is set in the attach FMH.

FROM(data-area)

specifies the data to be written to the terminal or logical unit, or sent to the partner transaction. This option may, when relevant, be omitted if ATTACHID is specified.

FROMLENGTH(data-value)

is a fullword alternative to FROMLENGTH.

FROMLENGTH(data-value)

specifies the length, as a halfword binary value, of the data to be written. If you use this option, you must also specify FROM. For a description of a safe upper limit, see "LENGTH options" on page 5.

INTO(data-area)

specifies the receiving field for the data read from the logical unit or terminal.

LEAVEKB

specifies that the keyboard is to remain locked at the completion of the data transfer.

LINEADDR(data-value)

specifies that the writing is to begin on a specific line of a 2260/2265 screen. The data value is a halfword binary value in the range 1 through 12 for a 2260, or 1 through 15 for a 2265.

MAXLENGTH(data-value)

is a fullword alternative to MAXLENGTH.

MAXLENGTH(data-value)

specifies the maximum amount (halfword binary value) of data that CICS is to recover in response to a CONVERSE command. If INTO is specified, MAXLENGTH overrides the use of TOLENGTH as an input to CICS. If SET is specified, MAXLENGTH provides a way for the program to limit the amount of data it receives at one time.

If the value specified is less than zero, zero is assumed.

If the length of data exceeds the value specified and the NOTRUNCATE option is not present, the data is truncated to that value and the LENGERR condition occurs. The data area specified in the TOLENGTH option is set to the original length of data.

If the length of data exceeds the value specified and the NOTRUNCATE option is present, CICS retains the remaining data and uses it to satisfy subsequent RECEIVE commands. The data area specified in the TOLENGTH option is set to the length of data returned.

If no argument is coded for MAXLENGTH, CICS defaults to TOLENGTH.

NOTRUNCATE

specifies that, when the data available exceeds the length requested, the remaining data is not to be discarded but retained for retrieval by subsequent RECEIVE commands.

PSEUDOBIN

specifies that the data being read and written is to be translated from System/7 pseudobinary representation to hexadecimal.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If both this option and CONVID are omitted, the principal facility for the task is used.

SET(ptr-ref)

specifies a pointer reference to be set to the address of data received from the conversation partner in an MRO conversation. The pointer reference, unless changed

by other commands or statements, is valid until the next CONVERSE (MRO) command or the end of task.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

STATE(cvda)

gets the state of the transaction program. The cvda values returned by CICS are:

ALLOCATED
FREE
PENDFREE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

STRFIELD

specifies that the data area specified in the FROM option contains structured fields. If this option is specified, the contents of all structured fields must be handled by the application program. The CONVERSE command, rather than a SEND command, must be used if the data area contains a read partition structured field. (Structured fields are described in the *CICS/ESA 3270 Data Stream Device Guide*.)

CTLCHAR and ERASE are mutually exclusive with STRFIELD, and their use with STRFIELD generates an error message.

TOLENGTH(data-area)

is a fullword alternative to TOLENGTH.

TOLENGTH(data-area)

specifies the length, as a halfword binary value, of the data to be received. If you specify INTO, but omit MAXLENGTH, "data-area" specifies the maximum length that the program accepts. If the value is less than zero, zero is assumed.

If the length of the data exceeds the value specified, but NOTTRUNCATE is omitted, the data is truncated to that value, and the LENGERR condition occurs. When the data is received, the data area is set to the length of the data.

For a description of a safe upper limit, see "LENGTH options" on page 5.

CONVERSE (non-VTAM) conditions

Some of the following conditions can occur in combination with others. CICS checks for these conditions in the following order:

1. INBFMH
2. EOC.

If more than one occurs, only the first is passed to the application program. EIBRCODE, however, is set to indicate all the conditions that occurred.

CBIDERR

occurs if the requested attach header control block named in ATTACHID cannot be found.

Default action: terminate the task abnormally.

EOC

occurs when a request/response unit (RU) is received with the end-of-chain indicator set. Field EIBEOC also contains this indicator.

Default action: ignore the condition.

EOF (not TCAM)

occurs when an end-of-file indicator is received.

Default action: terminate the task abnormally.

INBFMH

occurs if a request/response unit (RU) contains a function management header (FMH). Field EIBFMH contains this indicator and it should be used in preference to INBFMH. The IGNORE CONDITION command can be used to ignore the condition.

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- Data is discarded by CICS because its length exceeds the maximum that the program accepts and the NOTRUNCATE option is not specified.
- An out-of-range value is supplied in the FROMLENGTH option.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

RDATT (not TCAM)

occurs if the "receive" part of the conversation is terminated by the attention (ATTN) key rather than the return key.

Default action: ignore the condition.

TERMERR

occurs for a session-related error.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

WRBRK

occurs if the "send" part of the conversation is terminated by the attention (ATTN) key rather than the return key.

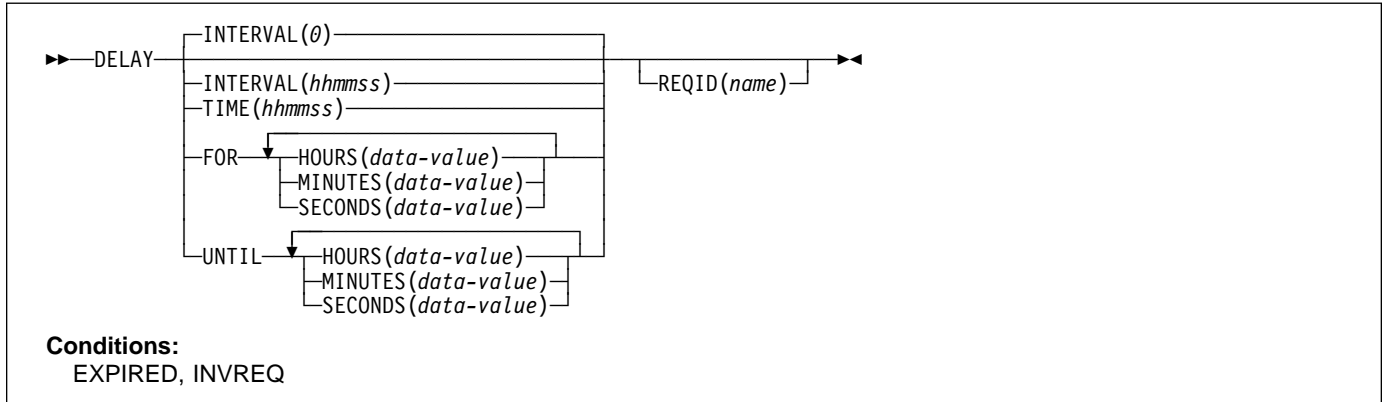
Default action: ignore the condition.

DELAY

Function

Delay the processing of a task.

Command syntax



Note for dynamic transaction routing

Using DELAY with REQID if later CANCELED could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

DELAY suspends the processing of the issuing task for a specified interval of time or until a specified time of day. It supersedes any previously initiated POST command for the task.

| The default is INTERVAL(0), but for C the default is FOR HOURS(0) MINUTES(0) SECONDS(0).

The following example shows you how to suspend the processing of a task for five minutes:

```
EXEC CICS DELAY
      INTERVAL(500)
      REQID('GXLBZQMR')
EXEC CICS DELAY FOR MINUTES(5)
```

The following example shows you how, at 09:00, to suspend the processing of a task until 12:45:

```
EXEC CICS DELAY
      TIME(124500)
      REQID('UNIQCODE')
```

There are two ways to enter the time under FOR or UNTIL.

- A combination of at least two of HOURS(0–99), MINUTES(0–59) and SECONDS(0–59). HOURS(1) SECONDS(3) would mean one hour and three seconds (the minutes default to zero).
- Any one of HOURS(0–99), MINUTES(0–5999), or SECONDS(0–359999). HOURS(1) means one hour. MINUTES(62) means one hour and two minutes.

SECONDS(3723) means one hour, two minutes, and three seconds.

DELAY options

FOR

specifies the duration of the delay.

HOURS(data-value)

a fullword binary value in the range 0–99.

INTERVAL(hhmmss)

specifies, in packed decimal format, the interval of time that is to elapse from the time when the DELAY command is issued. The **mm** and **ss** are in the range 0–59. The time specified is added to the current clock time by CICS when the command is executed to calculate the expiration time.

When using the C language, you are recommended to use the FOR/UNTIL HOURS, MINUTES, and SECONDS options as C does not provide a packed decimal data type. You may use INTERVAL, but if the value specified is **not** an integer constant, the application is responsible for ensuring that the value passed to CICS is in packed decimal format.

MINUTES(data-value)

specifies a fullword binary value in the range 0–59, when HOURS or SECONDS are also specified, or 0–5999 when MINUTES is the only option specified.

DELAY

REQID(name)

specifies a name (1–8 characters), which should be unique, to identify the DELAY request. Using this option to specify an application-defined name enables another transaction to cancel the DELAY request.

To enable other tasks to cancel unexpired DELAY requests, you must make the request identifier dynamically available. For example, storing it in a TS queue, whose name is known to other applications that may want to cancel the DELAY request, is one way you can pass a request identifier to other transactions.

SECONDS(data-value)

specifies a fullword binary value in the range 0–59, when HOURS or MINUTES are also specified, or 0–359 999 when SECONDS is the only option specified.

TIME(hhmmss)

specifies, in packed decimal format, the time when the task should resume processing.

When using the C language, you are recommended to use the FOR/UNTIL HOURS, MINUTES, and SECONDS options as C does not provide a packed decimal data type. You may use TIME, but if the value specified is **not** an integer constant, the application is responsible for ensuring that the value passed to CICS is in packed decimal format. See the section about expiration times in the *CICS/ESA Application Programming Guide*.

UNTIL

specifies the time at the end of the delay and when the task should resume processing.

DELAY conditions

EXPIRED

occurs if the time specified has already expired when the command is issued.

Default action: ignore the condition.

INVREQ

occurs in any of the following situations:

- The DELAY command is not valid for processing by CICS.
- Hours are out of range (RESP2=4)
- Minutes are out of range (RESP2=5)
- Seconds are out of range (RESP2=6).

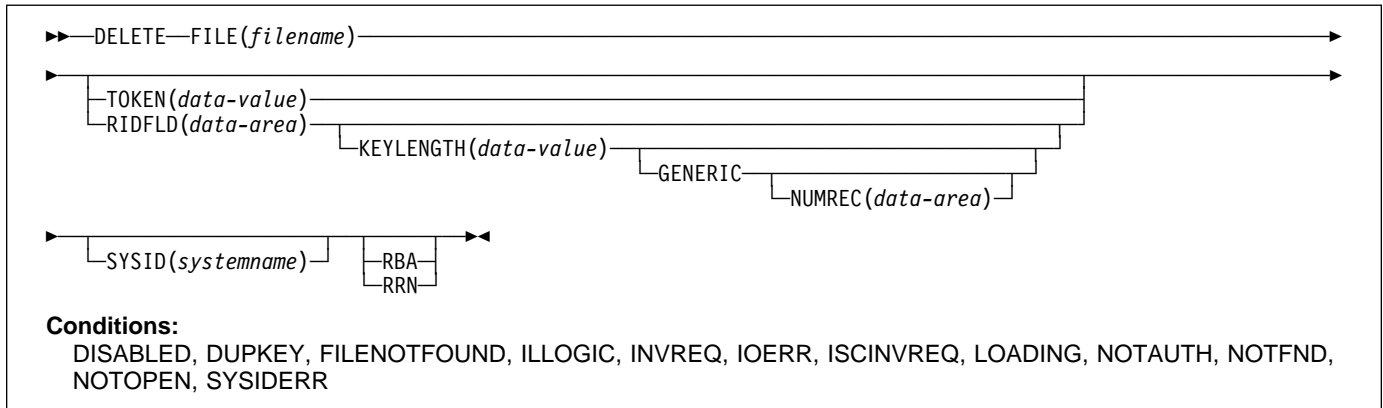
Default action: terminate the task abnormally.

DELETE

Function

| Delete a record from a file — VSAM KSDS, VSAM RRDS, and data tables only.

Command syntax



DELETE deletes a record from a file on a KSDS, a path over a KSDS, a CICS or user-maintained data table, or an RRDS. You cannot delete from a VSAM ESDS. **All references to KSDS apply equally to CICS maintained data tables and, except where stated otherwise, to paths over a KSDS.** The file may be on a local or a remote system. You identify, in the RIDFLD option, the specific record to be deleted.

You can delete a group of records in a similar way with a single invocation of this command, identifying the group by the GENERIC option (not available for user-maintained data tables or RRDS).

You can also use this command to delete a single record that has previously been retrieved for update (by a READ UPDATE command). In this case, you must not specify the RIDFLD option.

When this command is used to delete records from a CICS-maintained data table, the update is made to both the source VSAM KSDS data set and the in-memory data table.

When this command is used to delete records from a user-maintained data table, the update is made only to the in-memory data table.

The following example shows you how to delete a group of records in a VSAM data set:

```
EXEC CICS DELETE
      FILE('MASTVSAM')
      RIDFLD(ACCTNO)
      KEYLENGTH(1en)
      GENERIC
      NUMREC(NUMDEL)
```

DELETE options

FILE(filename)

specifies the name of the file to be accessed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

GENERIC (KSDS)

specifies that the search key is a generic key with a length specified in the KEYLENGTH option. The search for a record is satisfied when a record is found with a key that has the same starting characters (generic key) as those specified.

If you specify GENERIC, you must also specify RIDFLD.

You cannot use GENERIC for user-maintained data tables.

KEYLENGTH(data-value)

specifies the length (halfword binary) of the key that has been specified in the RIDFLD option, except when RBA or RRN is specified, in which case it is not valid. This option must be specified if GENERIC is specified, and it may be specified whenever a key is specified. However, if the length specified is different from the length defined for the data set and the operation is not generic, the INVREQ condition occurs.

The INVREQ condition also occurs if you specify GENERIC, and the KEYLENGTH is not less than that specified in the VSAM definition.

You should not specify a zero value of KEYLENGTH because the results of this are unpredictable.

DELETE

NUMREC(data-area) (KSDS)

specifies a halfword binary data area that CICS sets to the number of deleted records.

RBA (KSDS except for paths over KSDS)

specifies that the record identification field specified in the RIDFLD option contains a relative byte address. This option should be used only when deleting records using relative byte addresses instead of keys to identify the records.

You cannot use RBA for user-maintained data tables.

RIDFLD(data-area)

specifies the record identification field. When combined with RBA or RRN this is a 4-character field. The contents can be a key, a relative byte address, or a relative record number. For a relative byte address or a relative record number, the format of this field must be fullword binary. For a relative byte address, the RIDFLD can be greater than or equal to zero. For a relative record number, the RIDFLD can be greater than or equal to 1.

The contents must be a key for user-maintained data tables.

You must specify this option if you have also specified GENERIC.

RRN (VSAM RRDS)

specifies that the record identification field specified in the RIDFLD option contains a relative record number. This option should be used only with files referencing relative record data sets.

SYSID(systemname)

specifies the name (1–4 characters) of the system the request is directed to.

If you specify SYSID, and omit both RBA and RRN, you must also specify KEYLENGTH; it cannot be found in the FCT.

TOKEN(data-value)

specifies as a fullword binary value a unique request identifier for a DELETE, and is used to associate it with a previous READ for UPDATE request. This is an input value returned by the task to file control.

DELETE conditions

Note: RESP2 values are not set for files that are on remote systems.

DISABLED

occurs if a file is disabled (RESP2=50). A file may be disabled because:

- It was initially defined as disabled and has not since been enabled.
- It has been disabled by a SET FILE or a CEMT SET FILE command.

This condition cannot occur when deleting a record just read for update.

Default action: terminate the task abnormally.

DUPKEY

occurs if a record is accessed by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key follows (RESP2=140).

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if the file name referred to in the FILE option cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- Delete operations are not allowed according to the file entry specification in the FCT (RESP2=20).
- When the user-maintained data table was defined as a CICS file, the definition did not allow for delete operations (RESP2=20).
- A DELETE command is issued for a file referring to a VSAM ESDS (RESP2=21).
- A generic delete is issued for a file that is not a VSAM KSDS (RESP2=22).
- The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is greater than or equal to the length of a full key (RESP2=25).
- The KEYLENGTH option is specified (but the GENERIC option is not specified), and the specified length does not equal the length defined for the data set to which this file refers (RESP2=26).
- A DELETE command is issued for a file referring to a BDAM data set (RESP2=27).
- A DELETE command without the RIDFLD option is issued for a file for which no previous READ UPDATE command has been issued (RESP2=31).
- The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than zero (RESP2=42).
- The DELETE command does not conform to the format of DELETE for a user-maintained data table;

for example if GENERIC were specified (RESP2=44).

- A DELETE instruction includes a token whose value cannot be matched against any token in use for an existing READ for UPDATE request (RESP2=47).
- An attempt is made to function-ship a request which includes a TOKEN keyword (RESP2=48).

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the file control operation (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error.

(Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

LOADING

occurs if a delete request is issued for a user-maintained data table that is currently being loaded (RESP2=104).

A user-maintained data table cannot be modified during loading.

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTFND

occurs if an attempt to delete a record based on the search argument provided is unsuccessful (RESP2=80).

A NOTFND condition occurs, for VSAM only, if the RIDFLD specifies a track address outside the extent of a BDAM data set.

For user-maintained data tables, this condition occurs if an attempt to delete a record is unsuccessful because there is no entry with the specified key in the data table.

This can occur on an attempt to delete a record using a
DELETE without RIDFLD, if the delete is associated with
a READ UPDATE request for a record that this
transaction has deleted (using DELETE with RIDFLD)
after it was read for update (RESP2=80).

This does not mean that there is no such record in the source data set; it may be that such a record is present but was either rejected during initial loading by the user exit XDTRD, or was subsequently deleted from the user-maintained data table.

Default action: terminate the task abnormally.

NOTOPEN

occurs in any of the following situations (RESP2=60):

- The requested file is CLOSED and UNENABLED. The CLOSED, UNENABLED state is reached after a CLOSE request has been received against an OPEN ENABLED file and the file is no longer in use. This state can also be specified as the initial state by means of the FILSTAT parameter of the file control table TYPE=FILE, or by defining a file using the RDO options STATUS=UNENABLED and OPENTIME=FIRSTREF.
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.

This condition does not occur if the request is made to either a CLOSED, ENABLED file or a CLOSED, DISABLED file. In the first case, the file is opened as part of executing the request. In the second case, the DISABLED condition occurs.

This condition also cannot occur when deleting a record just read for update.

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed (RESP2=130).

Default action: terminate the task abnormally.

DELETEQ TD

Function

Delete all transient data.

Command syntax

```
▶▶—DELETEQ TD—QUEUE(name)—SYSID(systemname)—▶▶
```

Conditions:

DISABLED, INVREQ, ISCINVREQ, NOTAUTH, QIDERR, SYSIDERR

DELETEQ TD deletes all the transient data associated with a particular intrapartition destination (queue). All storage associated with the destination is released (deallocated). Note that you cannot use this command to delete an **extrapartition** transient data queue. An attempt to do so results in an INVREQ condition.

DELETEQ TD options

QUEUE(*name*)

specifies the symbolic name (1–4 alphanumeric characters) of the queue to be deleted. The name must have been defined in the DCT.

If SYSID is specified, the queue is assumed to be on a remote system irrespective of whether or not the name is defined in the DCT. Otherwise the entry in the DCT is used to find out whether the data set is on a local or a remote system.

SYSID(*systemname*) — remote systems only

specifies the name (1–4 characters) of the system the request is directed to.

NOTAUTH

occurs when a resource security check has failed on QUEUE(*name*).

Default action: terminate the task abnormally.

QIDERR

occurs if the symbolic destination to be used with DELETEQ TD cannot be found.

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed.

Default action: terminate the task abnormally.

DELETEQ TD conditions

DISABLED

occurs when the queue has been disabled.

Default action: terminate the task abnormally.

INVREQ

occurs if DELETEQ names an extrapartition queue.

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

DELETEQ TS

Function

Delete a temporary storage queue.

Command syntax

```
▶▶—DELETEQ TS—QUEUE(name)—SYSID(systemname)—▶▶
```

Conditions:

INVREQ, ISCINVREQ, NOTAUTH, QIDERR, SYSIDERR

Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

DELETEQ TS deletes all the temporary data associated with a temporary storage queue. All storage associated with the queue is freed.

You should delete temporary data as soon as possible to avoid using excessive amounts of storage.

When a recoverable temporary storage queue is deleted, you must issue a syncpoint before issuing a subsequent WRITEQ TS for the same queue.

QIDERR

occurs when the specified queue cannot be found in either main or auxiliary storage.

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed.

Default action: terminate the task abnormally.

DELETEQ TS options

QUEUE(*name*)

specifies the symbolic name (1–8 alphanumeric characters) of the queue to be deleted. The name may not consist solely of binary zeros and must be unique within the CICS system.

SYSID(*systemname*) — remote systems only

specifies the name (1–4 characters) of the system the request is directed to.

DELETEQ TS conditions

INVREQ

occurs when the queue was created by CICS internal code.

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on QUEUE(*name*).

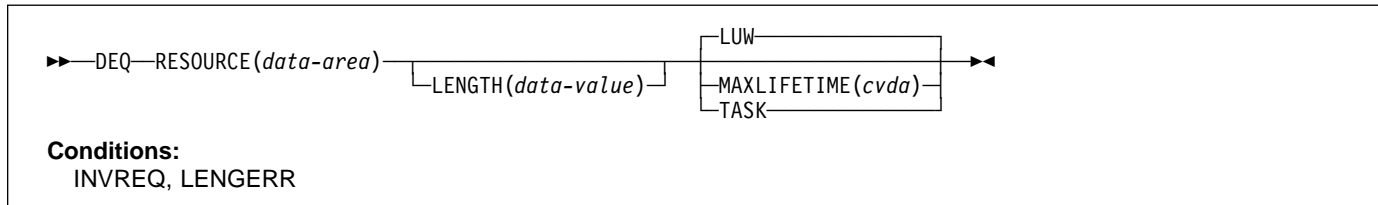
Default action: terminate the task abnormally.

DEQ

Function

Schedule use of a resource by a task (dequeue).

Command syntax



Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

DEQ causes a resource currently enqueued on by the task to be released for use by other tasks.

If a task enqueues on, but does not dequeue from, a resource, CICS automatically releases the resource during syncpoint processing or when the task is terminated. A resource in the context of this command is any string of 1–255 bytes, established by in-house standards, to protect against conflicting actions between tasks, or to cause single-threading within a program.

When issuing the DEQ command, the resource to be dequeued from must be identified by the method used when enqueueing on the resource. If no enqueue has been issued for the resource, the dequeue is ignored.

If more than one ENQ command is issued for a resource by a task, that resource remains owned by the task until the task issues a matching number of DEQ commands.

The following examples show how to dequeue from a resource:

```
EXEC CICS DEQ
      RESOURCE(RESNAME)
```

```
EXEC CICS DEQ
      RESOURCE(SOCSECNO)
      LENGTH(8)
```

DEQ options

LENGTH(*data-value*)

specifies that the resource to be dequeued from has a length given by the data value. The data value is a halfword binary value in the range 1 through 255. If the value you specify is outside this range, a LENGERR condition occurs. If the LENGTH option is specified in an ENQ command, it must also be specified in the DEQ command for that resource, and the values of these options must be the same.

MAXLIFETIME(*cvda*)

specifies the duration of the ENQ being released. The values passed to CICS are:

LUW ENQ was acquired with a duration of a logical unit of work. This is the default value.

TASK ENQ was acquired with a duration of a task.

RESOURCE(*data-area*)

specifies either an area whose address represents the resource to be dequeued from, or a variable that contains the resource (an employee name, for example). In the latter case, you must use the LENGTH option.

DEQ conditions

INVREQ

occurs if the MAXLIFETIME option is set with an incorrect CVDA (RESP2=2).

Default action: terminate the task abnormally.

LENGERR

means that the value you specified for the LENGTH option is outside the range 1 through 255 (RESP2=1).

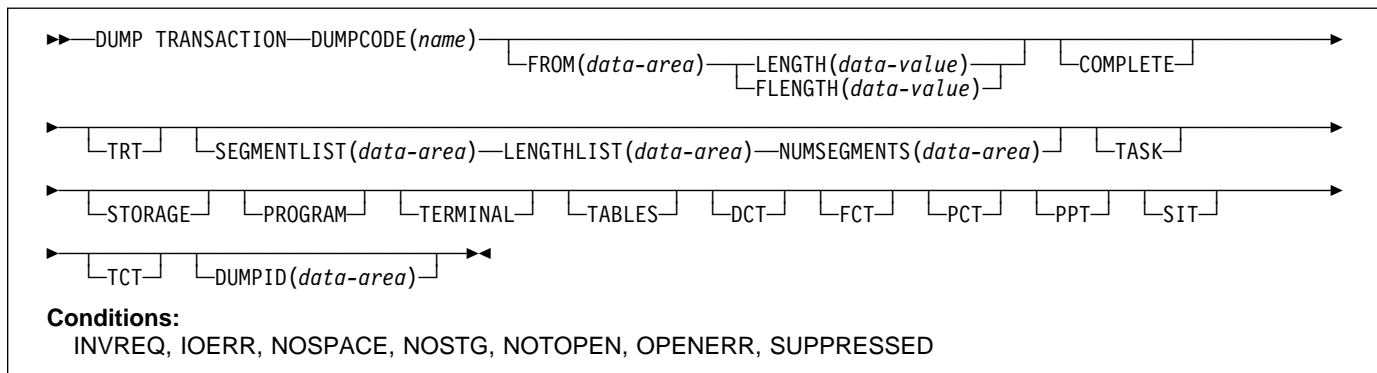
| Default action: terminate the task abnormally.

DUMP TRANSACTION

Function

Request a transaction dump.

Command syntax



DUMP TRANSACTION dumps all, a series, or any single main storage area related to a task, any or all of the CICS tables (DCT, FCT, PCT, PPT, SIT, or TCT), or all of these together.

Note that if you issue a DUMP TRANSACTION for a DUMPCODE that is defined in the transaction dump table with SYSDUMP, you also get a system dump.

The following example shows how to request a dump of all the task-related storage areas, the terminal control table, and a specified data area:

```
EXEC CICS DUMP TRANSACTION
      DUMPCODE('name')
      FROM(data-area)
      LENGTH(data-value)
```

This second example (written in PL/I) shows you a case in which five task-related storage areas are dumped:

```
DCL storage_address(5)  POINTER,
     storage_length(5)  FIXED BIN(31),
     nsegs              FIXED BIN(31);
storage_address(1) = ADDR(areal);
storage_length(1)  = CSTG(areal);
:
nsegs = 5;
EXEC CICS DUMP TRANSACTION
      DUMPCODE('name')
      SEGMENTLIST(storage_address)
      LENGTHLIST(storage_length)
      NUMSEGMENTS(nsegs);
```

If there is no entry in the system dump table for the specified DUMPCODE, a temporary entry is made. This entry is lost on the next CICS start. The system dump table is described in the *CICS/ESA Problem Determination Guide*.

DUMP TRANSACTION options

COMPLETE

dumps all main storage areas related to a task, all the CICS tables, and the DL/I control blocks.

DCT

dumps the destination control table.

DUMPCODE(name)

specifies a name (1–4 characters) that identifies the dump. If the name contains any leading or imbedded blanks, the dump is produced but INVREQ is raised. No entry is added to the system dump table.

If you omit all the options except DUMPCODE, you get the same TASK, but without the DL/I control blocks.

DUMPID(data-area)

is a number returned by the system that identifies the particular dump.

FCT

dumps the file control table.

FLENGTH(data-value)

specifies the length (fullword binary value) of the storage area (specified in the FROM option) that is to be dumped. The maximum length that you can specify is 16777215 bytes.

FLENGTH and LENGTH are mutually exclusive.

FROM(data-area)

dumps the specified data area, which must be a valid area; that is, storage allocated by the operating system within the CICS region. In addition, the following areas are dumped:

- Task control area (TCA) and, if applicable, the transaction work area (TWA).

- Common system area (CSA), including the user's portion of the CSA (CWA).

#

Apar 63695

#

Documentation for Apar 63695 added 6 Jan 1995 (TUCKER)

#

- If TRAN is specified for the TRTRANTY SIT parameter, only the trace entries associated with the current task are formatted. If TRTRANTY=ALL is specified, the entire internal trace table is formatted. This only applies when the CICS trace facility is active.
- Either the terminal control table terminal entry (TCTTE) or the destination control table entry associated with the requesting task.

Whenever the TCTTE is dumped, the terminal control table user area (if any) and the message control blocks (if any) associated with the TCTTE are dumped. The latter are used by basic mapping support.

LENGTH(data-value)

specifies the length (halfword binary) of the data area specified in the FROM option. For a description of a safe upper limit, see "LENGTH options" on page 5.

LENGTH and FLENGTH are mutually exclusive.

LENGTHLIST(data-area)

specifies a list of fullword binary values showing the lengths of the storage areas to be dumped. This corresponds to the list of segments specified in the SEGMENTLIST option. You must use both the SEGMENTLIST and NUMSEGMENTS options when you use the LENGTHLIST option.

NUMSEGMENTS(data-area)

specifies the number (fullword binary) of areas to be dumped. You must use both the SEGMENTLIST and LENGTHLIST options when you use the NUMSEGMENTS option.

PCT

formats a summary of each installed transaction definition.

PPT

.dumps the processing program table.

PROGRAM

specifies that program storage areas associated with this task are to be dumped, as follows:

- Task control area (TCA) and, if applicable, the transaction work area (TWA)
- Common system area (CSA), including the user's portion of the CSA (CWA)
- Trace entries relating to the task written to the internal trace table (only when the CICS trace facility is active)

- All program storage areas containing user-written application programs active on behalf of the requesting task
- Register save areas (RSAs) indicated by the RSA chain off the TCA
- Either the terminal control table terminal entry (TCTTE) or the destination control table entry associated with the requesting task.

Whenever the TCTTE is dumped, the terminal control table user area (if any) and the message control blocks (if any) associated with the TCTTE are dumped.

SEGMENTLIST(data-area)

specifies a list of addresses, which are the starting points of the segments to be dumped. Each segment is a task-related storage area. You must use both the NUMSEGMENTS and LENGTHLIST options when you use the SEGMENTLIST option.

SIT

.dumps the system initialization table.

STORAGE

specifies that storage areas associated with this task are to be dumped, as follows:

- Task control area (TCA) and, if applicable, the transaction work area (TWA)
- Common system area (CSA), including the user's portion of the CSA (CWA)
- Trace entries relating to the task written to the internal trace table (only when the CICS trace facility is active)
- All transaction storage areas
- Either the terminal control table terminal entry (TCTTE) or the destination control table entry associated with the requesting task.

Whenever the TCTTE is dumped, the terminal control table user area (if any) and the message control blocks (if any) associated with the TCTTE are dumped.

TABLES

.dumps the DCT, FCT, PCT, PPT, SIT, and the TCT.

TASK

specifies that storage areas associated with this task are to be dumped, as follows:

- A summary of the transaction environment associated with this task
- Common system area (CSA), including the user's portion of the CSA (CWA)
- Trace entries relating to the task written to the internal trace table (only when the CICS trace facility is active)
- All program storage areas containing user-written application programs active on behalf of the requesting task

DUMP TRANSACTION

- All transaction storage areas
- Either the terminal control table terminal entry (TCTTE) or the destination control table entry associated with the requesting task
- Register save areas (RSAs) indicated by the RSA chain off the TCA
- All terminal input/output areas (TIOAs) chained off the terminal control table terminal entry (TCTTE) for the terminal associated with the requesting task
- DL/I control blocks.

Whenever the TCTTE is dumped, the terminal control table user area (if any) and the message control blocks (if any) associated with the TCTTE are dumped.

TCT

dumps the terminal control table.

TERMINAL

specifies that storage areas associated with the terminal are to be dumped, as follows:

- Task control area (TCA) and, if applicable, the transaction work area (TWA)
- Common system area (CSA), including the user's portion of the CSA (CWA)
- Trace entries relating to the task written to the internal trace table (only when the CICS trace facility is active)
- All terminal input/output areas (TIOAs) chained off the terminal control table terminal entry (TCTTE) for the terminal associated with the requesting task as long as the request is not a write, or storage freezing is on for the task or the terminal
- Either the terminal control table terminal entry (TCTTE) or the destination control table entry associated with the requesting task.

Whenever the TCTTE is dumped, the terminal control table user area (if any) and the message control blocks (if any) associated with the TCTTE are dumped. The latter are used by basic mapping support.

TRT

dumps the trace entries relating to the task written to the internal trace table.

DUMP TRANSACTION conditions

INVREQ

occurs if an incorrect DUMPCODE is specified. DUMPCODE contains unprintable characters, or leading or imbedded blanks (RESP2=13).

The dump is produced but no entry is added to the system dump table.

Default action: terminate the task abnormally.

IOERR

occurs in any of the following situations:

- The SDUMP process is not authorized (RESP2=9).
- An error occurred during system dumping (RESP2=10).
- The CICS routine issuing the SDUMP is unable to establish a recovery routine (FESTAE) (RESP2=13).

Default action: terminate the task abnormally.

NOSPACE

occurs if the transaction dump is incomplete due to lack of space (RESP2=4).

Default action: terminate the task abnormally.

NOSTG

occurs if CICS has run out of working storage (RESP2=5).

Default action: terminate the task abnormally.

NOTOPEN

occurs if the current CICS dump data set is not open (RESP2=6).

Default action: terminate the task abnormally.

OPENERR

occurs if there is an error on opening, closing, or writing to the current CICS dump routine (RESP2=7).

Default action: terminate the task abnormally.

SUPPRESSED

occurs in any of the following situations:

- The transaction dump is suppressed by MAXIMUM in table (RESP2=1).
- The transaction dump is suppressed by NOTRANDUMP in table (RESP2=2).
- The transaction dump is suppressed by a user exit program (RESP2=3).

Default action: terminate the task abnormally.

ENDBR

Function

End browse of a file.

Command syntax

```
▶ ENDBR FILE(filename) [REQID(data-value)] [SYSID(systemname)] ◀
```

Conditions:

FILENOTFOUND, ILLOGIC, INVREQ, ISCINVREQ, NOTAUTH, SYSIDERR

ENDBR ends a browse on a file or data table (CICS or user-maintained) on a local or a remote system.

You should always issue an end browse (ENDBR) command before performing any update operations on the same data set (READ UPDATE, DELETE with RIDFLD, or WRITE), and before a syncpoint. You only need issue ENDBR after a successful STARTBR.

ENDBR options

FILE(filename)

specifies the name of the file being browsed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

REQID(data-value)

specifies a unique (halfword binary value) request identifier for a browse, used to control multiple browse operations on a data set. If this option is not specified, a default value of zero is assumed.

SYSID(systemname)

specifies the name (1–4 characters) of the system the request is directed to.

(See EIBRCODE in the EXEC interface block; refer to Appendix A, “EXEC interface block” on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs if the REQID, if any, does not match that of any successful STARTBR command (RESP2=35).

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed (RESP2=130).

Default action: terminate the task abnormally.

ENDBR conditions

Note: RESP2 values are not set for files that are on remote systems.

FILENOTFOUND

occurs if the name referred to in the FILE option cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

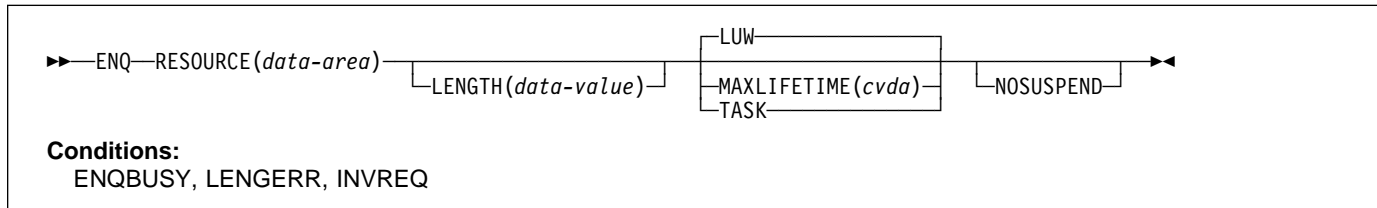
ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

ENQ

Function

Schedule use of a resource by a task (enqueue).



Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

ENQ causes further execution of the task issuing the ENQ command to be synchronized with the availability of the specified resource; control is returned to the task when the resource is available.

A resource in the context of this command is any string of 1–255 bytes, established by in-house standards, to protect against conflicting actions between tasks, or to cause single threading within a program.

If a task enqueues on a resource but does not dequeue from it, CICS automatically releases the resource during syncpoint processing (including DL/I, PCB, and TERM calls), or when the task is terminated. Option LUW on MAXLIFETIME forces the dequeue at the end of a logical unit of work (LUW). Option TASK on MAXLIFETIME forces the dequeue at the end of a task. If there are several LUWs in a task, the enqueue carries over the LUWs.

If more than one ENQ command is issued for the same resource by a given task, the resource remains owned by that task until the task issues a matching number of DEQ commands.

The resource to be enqueued on must be identified by one of the following methods:

- Specifying a data area that is the resource. It is the location (address) of the data area in storage that matters, not its contents. Two tasks, enqueueing on the same resource and using this method, must refer to the same location in storage. They could both, for example, refer to the same location in the CWA.

```
EXEC CICS ENQ
      RESOURCE (RESNAME)
```

- Specifying a data area that contains a unique argument (for example, an employee name) that represents the resource. It is the contents of the data value that matters, not its location. Two tasks, enqueueing on the same resource and using this method, can refer to the

same location or to different locations, but the contents of the locations must be the same. The length must be supplied in the LENGTH option.

```
EXEC CICS ENQ
      RESOURCE (SOCSECNO)
      LENGTH (8)
```

Resource unavailability

If a resource is not available when ENQUEUED, the application program is suspended until it becomes available. However, if the NOSUSPEND option has been specified and the resource is unavailable, the ENQBUSY condition is raised. It is also raised if you have previously issued #HANDLE CONDITION ENQBUSY, even if NOSUSPEND is #not coded. This allows the application program to handle the case of resource unavailability (by HANDLE CONDITION ENQUEUE) without waiting for the resource to become available.

ENQ options

LENGTH(data-value)

specifies that the resource to be enqueued on has a length given by the data value. The data value is a halfword binary value in the range 1–255. If the value you specify is outside this range, a LENGERR condition is issued. If the LENGTH option is specified in an ENQ command, it must also be specified in the DEQ command for that resource, and the values of these options must be the same. You must specify LENGTH when using the method that specifies a data value containing a unique argument, but not for the method that specifies a data area as the resource. It is the presence or absence of LENGTH that tells CICS which method you are using.

MAXLIFETIME(cvda)

specifies the duration of the ENQ to be automatically released by CICS. The values that can be passed to CICS are:

- | | |
|-------------|--|
| LUW | The duration of the ENQ is a logical unit of work. Examples are a syncpoint rollback or syncpoint, if the application does not issue a DEQ before the LUW ends. This is the default value. |
| TASK | The duration of the ENQ is a task. The enqueue carries over the LUWs within the task. Use MAXLIFETIME(TASK) with great care because other tasks issuing ENQ commands on the same resource could be suspended until the end of this task. |

There are two ways to code this option.

- You can assign a cvda value with the translator routine DFHVALUE. This allows you to change a cvda value in the program. For example:


```
MOVE DFHVALUE(LUW) TO AREA-A
EXEC CICS ENQ RESOURCE(RESNAME) MAXLIFETIME(AREA-A)
```
- If the required action is always the same, you can declare the value directly. For example:


```
EXEC CICS ENQ RESOURCE(RESNAME) LUW
or
EXEC CICS ENQ RESOURCE(RESNAME) TASK
```

NOSUSPEND

specifies that the application program is not to be suspended if the resource on the ENQ command is unavailable, but the ENQBUSY condition occurs.

RESOURCE(data-area)

Identifies the resource to be enqueued on by:

- Specifying an area whose address represents the resource.
- Specifying a variable that contains the resource (an employee name, for example). In this case, you must use the LENGTH option.

ENQ conditions**ENQBUSY**

occurs when an ENQ command specifies a resource that is unavailable and the NOSUSPEND option is specified.

If the NOSUSPEND option is not specified, and the ENQ command specifies a resource that is unavailable, the application program is suspended and the ENQBUSY condition is not raised.

Default action: ignore the condition.

INVREQ

occurs if the MAXLIFETIME option is set with an incorrect CVDA (RESP2=2).

Default action: terminate the task abnormally.

LENGERR

occurs when the value specified for the LENGTH option is outside the range 1 through 255 (RESP2=1).

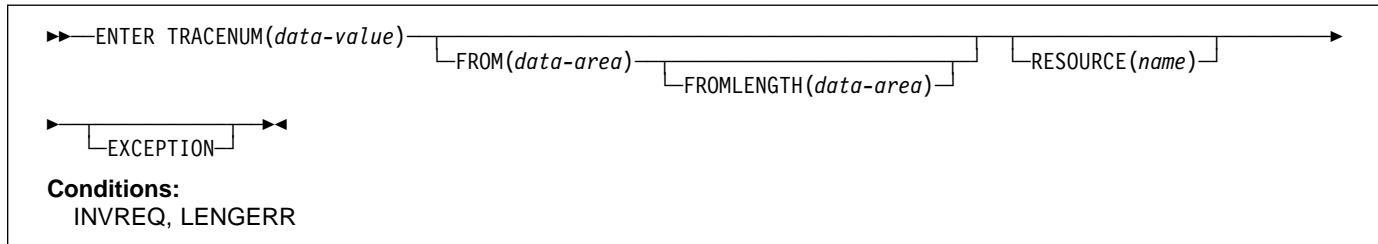
Default action: terminate the task abnormally.

ENTER TRACENUM

Function

Write a trace entry.

Command syntax



The ENTER TRACENUM command makes a trace entry in the currently active trace destinations. CICS writes the trace entry only if the master and user trace flags are on, unless you specify the EXCEPTION option, in which case a user trace entry is always written, even if the master and user trace flags are off. Exception trace entries are always written to the internal trace table (even if internal tracing is set off), but they are written to other destinations only if they are active.

You can use the exception trace option in an application program to write a trace entry when the program detects an exception or abnormal condition. To do this, include an ENTER TRACENUM(data-value) EXCEPTION command in your program's exception or abnormal condition error-handling routine.

To write an exception trace entry in an error situation when an application program has given up control, you can issue an ENTER TRACENUM(data-value) EXCEPTION command from a user-written program error program (PEP). See the *CICS/ESA Customization Guide* for programming information about modifying the DFHPEP program.

Note: ENTER TRACENUM replaces the earlier ENTER TRACEID command, which is still supported for compatibility with releases of CICS earlier than Version 3. You should use ENTER TRACENUM for all new programs, and whenever you apply maintenance to old programs.

For information about the trace entry format, see the *CICS/ESA Problem Determination Guide*.

The following COBOL example shows how to write a user trace entry with a trace identifier of 123, with trace data from a data area called USER-TRACE-ENTRY:

```

EXEC CICS ENTER TRACENUM(123)
      FROM(USER-TRACE-ENTRY)
END-EXEC.
    
```

ENTER TRACENUM options

EXCEPTION

specifies that CICS is to write a user exception trace entry. The EXCEPTION option overrides the master user trace flag, and CICS writes the trace entry even if the user trace flag is off. Exception trace entries are identified by the characters *EXCU when the trace entries are formatted by the trace utility program. See the *CICS/ESA Problem Determination Guide* for more information about user exception trace entries.

FROM(data-area)

specifies a data area whose contents are to be entered into the data field of the trace table entry. If you omit the FROM option, two fullwords of binary zeros are passed.

FROMLENGTH(data-area)

specifies a halfword binary data area containing the length of the trace data, in the range 0–4000 bytes. If FROMLENGTH is not specified, a length of 8 bytes is assumed.

RESOURCE(name)

specifies an 8-character name to be entered into the resource field of the trace table entry.

TRACENUM(data-value)

specifies the trace identifier for a user trace table entry as a halfword binary value in the range 0 through 199.

ENTER TRACENUM conditions

INVREQ

occurs in any of the following situations:

- TRACENUM is outside the range 0 through 199 (RESP2=1)
- There is no valid trace destination (RESP2=2)
- The user trace flag is set OFF and EXCEPTION has not been specified (RESP2=3).

Default action: terminate the task abnormally.

LENGERR

- | occurs when FROMLENGTH is outside the range 0 through 4000 (RESP2=4).
- | Default action: terminate the task abnormally.

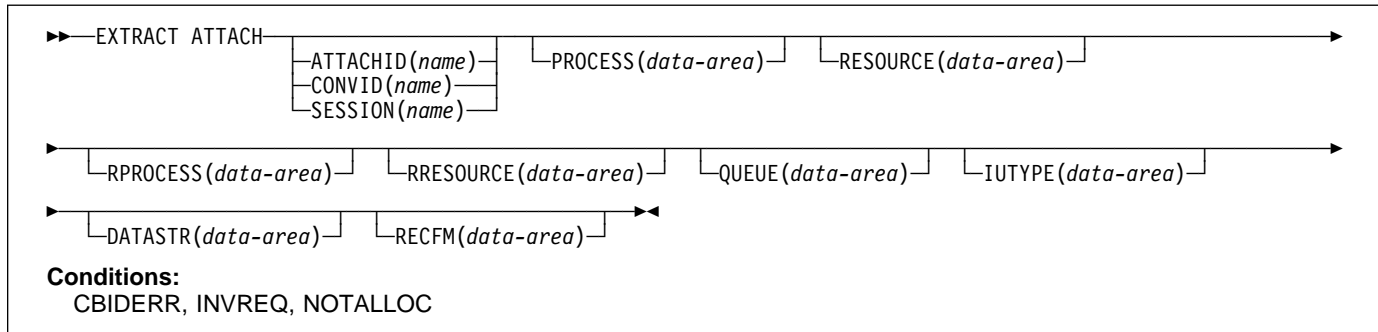
EXTRACT ATTACH (LUTYPE6.1)

EXTRACT ATTACH (LUTYPE6.1)

Function

Retrieve values from an LUTYPE6.1 attach header.

Command syntax



EXTRACT ATTACH retrieves a set of values that are held in an attach header control block, or that have been built previously. For the command to retrieve information from a received attach FMH (Function Management Header), EIBATT must have been set during a RECEIVE or CONVERSE command.

0-7	reserved - must be set to zero
8-11	0000 - user-defined
	1111 - SCS data stream
	1110 - 3270 data stream
	1101 - structured field
	1100 - logical record management
12-15	defined by the user if bits 8-11 are set to 0000; otherwise reserved (must be set to zero)

EXTRACT ATTACH (LUTYPE6.1) options

ATTACHID(name)

specifies that values are to be retrieved from an attach header control block. The name (1-8 characters) identifies this control block to the local task.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

DATASTR(data-area)

corresponds to the data stream profile field, ATTDSP, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is given by CICS to the data stream profile field in an attach FMH. For most CICS applications, the option can be omitted.

The value returned in the data area is a halfword binary value. Only the low-order byte is used; the SNA-defined meanings of the bits are as follows:

IUTYPE(data-area)

corresponds to the interchange unit field, ATTIU, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the interchange unit field in an attach FMH. For most CICS applications the option can be omitted.

The value returned in the data area is a halfword binary value. Only the low-order 7 bits are used; the SNA-defined meanings of the bits are as follows:

0-10	reserved - must be set to zero
11	0 - not end of multichain interchange unit
	1 - end of multichain interchange unit
12,13	reserved - must be set to zero
14,15	00 - multichain interchange unit
	01 - single-chain interchange unit
	10 - reserved
	11 - reserved

PROCESS(data-area)

corresponds to the process name, ATDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, a transaction running in one system can acquire a session to the second system and can identify the transaction to be attached; in the second system the identification is carried in the first chain of data sent across the session.

In general, the first four bytes of data identify the transaction to be attached. However an attach FMH, identifying the transaction to be attached, can be built and sent. The receiving CICS system uses just the first four bytes of the process name as a transaction name.

No significance is attached by CICS to process names in attach FMHs sent in chains of data other than the first.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the process name field in an attach FMH.

QUEUE(data-area)

corresponds to the queue name, ATTDQN, in an attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the queue name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the queue name field in an attach FMH.

RECFM(data-area)

corresponds to the deblocking algorithm field, ATTDDBA, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the deblocking algorithm field in an attach FMH. For most CICS applications, the option can be omitted.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the interchange unit field in an attach FMH.

The value returned in the data area is a halfword binary value. Only the low-order byte is used; the SNA-defined meanings of the bits are as follows:

0-7	reserved - must be set to zero
8-15	X'00' - reserved
	X'01' - variable-length
	variable-blocked
	X'02' - reserved
	X'03' - reserved
	X'04' - chain of RUs
	X'05' through X'FF' - reserved

RESOURCE(data-area)

corresponds to the resource name, ATTPRN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the resource name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the resource name field in an attach FMH.

RPROCESS(data-area)

corresponds to the return process name, ATTRDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return process name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return process name field in an attach FMH.

RRESOURCE(data-area)

corresponds to the return resource name, ATTRPRN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return resource name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return resource name field in an attach FMH.

SESSION(name)

specifies the symbolic identifier (1-4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

EXTRACT ATTACH (LUTYPE6.1) conditions

CBIDERR

occurs if the requested attach header control block cannot be found.

Default action: terminate the task abnormally.

INVREQ

occurs if incorrect data is found.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

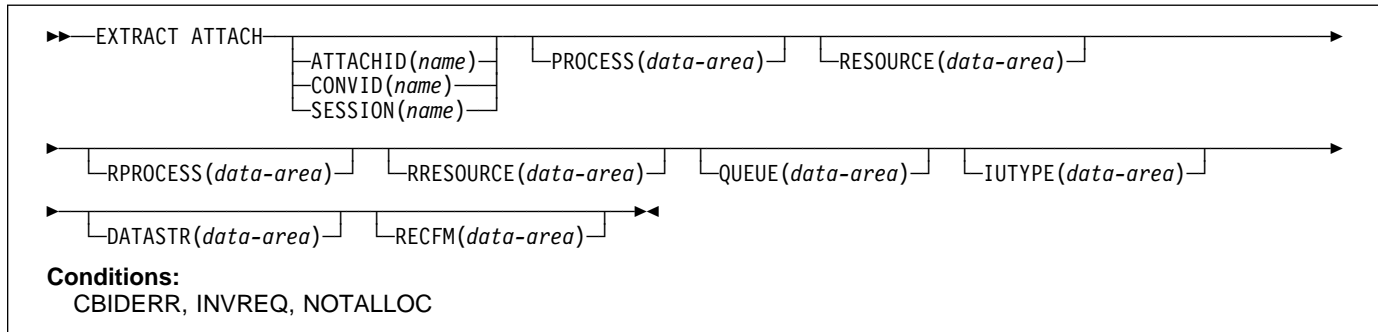
EXTRACT ATTACH (MRO)

EXTRACT ATTACH (MRO)

Function

Retrieve values from an MRO attach header.

Command syntax



EXTRACT ATTACH retrieves a set of values that are held in an attach header control block, or that have been built previously. For the command to retrieve information from a received attach FMH (Function Management Header), EIBATT must have been set during a RECEIVE or CONVERSE command.

For more information about MRO and IRC, see the *CICS/ESA Intercommunication Guide*.

EXTRACT ATTACH (MRO) options

ATTACHID(name)

specifies that values are to be retrieved from an attach header control block. The name (1–8 characters) identifies this control block to the local task.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

DATASTR(data-area)

corresponds to the data stream profile field, ATTDSP, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is given by CICS to the data stream profile field in an attach FMH. For most CICS applications, the option can be omitted.

The value returned in the data area is a halfword binary value. Only the low-order byte is used; the SNA-defined meanings of the bits are as follows:

0-7	reserved - must be set to zero
8-11	0000 - user-defined
	1111 - SCS data stream
	1110 - 3270 data stream
	1101 - structured field
	1100 - logical record management
12-15	defined by the user if bits 8-11 are set to 0000; otherwise reserved (must be set to zero)

IUTYPE(data-area)

corresponds to the interchange unit field, ATTIU, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the interchange unit field in an attach FMH. For most CICS applications the option can be omitted.

The value returned in the data area is a halfword binary value. Only the low-order 7 bits are used; the SNA-defined meanings of the bits are as follows:

0-10	reserved - must be set to zero
11	0 - not end of multichain interchange unit
	1 - end of multichain interchange unit
12,13	reserved - must be set to zero
14,15	00 - multichain interchange unit
	01 - single chain interchange unit
	10 - reserved
	11 - reserved

PROCESS(data-area)

corresponds to the process name, ATTDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, a transaction running in one system can acquire a session to the second system and can identify the transaction to be attached; in the second system the identification is carried in the first chain of data sent across the session.

In general, the first four bytes of data identify the transaction to be attached. However an attach FMH, identifying the transaction to be attached, can be built and sent. The receiving CICS system uses just the first four bytes of the process name as a transaction name.

No significance is attached by CICS to process names in attach FMHs sent in chains of data other than the first.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the process name field in an attach FMH.

QUEUE(data-area)

corresponds to the queue name, ATTDQN, in an attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the queue name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the queue name field in an attach FMH.

RECFM(data-area)

corresponds to the deblocking algorithm field, ATTDDBA, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no particular significance is attached by CICS to the deblocking algorithm field in an attach FMH. For most CICS applications, the option can be omitted.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the interchange unit field in an attach FMH.

The value returned in the data area is a halfword binary value. Only the low-order byte is used; the SNA-defined meanings of the bits are as follows:

0-7	reserved - must be set to zero
8-15	X'00' - reserved
	X'01' - variable-length variable-blocked
	X'02' - reserved
	X'03' - reserved
	X'04' - chain of RUs
	X'05' through X'FF' - reserved

RESOURCE(data-area)

corresponds to the resource name, ATTPRN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the resource name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the resource name field in an attach FMH.

RPROCESS(data-area)

corresponds to the return process name, ATTRDPN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return process name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return process name field in an attach FMH.

RRESOURCE(data-area)

corresponds to the return resource name, ATTRPRN, in an LUTYPE6.1 attach FMH.

For communication between two CICS systems, no significance is attached by CICS to the return resource name in an attach FMH.

For communication between a CICS system and another subsystem, refer to documentation supplied by the subsystem about how to use the return resource name field in an attach FMH.

SESSION(name)

specifies the symbolic identifier (1-4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

EXTRACT ATTACH (MRO) conditions

CBIDERR

occurs if the requested attach header control block cannot be found.

Default action: terminate the task abnormally.

INVREQ

occurs if incorrect data is found.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

EXTRACT ATTRIBUTES (APPC)

Function

Obtains the state of the APPC conversation.

Command syntax

```
▶—EXTRACT ATTRIBUTES—┐
                       └─CONVID(name)─┘—STATE(cvda)—▶
```

Conditions:

INVREQ, NOTALLOC

EXTRACT ATTRIBUTES extracts conversation state information for APPC mapped conversations.

EXTRACT ATTRIBUTES (APPC) options

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

By default, the principal facility is assumed.

STATE(*cvda*)

gets the state of the transaction program. The *cvda* values returned by CICS are:

```
ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND
```

EXTRACT ATTRIBUTES (APPC) conditions

INVREQ

occurs in any one of the following situations:

- The command is issued against a CPI-Communications conversation.
- The command is issued against an APPC basic conversation. (A GDS EXTRACT ATTRIBUTES should have been used instead.)
- A distributed program link server application explicitly, or implicitly by default, specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if the CONVID value specified in the command does not relate to a conversation owned by the application.

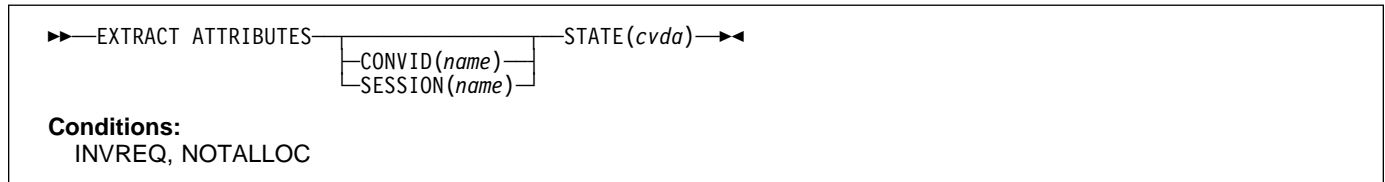
Default action: terminate the task abnormally.

EXTRACT ATTRIBUTES (MRO)

Function

Extract attributes from an MRO conversation.

Command syntax



EXTRACT ATTRIBUTES (MRO) extracts conversation state information for MRO conversations.

EXTRACT ATTRIBUTES (MRO) options

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

If both this option and CONVID are omitted, the principal facility for the task is used.

STATE(cvda)

gets the state of the transaction program. The cvda values returned by CICS are:

```

ALLOCATED
FREE
PENDFREE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSSEND

```

EXTRACT ATTRIBUTES (MRO) conditions

INVREQ

occurs in any one of the following situations:

- An incorrect command has been issued for the terminal or LU in use.
- A distributed program link server application explicitly, or implicitly by default, specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

EXTRACT LOGONMSG

Function

Access VTAM logon data.

Command syntax

```

▶▶ EXTRACT LOGONMSG INTO(data-area) LENGTH(data-area)
    SET(ptr-ref)
  
```

Condition: NOTALLOC

EXTRACT LOGONMSG accesses VTAM logon data. This data may have been specified by the terminal operator at logon or in the ISSUE PASS command, for example. This data is only available if the system initialization parameter LGNMSG=YES is specified. The data can only be extracted once. It is possible to force the first transaction that runs on the terminal to be that which issues EXTRACT LOGONMSG by using the the system initialization parameter GMTRAN.

All the logon data is extracted and its length placed in the field specified by the LENGTH option. Because the LENGTH option cannot be used to limit the amount of data extracted, it is recommended that a field of 256 bytes is always used for this option.

| If you use the SET option, the VTAM logon data is not freed
 | until the session terminates (CLSDST). If you use the INTO
 | option, the VTAM logon data is copied into user storage and
 | then freed.

EXTRACT LOGONMSG options

INTO(*data-area*)

specifies the receiving field for the data extracted.

LENGTH(*data-area*)

specifies the length, as a halfword binary value, of the data extracted. If no data is available, LENGTH is set to zero.

SET(*ptr-ref*)

specifies the pointer reference that is to be set to the address of the data extracted. The pointer reference, unless changed by other commands or statements, is valid until the next EXTRACT LOGONMSG command or the end of task.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

EXTRACT LOGONMSG condition

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

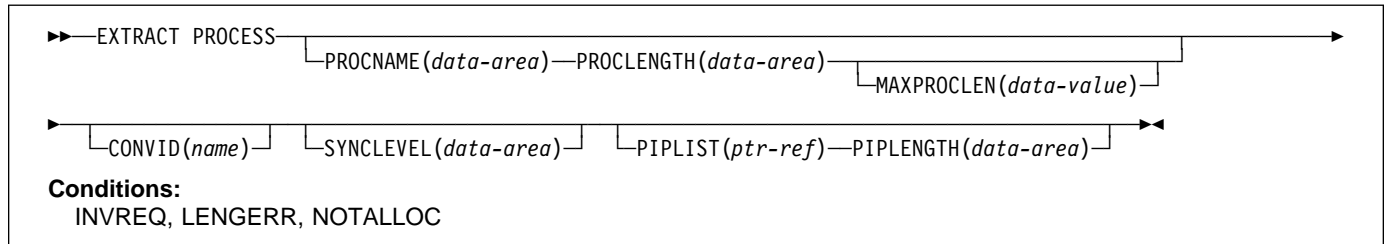
Default action: terminate the task abnormally.

EXTRACT PROCESS

Function

Retrieve values from APPC conversation attach header.

Command syntax



EXTRACT PROCESS lets an application program access conversation-related data, specified to CICS when the program is attached. The attach receiver does not have to execute an EXTRACT PROCESS command unless it requires this information.

The EXTRACT PROCESS command is valid only on an APPC conversation that is the principal facility for the task.

EXTRACT PROCESS options

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies the token representing the principal session.

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If this option is omitted, the principal facility is assumed.

MAXPROCLEN(data-value)

specifies the buffer length of PROCNAME. If MAXPROCLEN is not specified, the buffer is assumed to have 32 bytes.

PIPELENGTH(data-area)

specifies a halfword binary data area in which the total length of the process initialization parameter (PIP) list is returned.

PIPLIST(ptr-ref)

specifies a pointer reference that is set to the address of a CICS-provided data area containing a PIP list. This list contains variable-length records in the same format as the list in the CONNECT PROCESS command. A returned value of zero means that no PIP data has been received by CICS.

PROCLENGTH(data-area)

specifies a halfword data area that is set by CICS to the length of the process name. If PROCNAME is specified, this option must be specified.

PROCNAME(data-area)

specifies the data area to receive the process name specified by the remote system that caused the task to start. The data area can be 1–64 bytes long. The process name is padded on the right with blanks if it is too short. The PROCNAME data area should not be shorter than the MAXPROCLEN value.

SYNCLEVEL(data-area)

specifies a halfword data area that is set by CICS to the SYNCLEVEL value. For further information about synchronization levels, see the *CICS/ESA Intercommunication Guide*.

EXTRACT PROCESS conditions

INVREQ

occurs in any of the following situations:

- EXTRACT PROCESS has been used on a conversation other than APPC mapped (for example, LUTYPE6.1, APPC basic, or CPI Communications).
- EXTRACT PROCESS has been used on a conversation that was not started by input from the network, and whose session is not a principal facility.
- The command is issued against a CPI-Communications conversation.
- A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs if the actual length of PROCNAME is greater than MAXPROCLEN, or greater than 32 bytes if MAXPROCLEN is not specified.

Default action: terminate the task abnormally.

EXTRACT PROCESS

NOTALLOC

occurs if the CONVID value specified in the command does not relate to a conversation owned by the application.

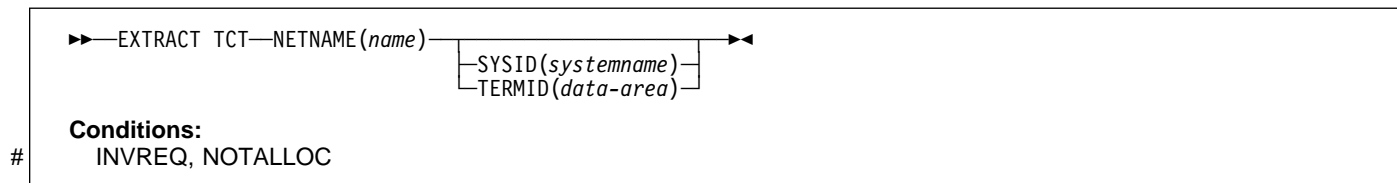
Default action: terminate the task abnormally.

EXTRACT TCT

Function

Convert an 8-character name to a 4-character name on an LUTYPE6.1 logical unit.

Command syntax



EXTRACT TCT converts the 8-character VTAM network name for a logical unit into the corresponding 4-character name it is known by in the local CICS system.

EXTRACT TCT options

NETNAME(*name*)

specifies the 8-character name of the logical unit in the VTAM network.

SYSID(*systemname*)

specifies the variable to be set to the equivalent local name of the system.

TERMID(*data-area*)

specifies the variable to be set to the equivalent local name of the terminal.

EXTRACT TCT conditions

INVREQ

occurs if NETNAME is not valid.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not

owned by the application.

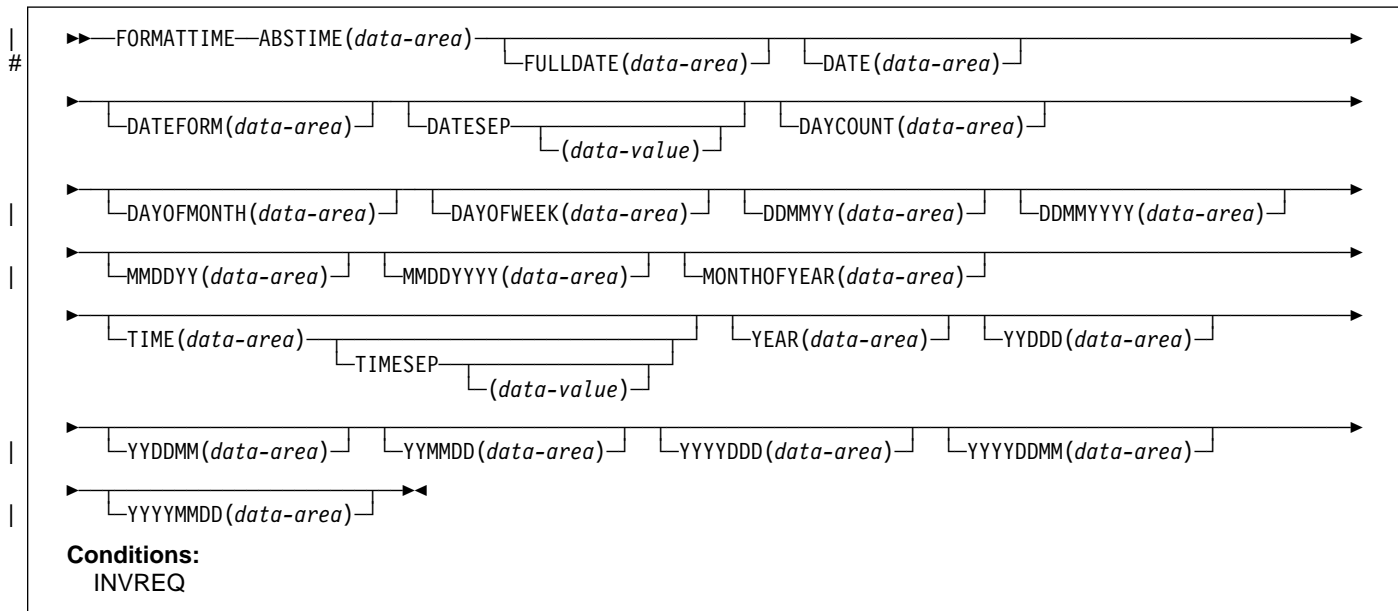
Default action: terminate the task abnormally.

FORMATTIME

Function

Transform absolute date and time into a specified format.

Command syntax



FORMATTIME transforms the absolute date and time into any of a variety of formats. Normally, the ABSTIME argument is the value returned by an ASKTIME ABSTIME command.

To obtain an elapsed time in a particular format, the ABSTIME data value can be the difference between two values returned by ASKTIME, and options such as DAYCOUNT(d) and TIME(t) can be specified.

If the milliseconds come to 500 or more, the returned seconds and, if necessary, the minutes and hours, are rounded up. The day, month, and year are, however, never rounded up.

```
# — Apar 76793 78317 —
# Documentation for Apar 76793 78317 added 28 Nov
# 1995 (TUCKER)
```

However, in the case where the ABSTIME argument contains a value representing the half-second before midnight, no rounding is performed, and the TIME parameter returns 23:59:59.

The following example shows the effect of some of the options of the command. Let "utime" contain the value 002837962864828 in milliseconds.

```
EXEC CICS ASKTIME ABSTIME(utime)
EXEC CICS FORMATTIME ABSTIME(utime)
        DATESEP('-') DDMMYY(date)
        TIME(time) TIMESEP
```

This gives the values 06-12-89 for "date" and 19:01:05 for "time".

FORMATTIME options

ABSTIME(data-area)

specifies the data value for the time, in packed decimal, since 00:00 hours on 1 January 1900 (in milliseconds rounded to the nearest hundredth of a second) that is to be converted to an alternative format.

The format of the parameter is:

```
COBOL: PIC S9(15) COMP-3
C:      char data_ref[8];
PL/I:   FIXED DEC(15);
ASM:    PL8
```

DATE(data-area)

specifies the variable that is to receive the date in the format specified in the DATFORM system initialization parameter. The returned value is in 8-character format. You should normally use this option only when a date is needed for output purposes. Where a date is needed for analysis, you should request the date in explicit form, for example, using the MMDDYY option.

DATEFORM(data-area)

specifies the format of the installation-defined date. CICS returns YYMMDD, DDMMYY, or MMDDYY (six

characters) according to the DATFORM system initialization parameter.

DATESEP(data-value)

specifies the character to be inserted as the separator between the year and the month, and between the day and the month; or between the year and the day if form YYDDD is specified.

If you omit this option, no separator is supplied. If you omit "data-value", a slash (/) is assumed as the separator.

DAYCOUNT(data-area)

returns the number of days since 1 January 1900 (day 1), as a fullword binary number. This is useful if you need to compare the current date with a previous date that has, for example, been stored in a data set.

DAYOFMONTH(data-area)

returns the number of the day in the month as a fullword binary number.

DAYOFWEEK(data-area)

returns the relative day number of the week as a fullword binary number: Sunday=0, Saturday=6. This number can be converted to a textual form of day in any language.

DDMMYY(data-area)

specifies the 8-character user field where CICS is to return the date, in day/month/year format (for example, 21/10/95). A separator is present if requested by the DATESEP option. If no separator is requested, the returned value is left-justified in the 8-character user field.

#

DDMMYYYY(data-area)

specifies the 10-character user field where CICS is to return the date, in day/month/year format (for example 17/06/1995). A separator is present if requested by the DATESEP option. If no separator is requested, the returned value is left-justified in the 10-character user field.

#

<p>Apar PQ00059</p> <p>Documentation for Apar PQ00059 added 20/03/97</p>

FULLDATE(data-area)

specifies the 10-character user field where CICS is to return the date, in the format specified in the DATFORM system initialization parameter, with the year expanded to 4 digits. A separator is present if requested by the DATESEP option. If no separator is requested, the returned value is left-justified in the 10-character user field. You should normally use this option only when a date is needed for output purposes. Where a date is needed for analysis, you should request the date in explicit form, for example, using the MMDDYYYY option.

MMDDYY(data-area)

specifies the 8-character user field in which CICS is to return the date, in month/day/year format (for example,

10/21/95). A separator is present if requested by the DATESEP option. If no separator is requested, the returned value is left-justified in the 8-character user field.

| MMDDYYYY(data-area)

specifies the 10-character user field where CICS is to return the date, in month/day/year format (for example 11/21/1995).

MONTHOFYEAR(data-area)

"data-area" is set to the relative month number of the year as a fullword binary number (January=1, December=12). You can convert this number, in your application program, to the name of the month in any language.

TIME(data-area)

"data-area" is set as an 8-character field to the current 24-hour clock time in the form hh:mm:ss, where the separator is specified by the TIMESEP option.

If the milliseconds come to 500 or more, the returned seconds and, if necessary, the minutes and hours, are rounded up. The day, month, and year are, however, never rounded up.

TIMESEP(data-value)

specifies the character to be used as the separator in the returned time. If you omit this option, no separator is assumed and six bytes are returned in an 8-character field. If you omit the "data-value", a colon (:) is used as a separator.

YEAR(data-area)

specifies the full 4-figure number of the year as a fullword binary number (for example, 1995, 2001).

YYDDD(data-area)

specifies the 6-character user field where CICS is to return the date, in year/day format (for example, 95/301). A separator is present if requested by the DATESEP option. If no separator is requested, the returned value is left-justified in the 6-character user field.

#

YYDDMM(data-area)

specifies the 8-character user field where CICS is to return the date, in year/day/month format (for example, 95/30/10). A separator is present if requested by the DATESEP option. If no separator is requested, the returned value is left-justified in the 8-character user field.

#

YYMMDD(data-area)

specifies the 8-character user field where CICS is to return the date, in year/month/day format (for example, 95/10/21).

| YYYYDDD(data-area)

specifies the 8-character user field where CICS is to return the date, in year/day format (for example 1995/200). A separator is present if requested by the DATESEP option. If no separator is requested, the

FORMATTIME

returned value is left-justified in the 8-character user
field.

| YYYYDDMM(data-area)

| specifies the 10-character user field where CICS is to
| return the date, in year/day/month format (for example
| 1995/21/06). A separator is present if requested by the
DATESEP option. If no separator is requested, the
returned value is left-justified in the 10-character user
field.

| YYYYMMDD(data-area)

| specifies the 10-character user field where CICS is to
| return the date, in year/month/day format (for example
| 1995/06/21). A separator is present if requested by the
DATESEP option. If no separator is requested, the
returned value is left-justified in the 10-character user
field.

FORMATTIME condition

INVREQ

occurs if the ABSTIME option is in an incorrect form
(RESP2=1).

Default action: terminate the task abnormally.

FREE**Function**

Return a terminal or logical unit.

Command syntax

```
▶—FREE—▶
```

Conditions:
NOTALLOC

FREE returns a terminal or logical unit when the transaction owning it no longer requires it. The principal facility is freed.

If you are running EDF, and the transaction frees the principal facility, EDF is terminated.

FREE conditions**NOTALLOC**

occurs if the task is not associated with the terminal.

Default action: terminate the task abnormally.

FREE (APPC)

FREE (APPC)

Function

Return an APPC mapped session to CICS.

Command syntax



FREE returns an APPC session to CICS when a transaction owning it no longer requires it. The session can then be allocated for use by other transactions.

| If you omit CONVID, the principal facility is freed. Facilities not freed explicitly are freed by CICS when the task terminates.

If you are running EDF, and the transaction frees the principal facility, EDF is terminated.

FREE (APPC) options

CONVID(name)

identifies the APPC mapped session to be freed. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

If this option is omitted, the principal facility is assumed.

STATE(cvda)

gets the state of the current conversation. The STATE option on a FREE command returns a cvda code of 00 if there is no longer an active conversation. The other output cvda values are:

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

FREE (APPC) conditions

INVREQ

occurs in any of the following situations:

- The CONVID value specified in the command relates to a basic (unmapped) APPC conversation.
- The CONVID value specified in the command relates to a CPI-Communications conversation.
- A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

| occurs if the specified CONVID value does not relate to a conversation owned by the application.

Default action: terminate the task abnormally.

FREE (LUTYPE6.1)

Function

Return LUTYPE6.1 sessions to CICS.

Command syntax



FREE returns an LUTYPE6.1 session to CICS when a transaction owning it no longer requires it. The session can then be allocated for use by other transactions.

If you omit both CONVID and SESSION, the principal facility is freed. Facilities not freed explicitly are freed by CICS when the task terminates.

If you are running EDF, and the transaction frees the principal facility, EDF is terminated.

FREE (LUTYPE6.1) options

CONVID(name)

identifies the LUTYPE6.1 session to be freed. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

FREE (LUTYPE6.1) conditions

INVREQ

occurs if the session specified in the command was allocated for a basic (unmapped) APPC conversation.

(See also EIBRCODE in Appendix A, “EXEC interface block” on page 343.)

Default action: terminate the task abnormally.

NOTALLOC

occurs if the session specified in the command is not owned by the application.

Default action: terminate the task abnormally.

FREE (MRO)

FREE (MRO)

Function

Return MRO sessions to CICS.

Command syntax



FREE returns an MRO session to CICS when a transaction owning it no longer requires it. The session can then be allocated for use by other transactions.

If you omit both CONVID and SESSION, the principal facility is freed. Facilities not freed explicitly are freed by CICS when the task terminates.

If you are running EDF, and the transaction frees the principal facility, EDF is terminated.

FREE (MRO) options

CONVID(name)

identifies the MRO session to be freed. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

STATE(cvda)

gets the state of the current conversation. The STATE on a FREE command returns a cvda code of 00 if there is no longer an active conversation. The other output cvda values are:

ALLOCATED
FREE
PENDFREE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

FREE (MRO) conditions

INVREQ

occurs in any one of the following situations:

- The session specified in the command was allocated for a basic (unmapped) APPC conversation
- The session is in the wrong state to be freed.

(See also EIBRCODE in Appendix A, “EXEC interface block” on page 343.)

Default action: terminate the task abnormally.

NOTALLOC

occurs if the session specified in the command is not owned by the application.

Default action: terminate the task abnormally.

FREEMAIN

Function

Release main storage acquired by a GETMAIN command.

Command syntax

```

  ► FREEMAIN DATA(data-area)
  └─┬─ DATAPOINTER(ptr-value)
    └─►

```

Conditions:
INVREQ

Note for dynamic transaction routing

Using FREEMAIN of storage GETMAINED with SHARED, or of a resource defined with RELOAD=YES that has been LOADED could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

FREEMAIN releases main storage previously acquired by a GETMAIN command issued by the application, or by a LOAD for a program, map, or table, defined with RELOAD=YES. If the task that GETMAINED the storage or LOADED the program does not release it, CICS releases it at task end, unless:

- The GETMAIN command specified the SHARED option
- The program is defined with RELOAD=YES
- The program is defined with RELOAD=NO but was LOADED with the HOLD option.

In the first two cases, the storage remains allocated until some other task issues a FREEMAIN to release it. In the third case, the program remains available until it is RELEASED by another task.

You can release CICS-key storage from a program only if it is being executed in CICS key. If the previously-acquired storage was obtained from CICS-key storage, and the program issuing the FREEMAIN is in user-key, an INVREQ condition occurs with a RESP2 value of 2.

Here are some examples of the use of FREEMAIN.

COBOL

```

DATA DIVISION.
WORKING-STORAGE SECTION.
77 AREA-POINTER    USAGE IS POINTER.
LINKAGE SECTION.
01 WORKAREA        PIC X(100).
PROCEDURE DIVISION.
EXEC CICS GETMAIN SET(AREA-POINTER)
LENGTH(100)
END-EXEC.

.
SET ADDRESS OF WORKAREA TO AREA-POINTER.

.
EXEC CICS FREEMAIN DATA(WORKAREA)
END-EXEC.
EXEC CICS RETURN
END-EXEC.

```

FREEMAIN

Alternatively, the previous COBOL example could free the storage by the following command:

```
EXEC CICS FREEMAIN DATAPOINTER(AREA-POINTER)
END-EXEC.
```

C

```
#pragma XOPTS(CICS);
#define MAINSIZE 100;
main()
{
    char          *buffer;
    struct eib_record dfheiptr;
    EXEC CICS ADDRESS EIB(dfheiptr);
    EXEC CICS GETMAIN SET(buffer)
                LENGTH(MAINSIZE);
    buffer[2] = 'a';
    .
    .
    EXEC CICS FREEMAIN DATA(buffer);
    EXEC CICS RETURN;
}
```

PL/I

```
DCL AREA_PTR    POINTER,
      WORKAREA  CHAR(100) BASED(AREA_PTR);
.
.
EXEC CICS GETMAIN SET(AREA_PTR) LENGTH(100);
.
EXEC CICS FREEMAIN DATA(WORKAREA);
```

Assembler

```
WORKAREA DS CL100
.
.
EXEC CICS GETMAIN SET(9) LENGTH(100)
USING WORKAREA,9
EXEC CICS FREEMAIN DATA(WORKAREA)
```

Alternatively, you can free storage using the DATAPOINTER option as shown in the following example:

```
WORKAREA DS CL100
.
EXEC CICS GETMAIN SET(9) LENGTH(100)
USING WORKAREA,9
.
.
DROP 9
.
EXEC CICS FREEMAIN DATAPOINTER(9)
```

FREEMAIN option

DATA(data-area)

specifies the data area of main storage to be released.

This storage must have been acquired by a previous GETMAIN command, except in the case of BMS pages. (For more guidance about BMS pages, see the description of the SET option in the *CICS/ESA Application Programming Guide*.)

Note that this option specifies the data area that was acquired by the GETMAIN command, not the pointer reference that was set to that address. You must use the DATAPOINTER option to specify a pointer-reference: DATA and DATAPOINTER are mutually exclusive. Therefore, in assembler language, "data-area" must be a relocatable expression that is a data reference; in COBOL or C it must be a data name; and in PL/I it must be a data reference. (See the *CICS/ESA Application Programming Guide* for a discussion of argument values.)

The length of storage released is the length obtained by the GETMAIN and not necessarily the length of the data area.

| DATAPOINTER(ptr-value)

specifies the address of the main storage to be released. This option is an alternative to the DATA option, and specifies the pointer reference that was returned by a GETMAIN command using the SET option.

The length of storage released is the length obtained by the GETMAIN.

FREEMAIN condition

INVREQ

occurs in any one of the following situations:

- The storage specified by the DATA or DATAPOINTER parameter is not storage acquired by a GETMAIN command (RESP2=1).
- The storage area specified by the DATA or DATAPOINTER parameter is in CICS-key storage, and the program issuing the FREEMAIN command is in user-key (RESP2=2).

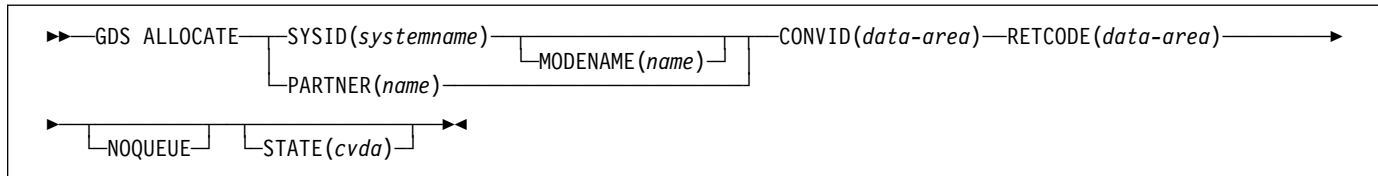
Default action: terminate the task abnormally.

GDS ALLOCATE

Function

Acquire a session to a remote system for use by APPC basic conversation (assembler-language and C programs only).

Command syntax



GDS ALLOCATE acquires a session to a remote system.

The return code is given in RETCODE (see Table 1 on page 103). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS ALLOCATE options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVID(data-area)

specifies the 4-character application data area that is to contain the token returned by an ALLOCATE command to identify the allocated conversation. This token is required in subsequent GDS commands issued on the conversation.

MODENAME(name)

specifies the name of the mode group from which the session is to be acquired. If you specify SYSID and omit MODENAME, CICS selects a modename from those defined for the system.

NOQUEUE

specifies that the request to allocate a session is not to be queued when a suitable APPC session cannot be acquired immediately. A session is acquired immediately only if it is a bound contention winner that is not already allocated to another conversation.

The return code in RETCODE indicates whether or not a session has been acquired.

If the NOQUEUE option is not used, a delay may occur before control is passed back to the application program. A delay can occur for any of the following reasons:

- All sessions for the specified SYSID and MODENAME are in use.
- The CICS allocation algorithm has selected a session that is not currently bound (in which case, CICS has to bind).

- The CICS allocation algorithm has selected a contention loser (in which case, CICS has to bid).

If there is a delay, the program waits until the session has been acquired.

PARTNER(name)

specifies the name (8 characters) of a set of definitions that include the names of a remote LU (NETNAME) and a communication profile to be used on the allocated session. For APPC basic conversations, the only relevant attribute set by the profile is MODENAME.

If you use this option as an alternative to SYSID and MODENAME, CICS uses the NETNAME and MODENAME from the PARTNER definition.

RETCODE(data-area)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

```

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND
  
```

SYSID(systemname)

specifies the remote system to which an APPC session is to be allocated. The name, which is 1–4 characters, identifies an entry (defined as an APPC connection) in the CICS terminal control table.

Table 1. GDS ALLOCATE return codes

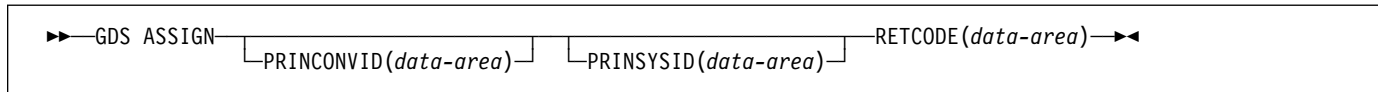
RETCODE (hexadecimal)	Description
01 0C 00	SYSID is unrecognized.
01 0C 04	SYSID is not an LUTYPE6.2 connection name.
01 04 04	NOQUEUE is specified but no bound connection-winner sessions are available.
01 04 08	MODENAME is not known.
01 04 0C	The MODENAME value is SNASVCMG which is restricted to use by CICS.
01 04 0C	VTAM has no class of service (COS) table for the MODENAME value.
01 04 10	The task was canceled during queuing of the command.
01 04 14	All modegroups are closed.
01 04 14	The requested modegroup is closed.
01 04 18	The requested modegroup is draining (closing down).
01 08 00	All sessions in the requested modegroup are unusable.
01 08 00	The connection is in quiesce state.
01 08 00	The connection is out of service.
01 08 00	The connection is not acquired.
01 08 00	The requested modegroup's local max (maximum permitted number of sessions) is 0.
01 08 00	The VTAM ACB is closed.
01 0C 14	The NETNAME specified in the PARTNER definition is not known.
02 0C 00	PARTNER is not known.
06 00 00	The PROFILE specified in the PARTNER definition is not known.

GDS ASSIGN

Function

Get identifier of principal facility in use by APPC basic conversation (assembler-language and C programs only).

Command syntax



GDS ASSIGN gets the identifier of the principal facility.

The return code is given in RETCODE (see Table 2). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS ASSIGN options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

PRINCONVID(data-area)

specifies a 4-byte data area in which the conversation token (CONVID) of the principal facility is to be returned.

PRINSYSID(data-area)

specifies a 4-byte data area in which the SYSID of the principal facility is to be returned.

RETCODE(data-area)

specifies the 6-byte application data area into which return code information is to be moved.

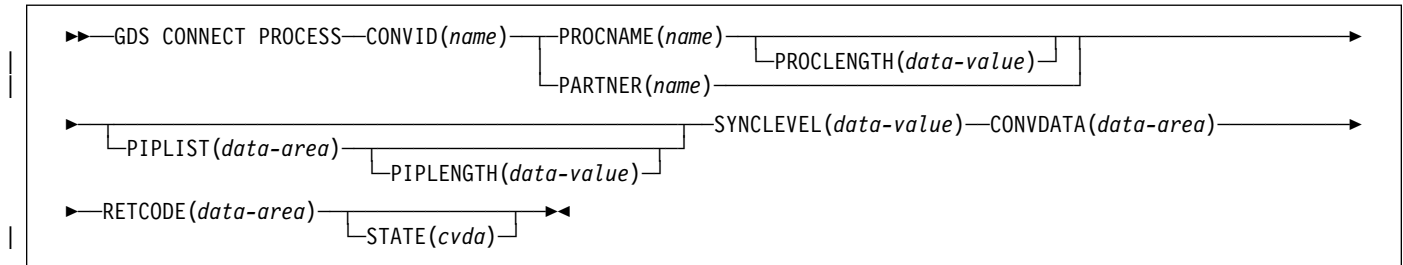
<i>Table 2. GDS ASSIGN return codes</i>	
RETCODE (hexadecimal)	Description
03 00	Principal facility is not APPC.
03 04	Principal facility is not basic.
04	No terminal principal facility exists.

GDS CONNECT PROCESS

Function

Initiate APPC basic conversation (assembler-language and C programs only).

Command syntax



EXEC CICS conditions are never raised on GDS commands.

The return code is given in RETCODE (see Table 3 on page 106). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS CONNECT PROCESS allows the application program to specify a partner application that is to run in the remote system.

GDS CONNECT PROCESS options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(data-area)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

PARTNER(name)

specifies the name (8 characters) of a set of definitions that includes the name (or extended name) of a remote partner transaction (TPNAME or XTPNAME). You can use this option as an alternative to PROCNAME and PROCLENGTH.

PIPLENGTH(data-value)

specifies the total length of the process initialization parameter (PIP) list specified on a CONNECT PROCESS command.

PIPLIST(data-area)

specifies the PIP data that is to be sent to the remote process. See the *CICS/ESA Distributed Transaction Programming Guide* for information about PIP data.

PROCLENGTH(data-value)

specifies the length (as a halfword binary value in the range 1–64) of the target process name.

PROCNAME(name)

specifies the name of the remote application. The APPC architecture allows names of lengths (1–64 bytes), but leaves each product free to set its own maximum. If the remote system is CICS, you can use the standard 4-character transaction ID. If the remote system is CICS/ESA Version 3.2.1 or higher, you can also use the TPNAME value in the transaction definition.

RETCODE(data-area)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

GDS CONNECT PROCESS

SYNCLEVEL(data-value)

specifies the synchronization level (halfword binary value) desired for the current conversation. The possible values are:

- 0 None
- 1 Confirm
- 2 Syncpoint.

Table 3. GDS CONNECT PROCESS return codes

RETCODE (hexadecimal)	Description
02 0C 00	PARTNER is not known.
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 0C	The SYNCLEVEL option specifies a value other than 0, 1, or 2.
03 0C	The SYNCLEVEL option requested either 1 or 2, but it was unavailable.
03 08	A state check occurred.
04	CONVID is for a session that is not allocated to the task, or that is a relay link.
05 00 00 00 00 20	PROCLENGTH is outside the range 1–64.
05 00 00 00 7F FF	The PIPELENGTH value is outside the range 4–763.
05 00 00 00 7F FF	The 2-byte length field (LL) for one of the PIPs is less than 4.
05 00 00 00 7F FF	The total of the LLs in PIP data is greater than the PIPELENGTH value.

GDS EXTRACT ATTRIBUTES

Function

Access state information on an APPC basic conversation (assembler-language and C programs only).

Command syntax

```

  ──▶ GDS EXTRACT ATTRIBUTES ── CONVID(name) ──┬── STATE(cvda) ──┬── CONVDATA(data-area) ── RETCODE(data-area) ──▶
  ───────────────────────────────────────────┘──────────────────────────────────────────┘

```

GDS EXTRACT ATTRIBUTES accesses state information about an APPC basic conversation.

The return code is given in RETCODE (see Table 4). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS EXTRACT ATTRIBUTES options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

Table 4. GDS EXTRACT ATTRIBUTES return codes

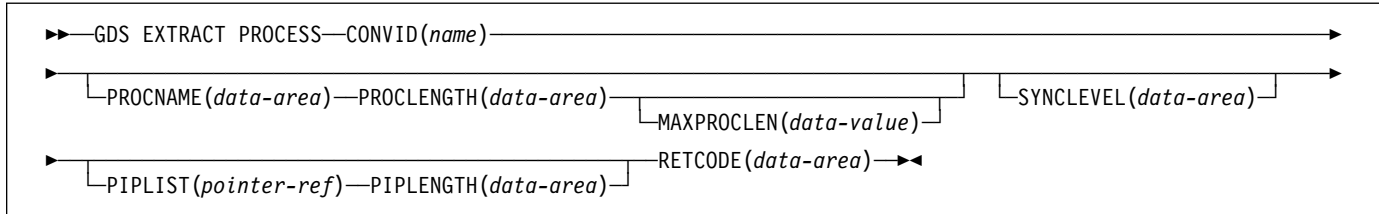
RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 01	INVREQ for a DPL server program.
03 04	CONVID is for a conversation that is not basic.
04	CONVID is for a session that is not allocated to the task, or that is a relay link.

GDS EXTRACT PROCESS

Function

Retrieve values from an APPC basic conversation (assembler-language and C programs only).

Command syntax



GDS EXTRACT PROCESS retrieves values from an APPC basic conversation. The data retrieved is valid only when the command is issued against an APPC basic principal facility.

The return code is given in RETCODE (see Table 5). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS EXTRACT PROCESS options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVID(name)

identifies the conversation the command relates to. The 4-character name identifies the token representing the principal session (returned by a previously executed GDS ASSIGN command).

MAXPROCLEN(data-value)

specifies the length (1–64 characters) of the PROCNAME data area. If MAXPROCLEN is not specified, the buffer is assumed to have 32 bytes.

PIPLENGTH(data-area)

specifies a halfword binary data area that is to receive the length of the PIPLIST received by a GDS EXTRACT PROCESS command.

PIPLIST(pointer-ref)

specifies the pointer reference that is to be set to the address of the PIPLIST received by a GDS EXTRACT PROCESS command. A zero setting indicates that no PIPLIST was received.

PROCLENGTH(data-area)

specifies a halfword binary data area that is set to the actual length of the process name.

PROCNAME(data-area)

specifies the application target data area (1–64 bytes) into which the process name, specified in the APPC

attach function management header, is to be moved. The area is padded with blanks, if necessary.

RETCODE(data-area)

specifies the 6-byte application data area into which return code information is to be moved.

SYNCLEVEL(data-area)

specifies a halfword binary data area that is set to indicate the synchronization level in effect for the current conversation. The possible values are:

- 0 None
- 1 Confirm
- 2 Syncpoint.

Table 5. GDS EXTRACT PROCESS return codes

RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 00	CONVID is for a session that is not the principal facility.
03 00	Principal facility was not started by terminal data.
03 04	CONVID is for a conversation that is not basic.
04	CONVID is for a session that is not allocated to the task, or that is a relay link.
05 00 00 00 00 20	PROCLENGTH value returned is greater than MAXPROCLEN value.

GDS FREE

Function

Return an APPC session to CICS (assembler-language and C programs only).

Command syntax

```
► GDS FREE—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)◄
```

GDS FREE returns the session to CICS. The issue of this command is valid only when the conversation is finished, that is, the conversation state is FREE.

The return code is given in RETCODE (see Table 6). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS FREE options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(*name*)

identifies the conversation to be freed. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The STATE on a FREE command returns a cvda code of 00 if there is no longer an active conversation. The other output cvda values are:

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSSEND

Table 6. GDS FREE return codes

RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check has occurred.
04	CONVID is for a session that is not allocated to the task, or that is a relay link.

GDS ISSUE ABEND

Function

Terminate APPC basic conversation abnormally (assembler-language and C programs only).

Command syntax

```
►—GDS ISSUE ABEND—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—◄
```

GDS ISSUE ABEND causes an APPC basic conversation to end immediately, regardless of the conversation state. The partner transaction is informed.

The return code is given in RETCODE (see Table 7). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS ISSUE ABEND options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

Table 7. GDS ISSUE ABEND return codes

RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check has occurred.
04	CONVID is for a session that is not allocated to the task, or that is a relay link.

GDS ISSUE CONFIRMATION

Function

Issue synchronization request on APPC basic conversation (assembler-language and C programs only).

Command syntax

```
▶—GDS ISSUE CONFIRMATION—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—◀
```

GDS ISSUE CONFIRMATION issues a synchronization request in response to a GDS SEND CONFIRM issued by a partner transaction.

The return code is given in RETCODE (see Table 8). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS ISSUE CONFIRMATION options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* values returned by CICS are:

ALLOCATED
 CONFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

Table 8. GDS ISSUE CONFIRMATION return codes

RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check has occurred.
03 14	The command was issued for a sync level 0 conversation.
04	CONVID is for a session that is not allocated to the task, or that is a relay link.

GDS ISSUE ERROR

Function

Inform APPC basic conversation partner of error (assembler-language and C programs only).

Command syntax

```
▶—GDS ISSUE ERROR—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—◀
```

GDS ISSUE ERROR informs the conversation partner that there is an error.

The return code is given in RETCODE, see below. For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS ISSUE ERROR options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

Table 9. GDS ISSUE ERROR return codes

RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check has occurred.
04	CONVID is for a session that is not allocated to task, or that is a relay link.

GDS ISSUE PREPARE

Function

Issue first flow of syncpoint request on APPC basic conversation (assembler-language and C programs only).

Command syntax

```
▶—GDS ISSUE PREPARE—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—▶
```

GDS ISSUE PREPARE issues first flow of syncpoint request.

The return code is given in RETCODE (see Table 10). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS ISSUE PREPARE options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

Table 10. GDS ISSUE PREPARE return codes

RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 0C	The command was issued on a conversation that is not sync-level 2.
03 24	A state error occurred.
04	CONVID is for a session that is not allocated to task, or that is a relay link.

GDS ISSUE SIGNAL

Function

Request change of direction from sending transaction APPC basic conversation (assembler-language and C programs only).

Command syntax

```
▶—GDS ISSUE SIGNAL—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—◀
```

GDS ISSUE SIGNAL requests a change of direction.

The return code is given in RETCODE (see Table 11). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS ISSUE SIGNAL options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* values returned by CICS are:

ALLOCATED
 CONFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

Table 11. GDS ISSUE SIGNAL return codes

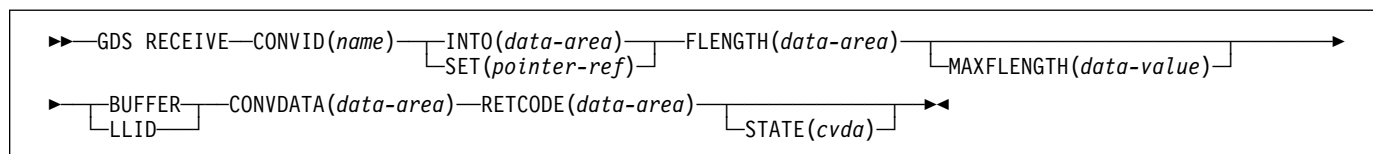
RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check has occurred.
04	CONVID is for a session that is not allocated to task, or that is a relay link.

GDS RECEIVE

Function

Receive data on an APPC basic conversation (assembler-language and C programs only).

Command syntax



GDS RECEIVE receives data and indicators from a partner transaction.

The return code is given in RETCODE (see Table 12 on page 116). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS RECEIVE options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

BUFFER

specifies that the length of the data passed to the application program in response to the RECEIVE command is to be restricted only by the length specified in the MAXFLENGTH option, and is not to be affected by GDS structured field boundaries. Control is returned to the application program when this length has been received, or when a synchronization request, change-direction, or end-bracket is received.

CONVDATA(data-area)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

FLENGTH(data-area)

specifies a fullword binary data area that is set to the length of the data made available to the application program.

INTO(data-area)

specifies the application target data area into which data is to be received from the application program connected

to the other end of the current conversation. The length of this area must not be less than the value specified in the MAXFLENGTH option.

LLID

specifies that the delimiter to be used by CICS to terminate the passing of data to the application program is the end of a GDS structured field, if this occurs before the MAXFLENGTH limit is reached.

MAXFLENGTH(data-value)

specifies, as a fullword binary value, either the length of the target data area specified in the INTO option, or the maximum length of data to be addressed by the pointer reference specified in the SET option. The length must not exceed 32 767 bytes. CICS does not receive more data than the MAXFLENGTH value allows.

RETCODE(data-area)

specifies the 6-byte application data area into which return code information is to be moved.

SET(pointer-ref)

specifies the pointer reference to be set to the address of data received from the application program connected to the other end of the current conversation. The pointer reference, unless changed by other commands or statements, is valid until the next RECEIVE (GDS or APPC) command, or the end of the task.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

GDS RECEIVE

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

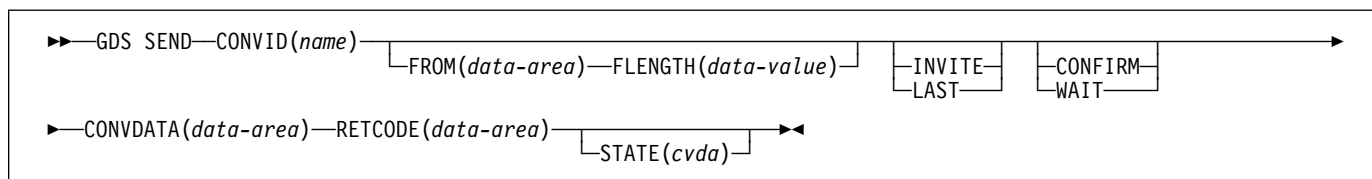
RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check occurred.
04	CONVID is for a session that is not allocated to the task, or that is a relay link.
05 00 00 00 7F FF	MAXFLENGTH is outside the range 0 through 32 767.

GDS SEND

Function

Send data on an APPC basic conversation (assembler-language and C programs only).

Command syntax



GDS SEND sends data.

The return code is given in RETCODE (see Table 13 on page 118). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS SEND options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONFIRM

allows an application working at synchronization level 1 or 2 to synchronize its processing with that of a process in a remote system. The actions taken to synchronize processing are defined by the application programs involved. The CONFIRM option causes RQD2 to be added to the data already sent, and forces a WAIT. On receipt of the indicator, the remote process takes the agreed actions and then sends a response. When the WAIT completes, CDBERR is set to X'00' if the appropriate response has been received.

CONVDATA(data-area)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

FLENGTH(data-value)

specifies the length (as a fullword binary value in the range 1–32 767) of the data specified in the FROM option.

FROM(data-area)

specifies the data that is to be sent.

INVITE

allows an application program to add a change-direction indicator to data already sent to a process in a connected APPC system. Control data is not transmitted by CICS until the subsequent execution of a WAIT or a SYNCPOINT command, unless CONFIRM or WAIT is also coded on the GDS SEND INVITE command.

LAST

allows an application program to add CEB to data already sent to a process in a connected APPC system. CEB is not transmitted by CICS until the subsequent execution of a WAIT or a SYNCPOINT command, unless CONFIRM or WAIT is also coded on the GDS SEND LAST command. Note that if one of these commands fails because of a conversation-related error, the conversation remains in bracket. In such a case, the application program should execute a GDS RECEIVE command. However, GDS SEND LAST WAIT (with no data) always causes the conversation to be deallocated.

RETCODE(data-area)

specifies the 6-byte application data area into which return code information is to be moved.

GDS SEND

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

WAIT

ensures that all data and indicators so far sent on a conversation are erased from the partner transaction.

If the WAIT option is not used, data from successive SEND commands is accumulated by CICS, together with any indicators, in an internal buffer. If the buffer becomes full, most of the accumulated data is transmitted to the remote system, but the accumulated indicators are not. Transmission of the accumulated data plus the indicators is forced by the WAIT or CONFIRM options of the GDS SEND command, or by a GDS WAIT command.

RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check has occurred.
03 14	The CONFIRM option has been used on a sync level 0 conversation.
03 10	LL error (incorrect or incomplete).
04	CONVID is for a session that is not allocated to the task, or that is a relay link.
05 00 00 00 7F FF	The FLENGTH value is outside the range 0 through 32 767.

GDS WAIT

Function

Ensure accumulated data transmitted on an APPC conversation (assembler-language and C programs only).

Command syntax

```
▶—GDS WAIT—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—▶
```

GDS WAIT ensures that the accumulated data has been sent.

The return code is given in RETCODE (see Table 14). For a list of return code values, see the *CICS/ESA Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

GDS WAIT options

In general, the arguments can be replaced by absolute or relocatable assembler-language expressions.

CONVDATA(*data-area*)

specifies the 24-byte application data area into which conversation-related information is to be moved. A description of the format of the data area is given in the discussion of CONVDATA fields in the *CICS/ESA Distributed Transaction Programming Guide*.

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed GDS ALLOCATE command, or the token representing the principal session (returned by a previously executed GDS ASSIGN command).

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information is to be moved.

STATE(*cvda*)

gets the state of the current conversation. The *cvda* values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

Table 14. GDS WAIT return codes

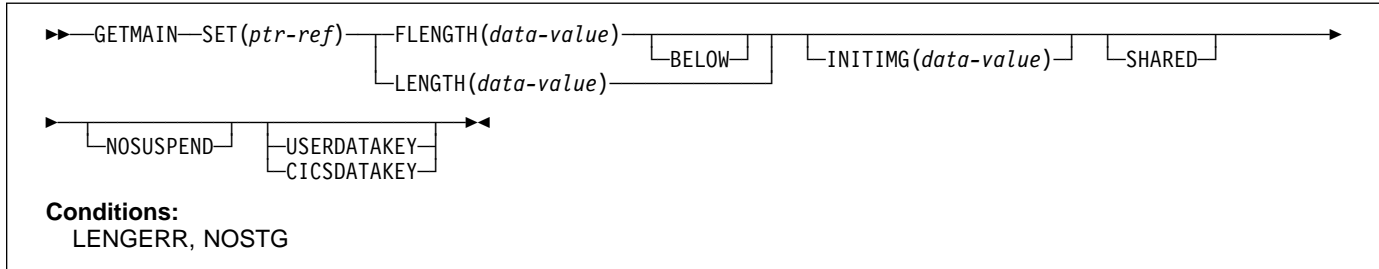
RETCODE (hexadecimal)	Description
03 00	CONVID is for a session that is not APPC.
03 00	CONVID is for a session that is in use by CPI Communications.
03 04	CONVID is for a conversation that is not basic.
03 08	A state check has occurred.
04	CONVID is for a session that is not allocated to task, or that is a relay link.

GETMAIN

Function

Get main storage.

Command syntax



Dynamic transaction routing

Using GETMAIN with SHARED could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

GETMAIN gets a main storage area of the size indicated by the FLENGTH option. (You can also use the LENGTH option, but this is supported for compatibility purposes and you are strongly recommended to use FLENGTH.) The address of the area is returned in the pointer reference supplied in the SET option.

CICS always allocates on double-word boundaries and rounds the requested length up to the nearest double-word multiple. Because there is no default initialization, you must use the INITIMG option if you require the storage to be initialized to a specific bit configuration.

CICS allocates storage from one of six different dynamic storage areas (DSAs):

- The CICS dynamic storage area (CDSA), below the 16MB line
- The user dynamic storage area (UDSA), below the 16MB line
- The shared dynamic storage area (SDSA), below the 16MB line
- The extended CICS dynamic storage area (ECDSA), above the 16MB line
- The extended user dynamic storage area (EUDSA), above the 16MB line
- The extended shared dynamic storage area (ESDSA), above the 16MB line

Note: There are two other dynamic storage areas—the read-only DSA (RDSA) and the extended read-only DSA (ERDSA)—but you cannot GETMAIN storage from these DSAs.

CICS determines whether to obtain the requested storage above or below the 16MB line, from one of the CICS- or user-key DSAs, or from one of the shared DSAs, according to the following options:

- The FLENGTH option with BELOW also specified
- The FLENGTH option alone, in conjunction with the addressing mode of the requesting program
- The LENGTH option
- The SHARED option.

The above- or below-the-line allocation is summarized in the following table:

Options	Address mode	
	24-bit	31-bit
FLENGTH BELOW	A DSA	A DSA
FLENGTH alone	A DSA	An EDSA
LENGTH	A DSA	A DSA

CICS decides whether to allocate storage from a CICS-key or user-key DSA, or from a shared DSA, according to the following options:

- The USERDATAKEY option on the GETMAIN command.
- The CICS DATAKEY option on the GETMAIN command.
- The TASK DATAKEY option on the transaction resource definition under which the requesting program is running if the USERDATAKEY or CICS DATAKEY option is omitted.
- The SHARED option on the GETMAIN command.

The data-key option on the GETMAIN command overrides the TASK DATAKEY option on the transaction resource

definition. The effect of the data-key options, without the SHARED option, is summarized in the following table:

Without SHARED option		
No data-key option	USERDATAKEY specified	CICSDATAKEY specified
Determined by TASKDATAKEY on transaction definition	User-key storage (from UDSA or EUDSA)	CICS-key storage (from CDSA or ECDSA)

The effect of the SHARED option is summarized in the following table:

With SHARED option		
No data-key option	USERDATAKEY specified	CICSDATAKEY specified
Determined by TASKDATAKEY on transaction definition	User-key storage (from SDSA or ESDSA)	CICS-key storage (from CDSA or ECDSA)

The storage that a task gets is available until it is released with a FREEMAIN command. For an area obtained without the SHARED option, only the task that acquired the storage may release it, and at task end CICS automatically releases such storage not already released. Note that any storage acquired with the SHARED option is accessible by all tasks, including those that are running with transaction isolation.

A SHARED area, on the other hand, is not released at task end and remains until explicitly freed; any task may issue the FREEMAIN. This means that you can use SHARED storage in task-to-task communication.

You cannot, however, use the storage obtained as a TIOA for subsequent terminal operations, because this could cause storage violations.

The following COBOL example shows how to get a 1024-byte area from user-key storage below the 16MB line (assuming that TASKDATAKEY(USER) is specified on the transaction resource definition), and initialize it to spaces:

```
EXEC CICS GETMAIN SET(PTR)
      FLENGTH(1024)
      BELOW
      INITIMG(BLANK)
```

You must define BLANK in your program as the character representing a space.

The following COBOL example shows how to get a 2048-byte area from CICS-key storage above the 16MB line (regardless of the TASKDATAKEY option specified on the transaction resource definition), and initialize it to spaces:

```
EXEC CICS GETMAIN SET(PTR)
      FLENGTH(2048)
      INITIMG(BLANK)
      CICSDATAKEY
```

Specifying CICSDATAKEY ensures that the requesting program obtains CICS-key storage from a CICS DSA, even if TASKDATAKEY(USER) is specified on the transaction resource definition.

GETMAIN options

BELOW

specifies that storage is to be obtained below the 16MB line, that is, from the CICS DSA.

CICSDATAKEY

specifies that CICS is to allocate storage from one of the CICS-key DSAs (the CDSA or ECDSA), overriding the TASKDATAKEY option specified on the transaction resource definition. If you do not specify the data key, CICS determines the type of storage (CICS-key or user-key) according to the TASKDATAKEY option on the transaction resource definition.

FLENGTH(data-value)

specifies the number of bytes of storage required, in fullword binary format.

The maximum length that you can specify is the value of the corresponding DSA limit parameter, either DSALIMIT or EDSALIMIT. These are the system initialization parameters that define the overall storage limit within which CICS can allocate and manage the individual DSAs.

If the length requested is bigger than the DSALIMIT or EDSALIMIT value, the LENGERR condition occurs. If it is not bigger than these limits, but is more than is available, NOSTG occurs.

INITIMG(data-value)

specifies an optional 1-byte initialization value. If you specify INITIMG, CICS sets every byte of the acquired storage to the bit string you provide. Otherwise, CICS does not initialize the storage. In COBOL programs only, you must use a data area rather than a data value to define the initialization bit string.

LENGTH(data-value)

specifies the number of bytes (unsigned halfword binary value) of storage required. LENGTH implies storage from below the 16MB line and has an upper limit of 65520 bytes. If you want storage above the 16MB line or a larger area, use FLENGTH.

If LENGTH is equal to zero, LENGERR occurs. If it is greater than the amount of storage available, NOSTG occurs.

Note: FLENGTH, with or without BELOW, is the recommended option: the LENGTH option is supported for compatibility purposes for those programs written to run under earlier releases of CICS.

NOSUSPEND

prevents CICS from suspending the task if no storage is available, and causes it to issue the NOSTG condition instead.

GETMAIN

SET(ptr-ref)

sets the pointer reference to the address of the acquired main storage. The pointer is set to the first byte of the storage area.

SHARED

prevents the automatic release of storage obtained by a GETMAIN command at the end of the task that requested it. This enables task-to-task communication. An area obtained with SHARED is not released until a corresponding FREEMAIN is issued, whether by the requesting task or some other task.

Be aware that if a task abends, any shared storage acquired is not automatically released.

USERDATAKEY

specifies that CICS is to allocate storage from one of the user-key DSAs (the UDSA, SDSA, EUDSA, or ESDSA), overriding the TASKDATAKEY option specified on the transaction resource definition. If you do not specify the data key, CICS determines the type of storage (CICS-key or user-key) according to the TASKDATAKEY option on the transaction resource definition.

GETMAIN conditions

LENGERR

occurs in any one of the following situations:

- The FLENGTH value is less than 1 or greater than the length of the target storage area from which the request is to be satisfied (RESP2=1). See the discussion about DSAs on page 120.
- The LENGTH value is zero.

Default action: terminate the task abnormally.

NOSTG

occurs when the storage requested is more than is currently available in the target DSA (RESP2=2). See the discussion about DSAs on page 120.

Default action: suspend the task until enough main storage is available to complete the request. However, the NOSUSPEND option overrides this default; it causes CICS to return control to the task immediately, without the requested storage.

HANDLE ABEND

Function

Handle an abnormal termination exit.

Command syntax



HANDLE ABEND is used to activate, cancel, or reactivate an exit for abnormal termination processing. You can suspend the command by means of the PUSH HANDLE and POP HANDLE commands as described in the *CICS/ESA Application Programming Guide*.

When a task terminates abnormally, CICS searches for an
active abend exit, starting at the logical level of the
application program in which the abend occurred, and
proceeding to successively higher levels. The first active
abend exit found, if any, is given control.

| The HANDLE ABEND command cannot intercept abends
| that are issued with the CANCEL option. Some internal
| abends generated by CICS are issued with the CANCEL
| option, for example the ASPx or APSJ abend codes.

When the label specified in a HANDLE ABEND LABEL command receives control, the registers are set as follows:

COBOL: Control returns to the HANDLE ABEND command with the registers restored; COBOL GO TO statement is then executed.

ASM: R15 - Abend label.
R0-14 - Contents at the time of last CICS service request.

If LABEL is specified, the addressing mode and execution key used are those of the program that issued the HANDLE ABEND command.

If PROGRAM is specified, the addressing mode is defined by the way the program is link-edited and the execution key is specified by the EXECKEY option on the program's resource definition.

If a COMMAREA has been established, it will be passed to
the specified PROGRAM.

The following example shows how to establish a program as an exit:

```
EXEC CICS HANDLE ABEND
      PROGRAM('EXITPGM')
```

HANDLE ABEND options

CANCEL

specifies that a previously established exit at the logical level of the application program in control is to be canceled.

LABEL(label)

specifies the program label to which control branches if abnormal termination occurs.

This option cannot be used for C or PL/I application programs.

PROGRAM(name)

specifies the name of the program to which control is to be passed if the task is terminated abnormally. If this program has not already been defined, the program will be autoinstalled in the event of the abend condition being raised.

The program named in this option should always terminate with an abend, except when handling abends generated as a result of application program logic.

RESET

specifies that an exit canceled by a HANDLE ABEND CANCEL command, or by CICS, is to be reactivated.

This option is usually issued by an abnormal termination exit routine.

HANDLE ABEND conditions

NOTAUTH

occurs when a resource security check has failed on PROGRAM(name).

Default action: terminate the task abnormally.

HANDLE ABEND

PGMIDERR

occurs in any of the following situations:

- | • The program has no entry in the PPT and
| autoinstall for programs is not active (RESP2=1)
- The program is disabled (RESP2=2)
- The installed program definition is for a remote
program (RESP2=9).

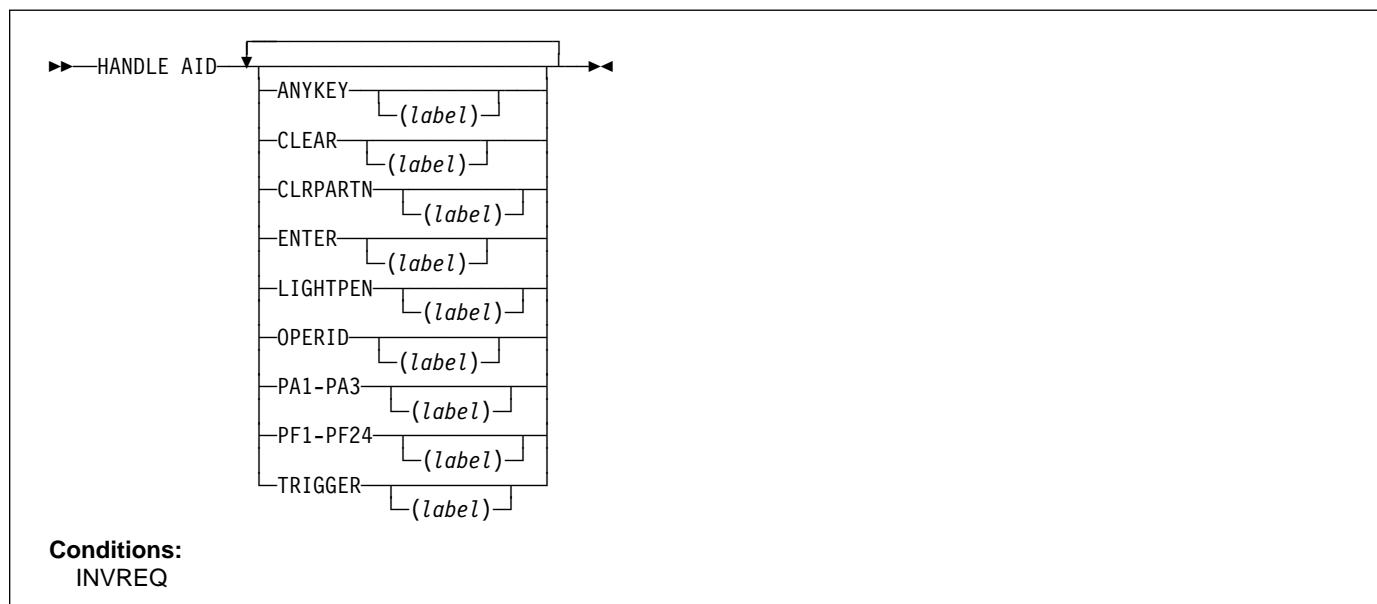
Default action: terminate the task abnormally.

HANDLE AID

Function

Handle attention identifiers (AIDs).

Command syntax



HANDLE AID is used to specify the label to which control is to be passed when an AID is received from a display device. Control is passed after the input command is completed; that is, after any data received in addition to the AID has been passed to the application program.

To cause an AID to be ignored, issue a HANDLE AID command that specifies the associated option *without* a label. This deactivates the effect of that option in any previously-issued HANDLE AID command.

If no HANDLE AIDs are in effect, that is none have been issued or all have been canceled, control returns to the application program at the instruction immediately following the input command. Look in EIBAID to determine which key was pressed.

No more than 16 options are allowed in the same command.

The C language does not support HANDLE AID.

The options that can be specified are:

- ANYKEY (any PA key, any PF key, or the CLEAR key, but not ENTER)
- CLEAR (for the key of that name)
- CLRPARTN (for the key of that name)
- ENTER (for the key of that name)
- LIGHTPEN (for a light-pen attention)

- OPERID (for the operator identification card reader, the magnetic slot reader (MSR), or the extended MSR (MSRE))
- PA1, PA2, or PA3 (any of the program access keys)
- PF1 through PF24 (any of the program function keys)
- TRIGGER (for a trigger field attention).

The following example shows a HANDLE AID command that specifies one label for the PA1 key; and a second label for CLEAR, PA2, PA3, and all the PF keys except PF10. If a PF10 AID is received or ENTER is pressed, control returns to the application program at the instruction immediately following the input command.

```
EXEC CICS HANDLE AID PA1(LAB1)
      ANYKEY(LAB2) PF10
```

If a task is initiated from a terminal by means of an AID, the first RECEIVE command in the task does not read from the terminal but copies only the input buffer (even if the length of the data is zero) so that control may be passed by means of a HANDLE AID command for that AID.

For the standard attention identifier list (DFHAID), and the standard attribute and printer control character list (DFHBMSCA), see Appendix J, "BMS-related constants" on page 375.

HANDLE AID

The execution key that the label receives control in, is the execution key that the program was running in when the HANDLE AID command was issued.

- | A print key specified by the system PRINT initialization
- | parameter takes precedence over a HANDLE AID command.

HANDLE AID condition

INVREQ

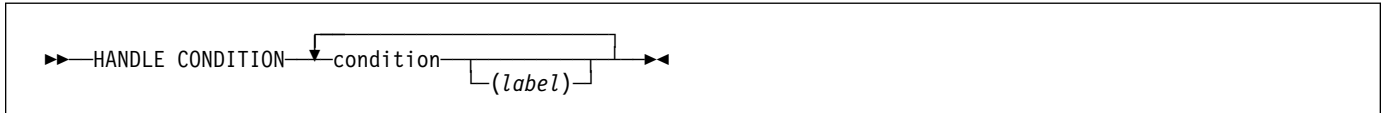
- | occurs if the command was issued by a distributed
- | program link server application (RESP2=200).
- | Default action: terminate the task abnormally.

HANDLE CONDITION

Function

Handle conditions.

Command syntax



You use HANDLE CONDITION to specify the label to which control is to be passed if a condition occurs. You must include the name of the condition and, optionally, a label to which control is to be passed if the condition occurs. You must ensure that the HANDLE CONDITION command is executed before the command that may give rise to the associated condition.

You cannot include more than sixteen conditions in the same command; the conditions should be separated by at least one space. You must specify additional conditions in further HANDLE CONDITION commands.

| If a condition occurs that is not specified in a HANDLE
 | CONDITION or IGNORE CONDITION command, the default
 | action is taken. If, however, the default action for a condition
 | not specified in a HANDLE CONDITION or IGNORE
 | CONDITION command terminates the task abnormally, and
 | the condition ERROR has been specified, the action for
 | ERROR is taken.

The following example shows you how to handle conditions, such as DUPREC, LENGERR, and so on, that can occur when you use a WRITE command to add a record to a data set.

Suppose that you want DUPREC to be handled as a special case; that you want standard system action (that is, to terminate the task abnormally) to be taken for LENGERR; and that you want all other conditions to be handled by the error routine ERRHANDL. You would code:

```
EXEC CICS HANDLE CONDITION
      ERROR(ERRHANDL)
      DUPREC(DUPRTN) LENGERR
```

The execution key that the label receives control in, is the execution key that the program was running in when the HANDLE CONDITION command was issued.

Scope

The HANDLE CONDITION command for a given condition applies only to the program in which it is specified. The HANDLE CONDITION command:

- Remains active while the program is being executed, or until:
 - An IGNORE CONDITION command for the same condition is encountered, in which case the HANDLE CONDITION command is overridden
 - Another HANDLE CONDITION command for the same condition is encountered, in which case the new command overrides the previous one.
 - # – A LINK command is executed to call another CICS program. The HANDLE CONDITION options are not inherited by the linked-to program.
- Is temporarily deactivated by the NOHANDLE or RESP option on a command.

Language considerations

| In an assembler-language application program, if a HANDLE
 | CONDITION and the command that causes the condition are
 | at the same logical level, the registers are restored to their
 | values in the application program at the point where the
 | command that caused the condition was issued. If, however,
 | the command that causes a condition occurs at a lower
 | logical level, the registers are restored to their current values
 | saved in DFHEISTG at the HANDLE CONDITION level. On
 | MVS/ESA, the addressing mode is set to that in effect at the
 | point where the HANDLE CONDITION command is issued.

In a PL/I application program, a branch to a label in an inactive procedure or in an inactive begin block, caused by a condition, produces unpredictable results.

The C language does not support HANDLE CONDITION.

HANDLE CONDITION

HANDLE CONDITION option

condition(label)

specifies the name of the condition; “label” specifies the location within the program to be branched to if the condition occurs.

If you omit “label”, any HANDLE CONDITION command for the condition is deactivated, and the default action is taken if the condition occurs. This is independent of the setting of the generalized ERROR condition.

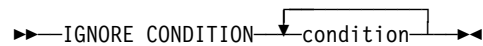
For more information about the conditions, see Appendix A, “EXEC interface block” on page 343.

IGNORE CONDITION

Function

Ignore conditions.

Command syntax



```
►—IGNORE CONDITION—↓condition—◄
```

For more information about the conditions, see Appendix A, “EXEC interface block” on page 343.

You use IGNORE CONDITION to specify that no action is to be taken if a condition occurs (that is, control returns to the instruction following the command that has failed to execute, and the EIB is set). Execution of a command could result in several conditions being raised. CICS checks these in a predetermined order and only the first one that is not ignored (by your IGNORE CONDITION command) is passed to your application program.

The IGNORE CONDITION command for a given condition applies only to the program in which it is specified, and it remains active while the program is being executed, or until a HANDLE CONDITION command for the same condition is encountered, in which case the IGNORE CONDITION command is overridden.

You cannot code more than sixteen conditions in the same command; the conditions should be separated by at least one space. You must specify additional conditions in further IGNORE CONDITION commands.

The C language does not support IGNORE CONDITION.

IGNORE CONDITION option

condition

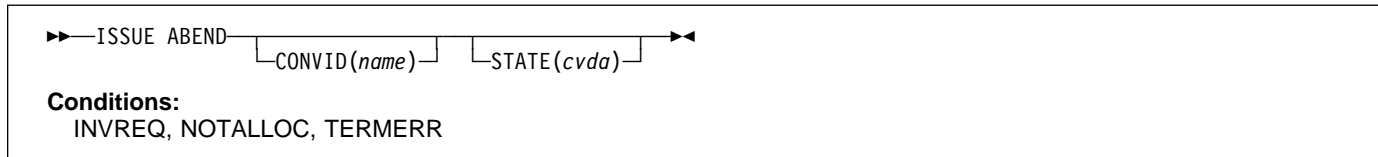
specifies the name of the condition to be ignored.

ISSUE ABEND

Function

Abend the mapped conversation with an APPC partner.

Command syntax



ISSUE ABEND abnormally ends the conversation. The partner transaction sees the TERMERR condition.

ISSUE ABEND options

CONVID(name)

identifies the conversation to be abended. The 4-character name identifies either the symbolic identifier returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the symbolic identifier representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If this option is omitted, the principal facility is assumed.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

```

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

```

ISSUE ABEND conditions

INVREQ

occurs in any of the following situations:

- ISSUE ABEND is used on any conversation other than an EXEC CICS APPC mapped conversation.
- A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if the CONVID value relates to a conversation that is not owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a session-related error. Any action on that conversation other than a FREE command causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

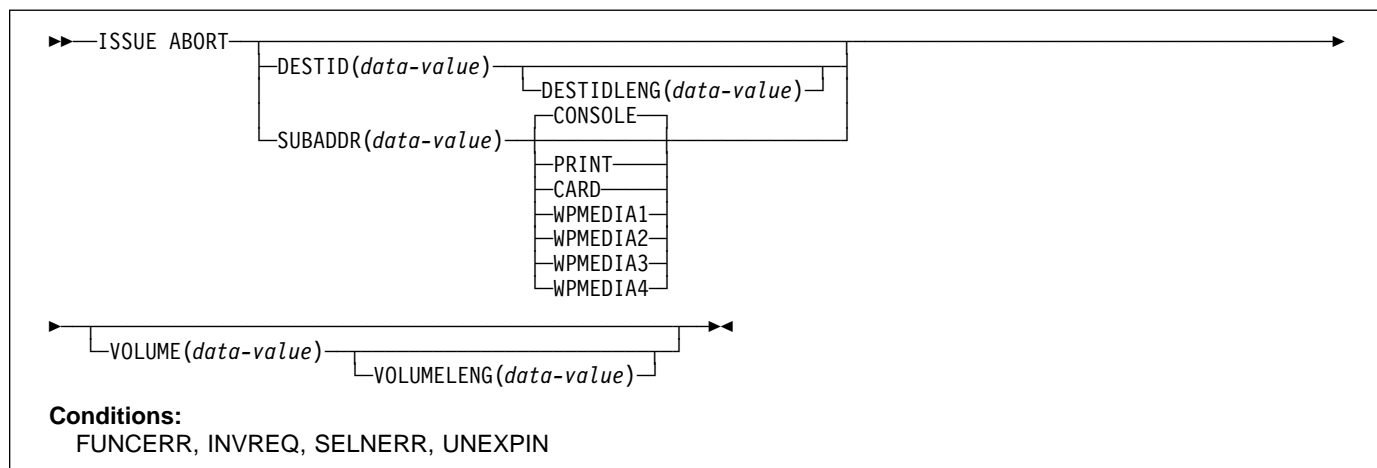
Default action: terminate the task abnormally with abend code ATNI.

ISSUE ABORT

Function

End processing of a data set abnormally.

Command syntax



ISSUE ABORT ends communication with a data set in an outboard controller, or the selected medium, abnormally. The data set specified in the DESTID option is deselected abnormally. The options CONSOLE, PRINT, CARD, and WPMEDIA1–4 are alternatives to DESTID and DESTIDLENG.

ISSUE ABORT options

CARD

specifies that the output medium is a card reader or card punch device. This option is not valid with DESTID and DESTIDLENG.

CONSOLE

specifies that the output medium is that provided for messages to the operator. This option is not valid with DESTID and DESTIDLENG. This refers to a programmable subsystem such as the IBM 3790 data communication system. It does not refer to a CICS or system console.

DESTID(data-value)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(data-value)

specifies the length (halfword binary value) of the name specified in the DESTID option.

PRINT

specifies that the output medium is a printer.

SUBADDR(data-value)

specifies the medium subaddress as a halfword binary value (in the range 0 through 15) which allows media of the same type, for example, “printer 1” or “printer 2”, to be defined. Value 15 means a medium of any type. The default is zero.

VOLUME(data-value)

specifies the name (1–6 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

VOLUMELENG(data-value)

specifies the length (halfword binary value) of the name specified in the VOLUME option.

WPMEDIA1 through WPMEDIA4

specifies that, for each specific LUTYPE4 device, a word-processing medium is defined to relate to a specific input/output device.

ISSUE ABORT conditions

FUNCERR

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

ISSUE ABORT

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

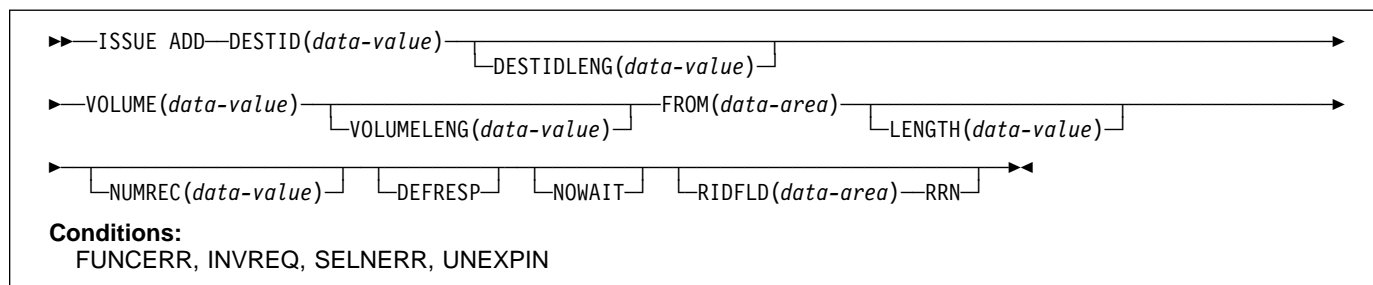
Default action: terminate the task abnormally.

ISSUE ADD

Function

Add a record to a data set.

Command syntax



ISSUE ADD adds records to a sequential or keyed direct data set in an outboard controller. The **FROM** option is used to specify the data to be written, and the **LENGTH** option specifies its length.

The **RIDFLD** option is only needed with this command when it applies to a DPCX/DXAM data set. In this case, it specifies the relative record number of the record to be added. When **RIDFLD** is used, **NUMREC** must be 1 (the default).

ISSUE ADD options

DEFRESP

specifies that all terminal control commands issued as a result of the **ISSUE ADD** command are to request a definite response from the outboard batch program, irrespective of the specification of message integrity for the CICS task (by the system programmer).

DESTID(data-value)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(data-value)

specifies the length (halfword binary value) of the name specified in the **DESTID** option.

FROM(data-area)

specifies the data to be written to the data set.

LENGTH(data-value)

specifies the length (halfword binary value) of the data to be written. For a description of a safe upper limit, see “**LENGTH** options” on page 5.

NOWAIT

specifies that the CICS task continues processing without waiting for the **ISSUE ADD** command to complete. If this option is not specified, the task activity is suspended until the command is completed.

NUMREC(data-value)

for a relative record data set, specifies as a halfword binary value the number of logical records to be added. Records are replaced sequentially starting with the one identified by the **RIDFLD** option.

For an indexed data set, **NUMREC** cannot be specified because only one record can be added.

RIDFLD(data-area)

specifies, for a relative record data set, a 4-character field as the relative record number (starting from zero) of the record. The **RRN** option is also required.

For a keyed direct data set, **RIDFLD** should specify a key.

RRN

specifies that the record identification field specified in the **RIDFLD** option contains a relative record number. This option is required for a relative record data set.

VOLUME(data-value)

specifies the name (1–6 characters) of a diskette in an outboard destination that contains the data set specified in the **DESTID** option.

VOLUMELENG(data-value)

specifies the length (halfword binary value) of the name specified in the **VOLUME** option.

ISSUE ADD conditions

FUNCERR

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

ISSUE ADD

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

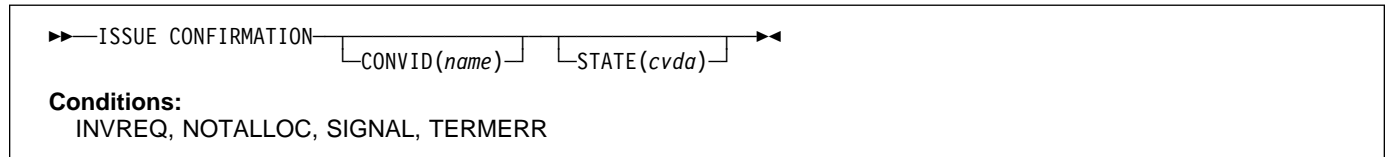
Default action: terminate the task abnormally.

ISSUE CONFIRMATION

Function

Issue a positive response to a SEND CONFIRM on an APPC mapped conversation.

Command syntax



ISSUE CONFIRMATION allows an application program to respond positively when the CONFIRM option has been specified on a SEND command executed by a partner transaction.

- Sync level 0
- Not APPC mapped.

- A distributed program link server application specified the function-shipping session on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

ISSUE CONFIRMATION options

CONVID(name)

identifies the conversation in which to send the response. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If this option is omitted, the principal facility is assumed.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

```

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

```

ISSUE CONFIRMATION conditions

INVREQ

occurs in any of the following situations:

- ISSUE CONFIRMATION is used on a conversation that is either of the following:

ISSUE CONFIRMATION

SIGNAL

occurs when an inbound SIGNAL data-flow control command is received from a partner transaction. EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

TERMERR

occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

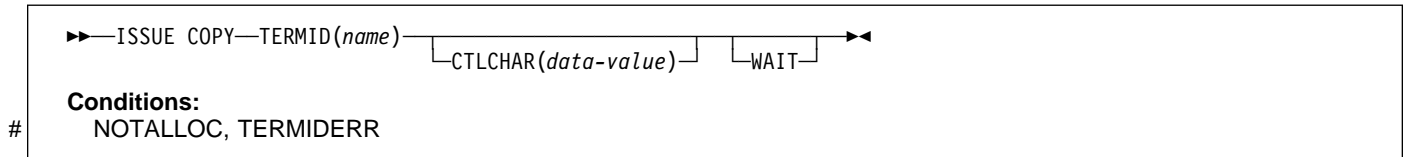
Default action: terminate the task abnormally with abend code ATNI.

ISSUE COPY (3270 display)

Function

Copy data from 3270 information display system (BTAM).

Command syntax



ISSUE COPY copies the format and data contained in the buffer of a specified terminal into the buffer of the terminal that started the transaction. Both terminals must be attached to the same remote control unit.

ISSUE COPY (3270 display) options

CTLCHAR(*data-value*)

specifies a 1-byte copy control character (CCC) that defines the copy function. A COBOL user must specify a data area containing this character. If the option is omitted, the contents of the entire buffer (including nulls) are copied.

TERMINID(*name*)

specifies the name (1–4 characters) of the terminal whose buffer is to be copied. The terminal must have been defined in the TCT.

WAIT

specifies that processing of the command must be completed before any subsequent processing is attempted.

If the WAIT option is not specified, control is returned to the application program when processing of the command has started. A subsequent input or output request (terminal control, BMS, or batch data interchange) to the terminal associated with the task causes the application program to wait until the previous request has been completed.

ISSUE COPY (3270 display) condition

NOTALLOC

- # occurs if the facility specified in the command is not owned by the application.
- # Default action: terminate the task abnormally.

TERMIDERR

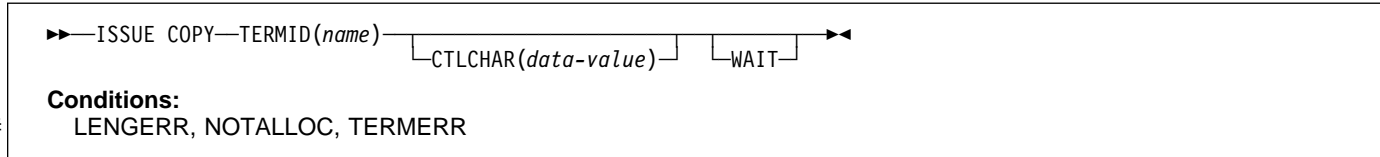
- occurs if the specified terminal identifier cannot be found in the terminal control table (TCT).
- Default action: terminate the task abnormally.

ISSUE COPY (3270 logical)

Function

Copy data from 3270 logical unit.

Command syntax



ISSUE COPY copies the format and data contained in the buffer of a specified terminal into the buffer of the terminal that started the transaction. Both terminals must be attached to the same remote control unit.

ISSUE COPY (3270 logical) options

CTLCHAR(*data-value*)

specifies a 1-byte copy control character (CCC) that defines the copy function. A COBOL user must specify a data area containing this character. If the option is omitted, the contents of the entire buffer (including nulls) are copied.

TERMINID(*name*)

specifies the name (1–4 characters) of the terminal whose buffer is to be copied. The terminal must have been defined in the TCT.

WAIT

specifies that processing of the command must be completed before any subsequent processing is attempted.

If the WAIT option is not specified, control is returned to the application program once processing of the command has started. A subsequent input or output request (terminal control, BMS, or batch data interchange) to the terminal associated with the task causes the application program to wait until the previous request has been completed.

TERMERR

occurs for a terminal-related error.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

ISSUE COPY (3270 logical) conditions

LENGERR

occurs if an out-of-range value is supplied.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

ISSUE DISCONNECT (default)

Function

Terminate a session between CICS and a logical unit or terminal.

Command syntax

▶—ISSUE DISCONNECT—▶

Conditions:
SIGNAL, TERMERR

ISSUE DISCONNECT terminates sessions between CICS and the following terminals or logical units:

- 3270-display logical unit (LUTYPE2)
- 3270-printer logical unit (LUTYPE3)
- LUTYPE4 logical unit
- 3270 SCS printer logical unit
- System/7
- 2260 or 2265 display station
- 3270 information display system (BTAM or TCAM)
- 3270 logical unit
- 3600 pipeline logical unit
- 3600(3601) logical unit
- 3600(3614) logical unit
- 3630 plant communication system
- 3650 interpreter logical unit
- 3650 host conversational (3270) logical unit
- 3650 host conversational (3653) logical unit
- 3650(3680) host command processor logical unit
- 3735 programmable buffered terminal
- 3740 data entry system
- 3767/3770 interactive logical unit
- 3770 batch logical unit
- 3790 logical units.

terminals only. Because of the asynchronous nature of this condition, the application program should check, using SEND CONFIRM or SYNCPOINT, to make sure any errors still outstanding have been resolved before it relinquishes control. If you wish to handle this condition, you must first issue a FREE command to free the session. If you do not do this, an INVREQ condition occurs, plus an ATCV abend if you do not handle this condition.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

ISSUE DISCONNECT (default) conditions

For most terminal and logical unit types, ISSUE DISCONNECT raises no conditions. Exceptions are:

SIGNAL

occurs only for an ISSUE DISCONNECT for LUTYPE4, 3600(3601), 3767 interactive, 3770 batch, and 3790 full-function logical units.

It occurs when an inbound SIGNAL data-flow control command is received from a logical unit or session. EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

TERMERR

occurs only for an ISSUE DISCONNECT for LUTYPE4 logical units.

It occurs for a terminal-related error, such as a session failure. This condition applies to VTAM-connected

ISSUE DISCONNECT (LUTYPE6.1)

ISSUE DISCONNECT (LUTYPE6.1)

Function

Disconnect an LUTYPE6.1 logical unit.

Command syntax

```
▶▶—ISSUE DISCONNECT—┬──SESSION(name)—▶▶
```

Conditions:

NOTALLOC, TERMERR

ISSUE DISCONNECT disconnects the unit, if DISCREQ=YES is set in DEFINE TERMTYPE for RDO.

ISSUE DISCONNECT (LUTYPE6.1) option

SESSION(*name*)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be disconnected. If this option is omitted, the principal facility for the task is disconnected.

ISSUE DISCONNECT (LUTYPE6.1) conditions

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a terminal-related error, such as a session failure.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

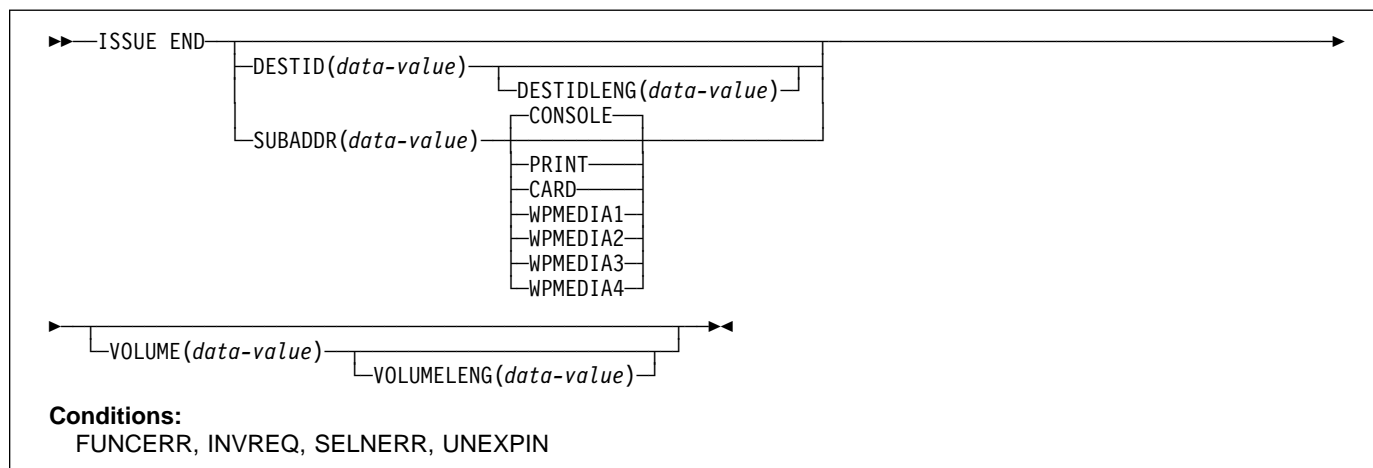
Default action: terminate the task abnormally withabend code ATNI.

ISSUE END

Function

End processing of a data set.

Command syntax



ISSUE END ends communication with a data set in an outboard controller or with the selected medium. The data set specified in the DESTID option, or the selected medium, is deselected normally. The options CONSOLE, PRINT, CARD, and WPMEDIA1–4 are alternatives to DESTID and DESTIDLENG.

ISSUE END options

CARD

specifies that the output medium is a card reader or card punch device. This option is not valid with DESTID and DESTIDLENG.

CONSOLE

specifies that the output medium is that provided for messages to the operator. This option is not valid with DESTID and DESTIDLENG. This refers to a programmable subsystem such as the IBM 3790 data communication system. It does not refer to a CICS or system console.

DESTID(data-value)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(data-value)

specifies the length (halfword binary value) of the name specified in the DESTID option.

PRINT

specifies that the output medium is a printer.

SUBADDR(data-value)

specifies the medium subaddress as a halfword binary value (in the range 0–15) that allows media of the same type, for example, “printer 1” or “printer 2”, to be defined. Value 15 means a medium of any type. The default is zero.

VOLUME(data-value)

specifies the name (1–6 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

VOLUMELENG(data-value)

specifies the length (halfword binary value) of the name specified in the VOLUME option.

WPMEDIA1 through WPMEDIA4

specifies that, for each specific LUTYPE4 device, a word-processing medium is defined to relate to a specific input/output device.

ISSUE END conditions

FUNCERR

occurs if there is an error during the execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

ISSUE END

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE ENDFILE

Function

Indicate the end-of-file condition to the 3740 data entry system.

Command syntax

```

  ┌───▶ ISSUE ENDFILE ───┐
  │                       │
  │ ┌─── ENDOUTPUT ───┐  │
  │ └──────────────────┘  │
  └────────────────────────┘
  
```

Conditions:
INVREQ, NOTALLOC

ISSUE ENDFILE indicates the end-of-file condition to the 3740.

ISSUE ENDFILE option

ENDOUTPUT

indicates the end-of-output condition as well as end of file.

| ISSUE ENDFILE conditions

| INVREQ

| occurs if a distributed program link server application attempted to send on its function shipping session, its principal facility (RESP2=200).

| Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

ISSUE ENDOUTPUT

Function

Indicate the end-of-output condition to the 3740 data entry system.

Command syntax



ISSUE ENDOUTPUT indicates the end-of-output condition to the 3740.

ISSUE ENDOUTPUT option

ENDFILE

indicates the end-of-file condition as well as end of output.

| ISSUE ENDOUTPUT conditions

| INVREQ

| occurs if a distributed program link server application attempted to send on its function shipping session, its principal facility (RESP2=200).
| Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.
Default action: terminate the task abnormally.

ISSUE EODS

Function

Send end-of-data-set function management header to the 3650 interpreter logical unit.

Command syntax

```
▶—ISSUE EODS—▶
```

Conditions:

INVREQ, NOTALLOC, TERMERR

ISSUE EODS issues the end-of-data-set management header.

ISSUE EODS conditions

| **INVREQ**

| occurs if a distributed program link server application
| attempted to send on its function shipping session, its
| principal facility (RESP2=200).

| Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not
owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a terminal-related error, such as a session
failure.

A CANCEL TASK request by a user node error program
(NEP) may cause this condition if the task has an
outstanding terminal control request active when the
node abnormal condition program handles the session
error.

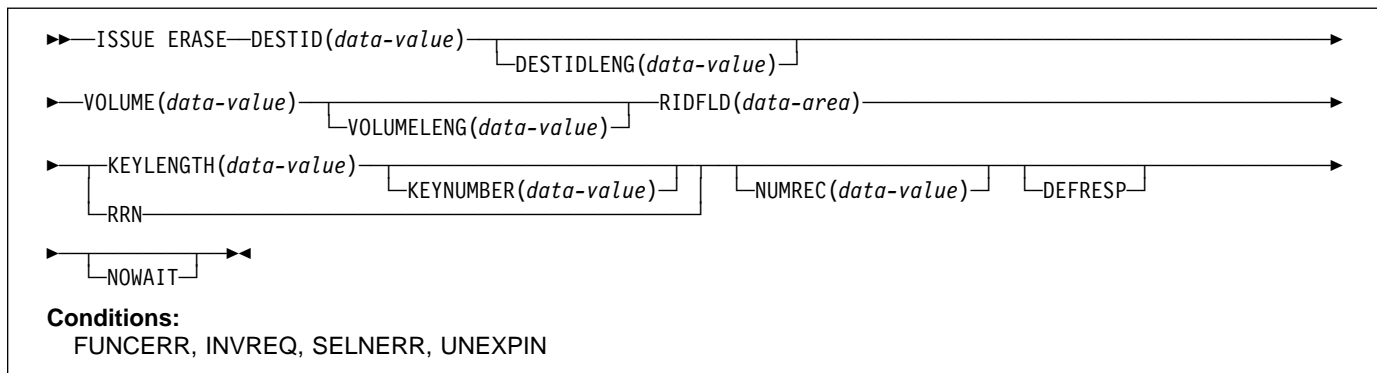
Default action: terminate the task abnormally withabend
code ATNI.

ISSUE ERASE

Function

Delete a record from a data set.

Command syntax



ISSUE ERASE deletes a record from a keyed direct data set in an outboard controller, or erases a record from a DPCX or DXAM relative record data set.

ISSUE ERASE options

DEFRESP

specifies that all terminal control commands issued as a result of the ISSUE ERASE command are to request a definite response from the outboard batch program, irrespective of the specification of message integrity for the CICS task (by the system programmer).

DESTID(*data-value*)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(*data-value*)

specifies the length (halfword binary value) of the name specified in the DESTID option.

KEYLENGTH(*data-value*)

specifies the length of the key specified in the RIDFLD option, as a halfword binary value.

KEYNUMBER(*data-value*)

specifies the number, as a halfword binary value, of the index to be used to locate the record. There can be eight indexes (1–8). The default is 1. This option applies only to DPCX or DXAM and is mutually exclusive with RRN.

NOWAIT

specifies that the CICS task continues processing without waiting for the ISSUE ERASE command to complete. If this option is not specified, the task activity is suspended until the command is completed.

NUMREC(*data-value*)

for a relative record data set, specifies as a halfword binary value the number of logical records to be deleted. Records are replaced sequentially starting with the one identified by the RIDFLD option.

For an indexed data set, NUMREC cannot be specified, because only one record is deleted.

RIDFLD(*data-area*)

specifies the record identification field. When combined with RRN this is a 4-character field.

For a relative record data set, the RIDFLD option specifies a fullword binary integer (the relative record number of the record starting from zero); and the RRN option is used.

For an indexed data set, the RIDFLD option specifies the key that is embedded in the data. The KEYLENGTH option is also required.

RRN

specifies that the record identification field specified in the RIDFLD option contains a relative record number. If the option is not specified, RIDFLD is assumed to specify a key.

VOLUME(*data-value*)

specifies the name (1–6 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

VOLUMELENG(*data-value*)

specifies the length (halfword binary value) of the name specified in the VOLUME option.

ISSUE ERASE conditions

FUNCERR

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE ERASEAUP

Function

Erase all unprotected fields of a 3270 buffer.

Command syntax

```

  >>—ISSUE ERASEAUP—┐
                       └─┬─ WAIT ─┘
  
```

Conditions:

INVREQ, NOTALLOC, TERMERR

ISSUE ERASEAUP erases unprotected fields by:

1. Clearing all unprotected fields to nulls (X'00')
2. Resetting modified data tags in each unprotected field to zero
3. Positioning the cursor to the first unprotected field
4. Restoring the keyboard.

You can use the ISSUE ERASEAUP command for the following types of 3270 logical units:

- 3270-display logical unit (LUTYPE2)
- 3270-printer logical unit (LUTYPE3)
- 3270 information display system (BTAM or TCAM)
- 3270 logical unit
- 3650 host conversational (3270) logical unit
- 3790 (3270-display) logical unit
- 3790 (3270-printer) logical unit.

NOTALLOC

- # occurs if the facility specified in the command is not owned by the application.
- # Default action: terminate the task abnormally.

TERMERR

occurs for a terminal-related error.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

ISSUE ERASEAUP option

WAIT

ensures that the erase is completed before control returns to the application program. If you omit WAIT, control returns to the application program as soon as ISSUE ERASEAUP starts processing.

ISSUE ERASEAUP conditions

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

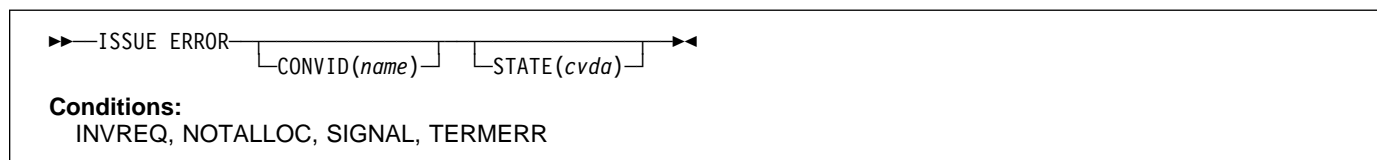
Default action: terminate the task abnormally.

ISSUE ERROR

Function

Inform APPC mapped conversation partner of error.

Command syntax



ISSUE ERROR allows an application program to inform a process in a connected APPC system that some program-detected error has occurred. For example, a remote CICS application is notified by having EIBERR set, with EIBERRCD=X'0889'. The actions required to recover from the error are the responsibility of logic contained in both application programs. The application program can use this command to respond negatively when the CONFIRM option has been specified on a SEND command executed by a process in a connected APPC system.

ISSUE ERROR options

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If this option is omitted, the principal facility is assumed.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

```

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

```

ISSUE ERROR conditions

INVREQ

occurs in any of the following situations:

- The command is not valid for the APPC conversation type in use.
- The command is issued against a CPI-Communications conversation.
- A distributed program link server application specified the function-shipping session on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if CONVID does not specify the name of a token that relates to a conversation owned by the application.

Default action: terminate the task abnormally.

SIGNAL

occurs when an inbound SIGNAL data-flow control command is received from a partner transaction. EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

TERMERR

occurs for a session-related error. Any action on that conversation other than a FREE command causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

ISSUE LOAD

ISSUE LOAD

Function

Specify the name of a program on 3650 interpreter logical unit.

Command syntax

```
▶—ISSUE LOAD—PROGRAM(name)—┐—CONVERSE—┘▶
```

Conditions:

NONVAL, NOSTART, NOTALLOC, TERMERR

ISSUE LOAD specifies the name of the 3650 application program that is to be loaded.

ISSUE LOAD options

CONVERSE

specifies that the 3650 application program is able to communicate with the host processor. If this option is not specified, the 3650 application program cannot communicate with the host processor.

PROGRAM(*name*)

specifies the name (1–8 characters) of the 3650 application program that is to be loaded.

ISSUE LOAD conditions

NONVAL

occurs if the 3650 application program name is not valid.

Default action: terminate the task abnormally.

NOSTART

occurs if the 3651 is unable to initiate the requested 3650 application program.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a terminal-related error.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

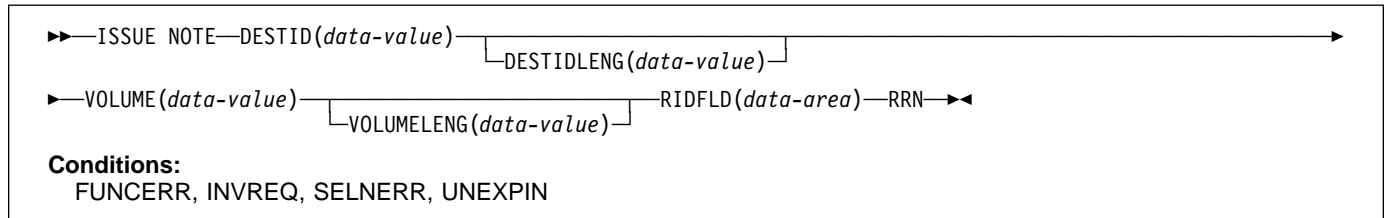
Default action: terminate the task abnormally withabend code ATNI.

ISSUE NOTE

Function

Request next record number.

Command syntax



ISSUE NOTE requests the number of the next record. It finds the relative record number of the next record in an addressed direct data set. The number is returned in the data area specified in the RIDFLD option. The RRR option must be specified, because a relative record number is involved.

ISSUE NOTE options

DESTID(*data-value*)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(*data-value*)

specifies the length (halfword binary value) of the name specified in the DESTID option.

RIDFLD(*data-area*)

specifies as a 4-character field a data area the relative record number of the next record is returned in.

RRR

specifies that the record identification field specified in the RIDFLD option contains a relative record number.

VOLUME(*data-value*)

specifies the name (1–6 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

VOLUMELENG(*data-value*)

specifies the length (halfword binary value) of the name specified in the VOLUME option.

ISSUE NOTE conditions

FUNCERR

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

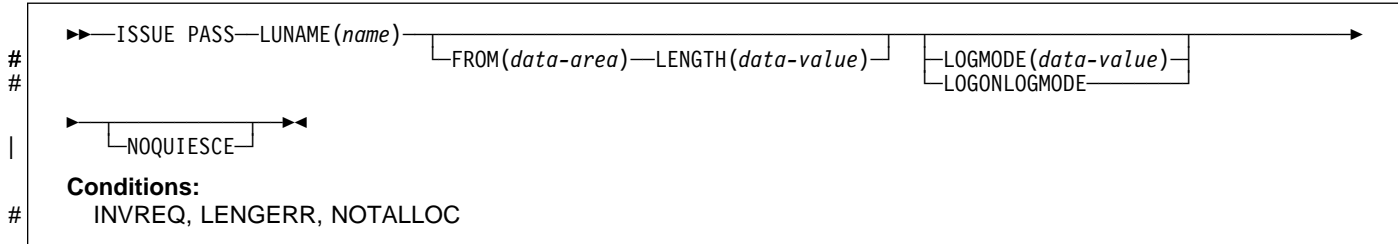
Default action: terminate the task abnormally.

ISSUE PASS

Function

VTAM application routing.

Command syntax



ISSUE PASS disconnects the terminal from CICS after the task has terminated, and transfers it to the VTAM application defined in the LUNAME option.

This command requires that AUTH=PASS is coded on the TYPETERM definition of the VTAM APPL macro for the CICS terminal-owning system that issues it, with DISCREQ=YES or RELREQ=YES in DEFINE TYPETERM for any terminal where this function might be used.

If the LUNAME specified is the name of another CICS system, you can use the EXTRACT LOGONMSG command to access the data referred to by this command.

Because of a VTAM limitation, the maximum length of the user data is restricted to 255 bytes.

Note: The system initialization parameter CLSDSTP=NOTIFY|NONOTIFY allows you to have the node error program (NEP) and the console notified of whether the PASS was successful or not. The NEP can be coded to reestablish a session ended by an unsuccessful PASS. For programming information about how to do this, see the section about NEP in the *CICS/ESA Customization Guide*.

ISSUE PASS options

FROM(data-area)

specifies the data area containing the logon user data that is to be passed to the application named in the LUNAME option. This option may be omitted if ATTACHID is specified on an LUTYPE6.1 command.

LENGTH(data-value)

specifies the length, as a halfword binary value, of the data issued.

LOGMODE(data-value)

Apar 88279

Documentation for Apar 84968 added 04/11/96

| specifies the name (1–8 characters) of the VTAM logon mode table entry used by VTAM to establish the new session.

Apar 84968

Documentation for Apar 88279 added 04/11/96

LOGONLOGMODE

specifies that the new session is to be established with the VTAM logon mode table entry in use when the session logged on.

Note: The logmode name saved is taken from the X'0D' control vector in the VTAM CINIT. This is the logmode name known in this system.

If persistent sessions (PSDINT=nnn in the SIT) is in use then the TYPETERM definition for any terminal to be ISSUE PASSEd should use RECOVOPTION(NONE) because the the logon LOGMODE name is not recovered across a persistent sessions restart.

If neither LOGMODE nor LOGONLOGMODE is supplied, the new session will be established with the default LOGMODE.

LUNAME(name)

specifies the name (1–8 characters) of the VTAM application to which the terminal is to be passed.

| NOQUIESCE

| specifies that the user can choose to recover from certain pass failures.

ISSUE PASS conditions

INVREQ

occurs if the command is not valid for the logical unit in use.

Default action: terminate the task abnormally.

LENGERR

occurs if an out-of-range value is supplied in the LENGTH option.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

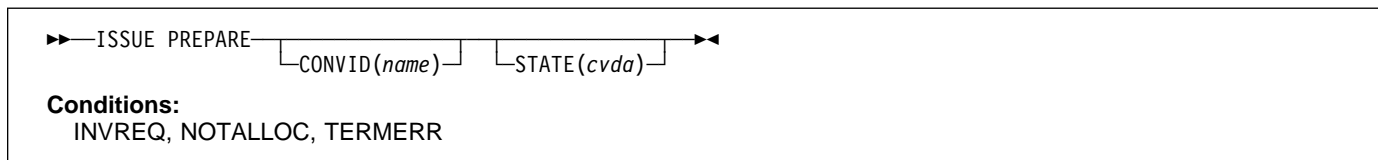
Default action: terminate the task abnormally.

ISSUE PREPARE

Function

Issue the first flow of a syncpoint request on an APPC mapped conversation.

Command syntax



ISSUE PREPARE applies only to distributed transaction processing over APPC links. It enables a syncpoint initiator to prepare a syncpoint slave for syncpointing by sending only the first flow (prepare-to-commit) of the syncpoint exchange. Depending on the reply from the syncpoint slave, the initiator can proceed with the syncpoint by issuing a SYNCPOINT command, or initiate back-out by issuing a SYNCPOINT ROLLBACK command.

ISSUE PREPARE options

CONVID(name)

specifies the symbolic identifier (1–4 characters) of an APPC mapped conversation. This option specifies the alternate facility to be used.

If this option is omitted, the principal facility is assumed.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

```

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

```

ISSUE PREPARE conditions

INVREQ

occurs in any of the following situations:

- The conversation is not an APPC mapped conversation.
- The conversation state is not valid for the request.
- The sync level of the conversation is not 2.
- A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if the CONVID value in the command does not relate to a conversation that is owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

ISSUE PRINT

Function

Print displayed data on first available printer.

Command syntax

▶—ISSUE PRINT—◀

Conditions:

INVREQ, NOTALLOC, TERMERR

ISSUE PRINT prints displayed data on the first available printer that can respond to a print request.

ISSUE PRINT can be used on a number of logical units, using the printers defined below:

- For a 3270 information display system (BTAM), the printer must be on the same control unit, have buffer capacity equal to or greater than that of the display, and have feature-print specified in the associated DFHTCT TYPE=TERMINAL table.
- For a 3270 logical unit or a 3650 host conversational (3270) logical unit, the printer must be defined by the PRINTTO or ALTPRT operands of the DFHTCT TYPE=TERMINAL table, by RDO, or by a printer supplied by the autoinstall user program.
- For a 3270-display logical unit with the PTRADAPT feature, used with a 3274 or 3276, the printer is allocated by the printer authorization matrix. The PTRADAPT feature is enabled by coding a DFHTCT TYPE=TERMINAL macro with TRMTYPE=LUTYPE2 and FEATURE=PTRADAPT.
- For a 3790 (3270-display) logical unit, the printer is allocated by the 3790.

The printer must be in service, not currently attached to a task, and owned by the same CICS system that owns the terminal running the transaction.

ISSUE PRINT conditions

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a terminal-related error.

An ISSUE PRINT on a BTAM 3270 information display system cannot raise the TERMERR condition.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

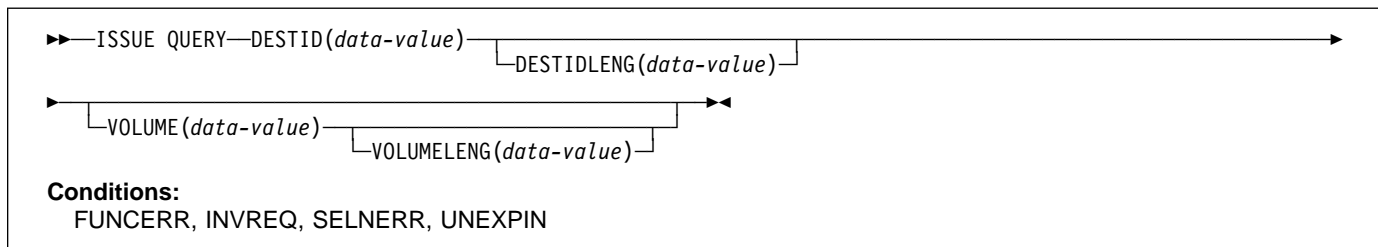
Default action: terminate the task abnormally with abend code ATNI.

ISSUE QUERY

Function

Interrogate a data set.

Command syntax



ISSUE QUERY interrogates a data set. It is used to request that a sequential data set in an outboard controller be transmitted to the host system. The application program should either follow this command with ISSUE RECEIVE commands to get the resulting inbound data, or terminate the transaction to allow CICS to start a new transaction to process the data.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE QUERY options

DESTID(data-value)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(data-value)

specifies the length (halfword binary value) of the name specified in the DESTID option.

VOLUME(data-value)

specifies the name (1–8 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

VOLUMELENG(data-value)

specifies the length (halfword binary value) of the name specified in the VOLUME option.

ISSUE QUERY conditions

FUNCERR

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

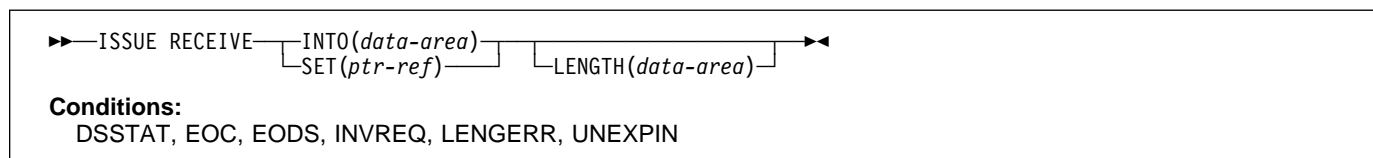
Default action: terminate the task abnormally.

ISSUE RECEIVE

Function

Read a record from a data set.

Command syntax



ISSUE RECEIVE reads a sequential data set in an outboard controller.

The INTO option specifies the area into which the data is to be placed. The LENGTH option must specify a data area that contains the maximum length of record that the program accepts. If the record length exceeds the specified maximum length, the record is truncated and the LENGERR condition occurs. After the retrieval operation, the data area specified in the LENGTH option is set to the record length (before any truncation occurred).

Alternatively, a pointer reference can be specified in the SET option. CICS then acquires an area of sufficient size to hold the record, and sets the pointer reference to the address of that area. After the retrieval operation, the data area specified in the LENGTH option is set to the record length.

The outboard controller might not send the data from the data set specified in the ISSUE QUERY command. The ASSIGN command must be used to get the value of DESTID (which identifies the data set that has actually been transmitted) and the value of DESTIDLENG (which is the length of the identifier in DESTID).

ISSUE RECEIVE options

INTO(data-area)

specifies the receiving field for the data read from the data set.

If you specify the ISSUE RECEIVE command with the INTO option, the parameter must be a data area that specifies the maximum length of data that the program is prepared to handle. If the value specified is less than zero, zero is assumed. If the length of the data exceeds the value specified, the data is truncated to that value and the LENGERR condition occurs. On completion of the retrieval operation, the data area is set to the original length of the data.

LENGTH(data-area)

specifies the length (halfword binary value) of the data received.

If you have specified SET, you must also specify LENGTH.

SET(ptr-ref)

specifies the pointer reference that is to be set to the address location of the data read from the data set.

If you specify the SET option, the parameter must be a data area. On completion of the retrieval operation, the data area is set to the length of the data.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

If you have specified SET, you must also specify LENGTH.

ISSUE RECEIVE conditions

DSSTAT

occurs when the destination status changes in one of the following ways:

- The data stream is abended.
- The data stream is suspended.

Default action: terminate the task abnormally.

EOC

occurs if the request/response unit (RU) is received with the end-of-chain (EOC) indicator set. Field EIBEOC also contains this indicator.

Default action: ignore the condition.

ISSUE RECEIVE

EODS

occurs when the end of the data set is encountered.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs if the length of the retrieved data is greater than the value specified by the LENGTH option.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

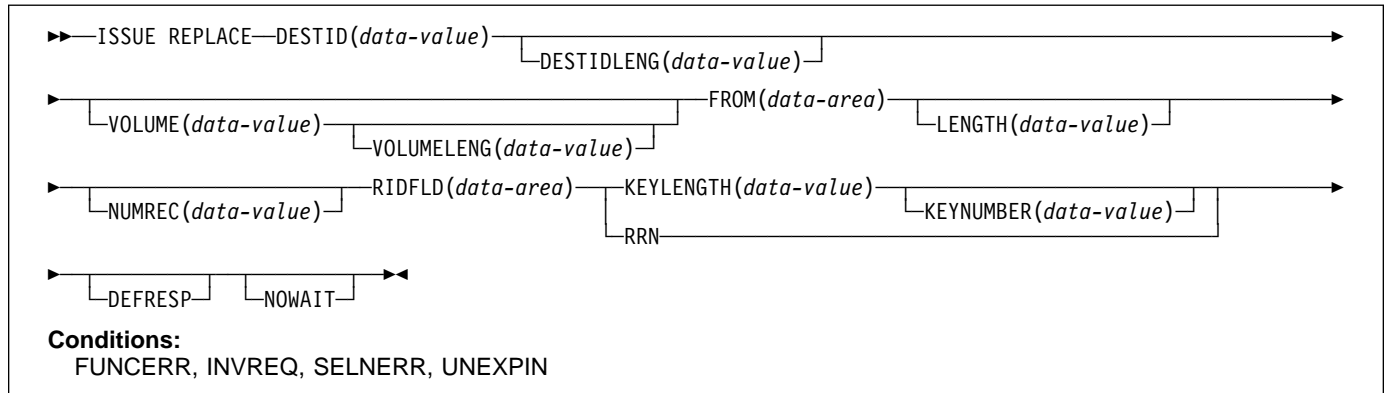
Default action: terminate the task abnormally.

ISSUE REPLACE

Function

Update a record in a data set.

Command syntax



ISSUE REPLACE updates (replaces) a record in either a relative (addressed direct) or an indexed (keyed direct) data set in an outboard controller.

ISSUE REPLACE options

DEFRESP

specifies that all terminal control commands issued as a result of the ISSUE REPLACE command request a definite response from the outboard batch program, irrespective of the specification of message integrity for the CICS task (by the system programmer).

DESTID(data-value)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(data-value)

specifies the length (halfword binary value) of the name specified in the DESTID option.

FROM(data-area)

specifies the data that is to be written to the data set.

KEYLENGTH(data-value)

specifies the length (halfword binary value) of the key specified in the RIDFLD option.

KEYNUMBER(data-value)

specifies the number, as a halfword binary value, of the index to be used to locate the record. There can be eight indexes (1 through 8). The default is 1. This option applies only to DPCX/DXAM and is mutually exclusive with RRN.

LENGTH(data-value)

specifies the length (halfword binary value) of the data to be written.

NOWAIT

specifies that the CICS task continues processing without waiting for the ISSUE REPLACE command to complete. If this option is not specified, the task activity is suspended until the command is completed.

NUMREC(data-value)

for a relative data set, specifies as a halfword binary value the number of logical records to be replaced. Records are replaced sequentially starting with the one identified by the RIDFLD option.

For an indexed data set, NUMREC cannot be specified because only one record is replaced.

RIDFLD(data-area)

specifies the record identification field. When combined with RRN this is a 4-character field.

For a relative record data set, the RIDFLD option specifies a fullword binary integer (the relative record number of the record starting from zero); and the RRN option is used.

For an indexed data set, the RIDFLD option specifies the key that is embedded in the data specified by the FROM option. The KEYLENGTH option is also required.

RRN

specifies that the record identification field specified in the RIDFLD option contains a relative record number. This option is required for a relative record data set.

If the option is not specified, RIDFLD is assumed to specify a key.

VOLUME(data-value)

specifies the name (1–8 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

ISSUE REPLACE

VOLUMELENG(data-value)

specifies the length (halfword binary value) of the name specified in the VOLUME option.

ISSUE REPLACE conditions

FUNCERR

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE RESET

Function

Relinquish use of a telecommunication line.

Command syntax

```
▶—ISSUE RESET—▶
```

Conditions:
INVREQ, NOTALLOC

ISSUE RESET applies only to the following devices:

- System/7
- 2260 or 2265 display station
- 2741 communication terminal
- 3270 information display system
- 3600 finance communication system
- 3735 programmable buffered terminal
- 3740 data entry system.

In each case, the device is connected with BTAM, and the next BTAM operation is a READ or WRITE INITIAL.

RESET for a VTAM logical unit is treated as an ISSUE DISCONNECT.

ISSUE RESET conditions

| INVREQ

| occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

| Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

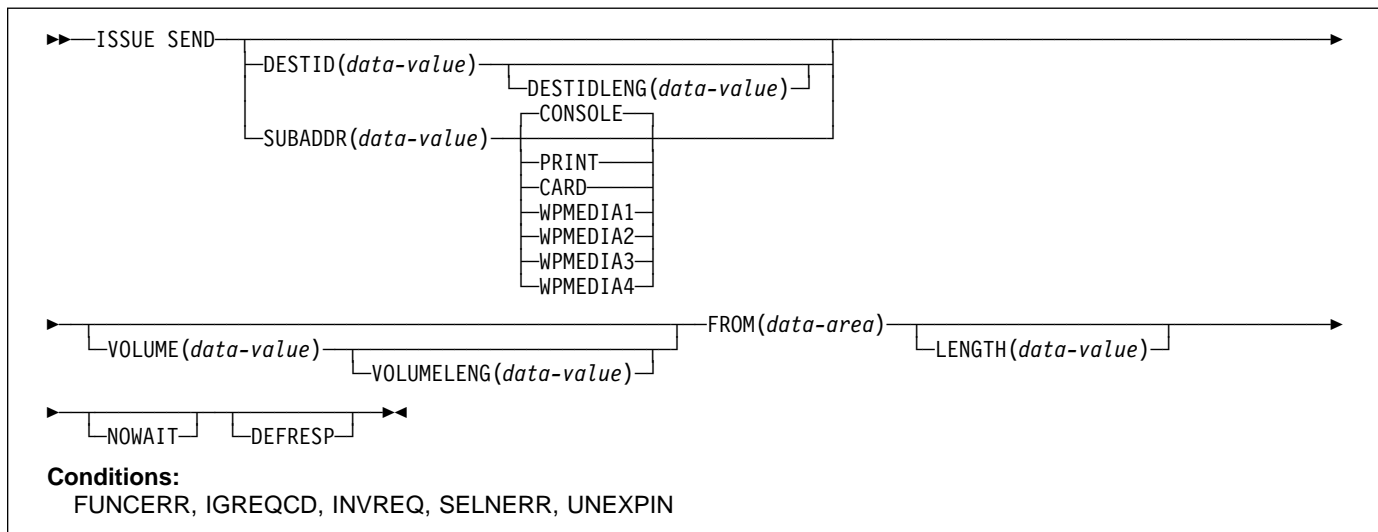
Default action: terminate the task abnormally.

ISSUE SEND

Function

Send data to a named data set or to a selected medium.

Command syntax



ISSUE SEND sends data to a named data set in an outboard controller, or to a selected medium in a batch logical unit or an LUTYPE4 logical unit. The options CONSOLE, PRINT, CARD, and WPMEDIA1–4 are alternatives to DESTID and DESTIDLENG.

ISSUE SEND options

CARD

specifies that the output medium is a card reader or card punch device. This option is not valid with DESTID and DESTIDLENG.

CONSOLE

specifies that the output medium is that provided for messages to the operator. This option is not valid with DESTID and DESTIDLENG. This refers to a programmable subsystem such as the IBM 3790 data communication system. It does not refer to a CICS or system console.

DEFRESP

specifies that all terminal control commands issued as a result of the ISSUE SEND command request a definite response from the outboard batch program, irrespective of the specification of message integrity for the CICS task (by the system programmer).

DESTID(data-value)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(data-value)

specifies the length (halfword binary value) of the name specified in the DESTID option.

FROM(data-area)

specifies the data to be written to the data set.

LENGTH(data-value)

specifies a halfword binary value that is the length of the data to be written.

NOWAIT

specifies that the CICS task continues processing without waiting for the ISSUE SEND command to complete. If this option is not specified, the task activity is suspended until the command is completed.

PRINT

specifies that the output is to the print medium.

SUBADDR(data-value)

specifies the medium subaddress as a halfword binary value (in the range 0–15) that allows media of the same type, for example, “printer 1” or “printer 2”, to be defined. Value 15 means a medium of any type. The default is zero.

VOLUME(data-value)

specifies the name (1–8 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

VOLUMELENG(data-value)

specifies the length of the name specified in the VOLUME option as a halfword binary value.

WPMEDIA1 through WPMEDIA4

specifies that for each specific LUTYPE4 device, a word processing medium is defined to relate to a specific input/output device.

ISSUE SEND conditions**FUNCERR**

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

IGREQCD

occurs when an attempt is made to execute an ISSUE SEND command after a SIGNAL RCD data-flow control code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

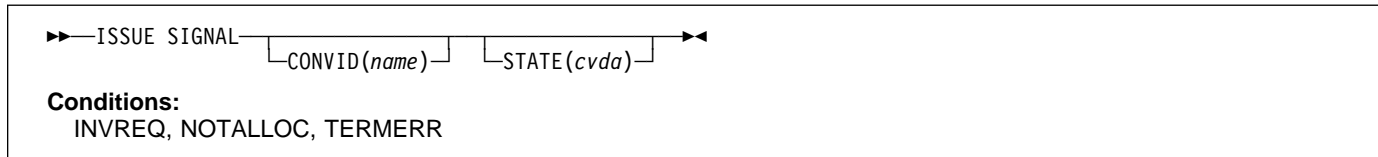
Default action: terminate the task abnormally.

ISSUE SIGNAL (APPC)

Function

Request change of direction from sending transaction on an APPC mapped conversation.

Command syntax



ISSUE SIGNAL, in a transaction in receive mode, signals to the sending transaction that a mode change is needed. It raises the SIGNAL condition on the next SEND, RECEIVE, or CONVERSE command executed in the sending transaction, and a previously executed HANDLE CONDITION command for this condition can be used either to take some action, or to ignore the request.

ISSUE SIGNAL (APPC) options

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If this option is omitted, the principal facility is assumed.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

ISSUE SIGNAL (APPC) conditions

INVREQ

occurs in any of the following situations:

- The command has been used on an APPC conversation that is not using the EXEC CICS interface, or is not a mapped conversation.
- A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if the CONVID value in the command does not relate to a conversation that is owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause TERMERR if the task has an outstanding terminal control request when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

ISSUE SIGNAL (LUTYPE6.1)

Function

Request change of direction from sending transaction on LUTYPE6.1 conversation.

Command syntax

```

  <<—ISSUE SIGNAL—>>
      |
      |—CONVID(name)—|
      |—SESSION(name)—|
  
```

Conditions:
NOTALLOC, TERMERR

ISSUE SIGNAL, in a transaction in receive mode, signals to the sending transaction that a mode change is needed. It raises the SIGNAL condition on the next SEND, RECEIVE, or CONVERSE command executed in the sending transaction, and a previously executed HANDLE CONDITION command for this condition can be used either to take some action, or to ignore the request.

If both CONVID and SESSION are omitted, the principal facility for the task is used.

ISSUE SIGNAL (LUTYPE6.1) options

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

SESSION(*name*)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used.

ISSUE SIGNAL (LUTYPE6.1) condition

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCVabend.

A CANCEL TASK request by a user node error program (NEP) may cause TERMERR if the task has an outstanding terminal control request when the node abnormal condition program handles the session error.

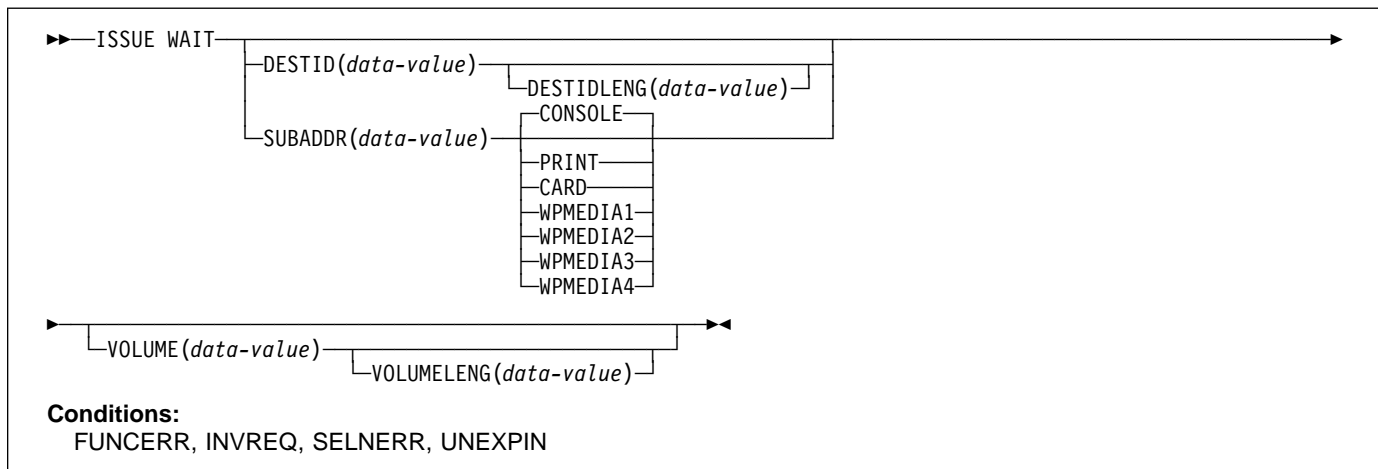
Default action: terminate the task abnormally with abend code ATNI.

ISSUE WAIT

Function

Wait for an operation to be completed.

Command syntax



ISSUE WAIT suspends task activity until the previous batch data interchange command is completed. This command is meaningful only when it follows an ISSUE ADD, ISSUE ERASE, ISSUE REPLACE, or ISSUE SEND command. The options CONSOLE, PRINT, CARD, and WPMEDIA1–4 are alternatives to DESTID and DESTIDLENG.

ISSUE WAIT options

CARD

specifies that the output medium is a card reader or card punch device. This option is not valid with DESTID and DESTIDLENG.

CONSOLE

specifies that the output medium is that provided for messages to the operator. This option is not valid with DESTID and DESTIDLENG.

This refers to a programmable subsystem such as the IBM 3790 data communication system. It does not refer to a CICS or system console.

DESTID(data-value)

specifies the name (1–8 characters) of the data set in the outboard destination.

DESTIDLENG(data-value)

specifies the length (halfword binary value) of the name specified in the DESTID option.

PRINT

specifies that the output is to the print medium.

SUBADDR(data-value)

specifies the medium subaddress as a halfword binary value (in the range 0–through 15) that allows media of the same type, for example, “printer 1” or “printer 2”, to be defined. Value 15 means a medium of any type. The default is zero.

VOLUME(data-value)

specifies the name (1–6 characters) of a diskette in an outboard destination that contains the data set specified in the DESTID option.

VOLUMELENG(data-value)

specifies the length of the name specified in the VOLUME option as a halfword binary value.

WPMEDIA1 through WPMEDIA4

specifies that, for each specific LUTYPE4 device, a word-processing medium is defined to relate to a specific input/output device.

ISSUE WAIT conditions

FUNCERR

occurs if there is an error during execution of the command. Destination selection is unaffected and other commands for the same destination may be successful.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

SELNERR

occurs if there is an error during destination selection. The destination is not selected and other commands for the same destination are unlikely to be successful.

Default action: terminate the task abnormally.

UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

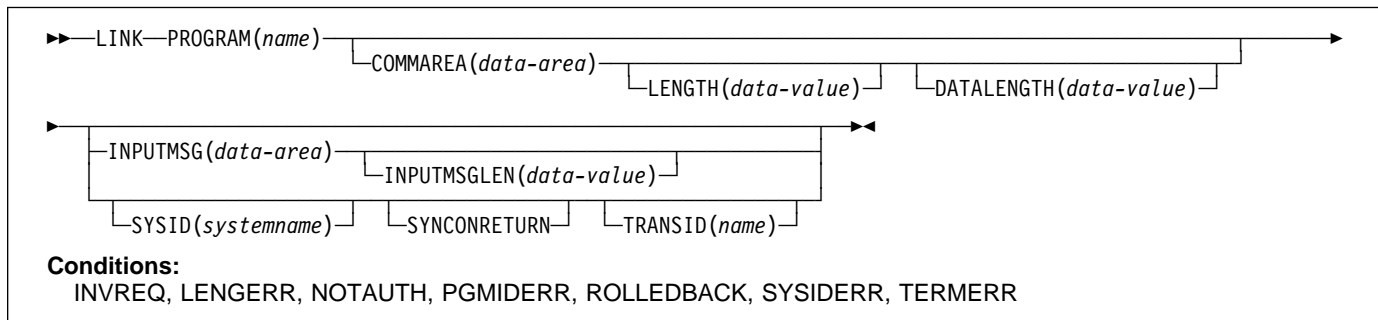
LINK

Function

Link to another program expecting return.

The external CICS interface provides an EXEC CICS LINK command that performs all six commands of the interface in one invocation.

Command syntax



LINK passes control from an application program at one logical level to an application program at the next lower logical level. If the requested program is not defined to CICS, and AUTOINSTALL is active, CICS supplies a definition for the program. If it is not a remote program in another CICS region, and the linked-to program is not already in main storage, CICS loads it. If the SYSID option specifies the name of a remote CICS region, CICS ships the link request to the remote region. When the RETURN command is executed in the linked-to program, control is returned to the program initiating the link at the next sequential executable instruction.

The following example shows how to request a link to an application program called PROG1:

```
EXEC CICS LINK PROGRAM('PROG1')
```

The linked-to program operates independently of the program that issues the LINK command with regard to handling # conditions, attention identifiers, abends, and execution key. For example, the effects of HANDLE CONDITION commands in the linking program are not inherited by the linked-to program, but the original HANDLE CONDITION commands are restored on return to the linking program. See the *CICS/ESA Application Programming Guide* for more information and an illustration of the concept of logical levels.

You can use the HANDLE ABEND command to deal with abnormal terminations in other link levels. See the *CICS/ESA Application Programming Guide* for further details about the relationship between LINK and HANDLE ABEND.

Distributed program link

If you specify a remote region name on the SYSID option (with or without the associated TRANSID and SYNCONRETURN options), the link is a *distributed program link* (DPL). In response to a distributed program link, the local CICS region (the *client region*) ships the link request to the remote region (the *server region*). The server region executes the linked-to program (the *server program*) on behalf of the program issuing the link request (the *client program*).

The SYSID and INPUTMSG options are mutually exclusive. If you specify both options on a LINK command, the translator issues error message DFH7230 (severity E) indicating conflicting options. See the DFH7xxx (DFHEXP command translator diagnostic) messages entry in the *CICS/ESA Messages and Codes* manual for an explanation of severity E for the different supported languages.

A server program running in the server region is restricted to a DPL subset of the CICS API. Briefly, the server program cannot issue:

- Terminal control commands that reference the principal facility
- Options of ASSIGN that return terminal attributes
- BMS commands
- Signon and signoff commands
- Batch data interchange commands
- Commands that address the TCTUA.

For details of the restricted DPL subset of the API, see Appendix G, "API restrictions for distributed program link" on page 367.

Abends in the server program: If a server program abends, the abend code is returned to the client program. If the client program is not written to handle the abend returned by the server program, the client program abends with the same abend code returned by the server program.

LINK options

COMMAREA(data-area)

specifies a communication area that is to be made available to the invoked program. In this option a pointer to the data area is passed. In a COBOL receiving program, you must give this data area the name DFHCOMMAREA in the receiving program. (See the section about passing data to other programs in the *CICS/ESA Application Programming Guide*.)

DATALENGTH(data-value)

specifies a halfword binary value that is the length of a contiguous area of storage, from the start of the COMMAREA, to be passed to the invoked program. If the amount of data being passed in a COMMAREA is small, but the COMMAREA itself is large so that the linked-to program can return the requested data, you should specify DATALENGTH in the interest of performance.

DATALENGTH cannot be used at the same time as INPUTMSG.

INPUTMSG(data-area)

specifies data to be supplied to the invoked program when it first issues a RECEIVE command. This data remains available until the execution of a RECEIVE or RETURN command. An invoked program can invoke a further program and so on, creating a chain of linked programs. If a linked-to chain exists, CICS supplies the INPUTMSG data to the first RECEIVE command executed in the chain. If control returns to the program that issued the LINK with INPUTMSG before the INPUTMSG data has been accepted by a RECEIVE command, CICS assumes that a RECEIVE command has been issued. This means that the original INPUTMSG data is no longer available.

INPUTMSG cannot be used at the same time as DATALENGTH.

See also the *CICS/ESA Application Programming Guide* for more information about the INPUTMSG option.

INPUTMSGLEN(data-value)

specifies a halfword binary value to be used with INPUTMSG.

LENGTH(data-value)

specifies a halfword binary value that is the length in bytes of the COMMAREA (communication area). This value may not exceed 32 500 bytes if the COMMAREA is to be passed between any two CICS servers (for any combination of product/version/release).

PROGRAM(name)

specifies the identifier (1–8 characters) of the program to which control is to be passed unconditionally. The specified name must have been defined as a program to CICS, though if AUTOINSTALL is active a definition is autoinstalled. If the SYSID option specifies a remote region, the linked-to program is the server program in the server region.

Note the use of quotes:

```
EXEC CICS LINK PROGRAM('PROGX')
```

PROGX is in quotes because it is the program name.

```
EXEC CICS LINK PROGRAM(DAREA)
```

DAREA is not in quotes because it is the name of a data area.

SYNCONRETURN

specifies that the server region named on the SYSID option is to take a syncpoint on successful completion of the server program.

```
# _____ Apar PQ03185 _____
# Documentation for Apar PQ03185 added 19/05/97
#
# Synconreturn is only applicable to remote links; it is
# ignored if the link is local.
```

SYSID(systemname)

specifies the system name of the CICS server region where the program link request is to be routed.

```
# If SYSID specifies a remote system, no reference is
# made to PROGRAM resource definitions held locally. If
# SYSID specifies the local system, CICS treats the LINK
# request as if SYSID was not specified.
```

TRANSID(name)

specifies the name of the mirror transaction that the remote region is to attach, and under which it is to run the server program. If you omit the TRANSID option, the server region attaches either CSMI, CPMI, or CVMI by default.

The transaction name you specify on the LINK command takes priority over any transaction specified on the program resource definition. Whilst you can specify your own name for the mirror transaction initiated by DPL requests, the transaction *must* be defined in the server region, and the transaction definition must specify the mirror program, DFHMIRS.

LINK

LINK conditions

INVREQ

occurs in any of the following situations:

- A LINK command with the INPUTMSG option is issued for a program that is not associated with a terminal, or that is associated with an APPC logical unit, or an IRC session (RESP2=8).
- A LINK command with the INPUTMSG option is issued for a program that is the subject of a DPL request; that is, SYSID is also specified (RESP2=19).
- The SYNCONRETURN option is specified but the program issuing the link request (the client program) is already in conversation with a mirror task in the remote region specified on the SYSID option.

#

Apar PQ25518

Documentation for Apar PQ25518 added
17/08/99

#

#

(That is, a logical unit-of-work (LUW) is in progress or the system initialization parameter MROFSE=YES has been specified in the client region) In this case, the client program is in an incorrect state to support the SYNCONRETURN option (RESP2=14).

- The program issuing the link request is already in conversation with a mirror task and the TRANSID specified is different from the transaction identifier of the active mirror (RESP2=15).
- The TRANSID specified is all blanks (RESP2=16).
- The program manager domain has not yet been initialized. This is probably due to a link request having been made in a first stage PLT (RESP2=30).

Default action: terminate the task abnormally.

Note: RESP2 values are not returned to the client for conditions occurring in a DPL server program.

LENGERR

occurs in any of the following situations:

- The length specified on the LENGTH option is greater than the length of the data area specified in the COMMAREA option, and while that data was being copied a destructive overlap occurred because of the incorrect length.
- The COMMAREA length is less than 0 or greater than 32767 (RESP2=11).
- The length specified on the DATALENGTH option is a negative value (RESP2=12).
- The length specified on the DATALENGTH option is greater than the length specified on the LENGTH option (RESP2=13).

- The COMMAREA address is zero, but the COMMAREA length is non zero (RESP2=26).
- The INPUTMSG length less than 0 or greater than 32767 (RESP2=27).

Default action: terminate the task abnormally.

Note: RESP2 values are not returned to the client for conditions occurring in a DPL server program.

NOTAUTH

occurs when a resource security check has failed on PROGRAM(name) (RESP2=101).

Default action: terminate the task abnormally.

PGMIDERR

occurs if a program:

- Has no entry in the PPT and either program autoinstall was switched off, or the program autoinstall control program indicated that the program should not be autoinstalled (RESP2=1)
- Is disabled (RESP2=2)
- Could not be loaded because
 - This was the first load of the program and the program load failed, usually because the load module could not be found.
 - This was a subsequent load of the program, but the first load failed.

In order to reset the load status the load module must be in the DFHRPL concatenation, and a SET PROGRAM NEWCOPY will be required (RESP2=3).

- The program autoinstall control program failed either because the program autoinstall control program is incorrect, incorrectly defined, or as a result of an abend in the program autoinstall control program. Program autoinstall is disabled and message DFHPG0202 or DFHPG0203 written to the CSPL (RESP2=21).
- The model returned by the program autoinstall control program was not defined in the PPT table, or was not enabled (RESP2=22).
- The program autoinstall control program returned invalid data (RESP2=23).
- Define for the program failed due to autoinstall returning an invalid program name or definition (RESP2=24).

Default action: terminate the task abnormally.

ROLLEDBACK

occurs if the SYNCONRETURN is specified and the server program is unable successfully to take a syncpoint. The server program has taken a rollback, and all changes made to recoverable resources in the remote region, within the current LUW, are backed out (RESP2=29).

Default action: terminate the task abnormally.

SYSIDERR

occurs in any of the following situations:

- The SYSID specified cannot be found in the intersystem table, or if the link to the specified system is unavailable (RESP2=18).
- The remote system specified by SYSID is an LUTYPE6.1-connected system. Distributed program link requests are not supported on LUTYPE6.1 connections (RESP2=20).

Notes:

1. There is no local queuing in the event of a SYSIDERR.
2. RESP2 values are not returned for conditions occurring on DPL requests.

Default action: terminate the task abnormally.

TERMERR

occurs if an irrecoverable error occurs during the conversation with the mirror (for example, if the session fails, or if the server region fails) (RESP2=17).

Default action: terminate the task abnormally.

Note: RESP2 values are not returned to the client for conditions occurring in a DPL server program.

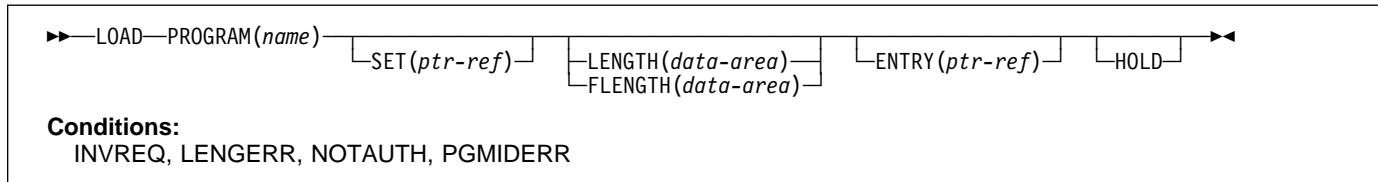
LOAD

LOAD

Function

Load a program from the CICS DFHRPL concatenation library into main storage.

Command syntax



Dynamic transaction routing

Using LOAD with HOLD, or using a resource that has been defined with RELOAD=YES, could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

LOAD makes available to the invoking task a copy of an application program, table, or map. If the program is defined with RELOAD=NO, it is only fetched from the library where it resides, if there is not a copy already in main storage. If the program is defined with RELOAD=YES, a new copy is always fetched from the library. (See the *CICS/ESA Application Programming Guide* for further details about maps.) Using LOAD can reduce system overhead. The following example shows how to load a user-prepared table called TB1:

```
EXEC CICS LOAD PROGRAM('TB1') SET(PTR)
```

LOAD options

ENTRY(ptr-ref)

specifies the pointer reference that is to be set to the address of the entry point in the program that has been loaded.

The top bit of the address is set on if the program is defined with AMODE=31.

For assembler programs without an explicit ENTRY defined in the linkedit definitions, the entry point returned depends on (1) whether there is a CICS stub, and (2) whether the LOAD command is issued from a PLT program:

- If there is a CICS stub, the entry point address is incremented for this stub unless the LOAD command is issued from a PLT program executed during the first phase of initialization or the final phase of shutdown.
- If there is not a CICS stub, the entry point address is the same as the load point address.

FLENGTH(data-area)

specifies a fullword binary area to be set to the length of the loaded program, table, or map. Use FLENGTH if the length of the loaded program is greater than 32KB.

HOLD

specifies that the loaded program, table, or map is not to be released (if still available) when the task issuing the LOAD command is terminated; it is to be released only in response to a RELEASE command from this task or from another task.

If you omit HOLD, the program, table, or map is released when the task that issued the load terminates or issues a RELEASE command.

If, however, the program is defined with RELOAD=YES, neither of the above apply. RELEASE does not work, and a FREEMAIN must be issued to get rid of the program.

LENGTH(data-area)

specifies a halfword binary value to be set to the length of the loaded program, table, or map. To avoid raising the LENGERR condition, use FLENGTH if the length of the loaded program is likely to be greater than 32KB.

PROGRAM(name)

specifies the identifier (1–8 characters) of a program, table, or map to be loaded. The specified name must have been defined as a program to CICS, though if AUTOINSTALL is active a definition is autoinstalled.

SET(ptr-ref)

specifies the pointer reference that is to be set to the address at which a program, table, or map is loaded.

LOAD conditions

INVREQ

Occurs if the program manager domain has not yet been initialized. This is probably due to a load request having been made in a first stage PLT (RESP2=30).

Default action: terminate the task abnormally.

LENGERR

occurs when LENGTH is used and the length of the loaded program is not less than 32KB. (RESP2=19).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on PROGRAM(name) (RESP2=101).

Default action: terminate the task abnormally.

PGMIDERR

occurs in any of the following situations:

- A program, table, or map has no entry in the PPT and either program autoinstall was switched off, or the program autoinstall control program indicated that the program should not be autoinstalled (RESP2=1).
- A program is disabled (RESP2=2).
- A program could not be loaded because
 - This was the first load of the program and the program load failed, usually because the load module could not be found.
 - This was a subsequent load of the program, but the first load failed.

In order to reset the load status the load module must be in the DFHRPL concatenation, and a SET PROGRAM NEWCOPY will be required (RESP2=3).

- The installed program definition is for a remote program (RESP2=9).
- The program autoinstall control program failed either because the program autoinstall control program is incorrect, incorrectly defined, or as a result of an abend in the program autoinstall control program. Program autoinstall is disabled and message DFHPG0202 or DFHPG0203 written to the CSPL (RESP2=21).
- The model returned by the program autoinstall control program was not defined in the PPT table, or was not enabled (RESP2=22).
- The program autoinstall control program returned invalid data (RESP2=23).
- Define for the program failed due to autoinstall returning an invalid program name or definition (RESP2=24).

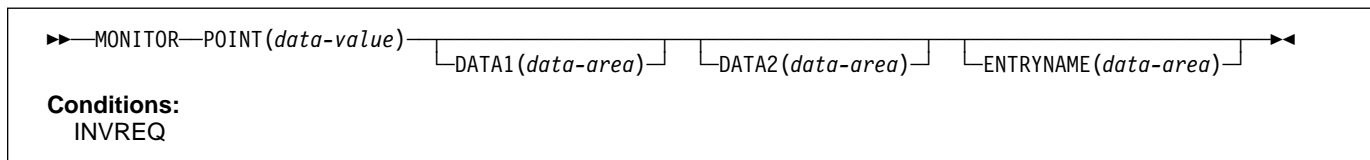
Default action: terminate the task abnormally.

MONITOR

Function

Code a user event-monitoring point.

Command syntax



MONITOR provides information about the performance of your application transactions. It replaces the monitoring aspects of ENTER TRACEID.

In addition to the monitoring data collected at predefined event monitoring points (EMPs) within CICS, a user application program can contribute data to user fields within the CICS monitoring records. You can do this by using the MONITOR command to invoke user-defined EMPs. At each of these user EMPs, you can add or change 1–16384 bytes of your own data in each performance monitoring record. In those 16384 bytes, you can have any combination of the following:

- 0 through 256 counters
- 0 through 256 clocks
- A single 8192-byte character string.

For example, you could use these user EMPs to count the number of times a certain event occurs, or to time the interval between two events.

Figure 4 gives examples of MONITOR commands (and of the MCT entries you need for them). See the *CICS/ESA Customization Guide* for more information about monitoring.

Notes:

1. Example 1 shows a user clock being started by an application identified as PROG3. This is the eleventh EMP in this application. To prevent confusion with the eleventh EMP in another application, this EMP is uniquely identified by the tag ENTRY3.11. The clock that is being started is the first clock in a string.
2. Example 2 shows the same user clock being stopped, by the same application, but from a different EMP. The EMP is uniquely identified by the tag ENTRY3.12.
3. Example 3 shows some user data being loaded into the 32-byte character string reserved for that purpose. The loading starts at offset 0, and the data is no more than 32 bytes in length.

```

1:
EXEC CICS MONITOR
      POINT(11)
      ENTRYNAME(ENTRY3)
      needing: DFHMCT TYPE=EMP,
              CLASS=PERFORM,
              ID=(ENTRY3.11),
              CLOCK=(1,CLOCKA),
              PERFORM=SCLOCK(1)

2:
EXEC CICS MONITOR
      POINT(12)
      ENTRYNAME(ENTRY3)
      needing: DFHMCT TYPE=EMP,
              CLASS=PERFORM,
              ID=(ENTRY3.12),
              PERFORM=PCLOCK(1)

3:
EXEC CICS MONITOR
      POINT(13)
      DATA1(address of data)
      DATA2(length of data)
      ENTRYNAME(ENTRY3)
      needing: DFHMCT TYPE=EMP,
              CLASS=PERFORM,
              ID=(ENTRY3.13),
              PERFORM=MOVE(0,32)

```

Figure 4. Examples of coding user EMPs

MONITOR options

DATA1(data-area)

specifies a 4-byte variable whose contents depend on the type of user EMP being used:

- If the user EMP contains an ADDCNT, SUBCNT, NACNT, EXCNT, or ORCNT option, the DATA1 variable is an area used as defined by the MCT user EMP definition.
- If the MCT user EMP definition contains an MLTCNT option, the DATA1 variable is an area with the address of a series of adjacent fullwords containing the values to be added to the user count fields defined in the MCT user EMP definition.
- If the MCT user EMP definition contains a MOVE option, the DATA1 variable is an area with the address of the character string to be moved.

See the &dfha500I. for details of user EMP options.

DATA2(data-area)

specifies a 4-byte variable whose contents depend on the type of user EMP being used:

- If the user EMP contains an ADDCNT, SUBCNT, NACNT, EXCNT, or ORCNT option, the DATA2 variable is an area used as defined by the MCT user EMP definition.
- If the MCT user EMP definition contains an MLTCNT option, the DATA2 variable is an area with the number of user count fields to be updated. The number specified in DATA2 overrides the default value defined in the MCT for the operation. A value of zero instructs monitoring to use the default. If DATA2 is not specified, the MLTCNT operation raises an INVREQ condition although the operation was successful.
- If the MCT user EMP definition contains a MOVE option, the DATA2 variable is an area with the length of the character string to be moved. The number specified in DATA2 will override the default value defined in the MCT for the operation. A value of zero instructs monitoring to use the default. If DATA2 is not specified, the MOVE operation raises an INVREQ although the operation was successful.

See the &dfha500I. for details of user EMP options.

ENTRYNAME(data-area)

is the monitoring point entry name that qualifies the POINT value and is defined in the monitoring control table (MCT). ENTRYNAME defaults to USER if not specified.

POINT(data-value)

specifies the monitoring point identifier as defined in the MCT, and is in the range 0 through 255. Note, however, that point identifiers in the range 200 through 255 are reserved for use by IBM program products.

MONITOR condition

INVREQ

occurs in any of the following situations:

- Your POINT value is outside the range 1 through 255 (RESP2=1)
- Your POINT value is not defined in the MCT (RESP2=2)
- Your DATA1 value is not valid (RESP2=3)
- Your DATA2 value is not valid (RESP2=4)
- You did not specify DATA1 for an MCT operation that required it (RESP2=5)
- You did not specify DATA2 for an MCT operation that required it (RESP2=6)

Default action: terminate the task abnormally.

POINT

POINT

Function

Get information about an LUTYPE6.1 logical unit.

Command syntax



POINT gets information about a named facility, such as whether it owns the given facility.

This command can be used on an MRO session.

POINT options

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If both this option and CONVID are omitted, the principal facility for the task is used.

POINT condition

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

POP HANDLE

Function

Restore the stack.

Command syntax

▶—POP HANDLE—◀

Conditions:
INVREQ

POP HANDLE enables you to restore the effect of IGNORE CONDITION, HANDLE ABEND, HANDLE AID, and HANDLE CONDITION commands to the state they were in before a PUSH HANDLE command was executed at the current link level. This can be useful, for example, during a branch to a subroutine embedded in a main program.

```
# Normally, when a CICS program calls a subroutine (at the
# same logical level), the program or routine that receives
# control inherits the current HANDLE commands. These
# commands may not be appropriate within the called program.
# The called program can use PUSH HANDLE to suspend
# existing HANDLE commands, and before returning control to
# the caller, can then restore the original commands using the
# POP HANDLE command.
```

```
# Note: When a CICS program uses EXEC CICS LINK to call
# another CICS program, the HANDLE CONDITION
# options are NOT inherited by the linked-to program,
# but CICS will search preceding logical levels for a
# HANDLE ABEND exit. See CICS/ESA Application
# Programming Guide for further details about the
# relationship between LINK and HANDLE ABEND.
```

You can nest PUSH HANDLE ... POP HANDLE command sequences within a task. Each POP HANDLE command restores a set of specifications.

The C language does not support POP HANDLE.

POP HANDLE condition

INVREQ

occurs if no matching PUSH HANDLE command has been executed at the current link level.

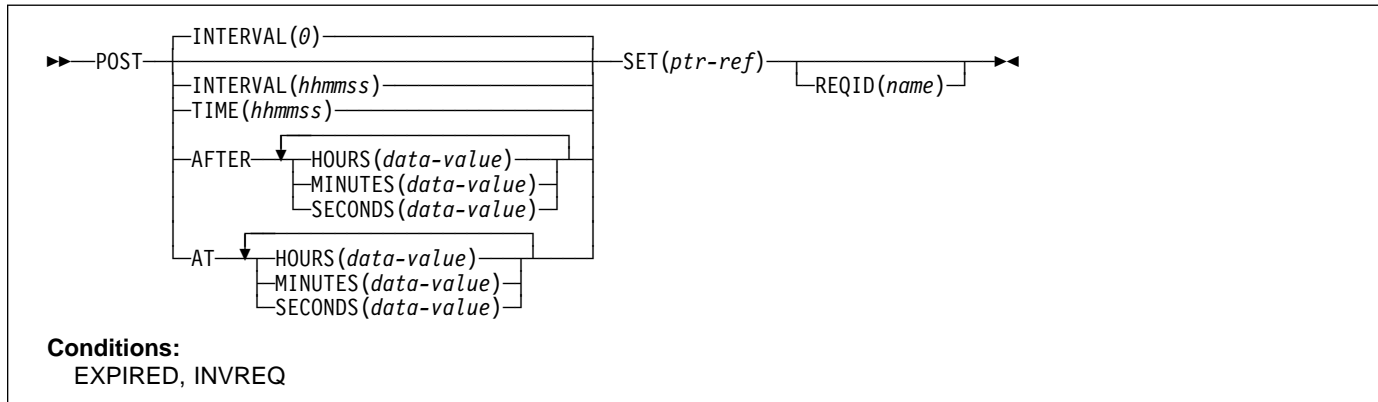
Default action: terminate the task abnormally.

POST

Function

Request notification when a specified time has expired.

Command syntax



Note for dynamic transaction routing

Using POST if later CANCELED by another task could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

POST requests notification that a specified time has expired. In response to this command, CICS makes a timer-event control area available for testing. This 4-byte control area is initialized to binary zeros, and the pointer reference specified in the SET option is set to its address.

When the time you specify has expired, the timer-event control area is posted; that is, its first byte is set to X'40' and its third byte to X'80'. You can test posting in either of the following ways:

- By checking the timer-event control area at intervals. You must give CICS the opportunity to post the area; that is, the task must relinquish control of CICS before you test the area. Normally, this condition is satisfied as a result of other commands being issued; if a task is performing a long internal function, you can force control to be relinquished by issuing a SUSPEND command.
- By suspending task activity by a WAIT EVENT or WAIT EXTERNAL command until the timer-event control area is posted. This action is similar to issuing a DELAY command but, with a POST and WAIT EVENT or WAIT EXTERNAL command sequence, you can do some processing after issuing the POST command; a DELAY command suspends task activity at once. No other task should attempt to wait on the event set up by a POST command.
- By using WAITCICS.

The timer-event control area can be released for a variety of reasons (see below). If this happens, the result of any other task issuing a WAIT command on the event set up by the POST command is unpredictable.

However, other tasks can cancel the event if they have access to the REQID associated with the POST command. (See the CANCEL command and the description of the REQID option.) A timer-event control area provided for a task is not released or altered (except as described above) until one of the following events occurs:

- The task issues a subsequent DELAY, POST, or START command.
- The task issues a CANCEL command to cancel the POST command.
- The task is terminated, normally or abnormally.
- Any other task issues a CANCEL command for the event set up by the POST command.

A task can have only one POST command active at any given time. Any DELAY, POST, or START command supersedes a POST command previously issued by the task.

The default is INTERVAL(0), but for C the default is AFTER HOURS(0) MINUTES(0) SECONDS(0).

The following example shows you how to request a timer-event control area for a task, to be posted after 30 seconds:

```
EXEC CICS POST
  INTERVAL(30)
  REQID('RBL3D')
  SET(PREF)
```

The following example shows you how to provide a timer-event control area for the task, to be posted when the specified time of day is reached. Because no request identifier is specified in the command, CICS automatically assigns one and returns it to the application program in the EIBREQID field in the EIB.

```
EXEC CICS POST
  TIME(PACKTIME)
  SET(PREF)
```

There are two ways to enter the time under AFTER and AT.

1. A combination of at least two of HOURS(0–99), MINUTES(0–59), and SECONDS(0–59). HOURS(1) SECONDS(3) would mean one hour and three seconds (the minutes default to zero).
2. As one of HOURS(0–99), MINUTES(0–5999), or SECONDS(0–359999). HOURS(1) means one hour. MINUTES(62) means one hour and two minutes. SECONDS(3723) means one hour, two minutes, and three seconds.

POST options

AFTER

specifies the interval of time to elapse.

AT

specifies the time of expiring.

HOURS(data-value)

specifies a fullword binary value in the range 0–99.

INTERVAL(hhmmss)

specifies an interval of time that is to elapse from the time at which the POST command is issued. The **mm** and **ss** are in the range 0–59. The time specified is added to the current clock time by CICS when the command is executed to calculate the expiration time.

This option is used to specify when the posting of the timer-event control area should occur.

When using the C language, you are recommended to use the AFTER/AT HOURS, MINUTES, and SECONDS options as C does not provide a packed decimal data type. You may use INTERVAL, but if the value specified is **not** an integer constant, the application is responsible for ensuring that the value passed to CICS is in packed decimal format.

MINUTES(data-value)

specifies a fullword binary value in the range 0–59, when HOURS or SECONDS are also specified, or 0–5999 when MINUTES is the only option specified.

REQID(name)

specifies a name (1–8 characters), which should be unique, to identify the POST request. Using this option to specify an application-defined name is one way to enable another transaction to cancel the POST request.

If you do not specify your own REQID, CICS generates a unique request identifier for you in the EIBREQID field of the EXEC interface block. This, like your own REQID, can be used by another transaction to cancel the POST request.

To enable other tasks to cancel unexpired POST requests, you must make the request identifier dynamically available. For example, storing it in a TS queue, whose name is known to other applications that may want to cancel the POST request, is one way you can pass a request identifier to other transactions.

SECONDS(data-value)

specifies a fullword binary value in the range 0–59, when HOURS or MINUTES are also specified, or 0–359999 when SECONDS is the only option specified.

SET(ptr-ref)

specifies the pointer reference to be set to the address of the 4-byte timer-event control area generated by CICS. This area is initialized to binary zeros; on expiration of the specified time, the first byte is set to X'40', and the third byte to X'80'.

The timer-event control area always resides below the 16MB line in shared dynamic storage (SDSA).

TIME(hhmmss)

specifies the time when the posting of the timer-event control area should occur.

When using the C language, you are recommended to use the AFTER/AT HOURS, MINUTES, and SECONDS options as C does not provide a packed decimal data type. You may use TIME, but if the value specified is **not** an integer constant, the application is responsible for ensuring that the value passed to CICS is in packed decimal format. See the section about expiration times in the *CICS/ESA Application Programming Guide*.

POST conditions

EXPIRED

occurs if the time specified has already expired when the command is issued.

Default action: ignore the condition.

INVREQ

occurs in any of the following situations:

- The POST command is not valid for processing by CICS
- Hours are out of range (RESP2=4)
- Minutes are out of range (RESP2=5)
- Seconds are out of range (RESP2=6).

Default action: terminate the task abnormally.

PURGE MESSAGE

PURGE MESSAGE

Function

Discontinue building a BMS logical message.

Command syntax

▶—PURGE MESSAGE—▶

Conditions:

Full BMS: INVREQ, TSIOERR

PURGE MESSAGE discontinues the building of a BMS logical message. It deletes the current logical message, including any pages of device-dependent data stream already written to CICS temporary storage. The application program may then build a new logical message.

The portions of the logical message already built in main storage or in temporary storage are deleted.

See Appendix K, “BMS macro summary” on page 381 for map definition macros.

PURGE MESSAGE is only available on full-function BMS. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

PURGE MESSAGE condition

INVREQ

occurs if the command was called in a distributed program link server program (RESP2=200).

Default action: terminate the task abnormally.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

PUSH HANDLE

Function

Suspend the stack.

Command syntax

▶▶—PUSH HANDLE—◀◀

PUSH HANDLE enables you to suspend the current effect of IGNORE CONDITION, HANDLE ABEND, HANDLE AID, and HANDLE CONDITION commands. This can be useful, for example, during a branch to a subroutine embedded in a main program.

Normally, when a CICS program calls a subroutine (at the
same logical level), the program or routine that receives
control inherits the current HANDLE commands. These
commands may not be appropriate within the called program.
The called program can use PUSH HANDLE to suspend
existing HANDLE commands.

Note: When a CICS program uses EXEC CICS LINK to call
another CICS program, the HANDLE CONDITION
options are NOT inherited by the linked-to program,
but CICS will search preceding logical levels for a
HANDLE ABEND exit. See *CICS/ESA Application
Programming Guide* for further details about the
relationship between LINK and HANDLE ABEND.

You can nest PUSH HANDLE ... POP HANDLE command sequences within a task. Each PUSH HANDLE command stacks a set of specifications.

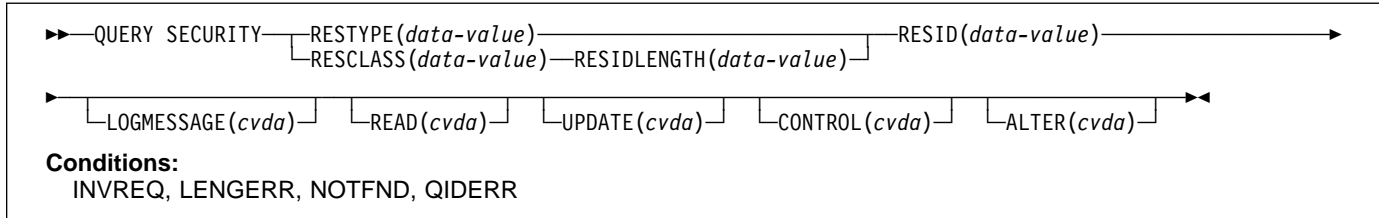
The C language does not support PUSH HANDLE.

QUERY SECURITY

Function

To query the security authorization of the user.

Command syntax



QUERY SECURITY allows the application to determine whether the user has access to resources defined in the external security manager (ESM). These resources can be:

- In CICS resource classes
- In user-defined resource classes.

The user in this context is the user invoking the transaction that contains the QUERY SECURITY command.

For more information on the use of the QUERY SECURITY command, see the *CICS/ESA CICS-RACF Security Guide*.

QUERY SECURITY options

ALTER(cvda)

enables you to query whether the user has ALTER authority for the named resource. The cvda values returned by CICS are ALTERABLE and NOTALTERABLE.

CONTROL(cvda)

enables you to query whether the user has CONTROL authority for the named resource. The cvda values returned by CICS are CTRLABLE and NOTCTRLABLE.

LOGMESSAGE(cvda)

enables you to inhibit security violation messages. The values passed to CICS are LOG (the default value), or, to inhibit messages, NOLOG.

READ(cvda)

enables you to query whether the user has READ authority for the named resource. The cvda values returned by CICS are READABLE and NOTREADABLE. READ access authority usually permits nondestructive use of a resource as, for example, in the case of READ and INQUIRE commands.

RESCLASS(data-value)

specifies an 8-character field identifying the name of a valid resource class, that could be non-CICS, in the ESM. The class name identified by RESCLASS is treated literally with no translation.

If the ESM is RACF, the class can be CICS-supplied or user-defined. RESCLASS enables you to define more narrowly the authorization to be queried; for example, you can query at the record or field level.

The responses returned by the command reflect the definition of the RESID resource as defined in the specified RESCLASS.

RESID(data-value)

specifies the name of the CICS or user-defined resource that you want to query the users access to. The value is a character string (1-12 characters for a CICS resource, and 1-246 characters for a user-defined resource,

#

Apar PQ18509

#

Documentation for Apar PQ18509 added 28/08/98

#

#

#

unless you are using the COBOL3 or OOCOBOL translator option in which case the maximum length is 160 characters).

|

|

|

|

|

Note that the actual resource checked depends on whether RESCLASS or RESTYPE is specified in the command and whether prefixing is active (SECPRFX=YES specified in the System Initialization Table).

|

|

|

|

|

|

If RESCLASS is specified, the resource checked is always the actual RESID data-value, whether or not prefixing is on or off. IF RESTYPE is specified and SECPRFX=NO, the resource checked is the RESID data-value as specified, however if SECPRFX=YES, the resource checked is the RESID data-value prefixed with the Region USERID.

RESIDLENGTH(data-value)

specifies the length, as a fullword binary, of the resource identifier in RESID. You only use this parameter when specifying the RESCLASS option.

RESTYPE(data-value)

specifies the type of resource (1–12 characters) you want to query the user's access to.

The responses returned by the command reflect the results that would be obtained if an actual attempt was made to access the specified CICS resource.

The value you specify for RESTYPE must be one of the following resource types:

- FILE
- JOURNALNUM
- PROGRAM
- PSB
- SPCOMMAND1
- TDQUEUE
- TRANSACTION
- TRANSATTACH
- TSQUEUE

¹ See *CICS/ESA CICS-RACF Security Guide*

Apar 61283

Documentation for Apar 61283 added 13 Oct 1994 (TUCKER)

With dynamic transaction routing, it is not necessary to install transaction definitions in terminal owning regions. A QUERY SECURITY command with a RESTYPE of TRANSATTACH returns the NOTFND condition if the transaction is not installed. Programmers, however, should be aware that the transaction may be routed dynamically.

UPDATE(cvda)

enables you to query whether the user has UPDATE authority for the named resource. The cvda values returned by CICS are UPDATABLE and NOTUPDATABLE. UPDATE access authority usually permits destructive use of a resource as, for example, in the case of WRITE, DELETE, or UPDATE commands.

QUERY SECURITY conditions

INVREQ

occurs in any of the following circumstances:

- The cvda value is not valid for the LOGMESSAGE (RESP2=7)
- The RESID is invalid or filled with blanks (RESP2=9)
- The external security manager (ESM) is inactive or not present (RESP2=10).

Default action: terminate the task abnormally.

LENGERR

occurs if the RESIDLENGTH value is not valid, that is, not in the range 1 through 246 (RESP2=6).

Default action: terminate the task abnormally.

NOTFND

occurs in any of the following circumstances:

- The RESID is not valid (RESP2=1)
- The RESTYPE is not valid (RESP2=2)
- The RESID value for RESTYPE (SPCOMMAND) not valid (RESP2=3)

QUERY SECURITY

- The RESCLASS is not defined to the external security manager (ESM) (RESP2=5)
- #
- The resource is not protected (RESP2=8). This is only returned when QUERY SECURITY is used with the RESCLASS option (and never occurs with RESTYPE).
- #
- #

Possible causes include:

- RESCLASS not active
- No profile found
- ESM not active.

| Default action: terminate the task abnormally.

QIDERR

occurs if an indirect queue name associated with the given RESID is not found (RESP2=1).

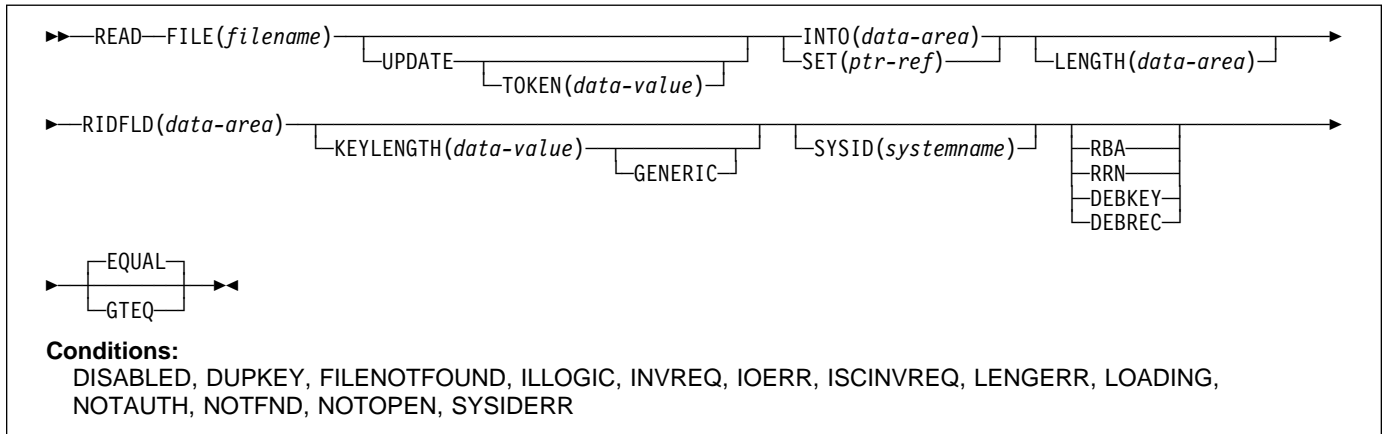
| Default action: terminate the task abnormally.

READ

Function

Read a record from a file.

Command syntax



READ reads a record from a file on a local or a remote system.

For both UPDATE and non-UPDATE commands, you must identify the record to be retrieved by the record identification field specified in the RIDFLD option. Immediately upon completion of a READ UPDATE command, the RIDFLD data area is available for reuse by the application program.

When this command reads a CICS-maintained data table, a READ request with UPDATE or RBA is always satisfied by a call to VSAM. For a non-UPDATE key READ, an attempt is first made to satisfy the request from the in-memory table. If the record is not found, a call is made to VSAM unless it is known that the record is not in the same data set either. See the SDTG for more detail.

A full key direct read that is neither a generic read nor a READ UPDATE, is satisfied by a reference to the data table. If the record is not found in the table, the source data set is accessed, unless the table is known to be complete, that is, all records in the source are also present in the table (which is the case if loading is finished and none has been rejected by a user exit). The details of the command for a CICS-maintained table are the same as for a VSAM KSDS.

| When this command reads a user-maintained data table,
 | only the data table is accessed once loading is complete; the
 | VSAM file is not changed in any way.

The following example shows you how to read a record from a file named MASTER into a specified data area:

```
EXEC CICS READ
  INTO(RECORD)
  FILE('MASTER')
  RIDFLD(ACCTNO)
```

The following example shows you how to read a record for update from a VSAM file using a generic key and specifying a greater-or-equal key search.

```
EXEC CICS READ
  INTO(RECORD)
  LENGTH(RECLEN)
  FILE('MASTVSAM')
  RIDFLD(ACCTNO)
  KEYLENGTH(4)
  GENERIC
  GTEQ
  UPDATE
```

READ options

DEBKEY (blocked BDAM)

specifies that deblocking is to occur by key. If neither DEBREC nor DEBKEY is specified, deblocking does not occur.

DEBREC (blocked BDAM)

specifies that deblocking is to occur by relative record (relative to zero). If neither DEBREC nor DEBKEY is specified, deblocking does not occur.

EQUAL (VSAM KSDS, paths and data tables)

specifies that the search is satisfied only by a record having the same key (complete or generic) as that specified in the RIDFLD option.

FILE(filename)

specifies the of the file to be accessed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the

READ

FCT entry is used to find out whether the data set is on a local or a remote system.

GENERIC (VSAM KSDS, paths and data tables)

specifies that the search key is a generic key whose length is specified in the KEYLENGTH option. The search for a record is satisfied when a record is found that has the same starting characters (generic key) as those specified.

For UMTs, GENERIC cannot be specified with UPDATE.

GTEQ (VSAM KSDS, paths and data tables)

specifies that, if the search for a record having the same key (complete or generic) as that specified in the RIDFLD option is unsuccessful, the first record having a greater key is retrieved.

For UMTs, GENERIC cannot be specified with UPDATE.

INTO(data-area)

specifies the data area into which the record retrieved from the data set is to be written.

KEYLENGTH(data-value)

specifies the length (halfword binary) of the key that has been specified in the RIDFLD option, except when RBA or RRN is specified, in which case KEYLENGTH is not valid. This option must be specified if GENERIC is specified, and it can be specified whenever a key is specified. However, if the length specified is different from the length defined for the data set and the operation is not generic, the INVREQ condition occurs.

The INVREQ condition also occurs if GENERIC is specified and the KEYLENGTH is not less than that specified in the VSAM definition.

If KEYLENGTH(0) is used with the object of reading the first record in the data set, the GTEQ option must also be specified. If EQUAL is specified either explicitly or by default with KEYLENGTH(0), the results of the READ are unpredictable.

LENGTH(data-area)

specifies the length, as a halfword binary value, of the data area where the record is to be put. On completion of the READ command, the LENGTH parameter contains the actual length of the record.

LENGTH must be specified with the INTO option on READ commands involving variable-length records. It need not be specified for fixed-length records, but its inclusion is recommended because:

- It causes a check to be made that the record being read is not too long for the available data area
- When reading fixed-length records into an area longer than the record being accessed, the LENGERR condition is raised for VS COBOL II, C, PL/I, and assembler-language applications if the LENGTH option is not specified. If the length specified exceeds the file record length, CICS uses the longer length for the move. If the target area in

the application program is not large enough, storage is overlaid beyond the target area.

When reading into a target data area longer than the record being read, the contents of the target data area, from the end of the retrieved record to the end of the target data area, are unpredictable.

If you specify the INTO option, the LENGTH argument must be a data area that specifies the largest record the program accepts. If the retrieved record is longer than the value specified in the LENGTH option, the record is truncated to the specified value and the LENGERR condition occurs. In this case, the LENGTH data area is set to the length of the record prior to truncation.

If you specify the SET option, the LENGTH option need not be specified but, if it is, the argument must be a data area.

RBA (VSAM ESDS, KSDS, or CMT)

(base data sets only, not paths) specifies that the record identification field specified in the RIDFLD option contains a relative byte address. This option should only be used when reading records from an ESDS base or when reading from a KSDS base and using relative byte addresses instead of keys to identify the records.

| You cannot use RBA for user-maintained data tables.

RIDFLD(data-area)

specifies the record identification field. When combined with RBA or RRN this is a 4-character field. The contents can be a key, a relative byte address, or relative record number (for VSAM data sets), or a block reference, a physical key, and a deblocking argument (for BDAM data sets). For a relative byte address or a relative record number, the format of this field must be fullword binary. For a relative byte address, the RIDFLD can be greater than or equal to zero. For a relative record number, the RIDFLD can be greater than or equal to 1.

| Immediately upon completion of the command, the RIDFLD data area is available for reuse by the application program, even if UPDATE was specified.

Make sure that the variable specified by RIDFLD is not shorter than the KEYLENGTH specified in this command or, if KEYLENGTH is not specified, the key length of the file you are reading; otherwise, the results are unpredictable.

RRN (VSAM RRDS)

specifies that the record identification field specified in the RIDFLD option contains a relative record number. This option should only be used with files referencing relative record data sets.

SET(ptr-ref)

indicates that CICS is to supply a buffer where the record is read, and specifies the pointer reference that is to contain the address of the retrieved record.

In assembler language, if the DUPKEY condition occurs, the specified register has not been set, but can be loaded from DFHEITP1.

The pointer reference is valid until the next READ command for the same file or until completion of a corresponding REWRITE, DELETE, or UNLOCK command, or a SYNCPOINT in the case of READ UPDATE SET. If you want to retain the data within the field addressed by the pointer, it should be moved to your own area.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, the address of the data is below the 16MB line.

If TASKDATAKEY (USER) is specified for the executing transaction, the data returned is in a user-key; otherwise it is in a CICS-key.

If SET is not specified, LENGTH must either be specified explicitly or must be capable of being defaulted from the INTO option using the length attribute reference in assembler language, or STG and CSTG in PL/I. LENGTH must be specified explicitly in OS/VS COBOL and C.

SYSID(systemname)

specifies the name of the system the request is directed to.

If you specify SYSID, and omit both RBA and RRN, you must also specify LENGTH and KEYLENGTH; they cannot be found in the FCT.

| **TOKEN(data-value)**

specifies as a fullword binary value a unique request identifier for a READ, used to control multiple UPDATE operations on a data set. This is an output value provided by file control to the requesting task.

UPDATE

specifies that the record is to be obtained for updating or (for VSAM and user-maintained data tables) deletion. If this option is omitted, a read-only operation is assumed.

UPDATE guarantees read integrity.

You cannot use UPDATE for user-maintained data tables if you also have either GENERIC or GTEQ.

READ conditions

Note: RESP2 values are not set for files that are on remote systems.

DISABLED

occurs if a file is disabled (RESP2=50). A file may be disabled because:

- It was initially defined as disabled and has not since been enabled

- It has been disabled by a SET FILE or a CEMT SET FILE command.

Default action: terminate the task abnormally.

DUPKEY (VSAM)

occurs if a record is retrieved by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key follows (RESP2=140).

In assembler language, if the SET option is being used, the specified register has not been set, but can be loaded from DFHEITP1.

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if a file name referred to in the FILE option cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

| For user-maintained data tables, this condition occurs
| only for a non-UPDATE READ during loading when
| CICS has attempted to retrieve the record from the
| source data set.

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- READ is not allowed according to the file entry specification in the FCT (RESP2=20).
- A READ command with the UPDATE option is issued to a file where update operations are not allowed according to the file entry specification in the FCT (RESP2=20).
- The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is greater than or equal to the length of a full key (RESP2=25).
- The KEYLENGTH option is specified (but the GENERIC option is not specified), and the specified length does not equal the length defined for the data set to which this file refers (RESP2=26).
- Following a READ UPDATE command for a file, another READ UPDATE command is issued for a file referencing the same data set before exclusive control is released by a REWRITE, UNLOCK, or DELETE command (RESP2=28).
- A BDAM key conversion error occurred (RESP2=40).

READ

- The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than or equal to zero (RESP2=42).
- The command does not conform to the format of READ for a user-maintained data table; for example if GTEQ is specified with UPDATE (RESP2=44).
- An attempt is made to function-ship an UPDATE request which includes a TOKEN keyword (RESP2=48).

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the READ operation (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error.

For user-maintained data tables, this condition occurs only for a non-UPDATE READ during loading when CICS has attempted to retrieve the record from the source data set.

(Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- Neither the LENGTH option nor the SET option is specified on a READ command for a file with variable-length records or for a BDAM file with variable-length or undefined-format records (RESP2=10).
- The length of a record read with the INTO option specified exceeds the value specified in the LENGTH option; the record is truncated, and the data area supplied in the LENGTH option is set to the actual length of the record (RESP2=11).
- An incorrect length is specified for a file with fixed-length records (RESP2=13).

Default action: terminate the task abnormally.

LOADING

occurs if a READ UPDATE is issued for a user-maintained data table that is currently being loaded (RESP2=104). A user-maintained data table cannot be modified during loading.

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTFND

occurs if an attempt to retrieve a record based on the search argument provided is unsuccessful (RESP2=80).

For user-maintained data tables, this condition occurs if an attempt to read a record is unsuccessful because there is no entry with the specified key in the data table (RESP2=80). This does not mean that there is no such record in the source data set; it may be that such a record is present but was either rejected during initial loading by the user exit XDTRD, or was subsequently deleted from the user-maintained data table.

Default action: terminate the task abnormally.

NOTOPEN

occurs in any of the following situations (RESP2=60):

- The requested file is CLOSED and UNENABLED. The CLOSED, UNENABLED state is reached after a CLOSE request has been received against an OPEN ENABLED file and the file is no longer in use. This state can also be specified as the initial state by means of the FILSTAT parameter of the file control table TYPE=FILE, or by defining a file using the RDO options STATUS=UNENABLED and OPENTIME=FIRSTREF.
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.

This condition does not occur if the request is made to either a CLOSED, ENABLED file or to a CLOSED, DISABLED file. In the first case, the file is opened as part of executing the request. In the second case, the DISABLED condition occurs.

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed (RESP2=130).

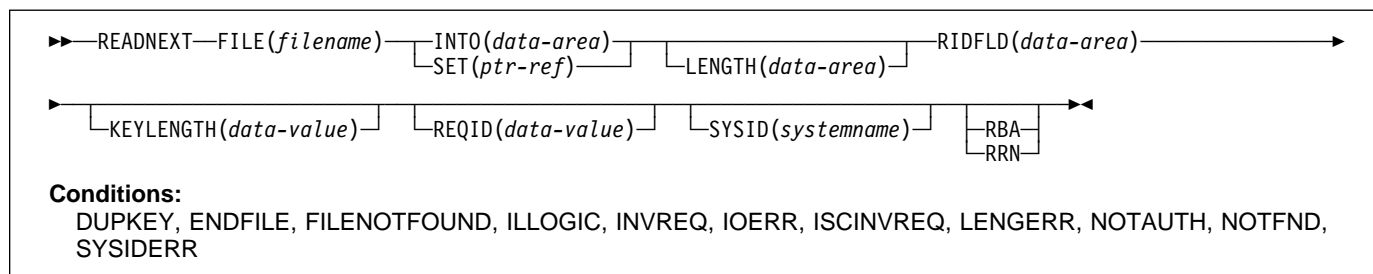
Default action: terminate the task abnormally.

READNEXT

Function

Read next record during a browse of a file.

Command syntax



READNEXT can be used repeatedly to read records in sequential order from a file on a local or a remote system. Such a series of sequential read commands is known as a **browse** of the file. A browse can also consist of a sequence of READNEXT and READPREV commands in any order. A browse must be initiated with the STARTBR command, to identify the starting point of the browse, and terminated with the ENDBR command.

You must provide, in the RIDFLD option, a data area that is sufficiently large to contain a complete identifier (full key, RBA, or RRN) of records in the file. This data area can be used both as an output and as an input parameter.

It is used as an output parameter when CICS, on completion of each READNEXT command, places the complete identifier of the record just retrieved into the RIDFLD data area. CICS then holds this identifier to mark the point from which the subsequent READNEXT is to continue.

Except for BDAM, it may also be used as an input parameter. Modifying the RIDFLD before issuing the next READNEXT command causes that command to reposition the browse to the new identifier, from which it continues in the usual way. If the browse was started with the GENERIC option, the modified RIDFLD must be generic. If the browse was started with the GTEQ option, the next record returned is the first record in the data set with a key greater than or equal to the modified RIDFLD.

| A READNEXT command following a READPREV, or a
 | STARTBR or RESETBR that specified a 'lost' key value, is
 | treated as though the RIDFLD value has been modified, and
 | results in a reposition (as above).

READNEXT options

FILE(filename)

specifies the of the file to be browsed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

INTO(data-area)

specifies the data area into which the record retrieved from the data set is to be written.

KEYLENGTH(data-value)

specifies the length (halfword binary) of the key that has been specified in the RIDFLD option, except when RBA or RRN is specified, in which case KEYLENGTH is not valid.

If the browse was started without the GENERIC option (that is a full key browse) and if the length specified is different from the length defined for the data set, the INVREQ condition occurs.

If the browse was started with the GENERIC option (that is, a generic key browse), and if the length specified is greater than the length specified for the data set, the INVREQ condition occurs.

If GTEQ and GENERIC were specified on the most recent STARTBR or RESETBR command, issuing READNEXT with KEYLENGTH(0) specified repositions the BROWSE at the start of the file. If EQUAL had been specified, the effect of READNEXT KEYLENGTH(0) would be unpredictable.

For a generic browse, CICS maintains a current key length for the browse. The current key length is initialized to be the value specified as KEYLENGTH on the STARTBR command.

The current key length may be modified by specifying KEYLENGTH on a READNEXT or RESETBR command.

READNEXT

If the current key length is changed, this causes the browse to be repositioned. The browse is repositioned to the key whose initial characters match the value specified in the RIDFLD for the current key length.

The current key length is zero after a request that specifies KEYLENGTH(0).

IF KEYLENGTH is omitted on a READNEXT command, the current key length remains the same and the browse continues without repositioning.

If KEYLENGTH is specified on a READNEXT command and is equal to the current key length, this is treated as being no change and the browse is not repositioned. The one exception to this is when KEYLENGTH(0) is specified, which always causes the browse to be repositioned to the beginning of the file.

KEYLENGTH can be specified during a generic browse with a value equal to the full key length. This does not cause the current key length to change and the browse is not repositioned. The ability to specify the full key length during a generic browse is provided to allow requests that specify SYSID to be able to tell the function-shipping transformers how long the key is, so that the transformers can ship the key to the file-owning region.

A browse can be repositioned by modifying the RIDFLD data area. A generic browse is repositioned only if the modification to RIDFLD changes the part of RIDFLD corresponding to the current key length. A consequence of this is that the browse can not be repositioned by modifying the RIDFLD data area if the current key length is zero.

LENGTH(data-area)

specifies the length, as a halfword binary value, of the record to be retrieved. On completion of a retrieval operation, the LENGTH parameter contains the actual length of the retrieved record.

This option must be specified if SYSID is specified. It must also be specified with the INTO option on commands involving variable-length records. It need not be specified for fixed-length records, but its inclusion is recommended because:

- It causes a check to be made that the record being read is not too long for the available data area.
- When browsing fixed-length records into an area longer than the record being accessed, the LENGERR condition is raised for assembler-language, VS COBOL II, and PL/I applications if the LENGTH option is not specified. If the length specified exceeds the file record length, CICS uses the longer length for the move. If the target area in the application program is not large enough, storage is overlaid beyond the target area.

When browsing into a target data area longer than the record being read, the contents of the target data area,

from the end of the retrieved record to the end of the target data area, are unpredictable.

If you specify the INTO option, the LENGTH argument must be a data area that specifies the largest record the program accepts. If the retrieved record is longer than the value specified in the LENGTH option, the record is truncated to the specified value and the LENGERR condition occurs. In this case, the LENGTH data area is set to the length of the record prior to truncation.

If you specify the SET option, the LENGTH option need not be specified but, if it is, the argument must be a data area.

Note that a file control command issued against a variable-length record in a file defined on the local CICS system fails with a LENGERR condition if a length is not specified. However, if the same command is issued against a file defined on a remote system, the command does not fail. The transformer which passes CICS commands from one CICS region to another has to pass a default value for the length argument if a value is not specified. This is because, if the file is remote, the transformer cannot identify the record as fixed or variable, and it is valid not to pass a length when issuing a request against a fixed-length record.

RBA (VSAM ESDS, KSDS, or CMT)

(base data sets only, not paths)
specifies that the record identification field specified in the RIDFLD option contains a relative byte address.

This option must be specified when the STARTBR or RESETBR command specified the RBA option.

| You cannot use RBA for user-maintained data tables.

REQID(data-value)

specifies as a halfword binary value a unique request identifier for a browse, used to control multiple browse operations on a data set. If this option is not specified, a default value of zero is assumed.

RIDFLD(data-area)

specifies the record identification field. When combined with RBA or RRN this is a 4-character field. The contents can be a key, a relative byte address, or relative record number (for VSAM data sets), or a block reference, physical key, and deblocking argument (for BDAM data sets). For a relative byte address or a relative record number, the format of this field must be fullword binary. For a relative byte address, the RIDFLD can be greater than or equal to zero. For a relative record number, the RIDFLD can be greater than or equal to 1.

Even if the browse is generic, this field should always be large enough to accommodate the complete record identifier. This is because, on completion of the READNEXT command, the field is updated by CICS with the complete identification of the record retrieved.

RRN (VSAM RRDS)

specifies that the record identification field specified in the RIDFLD option contains a relative record number.

SET(ptr-ref)

specifies the pointer reference that is to be set to the address of the retrieved record.

In assembler language, if the DUPKEY condition occurs, the register specified will not have been set, but can be loaded from DFHEITP1.

The pointer reference is valid until the next READNEXT or READPREV command specifying SET for the same file. The pointer is no longer valid after an ENDBR or SYNCPOINT command. If you want to retain the data within the field addressed by the pointer, you must move it to your own area.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, the address returned in the SET pointer is below the 16MB line.

If TASKDATAKEY(USER) is specified for the running task, the data returned is in a user-key; otherwise it is in a CICS-key.

SYSID(systemname)

specifies the name of the system to which the request is directed.

If you specify SYSID, and omit both RBA and RRN, you must also specify LENGTH and KEYLENGTH; they cannot be found in the FCT.

READNEXT conditions

Note: RESP2 values are not set for files that are on remote systems.

DUPKEY (VSAM)

occurs if a record is retrieved by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key follows (RESP2=140). It does not occur as a result of a READNEXT command that reads the last of the records having the nonunique key.

In assembler language, if the SET option is being used, the register specified will not have been set, but can be loaded from DFHEITP1.

Default action: terminate the task abnormally.

ENDFILE

occurs if an end-of-file condition is detected during the browse (RESP2=90).

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if a file name referred to in the FILE option cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- The KEYLENGTH option is specified for a generic browse (that is one where GENERIC was specified on the STARTBR or the last RESETBR) and the value of KEYLENGTH was greater than the full key length (RESP2=25).
- The KEYLENGTH option is specified for a nongeneric browse, and the specified length does not equal the length defined for the data set to which this file refers (RESP2=26).
- The REQID, if any, does not match that of any successful STARTBR command (RESP2=34).
- The type of record identification (for example, key or relative byte address) used to access a data set during the browse is changed by the READNEXT command (RESP2=37).
- The KEYLENGTH option is specified for a generic browse (that is one where GENERIC was specified on the STARTBR or the last RESETBR) and the value of KEYLENGTH was less than zero (RESP2=42).

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the READNEXT command (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error.

(Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

READNEXT

LENGERR

occurs in any of the following situations:

- Neither the LENGTH nor the SET option is specified for a file with variable-length records or a BDAM file with undefined-format records (RESP2=10).
- The length of the record read with the INTO option specified exceeds the value specified in the LENGTH option; the record is truncated, and the data area supplied in the LENGTH option is set to the actual length of the record (RESP2=11).
- An incorrect length is specified for a file with fixed-length records (RESP2=13).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTFND

occurs if an attempt to retrieve a record based on the search argument provided is unsuccessful (RESP2=80). This may occur if the READNEXT command immediately follows a STARTBR command that specified the key of the last record in the data set (a complete key of X'FF's).

Default action: terminate the task abnormally.

SYSIDERR

| occurs when the SYSID option specifies a name that is
| neither the local system nor a remote system (made
| known to CICS by defining a CONNECTION).
SYSIDERR also occurs when the link to the remote
system is closed (RESP2=130).

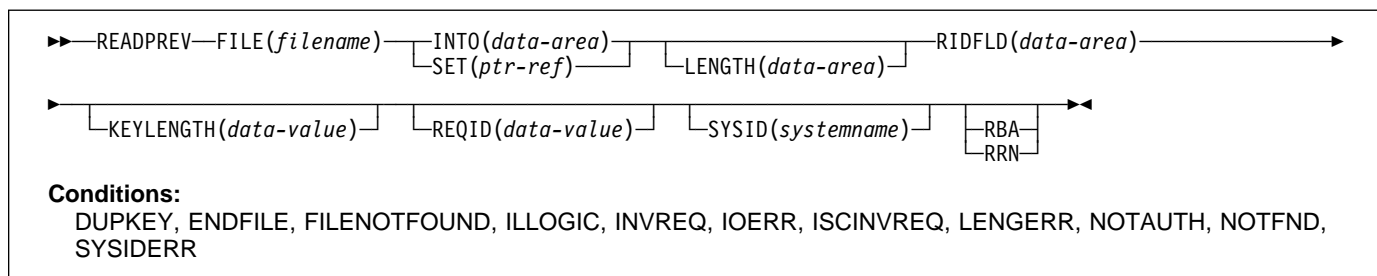
Default action: terminate the task abnormally.

READPREV

Function

Read previous record during a file browse; VSAM and data tables only.

Command syntax



READPREV can be used repeatedly to read records in reverse sequential order from a VSAM file on a local or a remote system.

Such a series of sequential read commands is known as a **browse** of the file. A browse may also consist of a sequence of READNEXT and READPREV commands in any order. A browse must be initiated with the STARTBR command, to identify the start of the browse, and terminated with the ENDBR command.

You must provide, in the RIDFLD option, a data area that is sufficiently large to contain a complete identifier (full key, RBA, or RRN) of records in the file. This data area is used both as an output and as an input parameter.

It is used as an output parameter when CICS, on completion of each READPREV command, places the complete identifier of the record just retrieved into the RIDFLD data area. CICS then holds this identifier to mark the point from which the subsequent READPREV is to continue.

It may also be used as an input parameter. Modifying the RIDFLD before issuing the next READPREV command causes that command to reposition the browse to the new identifier, from which it continues in the usual way. The modified record identifier must always be a full key, RBA, or RRN. A generic key may not be specified, nor may a browse that was started with the GENERIC option include a READPREV command.

If you include a READPREV command immediately following a STARTBR command, your STARTBR command RIDFLD must specify the key of a record that exists on the data set; otherwise the NOTFND condition will occur.

A READPREV command following a READNEXT, or a STARTBR or RESETBR that did not specify a 'lost' key value, is treated as though the RIDFLD value had been modified and results in a reposition (as above).

READPREV options

FILE(filename)

specifies the of the file being browsed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

INTO(data-area)

specifies the data area into which the record retrieved from the data set is to be written.

KEYLENGTH(data-value)

specifies the length (halfword binary) of the key that has been specified in the RIDFLD option, except when RBA or RRN is specified, in which case KEYLENGTH is not valid. If the length specified is different from the length defined for the data set, the INVREQ condition occurs.

LENGTH(data-area)

specifies the length, as a halfword binary value, of the record to be retrieved. On completion of a retrieval operation, LENGTH contains the actual length of the retrieved record.

This option must be specified if SYSID is specified. It must also be specified with the INTO option on READPREV commands involving variable-length records. It need not be specified for fixed-length records, but its inclusion is recommended because:

- It causes a check to be made that the record being read is not too long for the available data area.
- When browsing fixed-length records into an area longer than the record being accessed, the LENGERR condition is raised for VS COBOL II, C, PL/I, and assembler-language applications if the LENGTH option is not specified.

READPREV

- When reading fixed-length records into an area longer than the record being accessed, the LENGERR condition is raised for VS COBOL II, PL/I, and assembler-language applications if the LENGTH option is not specified. If the length specified exceeds the file record length, CICS uses the longer length for the move. If the target area in the application program is not large enough, storage is overlaid beyond the target area.

When browsing into a target data area longer than the record being read, the contents of the target data area, from the end of the retrieved record to the end of the target data area, are unpredictable.

If you specify the INTO option, LENGTH specifies the largest record the program accepts. If the retrieved record is longer than the value specified in the LENGTH option, the record is truncated to the specified value and the LENGERR condition occurs. In this case, the LENGTH data area is set to the length of the record prior to truncation.

If you specify the SET option, LENGTH need not be specified.

RBA (VSAM ESDS, KSDS, or CMT)

(base data sets only, not paths)
specifies that the record identification field specified in the RIDFLD option contains a relative byte address. You must specify this option after any STARTBR or RESETBR command that specified the RBA option.

| You cannot use RBA for user-maintained data tables.

REQID(data-value)

specifies as a halfword binary value a unique request identifier for a browse, used to control multiple browse operations on a data set. If this option is not specified, a default value of zero is assumed.

RIDFLD(data-area)

specifies the record identification field. When combined with RBA or RRN this is a 4-character field. The contents can be a key, a relative byte address, or relative record number. For a relative byte address or a relative record number, the format of this field must be fullword binary. For a relative byte address, the RIDFLD can be greater than or equal to zero. For a relative record number, the RIDFLD can be greater than or equal to 1.

On completion of the READPREV command, this field is updated by CICS with the complete identification of the record retrieved.

RRN (VSAM RRDS)

specifies that the record identification field specified in the RIDFLD option contains a relative record number.

SET(ptr-ref)

specifies the pointer reference that is to be set to the address of the retrieved record.

In assembler language, if the DUPKEY condition occurs, the specified register has not been set, but can be loaded from DFHEITP1.

The pointer reference is valid until the next READ command for the same file. If you want to retain the data within the field addressed by the pointer, it should be moved to your own area.

If DATALOCATION(ANY) is associated with the application program, the address returned in the SET pointer can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the address returned in the SET pointer is below the 16MB line.

If TASKDATAKEY(USER) is specified for the running task, the data returned is in a user-key; otherwise it is in a CICS-key.

SYSID(systemname)

specifies the name of the system to which the request is directed.

If you specify SYSID, and omit both RBA and RRN, you must also specify LENGTH and KEYLENGTH.

READPREV conditions

Note: RESP2 values are not set for files that are on remote systems.

DUPKEY

occurs if a record is retrieved by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key exists (RESP2=140).

In assembler language, if the SET option is being used, the specified register has not been set, but can be loaded from DFHEITP1.

Default action: terminate the task abnormally.

ENDFILE

occurs if an end-of-file condition is detected during a browse (RESP2=90).

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if a file name referred to in the FILE option cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- A READPREV command is issued for a file for which the previous STARTBR or RESETBR command has the GENERIC option (RESP2=24).
- The KEYLENGTH option is specified and the specified length does not equal the length defined for the data set for this file refers to (RESP2=26).
- The type of record identification (for example, key or relative byte address) used to access a data set during the browse is changed (RESP2=37).
- A READPREV is issued for a BDAM file (RESP2=39).
- The REQID, if any, does not match that of any successful STARTBR command (RESP2=41).

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the browse (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error. (Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- Neither the LENGTH nor the SET option is specified for a file with variable-length records (RESP2=10).
- The length of the record read with the INTO option specified exceeds the value specified in the LENGTH option; the record is truncated, and the data area supplied in the LENGTH option is set to the actual length of the record (RESP2=11).
- An incorrect length is specified for a file with fixed-length records (RESP2=13).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTFND

occurs if an attempt to retrieve a record based on the search argument provided is unsuccessful (RESP2=80). One place where this may occur is where the READPREV command immediately follows a STARTBR

or RESETBR command that specified GTEQ and the key of a record that does not exist on the data set.

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that has not been defined to CICS as a remote system (defined by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed (RESP2=130).

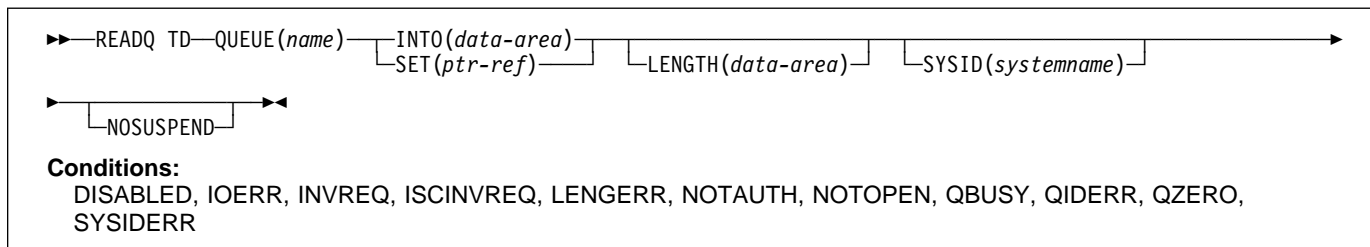
Default action: terminate the task abnormally.

READQ TD

Function

Read data from the transient data queue.

Command syntax



READQ TD reads transient data from a queue (after which the record is no longer available).

If you are using automatic transaction initiation (ATI) (see the section on ATI in the *CICS/ESA Application Programming # Guide* for introductory information) your application should # test for the QZERO condition to ensure that termination of an automatically initiated task occurs only when the queue is empty.

The following example shows how to read a record from an intrapartition data set (queue), which in this case is the control system message log (CSML), into a data area specified in the request:

```
EXEC CICS READQ TD
      QUEUE('CSML')
      INTO(DATA)
      LENGTH(LDATA)
```

The following example shows how to read a record from an extrapartition data set (queue) having fixed-length records into a data area provided by CICS; the pointer reference specified by the SET option is set to the address of the storage area reserved for the data record. It is assumed that the record length is known.

```
EXEC CICS READQ TD
      QUEUE(EX1)
      SET(PREF)
```

If the READQ TD command attempts to access a record in a logically recoverable intrapartition queue that is being written to, or deleted by, another task, and there are no more committed records, the command waits until the queue is no longer being used for output. If, however, the NOSUSPEND option has been specified, the QBUSY condition is raised.

READQ TD options

INTO(data-area)

specifies the user data area into which the data read from the transient data queue is to be placed.

LENGTH(data-area)

specifies the length, as a halfword binary value, of the record to be read.

If you specify the INTO option, LENGTH specifies the maximum length of data that the program accepts. If the value specified is less than zero, zero is assumed. If the length of the data exceeds the value specified, the data is truncated to that value and the LENGERR condition occurs. On completion of the retrieval operation, the data area is set to the original length of the data.

NOSUSPEND

specifies that if the application program attempts to read from a queue that is already being used for output, the task is not suspended until the queue becomes available. Instead, the QBUSY condition is raised.

This option only applies to intrapartition queues.

QUEUE(name)

specifies the symbolic name (1–4 alphanumeric characters) of the queue to be read from. The name must have been defined in the DCT.

If SYSID is specified, the queue is assumed to be on a remote system irrespective of whether or not the name is defined as remote in the DCT. Otherwise the entry in the DCT is used to find out whether the data set is on a local or a remote system.

SET(ptr-ref)

specifies a pointer reference that is to be set to the address of the data read from the queue. CICS acquires an area large enough to hold the record and sets the pointer reference to the address of that area. The area is retained until another transient data command is executed. The pointer reference, unless

changed by other commands or statements, is valid until the next READQ TD command or the end of task.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

SYSID(systemname) — remote systems only

specifies the name (1–4 characters) of the system to which the request is directed.

READQ TD conditions

DISABLED

occurs when the queue has been disabled.

Default action: terminate the task abnormally.

INVREQ

occurs if READQ names an extrapartition queue that has been opened for output. This condition cannot occur for intrapartition queues.

Default action: terminate the task abnormally.

IOERR

occurs when an input/output error occurs and the data record in error is skipped.

This condition occurs as long as the queue can be read; a QZERO condition occurs when the queue cannot be read.

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LENGERR

occurs if READQ names an INTO area that cannot hold all the data that is to be returned to the application. The check is made after the XTDIN exit has been invoked.

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on QUEUE(name).

Default action: terminate the task abnormally.

NOTOPEN

occurs if the destination is closed. This condition applies to extrapartition queues only.

Default action: terminate the task abnormally.

QBUSY

occurs if a READQ TD command attempts to access a record in a logically recoverable intrapartition queue that is being written to or is being deleted by another task, and there are no more committed records.

The NOSUSPEND option must be specified for this condition to be raised.

This condition applies only to intrapartition queues.

Default action: ignore the condition.

QIDERR

occurs if the symbolic destination to be used with READQ TD cannot be found.

Default action: terminate the task abnormally.

QZERO

occurs when the destination (queue) is empty or the end of the queue has been reached.

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed.

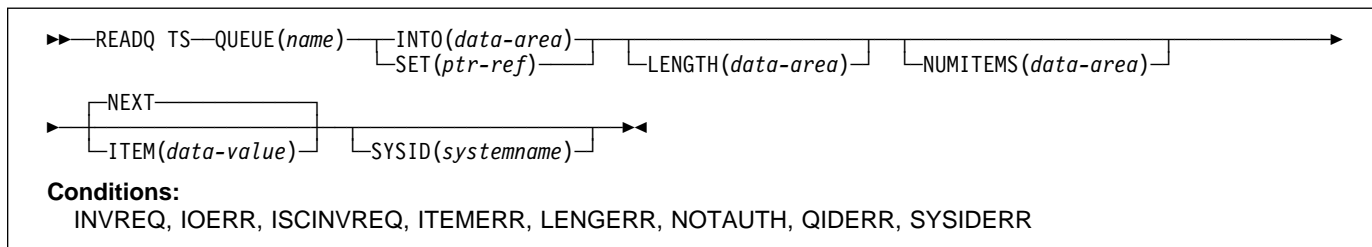
Default action: terminate the task abnormally.

READQ TS

Function

Read data from temporary storage queue.

Command syntax



Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

READQ TS retrieves data from a temporary storage queue in main or auxiliary storage.

The following example shows how to read the first (or only) record from a temporary storage queue into a data area specified in the request; the LENGTH data area is given the value of the length of the record.

```
EXEC CICS READQ TS
      ITEM(1)
      QUEUE(UNIQNAME)
      INTO(DATA)
      LENGTH(LDATA)
```

The following example shows how to read the next record from a temporary storage queue into a data area provided by CICS; the pointer reference specified by the SET option is set to the address of the storage area reserved for the data record, and the LENGTH data area is given the value of the length of the record.

```
EXEC CICS READQ TS
      QUEUE(DESCRQ)
      SET(PREF)
      LENGTH(LENG)
      NEXT
```

READQ TS options

INTO(data-area)

specifies the data area into which the data is to be written. The data area can be any variable, array, or structure.

ITEM(data-value)

provides a halfword binary value that specifies the item number of the logical record to be retrieved from the queue.

LENGTH(data-area)

specifies the length, as a halfword binary value, of the record to be read.

If you specify the INTO option,

#

Apar 75415

Documentation for Apar 75415 added 19 Sep 1995 (TUCKER)

#

#

#

#

#

LENGTH need not be specified if the length can be generated by the compiler from the INTO variable. If you specify both INTO and LENGTH, LENGTH specifies the maximum length of data that the program accepts. If the value specified is less than zero, zero is assumed. If the length of the data exceeds the value specified, the data is truncated to that value and the LENGERR condition occurs.

On completion of the retrieval operation, the data area is set to the original length of the data record read from the queue.

If you specify the SET option,

#

Apar 75415

Documentation for Apar 75415 added 19 Sep 1995 (TUCKER)

#

#

#

the LENGTH must be specified.

NEXT

specifies retrieval for the next sequential logical record following the last record retrieved (by any task), or the first record if no previous record has been retrieved.

NUMITEMS(data-area)

specifies a halfword binary field into which CICS stores a number indicating how many items there are in the queue. This only occurs if the command completes normally.

QUEUE(name)

specifies the symbolic name (1–8 characters) of the queue to be read from.

SET(ptr-ref)

specifies the pointer reference that is set to the address of the retrieved data. The pointer reference, unless changed by other commands or statements, is valid until the next READQ TS command or the end of task.

If the application program is defined with DATALOCATION(ANY), the address of the data can be above or below the 16MB line. If the application program is defined with DATALOCATION(BELOW), the address of the data is below the 16MB line.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

SYSID(systemname) — remote systems only

specifies the name (1–4 characters) of the system to which the request is directed.

This condition only applies to the INTO option and cannot occur with SET.

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on QUEUE(name).

Default action: terminate the task abnormally.

QIDERR

occurs when the queue specified cannot be found, either in main or in auxiliary storage.

Default action: terminate the task abnormally.

READQ TS conditions**INVREQ**

occurs when the queue was created by CICS internal code.

Default action: terminate the task abnormally.

IOERR

occurs when there is an irrecoverable input/output error.

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

ITEMERR

occurs in any of the following situations:

- The item number specified is invalid (that is, outside the range of item numbers written to the queue).
- An attempt is made to read beyond the end of the queue using the NEXT (default) option.

Default action: terminate the task abnormally.

LENGERR

occurs when the length of the stored data is greater than the value specified by the LENGTH option.

READQ TS

SYSIDERR

| occurs when the SYSID option specifies a name that is
| neither the local system nor a remote system (made
| known to CICS by defining a CONNECTION).
SYSIDERR also occurs when the link to the remote
system is closed.

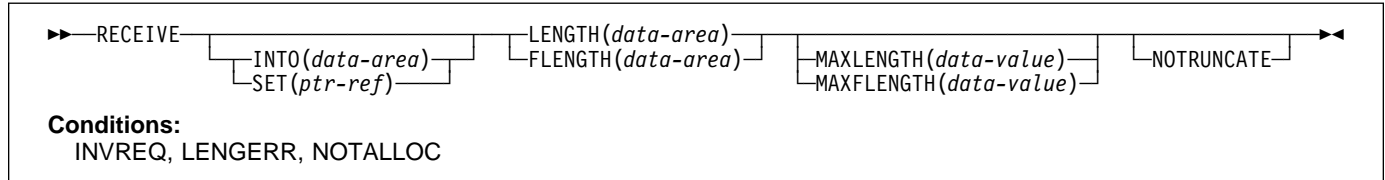
Default action: terminate the task abnormally.

RECEIVE (VTAM default)

Function

Receive data from standard CICS terminal support (BTAM or TCAM) or from a task that is not attached to a terminal.

Command syntax



This form of the RECEIVE command is used by all CICS-supported terminals for which the other RECEIVE descriptions are not appropriate.

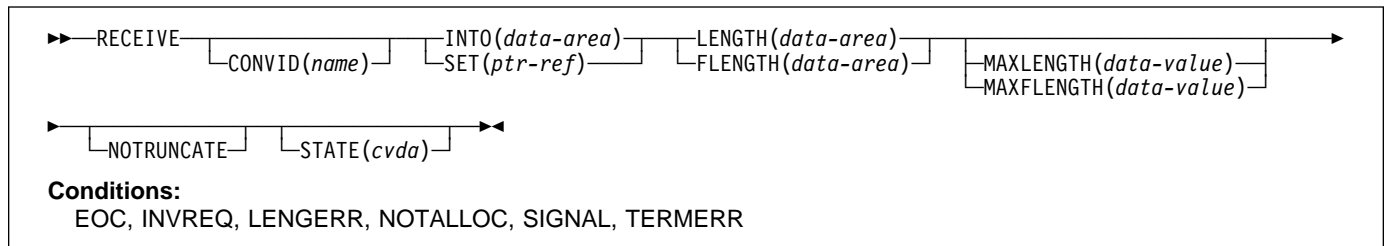
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (APPC)

Function

Receive data on an APPC mapped conversation.

Command syntax



RECEIVE receives data from the conversation partner in an APPC mapped conversation.

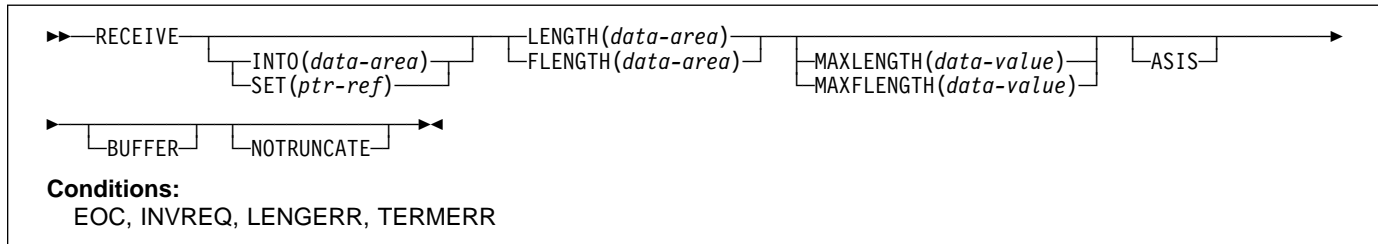
RECEIVE (VTAM)

RECEIVE (LUTYPE2/LUTYPE3)

Function

Receive data from a 3270-display logical unit (LUTYPE2) or a 3270-printer logical unit (LUTYPE3).

Command syntax



RECEIVE receives data from the terminal.

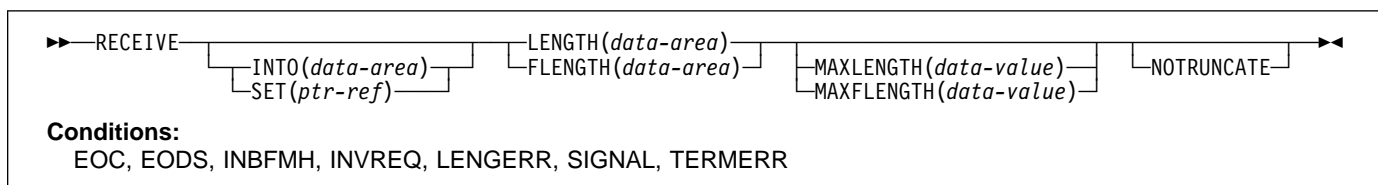
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) (and BUFFER has not been specified), you can omit both the INTO and SET options.

RECEIVE (LUTYPE4)

Function

Receive data from an LUTYPE4 logical unit.

Command syntax



RECEIVE receives data from the terminal.

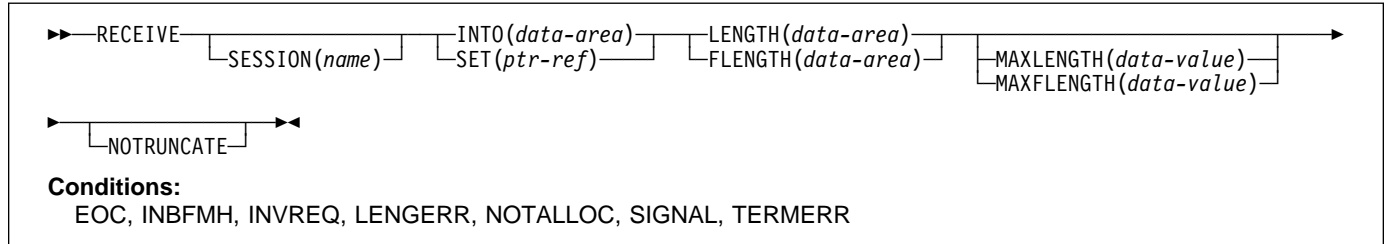
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (LUTYPE6.1)

Function

Receive data on an LUTYPE6.1 session.

Command syntax



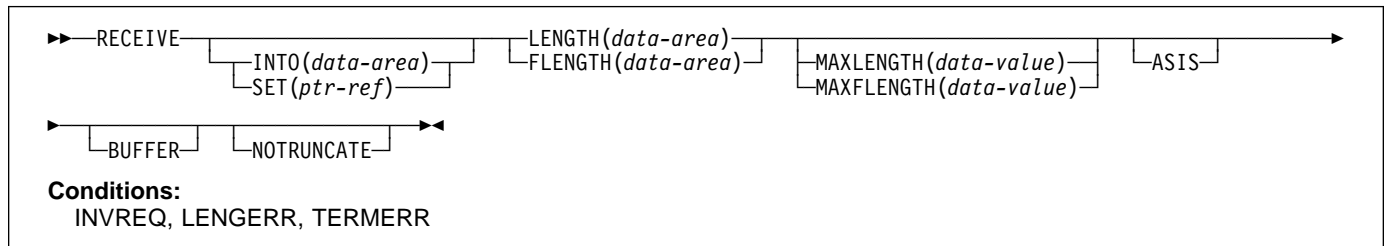
RECEIVE receives data from the conversation partner in an LUTYPE6.1 conversation.

RECEIVE (3270 logical)

Function

Receive data from a 3270 logical unit.

Command syntax



RECEIVE receives data from a terminal.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) (and BUFFER has not been specified), you can omit both the INTO and SET options.

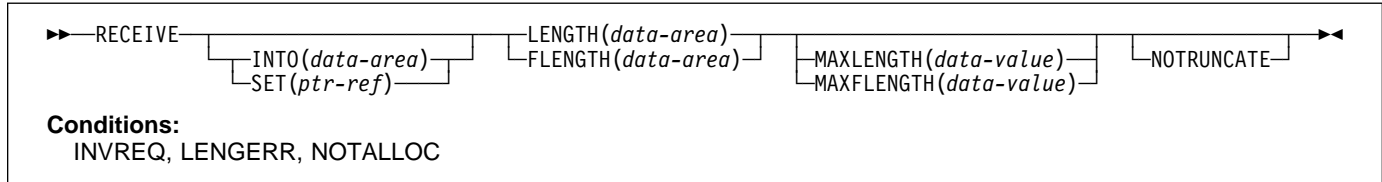
RECEIVE (VTAM)

RECEIVE (3600 pipeline)

Function

Receive initial input data from a 3600 pipeline logical unit. Subsequent RECEIVES for further input data are not allowed.

Command syntax



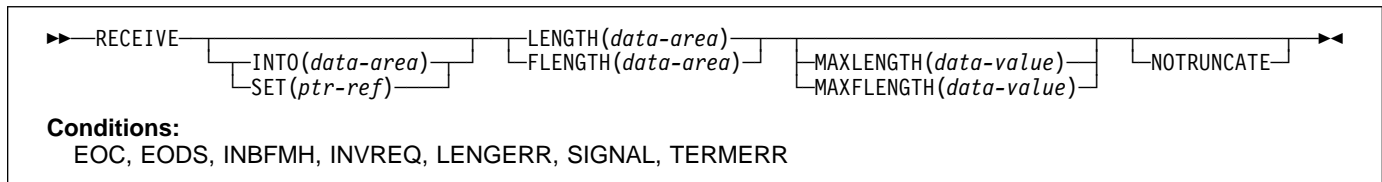
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (3600-3601)

Function

Receive data from a 3600 (3601) logical unit

Command syntax



RECEIVE receives data from the terminal. This form of RECEIVE also applies to the 3630 plant communication system.

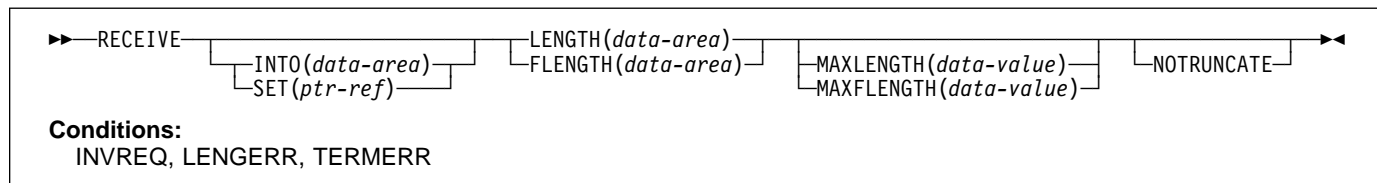
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (3600-3614)

Function

Receive data from a 3600 (3614) logical unit.

Command syntax



RECEIVE receives data from the terminal.

The data-stream and communication format used between a CICS application program and a 3614 is determined by the 3614. The application program is therefore device dependent when handling 3614 communication.

For further information about designing 3614 application programs for CICS, refer to the *CICS/OS/VS IBM 4700/3600/3630 Guide*.

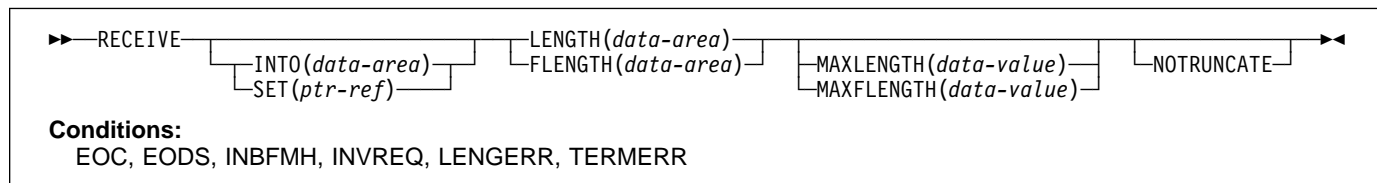
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (3650)

Function

Receive data from 3650 logical units.

Command syntax



RECEIVE receives data from the terminal. This form of RECEIVE also applies to the following 3650 devices:

- Interpreter logical unit
- Host conversational (3270) logical unit
- Host conversational (3653) logical unit
- 3650/3680 command processor logical unit.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

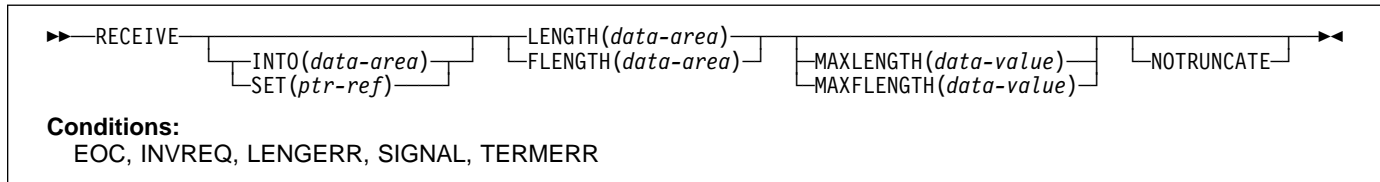
RECEIVE (VTAM)

RECEIVE (3767)

Function

Receive data from a 3767 interactive logical unit.

Command syntax



RECEIVE receives data from the terminal. This form of RECEIVE also applies to the 3770 interactive logical unit.

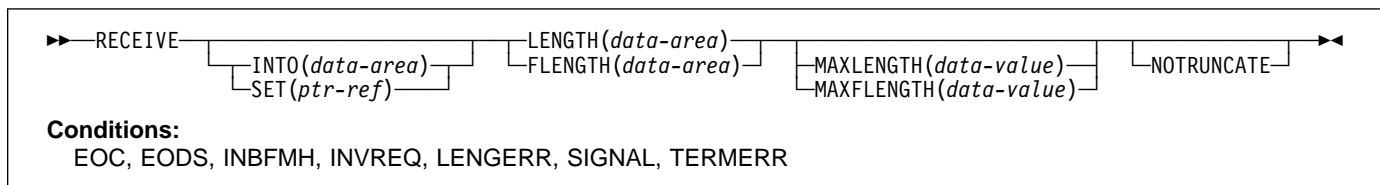
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (3770)

Function

Receive data from a 3770 batch logical unit.

Command syntax



RECEIVE receives data from the terminal.

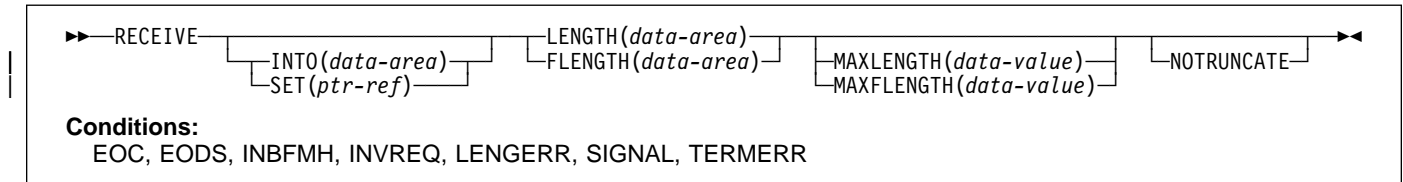
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (3790 full-function or inquiry)

Function

Receive data from a 3790 full-function or inquiry logical unit.

Command syntax



RECEIVE receives data from the terminal. This form of RECEIVE also applies to the following devices:

- 3650/3680 full-function logical unit
- 3770 full-function logical unit.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (VTAM)

RECEIVE (VTAM) options

ASIS

specifies that lowercase characters in the 3270 input data stream are not translated to uppercase; this allows the current task to receive a message containing both uppercase and lowercase data.

This option has no effect on the first RECEIVE command of a transaction, because terminal control performs a READ INITIAL and uses the terminal defaults to translate the operation data.

This option has no effect if the screen contains data prior to a transaction being initiated. This data is read and translated in preparation for the next task and the first RECEIVE command in that task retrieves the translated data.

Note: If you are using a katakana terminal, you might see some messages containing mixed English and katakana characters. That is because katakana terminals cannot display mixed-case output. Uppercase characters in the data stream appear as uppercase English characters, but lowercase characters appear as katakana characters. If this happens, ask your system programmer to specify MSGCASE=UPPER in the system initialization parameters so that messages contain uppercase characters only.

BUFFER

specifies that the contents of the 3270 buffer are to be read, beginning at buffer location 1 and continuing until all contents of the buffer have been read. All character and attribute sequences (including nulls) appear in the input data stream in the same order as they appear in the 3270 buffer.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If this option is omitted, the principal facility is assumed.

FLENGTH(data-area)

A fullword alternative to LENGTH.

INTO(data-area)

specifies the receiving field for the data read from the logical unit or terminal, or the application target data area into which data is to be received from the application program connected to the other end of the current conversation.

LENGTH(data-area)

specifies the length, as a halfword binary value, of the data received.

If you specify the INTO option, but omit the MAXLENGTH option, the argument must be a data area that specifies the maximum length that the program accepts. If the value specified is less than zero, zero is assumed.

If you specify the SET option, the argument must be a data area. When the data has been received, the data area is set to the length of the data.

MAXLENGTH(data-value)

A fullword alternative to MAXLENGTH.

MAXLENGTH(data-value)

specifies the maximum amount (halfword binary value) of data that CICS is to recover. If INTO is specified, MAXLENGTH overrides the use of LENGTH as an input to CICS. If SET is specified, MAXLENGTH provides a way for the program to limit the amount of data it receives at one time.

If the length of data exceeds the value specified and the NOTRUNCATE option is not present, the data is truncated to that value and the LENGERR condition occurs. The data area specified in the LENGTH option is set to the original length of data.

If the length of data exceeds the value specified and the NOTRUNCATE option is present, CICS retains the remaining data and uses it to satisfy subsequent RECEIVE commands. The data area specified in the LENGTH option is set to the length of data returned.

If this option is omitted, the value indicated in the LENGTH option is assumed.

NOTRUNCATE

specifies that, when the data available exceeds the length requested, the remaining data is not to be discarded but is to be retained for retrieval by subsequent RECEIVE commands.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If this option is omitted, the principal facility for the task is used.

SET(ptr-ref)

specifies the pointer reference that is to be set to the address of the data read from the logical unit or terminal, or the partner transaction. The pointer reference is valid until the next receive command or the end of task.

#

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned

is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

RECEIVE (VTAM) conditions

Some of the following conditions may occur in combination with others. CICS checks for these conditions in the following order:

1. EODS
2. INBFMH
3. EOC.

If more than one occurs, only the first is passed to the application program. EIBRCODE, however, is set to indicate all the conditions that occurred.

EOC

occurs when a request/response unit (RU) is received with end-of-chain-indicator set. Field EIBEOC also indicates this condition.

Default action: ignore the condition.

EODS (interpreter logical unit only)

occurs when an end-of-data-set indicator is received.

Default action: terminate the task abnormally.

INBFMH

occurs if a request/response unit (RU) contains a function management header (FMH). Field EIBFMH contains this indicator and it should be used in preference to INBFMH. The IGNORE CONDITION command can be used to ignore the condition.

Default action: terminate the task abnormally.

INVREQ

occurs if:

- The command is used on an APPC conversation that is not using the EXEC CICS interface or that is not a mapped conversation.

- A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs if data is discarded by CICS because its length exceeds the maximum the program accepts and the NOTRUNCATE option is not specified.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the RECEIVE command is issued by a transaction that has been started as a nonterminal task by the START command, or if the CONVID value or facility specified in the command does not relate to a conversation owned by the application.

Default action: terminate the task abnormally.

SIGNAL

occurs when an inbound SIGNAL data-flow control command is received from a partner transaction. EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

TERMERR

occurs for a session-related or terminal-related error. Any action on that conversation other than a FREE causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

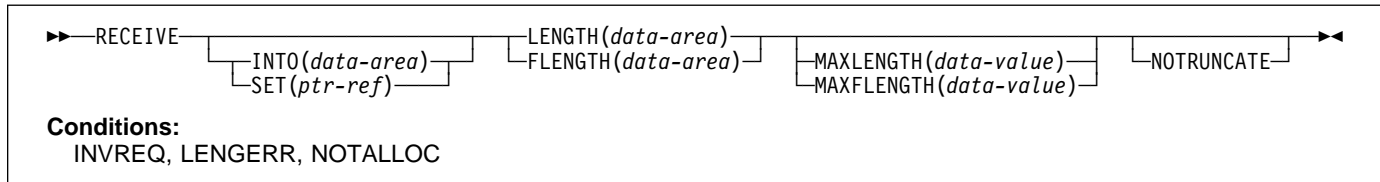
RECEIVE (non-VTAM)

RECEIVE (non-VTAM default)

Function

Receive data from standard CICS terminal support (BTAM or TCAM) or from a task that is not attached to a terminal.

Command syntax



| This form of the RECEIVE command is used by all CICS-supported terminals for which the other RECEIVE descriptions are not appropriate.

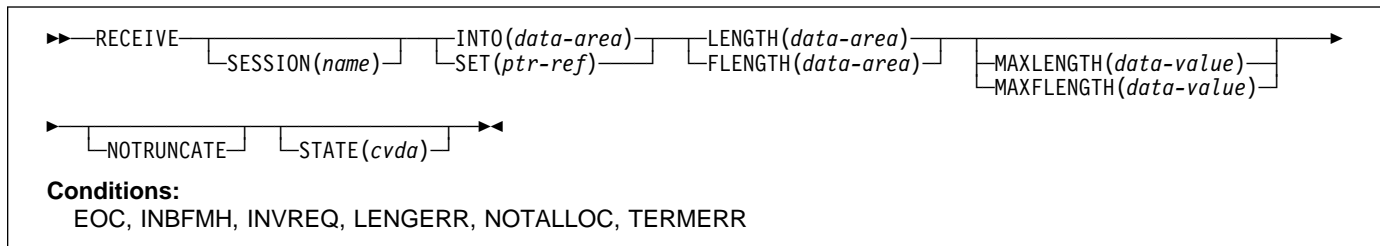
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (MRO)

Function

Receive data on an MRO conversation.

Command syntax



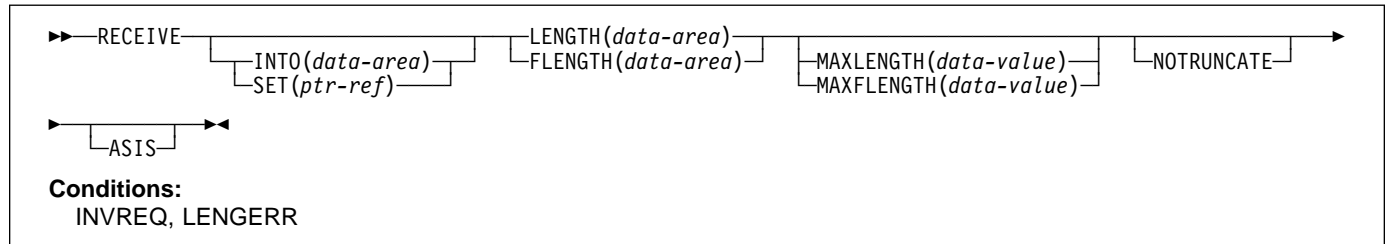
RECEIVE receives data from the conversation partner in an MRO conversation.

RECEIVE (System/3)

Function

Receive data from a System/3.

Command syntax



RECEIVE receives data from the terminal. This form of the RECEIVE command also applies to the following devices:

- 2770 data communication system
- 2780 data transmission terminal
- 3660 supermarket scanning system
- 3780 communication terminal.

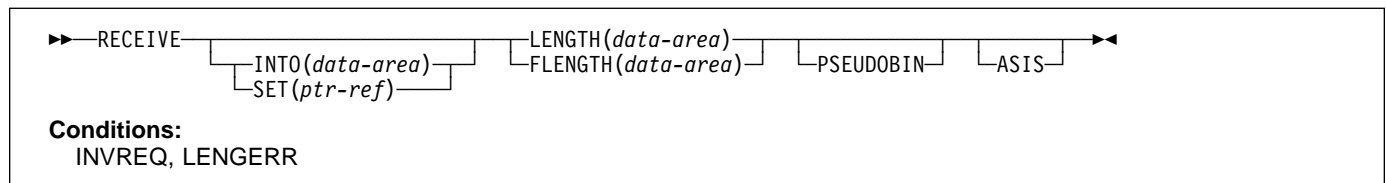
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (System/7)

Function

Receive data from a System/7.

Command syntax



RECEIVE receives data from the terminal. For a description of running with a System/7, see “CONVERSE (System/7)” on page 54.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

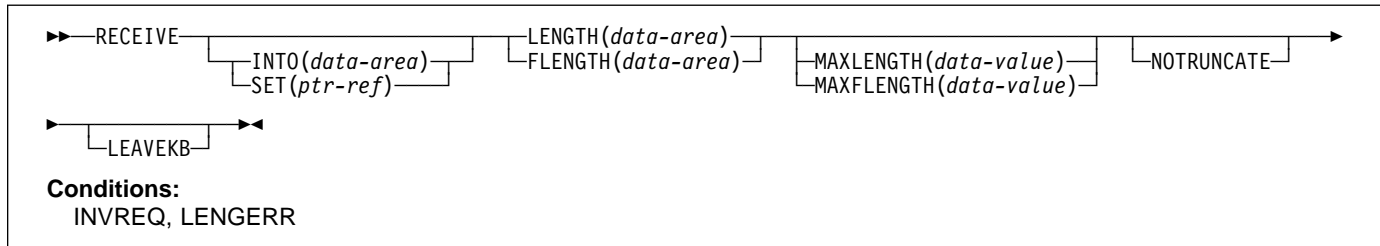
RECEIVE (non-VTAM)

RECEIVE (2260)

Function

Receive data from a 2260 or 2265 display station.

Command syntax



RECEIVE receives data from the terminal.

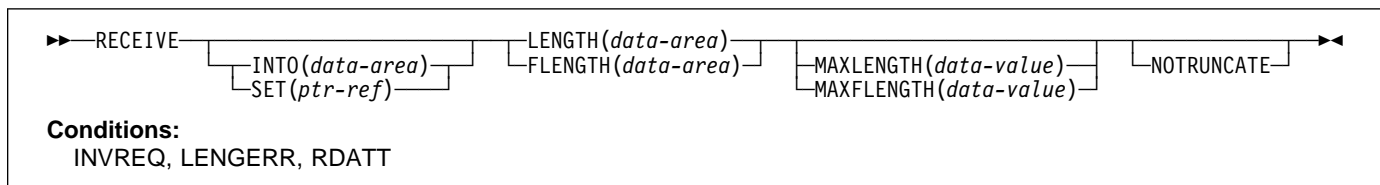
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (2741)

Function

Receive data from a 2741 communication terminal.

Command syntax



RECEIVE receives data from the terminal. For more information about using a 2741 communication terminal, see "CONVERSE (2741)" on page 56.

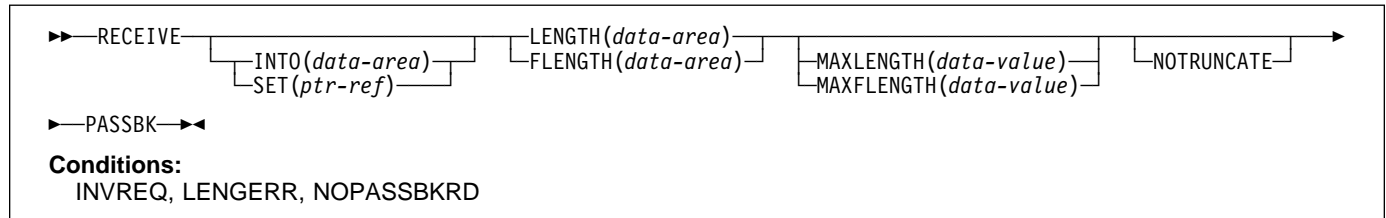
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (2980)

Function

Receive data from a 2980 general banking terminal system.

Command syntax



RECEIVE receives data from the terminal.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

Passbook control

All input and output requests to the passbook area of a 2980 are dependent on the presence of a passbook. The PASSBK option is used to specify that communication is with a passbook. The conditions NOPASSBKRD (RECEIVE) and NOPASSBKWR (SEND) occur on input and output requests respectively when a passbook is not present. These conditions can be handled by a HANDLE CONDITION command and appropriate handling routines.

If the passbook is present on an input request, the application program generally writes back to the passbook area to update the passbook. If the NOPASSBKWR condition occurs, CICS allows immediate output to the terminal. In a routine for the NOPASSBKWR condition, the application program should send an error message to the journal area of the terminal to inform the 2980 operator of this error condition. To allow the operator to insert the required passbook, CICS causes the transaction to wait 23.5 seconds before continuing.

On regaining control from CICS after sending the error message, the application program can attempt again to update the passbook when it has ensured that the print element is positioned correctly in the passbook area. This is generally accomplished by issuing two carrier returns followed by the number of tabs required to move the print element to the correct position.

If the NOPASSBKWR condition occurs during the second attempt to write to the passbook area, the application program can send another error message or take some alternative action (for example, place the terminal "out of service"). The presence of the Auditor Key on a 2980 Administrative Station Model 2 is controlled by the SEND PASSBK command and may be used in a manner similar to that described above.

Output control

The unit of transmission for a 2980 is called a **segment**. However, for the passbook and journal areas, CICS allows an application program to send messages that exceed the buffer size. For the passbook area, the maximum length of message is limited to one line of a passbook to avoid spacing ("indexing") past the bottom of the passbook. For the journal area, the maximum length of message is specified in the LENGTH option of the SEND command.

For example, consider a 2972 buffer size of 48 characters and a 2980 Teller Station Model 4 passbook print area of 100 characters per line. The application program can send a message of 100 characters to this area; CICS segments the message to adjust to the buffer size. The application program must insert the passbook indexing character (X'25') as the *last* character written in one output request to the passbook area. This is done to control passbook indexing and thereby achieve positive control of passbook presence.

RECEIVE (non-VTAM)

If a message contains embedded passbook indexing characters, and segmentation is necessary because of the length of the message, the output is terminated if the passbook spaces beyond the bottom of the passbook; the remaining segments are not printed.

Output to a common buffer

The SEND CBUFF command is used to transmit data to a common buffer. The data is translated to the character set of the receiving 2980 model. If more than one 2980 model type is connected to the 2972 control unit, the lengths are truncated if they exceed the buffer size.

The DFH2980 structure

The DFH2980 structure contains constants that may only be used when writing COBOL or PL/I application programs for the 2980. The structure is obtained by copying DFH2980 into the application program.

For COBOL, DFH2980 is copied into the working-storage section; for PL/I, DFH2980 is included using a %INCLUDE statement.

The station identification is given in the field STATIONID, whose value must be determined by the ASSIGN command. To test whether a normal or alternate station is being used, the STATIONID field is compared with values predefined in DFH2980.

The values are:

STATION-

STATION_

where normal stations. (The break symbol is hyphen (-) for COBOL, and underscore (_) for PL/I.)

The teller identification on a 2980 Teller Station Model 4 is given in the 1-byte character field TELLERID. An ASSIGN command must be used to find out the TELLERID value.

Tab characters (X'05') must be included in the application program. The number of tabs required to position the print element to the first position of a passbook area is given in the field NUMTAB. An ASSIGN command must be used to find out the NUMTAB value. The value of NUMTAB is specified by the system programmer and may be unique to each terminal.

Other tab characters are inserted as needed to control formatting.

Any of the DFH2980 values TAB-ZERO through TAB-NINE for COBOL and PL/I, may be compared with NUMTAB to find out the number of tab characters that need to be inserted in an output message to get correct positioning of the print element. The tab character is included in DFH2980 as TABCHAR.

Thirty special characters are defined in DFH2980. Twenty-three of these can be referred to by the name SPECCHAR-# or SPECCHAR_# (for American National Standard COBOL or PL/I) where # is an integer (0 through 22). The seven other characters are defined with names that imply their usage, for example, TABCHAR.

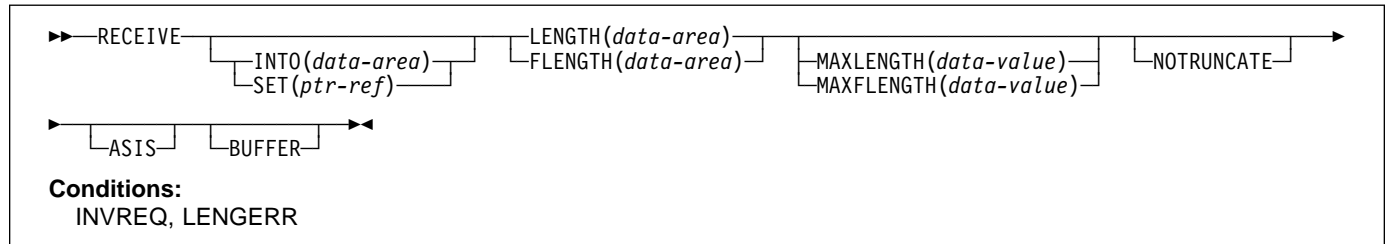
Several other characters defined in DFH2980, such as HOLDPCF or TCTTEPCR, are intended for use in application programs using CICS macros, and should not be required in application programs using CICS commands.

RECEIVE (3270 display)

Function

Receive data from a 3270 information display system (BTAM or TCAM).

Command syntax



RECEIVE receives data from the terminal.

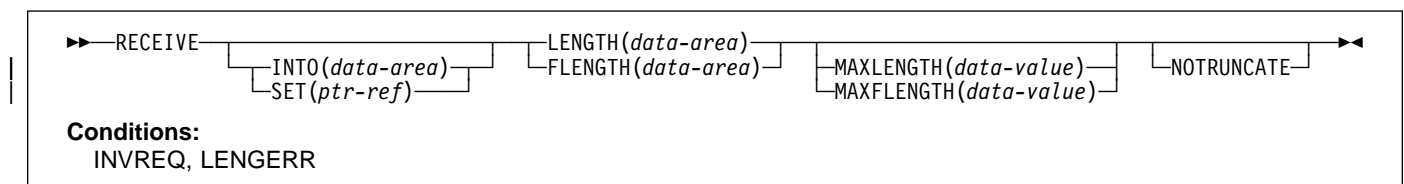
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) (and BUFFER has not been specified), you can omit both the INTO and SET options.

RECEIVE (3600 BTAM)

Function

Receive data from a 3600 finance communication system (BTAM) - only valid for remote terminals. This form of the RECEIVE command also applies to the 4700 finance communication system. For more information about using the 3600 or 4700 finance communication system, see "CONVERSE (3600 BTAM)" on page 58 and "BTAM programmable terminals" on page 358.

Command syntax



RECEIVE receives data from the terminal. For more information about using the 3600 or 4700 finance communication system, see "CONVERSE (3600 BTAM)" on page 58.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

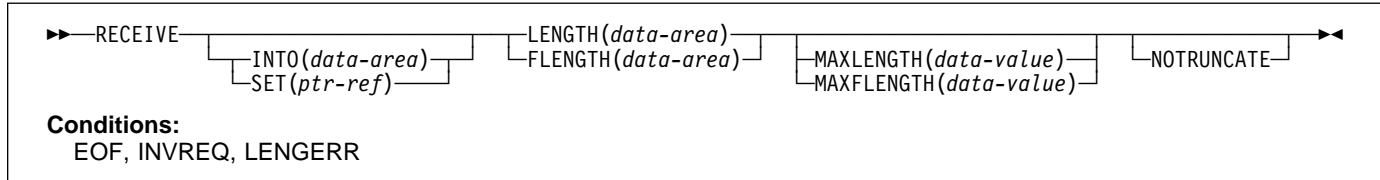
RECEIVE (non-VTAM)

RECEIVE (3735)

Function

Receive data from a 3735 Programmable Buffered Terminal.

Command syntax



RECEIVE receives data from the terminal. The 3735 Programmable Buffered Terminal may be serviced by CICS in response to terminal-initiated input (automatic answering), or as a result of an automatic (automatic calling) or time-initiated transaction.

For more information about using the 3735 terminal, see “CONVERSE (3735)” on page 60.

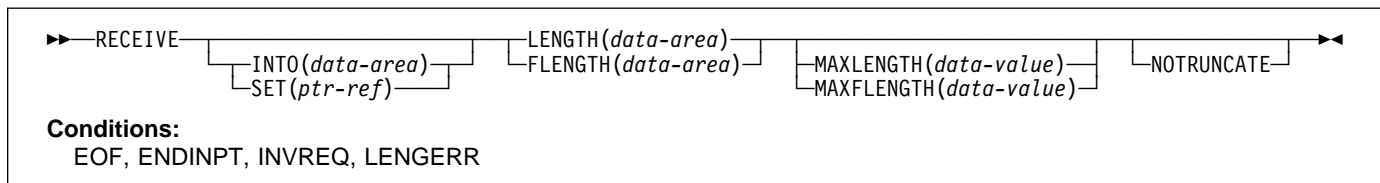
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (3740)

Function

Receive data from a 3740 data entry system.

Command syntax



RECEIVE receives data from the terminal.

In batch mode, many files are exchanged between the 3740 and CICS in a single transmission. The transmission of an input batch must be complete before an output transmission can be started.

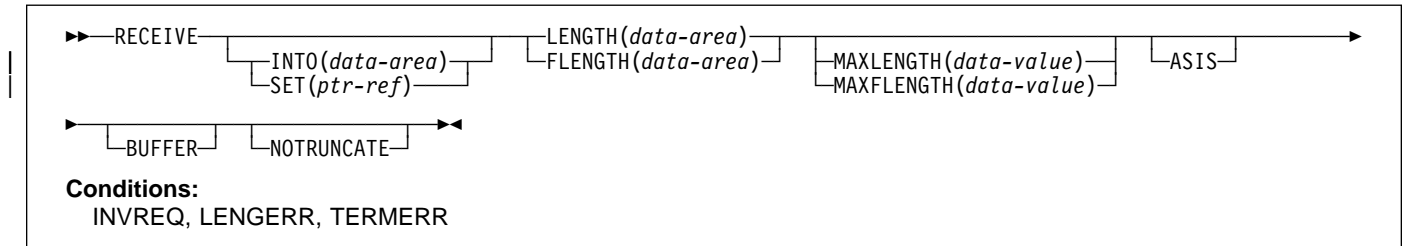
If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) you can omit both the INTO and SET options.

RECEIVE (3790 3270-display)

Function

Receive data from a 3790 (3270-display) logical unit.

Command syntax



RECEIVE receives data from the terminal.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID) (and BUFFER has not been specified), you can omit both the INTO and SET options.

RECEIVE (non-VTAM)

RECEIVE (non-VTAM) options

If you specify the INTO option, but omit the MAXLENGTH option, the argument must be a data area that specifies the maximum length that the program accepts. If the value specified is less than zero, zero is assumed.

If the length of the data exceeds the value specified, but the NOTRUNCATE option is not specified, the data is truncated to that value and the LENGERR condition occurs. When the data has been received, the data area is set to the original length of the data.

If you specify the SET option, the argument must be a data area. When the data has been received, the data area is set to the length of the data.

For a description of a safe upper limit, see “LENGTH options” on page 5.

ASIS

indicates that output is to be sent in transparent mode (with no recognition of control characters and accepting any of the 256 possible combinations of eight bits as valid transmittable data).

This option has no effect on the first RECEIVE command of a transaction, because terminal control performs a READ INITIAL operation and uses the terminal defaults to translate the data.

This option has no effect if the screen contains data prior to a transaction being initiated. This data is read and translated in preparation for the next task and the first RECEIVE command in that task retrieves the translated data.

Note: If you are using a katakana terminal, you might see some messages containing mixed English and katakana characters. That is because katakana terminals cannot display mixed-case output. Uppercase characters in the data stream appear as uppercase English characters, but lowercase characters appear as katakana characters. If this happens, ask your system programmer to specify MSGCASE=UPPER in the system initialization parameters so that messages contain uppercase characters only.

BUFFER (not TCAM)

specifies that the contents of the 3270 buffer are to be read, beginning at buffer location one and continuing until all contents of the buffer have been read. All character and attribute sequences (including nulls) appear in the input data stream in the same order that they appear in the 3270 buffer.

FLENGTH(data-area)

A fullword alternative to LENGTH.

INTO(data-area)

specifies the receiving field for the data read from the terminal or logical unit, or the application target area receiving the data from the application program connected to the other end of the current conversation.

LEAVEKB

specifies that the keyboard is to remain locked at the completion of the data transfer.

LENGTH(data-area)

specifies the length, as a halfword binary value, of the data transmitted.

If you specify the INTO option, but omit the MAXLENGTH option, the argument must be a data area that specifies the maximum length that the program accepts. If the value specified is less than zero, zero is assumed.

If the length of the data exceeds the value specified, but the NOTRUNCATE option is not specified, the data is truncated to that value and the LENGERR condition occurs. When the data has been received, the data area is set to the original length of the data.

If you specify the SET option, the argument must be a data area. When the data has been received, the data area is set to the length of the data.

For a description of a safe upper limit, see “LENGTH options” on page 5.

MAXFLENGTH(data-value)

A fullword alternative to MAXLENGTH.

MAXLENGTH(data-value)

specifies the maximum amount (halfword binary value) of data that CICS is to recover. If INTO is specified, MAXLENGTH overrides the use of LENGTH as an input to CICS. If SET is specified, MAXLENGTH provides a way for the program to limit the amount of data it receives at one time.

If the value specified is less than zero, zero is assumed.

If the length of data exceeds the value specified and the NOTRUNCATE option is not present, the data is truncated to that value and the LENGERR condition occurs. The data area specified in the LENGTH option is set to the original length of data.

If the length of data exceeds the value specified and the NOTRUNCATE option is present, CICS retains the remaining data and uses it to satisfy subsequent RECEIVE commands. The data area specified in the LENGTH option is set to the length of data returned.

If this option is omitted, the value indicated in the LENGTH option is assumed.

NOTRUNCATE

specifies that, when the data available exceeds the length requested, the remaining data is not to be discarded but is to be retained for retrieval by subsequent RECEIVE commands.

PASSBK

specifies that communication is with a passbook.

PSEUDOBIN

specifies that the data being read is to be translated from System/7 pseudobinary representation to hexadecimal.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If this option is omitted, the principal facility for the task is used.

SET(ptr-ref)

specifies a pointer reference that is to be set to the address of data received from the conversation partner in an MRO conversation. The pointer reference is valid until the next receive command or the end of task.

If DATALOCATION(ANY) is associated with the application program, the address of the data can be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
FREE
PENDFREE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSSEND

RECEIVE (non-VTAM) conditions

The following conditions can occur in combination with others. CICS checks for these conditions in the order:

1. INBFMH
2. EOC.

If more than one occurs, only the first is passed to the application program. EIBRCODE, however, is set to indicate all the conditions that occurred.

ENDINPT (not TCAM)

occurs when an end-of-input indicator is received.

Default action: terminate the task abnormally.

EOC

occurs when a request/response unit (RU) is received with the end-of-chain indicator set. Field EIBEOC also contains this indicator.

Default action: ignore the condition.

EOF (not TCAM)

occurs when an end-of-file indicator is received.

Default action: terminate the task abnormally.

INBFMH

occurs if a request/response unit (RU) contains a function management header (FMH). Field EIBFMH contains this indicator and it should be used in preference to INBFMH. The IGNORE CONDITION command can be used to ignore the condition.

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs if data is discarded by CICS because its length exceeds the maximum the program accepts and the NOTRUNCATE option is not specified.

Default action: terminate the task abnormally.

NOPASSBKRD

occurs if no passbook is present.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

RDATT (not TCAM)

occurs if a RECEIVE command is terminated by the attention (ATTN) key rather than the return key.

Default action: ignore the condition.

TERMERR

occurs for a terminal-related error, such as a session failure. This condition applies to VTAM-connected terminals only.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

RECEIVE MAP

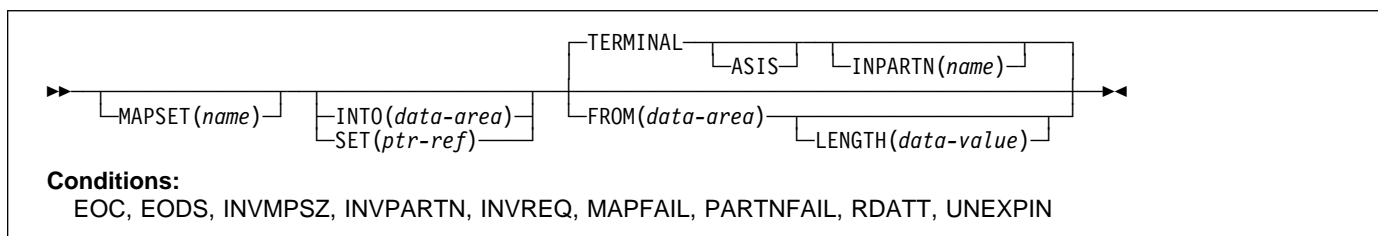
Function

Receive screen input into an application data area. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

Command syntax

```
▶▶—RECEIVE MAP(name)—▶▶
```

Minimum BMS:



Note: INPARTN is supported by Standard and full BMS

RECEIVE MAP maps input data from a terminal into a data area in an application program.

Data from certain logical units is not mapped, but is left unaltered. Refer to the appropriate CICS subsystem guide to see if this is true for a particular logical unit.

Following a RECEIVE MAP command, the inbound cursor position is placed in EIBCPOSN, and the terminal attention identifier (AID) placed in EIBAID.

See Appendix K, “BMS macro summary” on page 381 for map definitions.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID), you can omit both the INTO and the SET options.

RECEIVE MAP options

ASIS

specifies that lowercase characters in the 3270 input data stream are not translated to uppercase; this allows the current task to receive a message containing both uppercase and lowercase data.

This option has no effect on the first RECEIVE command of a transaction, or if the screen contains data prior to a transaction being initiated. For example, if a transaction is initiated by another transaction, and begins by receiving data originally output by that transaction, it

cannot suppress uppercase translation on the data. This data is read and translated in preparation for the next task and the first RECEIVE command in that task retrieves the translated data.

Note: If you are using a katakana terminal, you might see some messages containing mixed English and katakana characters. That is because katakana terminals cannot display mixed-case output. Uppercase characters in the data stream appear as uppercase English characters, but lowercase characters appear as katakana characters. If this happens, ask your system programmer to specify MSGCASE=UPPER in the system initialization parameters so that messages contain uppercase characters only.

FROM(*data-area*)

specifies the data area containing the data to be mapped by a RECEIVE MAP command. This includes the 12-byte prefix generated by the TIOAPFX=YES option on the DFHMDI and DFHMSD BMS map definitions (see pages 395 and 401).

INPARTN(*name*)

specifies the name (1–2 characters) of the partition in which the terminal operator is expected to enter data. If the terminal operator enters data in some other partition, the INPARTN partition is activated, the keyboard is unlocked for the partition, and an error message is output to any error message partition. This option is ignored if the terminal does not support partitions, or if there is no application partition set.

INTO(data-area)

specifies the data area into which the mapped data is to be written. If this field is not specified, the name defaults to the name of the map suffixed with an I.

LENGTH(data-value)

specifies the length of the data to be formatted as a halfword binary value. It must not exceed the length of the FROM data area, but this should include the length of the 12-byte prefix generated by the TIOAPFX=YES option on the DFHMDI and DFHMSD BMS map definitions (see pages 395 and 401).

For a description of a safe upper limit, see "LENGTH options" on page 5.

MAP(name)

specifies the name (1–7 characters) of the map to be used.

MAPSET(name)

specifies the unsuffixed name (1–7 characters) of the mapset to be used. The mapset must reside in the CICS program library. The mapset can be defined either by using RDO or by program autoinstall when the mapset is first used. If this option is not specified, the name given in the MAP option is assumed to be that of the mapset.

SET(ptr-ref)

specifies the pointer that is to be set to the address of the 12-byte prefix to the mapped data.

The pointer reference is valid until the next receive command or the end of task.

If DATALOCATION(ANY) is associated with the application program, the address of the data may be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

TERMINAL

specifies that input data is to be read from the terminal that originated the transaction.

RECEIVE MAP conditions

Some of the following conditions can occur in combination. If more than one occurs, only the first is passed to the application program.

EIBRCODE, however, is set to indicate all the conditions that occurred.

EOC

occurs if the request/response unit (RU) is received with the end-of-chain (EOC) indicator set. It applies only to logical units.

Default action: ignore the condition.

EODS

occurs if no data is received (only an FMH). It applies only to 3770 batch LUs and to 3770 and 3790 batch data interchange LUs.

Default action: terminate the task abnormally.

INVMPSZ

occurs if the specified map is too wide or too long for the terminal.

Default action: terminate the task abnormally.

INVPARTN

occurs if the specified partition is not defined in the partition set associated with the application program.

Default action: terminate the task abnormally.

INVREQ

occurs if a RECEIVE MAP command is issued in a nonterminal task; these tasks do not have a TIOA or a TCTTE.

Default action: terminate the task abnormally.

MAPFAIL

occurs if the data to be mapped has a length of zero or does not contain a set-buffer-address (SBA) sequence. It applies only to 3270 devices. The receiving data area contains the unmapped input data stream. The amount of unmapped data moved to the user's area is limited to the length specified in the LENGTH option. The input map is not set to nulls.

This condition also arises if a program issues a RECEIVE MAP command to which the terminal operator responds by pressing a CLEAR or PA key, or by pressing ENTER or a PF key without entering data.

Default action: terminate the task abnormally.

PARTNFAIL

occurs if the terminal operator attempts to enter data more than three times in a partition other than that specified by the INVPARTN option.

Default action: terminate the task abnormally.

RDATT

occurs if a RECEIVE MAP command is terminated by the operator using the ATTN key rather than the RETURN key. It applies only to the 2741 Communications Terminal, and only if 2741 read attention support has been generated for CICS.

Default action: ignore the condition.

RECEIVE MAP

UNEXPIN

occurs when unexpected or unrecognized data is received. This only applies to batch data interchange terminals.

Default action: terminate the task abnormally.

RECEIVE MAP MAPPINGDEV

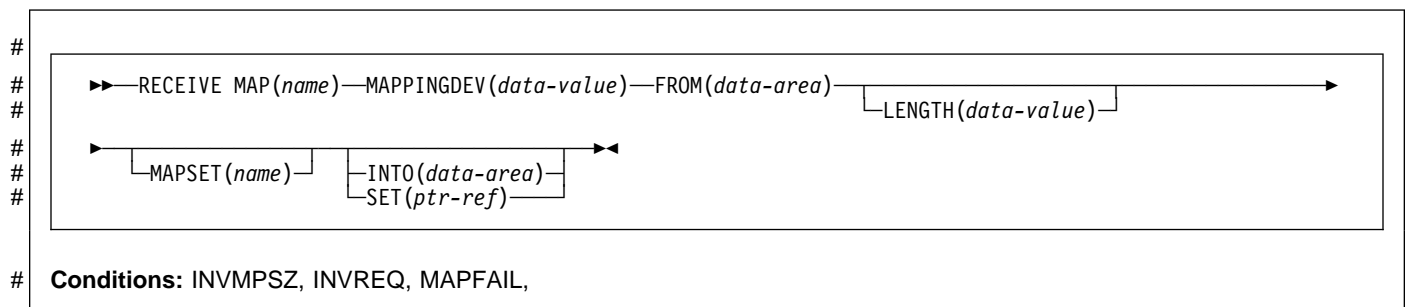
Function

Receive screen input into an application data area, without reference to the principal facility, if any, using terminal characteristics obtained from the MAPPINGDEV parameter.

Apar PQ06521

RECEIVE MAP MAPPINGDEV added by APAR PQ06521 12/08/97

Command syntax



#

Usage

RECEIVE MAP MAPPINGDEV allows the mapping of input data from a 3270 terminal that is not necessarily the principal facility of the transaction.

MAPPINGDEV specifies the name of a 3270 terminal whose BMS characteristics were used to create the input data stream. This may be a terminal from which the data was originally received using a RECEIVE command.

Parameters

AID(*data-value*)

specifies the one-byte data area containing the value of the 3270 attention identifier (AID) to be used when performing the mapping operation. Usually this will be the value contained in EIBAID following the RECEIVE operation that originally received the datastream from the terminal.

The value specified is moved into field EIBAID in the EXEC interface block on completion of the operation. No check is made that the AID value specified is valid.

If AID(*data-value*) is not specified, then the AID value defaults to X'7D' (the Enter key).

If the AID byte (either explicitly, or by default) indicates an operation other than CLEAR, PA1, PA2, or PA3, and CURSLOC=YES is specified for the map, then the field containing the cursor is flagged by setting the X'02' bit in its flag byte.

If the AID (whether specified explicitly, or by default) is the subject of a HANDLE AID command, the specified branch will be taken in the usual way.

CURSOR(*data-value*)

specifies an unsigned halfword binary field containing the cursor position (relative to zero) to be used. Usually this will be the value contained in EIBCPOSN following the RECEIVE operation that originally received the datastream from the terminal.

The value specified is moved into EIBCPOSN in the EXEC interface block on completion of the operation. No check is made that the CURSOR value specified is valid.

If CURSOR(*data-value*) is not specified, then the cursor value defaults to X'0000'.

FROM(*data-area*)

specifies the data area containing the data to be mapped. This must be in the format of a TIOA and must contain a 12-byte prefix.

INTO(*data-area*)

specifies the data area into which the mapped data is to be written. If this field is not specified, the name defaults to the name of the map suffixed with an I.

LENGTH(*data-value*)

specifies the length of the data to be formatted as a halfword binary value. It must not exceed the length of the FROM data area, but this should include the length of the 12-byte prefix generated by the TIOAPFX=YES option on the DFHMDI and DFHMSD BMS map definitions (see pages 395 and 401).

RECEIVE MAP MAPPINGDEV

For a description of a safe upper limit, see "LENGTH options" on page 5. # Default action: terminate the task abnormally.

MAP(*name*)
specifies the name (1–7 characters) of the map to be used.

MAPPINGDEV(*data-value*)
specifies the name of a 3270 terminal whose characteristics match those of the terminal from which the data was originally received using a RECEIVE command.

MAPSET(*name*)
specifies the unsuffixed name (1–7 characters) of the mapset to be used. The mapset must reside in the CICS program library. The mapset can be defined either by using RDO or by program autoinstall when the mapset is first used. If this option is not specified, the name given in the MAP option is assumed to be that of the mapset.

SET(*ptr-ref*)
specifies the pointer that is to be set to the address of the 12-byte prefix to the mapped data.
The pointer reference is valid until the next RECEIVE or RECEIVE MAP command, or until the end of the transaction, unless FREEMAINed by the application.
If "TASKDATALOC(ANY)" is specified for the running task, the data returned may be above or below the 16MB line.
If "TASKDATALOC(BELOW)" is specified for the running task, the data returned is below the 16MB line.
If "TASKDATAKEY(USER)" is specified for the running task, and storage protection is active, the data returned is in user-key. If "TASKDATAKEY(CICS)" is specified and storage protection is active, the data returned is in CICS-key.

Error Conditions

Some of the following conditions may occur in combination.
If more than one occurs, only the first is passed to the application program.

INVMPSZ

occurs if the specified map is too wide or too long for the terminal.
Default action: terminate the task abnormally.

INVREQ

occurs if the terminal specified by MAPPINGDEV does not exist, does not support BMS, or is not a 3270 printer or display.
Default action: terminate the task abnormally.

MAPFAIL

occurs if the data to be mapped has a length of zero or does not contain a set-buffer-address (SBA) sequence.

RECEIVE PARTN

Function

Receive data from an 8775 terminal partition. This command is only available on standard and full BMS. For further information about BMS, see the dfhp33h.CICS/ESA *Application Programming Guide*.

Command syntax

```
▶▶RECEIVE PARTN(data-area)▶▶
```

Standard and full BMS:

```
▶▶INTO(data-area)LENGTH(data-value)ASIS▶▶
   SET(ptr-ref)
```

Conditions:

EOC, EODS, INVPARTN, INVREQ, LENGERR

RECEIVE PARTN reads data from a partition on an 8775 terminal. It indicates which partition the data came from, and puts the data into the INTO or the SET data area. You can then treat the data as though it had originated from a terminal in base (unpartitioned) state.

Following a RECEIVE PARTN command, the inbound cursor position is placed in EIBCPOSN, and the terminal attention identifier (AID) placed in EIBAID. EIBAID and EIBCPOSN are also updated at task initiation for non-ATI tasks as well as after each terminal control and BMS input.

See Appendix K, “BMS macro summary” on page 381 for map definitions.

If data is to be received, you must specify either the INTO or the SET option. If a RECEIVE is issued purely to detect an attention identifier (AID), you can omit both the INTO and the SET options.

RECEIVE PARTN options

ASIS

specifies that lowercase characters in the 3270 input data stream are not translated to uppercase; this allows the current task to receive a message containing both uppercase and lowercase data.

The ASIS option has no effect on the first RECEIVE command of a transaction, or if the screen contains data prior to a transaction being initiated. For example, if a transaction is initiated by another transaction, and begins by receiving data originally output by that transaction, it cannot suppress uppercase translation on the data. This data is read and translated in preparation for the next task and the first RECEIVE command in that task retrieves the translated data.

Note: If you are using a katakana terminal, you might see some messages containing mixed English and katakana characters. That is because katakana terminals cannot display mixed-case output. Uppercase characters in the data stream appear as uppercase English characters, but lowercase characters appear as katakana characters. If this happens, ask your system programmer to specify MSGCASE=UPPER in the system initialization parameters so that messages contain uppercase characters only.

INTO(*data-area*)

specifies the area into which the input data stripped of partition controls is to be written. The length of this area must be specified by the LENGTH option. If the area is not large enough to hold the input data, the input data is truncated, and the LENGERR condition raised. The length option data area is set to the length of data received, prior to any truncation.

LENGTH(*data-value*)

specifies the length of the data to be formatted as a halfword binary value. It must be set to the length of any INTO area prior to the command. After the command, BMS sets the LENGTH option to the length of data received prior to any truncation if the INTO area is too small.

For a description of a safe upper limit, see “LENGTH options” on page 5.

PARTN(*data-area*)

is set to the name (1–2 characters) of the input partition.
| The partition can be defined either by using RDO or by
| program autoinstall when the partition is first used.

SET(*ptr-ref*)

specifies the pointer that is to be set to the address of the 12-byte prefix to the mapped data. The pointer

RECEIVE PARTN

reference is valid until the next receive command or the
end of task.

If DATALOCATION(ANY) is associated with the application program, the address of the data may be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

RECEIVE PARTN conditions

Some of the following conditions can occur in combination. If more than one occurs, only the first one is passed to the application program.

EOC

occurs if the request/response unit (RU) is received with the end-of-chain (EOC) indicator set. It applies only to logical units.

Default action: ignore the condition.

EODS

occurs if no data is received (only an FMH). It applies only to 3770 batch LUs and to 3770 and 3790 batch data interchange LUs.

Default action: terminate the task abnormally.

INVPARTN

occurs if the specified partition is not defined in the partition set associated with the application program.

Default action: terminate the task abnormally.

INVREQ

occurs if a RECEIVE PARTN command is issued in a nonterminal task; these tasks do not have a TIOA or a TCTTE.

Default action: terminate the task abnormally.

LENGERR

occurs if the INTO area of a RECEIVE PARTN command is not large enough to hold the input data.

Default action: truncate the data to fit within the INTO area.

RELEASE

Function

Release a loaded program, table, or mapset.

Command syntax

```
▶▶—RELEASE—PROGRAM(name)—▶▶
```

Conditions:

INVREQ, NOTAUTH, PGMIDERR

Note for dynamic transaction routing

Using RELEASE of a program LOADED with HOLD could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

RELEASE releases the program, table, or mapset previously loaded by a LOAD command. This means that the issuing task can no longer use the resource unless another LOAD is issued.

If the HOLD option is specified in the LOAD command, the loaded resource is not released at the end of the task. It can only be released by a RELEASE command. This RELEASE command may be issued by the task that loaded the resource or by any other task.

If the HOLD option is not specified in the LOAD command, the loaded resource is released at the end of the task. It may, however, be released before this by the task that loaded the resource issuing a RELEASE command.

The following example shows how to release an application program, called PROG4, loaded in response to a LOAD command:

```
EXEC CICS RELEASE PROGRAM('PROG4')
```

RELEASE option

PROGRAM(name)

specifies the identifier (1–8 characters) of a program, table, or mapset to be deleted. The specified name must have been defined as a program to CICS, though if AUTOINSTALL is active a definition is autoinstalled.

RELEASE conditions

INVREQ

occurs in any of the following situations:

- An invalid attempt is made by the program to release itself (RESP2=5). A RELEASE command for the program that contains this command is allowed only when a corresponding LOAD command for the program has been issued from the same task, or when a LOAD command with the HOLD option has been issued from another task.
- The command is issued for a program that is not loaded (RESP2=6).

Apar PQ28669

Documentation for Apar PQ28669 added
09/09/99

Either the command is issued for a program that was loaded, without the HOLD option, by another task, or the program has been enabled as a global user exit (RESP2=7).

- The program is defined with RELOAD=YES. It must be released by a FREEMAIN rather than a RELEASE command (RESP2=17).
- The program manager domain has not yet been initialized. This is probably due to a release request having been made in a first stage PLT (RESP2=30).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on PROGRAM(name).

Default action: terminate the task abnormally.

RELEASE

PGMIDERR

occurs in any of the following situations:

- A program, table, or mapset has no entry in the PPT (RESP2=1)
- A program, table, or mapset is disabled (RESP2=2)
- The installed program definition is for a remote program (RESP2=9).

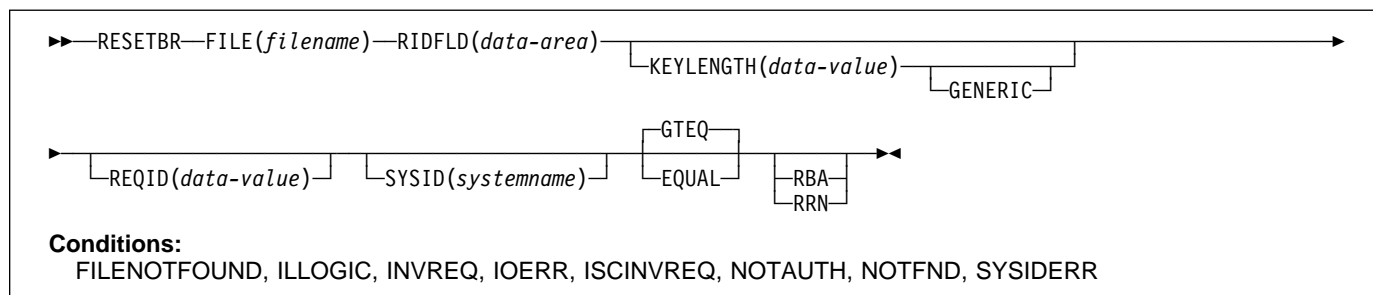
Default action: terminate the task abnormally.

RESETBR

Function

Reset start of browse.

Command syntax



RESETBR specifies, during a browse, the record in a file or data table on a local or a remote system, where you want the browse to be repositioned.

When browsing a VSAM file or data table, you can use this command not only to reposition the browse (which can be achieved more simply by modifying the RIDFLD data area on a READNEXT or READPREV command), but also to change its characteristics from those specified on STARTBR, without ending the browse. The characteristics that may be changed are those specified by the GENERIC, GTEQ, and RBA options.

When browsing a BDAM file, you can include this command at any time prior to issuing any other browse command. It is similar to an ENDBR-STARTBR sequence (but with less function), and gives the BDAM user the sort of skip sequential capability that is available to VSAM users through use of the READNEXT command.

RESETBR options

EQUAL (VSAM and data table)

specifies that the search is satisfied only by a record having the same key (complete or generic) as that specified in the RIDFLD option.

FILE(filename)

specifies the of the file to be accessed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

GENERIC (VSAM KSDS, path or data table)

specifies that the search key is a generic key whose length is specified in the KEYLENGTH option. The search for a record is satisfied when a record is found

that has the same starting characters (generic key) as those specified.

GTEQ (VSAM and data table)

specifies that if the search for a record having the same key (complete or generic) as that specified in the RIDFLD option is unsuccessful, the first record having a greater key is retrieved. Use this option only with keyed or RRN.

KEYLENGTH(data-value)

specifies the length (halfword binary) of the key that has been specified in the RIDFLD option, except when RBA or RRN is specified, in which case KEYLENGTH is not valid.

This option must be specified if GENERIC is specified, and it can be specified whenever a key is specified. If the length specified is different from the length defined for the data set and the operation is not generic, the INVREQ condition occurs.

The INVREQ condition also occurs if a RESETBR command specifies GENERIC, and the KEYLENGTH is not less than that specified in the VSAM definition.

If KEYLENGTH(0) is used with the object of reading the first record in the data set, the GTEQ option must also be specified. If EQUAL is specified either explicitly or by default with KEYLENGTH(0), the results of the STARTBR are unpredictable.

RBA (VSAM ESDS, KSDS, or CMT)

(base data sets only, not paths)
specifies that the record identification field specified in the RIDFLD option contains a relative byte address. Use this option only when browsing a KSDS and using relative byte addresses instead of keys to identify the records.

| You cannot use RBA for user-maintained data tables.

RESETBR

REQID(data-value)

specifies as a halfword binary value a unique request identifier for a browse, used to control multiple browse operations on a data set. If this option is not specified, a default value of zero is assumed.

RIDFLD(data-area)

specifies the record identification field. When combined with RBA or RRN this is a 4-character field. The contents can be a key, a relative byte address, or relative record number (for VSAM data sets), or a block reference, physical key, and a deblocking argument (for BDAM data sets). For a relative byte address or a relative record number, the format of this field must be fullword binary. For a relative byte address, the RIDFLD can be greater than or equal to zero. For a relative record number, the RIDFLD can be greater than or equal to 1.

For VSAM, a full record id of X'FF's indicates that the browse is to be positioned at the end of the data set in preparation for a backwards browse using READPREV commands.

RRN (VSAM RRDS)

specifies that the record identification field specified in the RIDFLD option contains a relative record number.

SYSID(systemname)

specifies the name of the system to which the request is directed.

If you specify SYSID, and omit both RBA and RRN, you must also specify KEYLENGTH; it cannot be found in the FCT.

RESETBR conditions

Note: RESP2 values are not set for files that are on remote systems.

FILENOTFOUND

occurs if a file name referred to in the FILE option cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is greater than or equal to the length of a full key (RESP2=25).

- The KEYLENGTH option is specified (but the GENERIC option is not specified), and the specified length does not equal the length defined for the data set to which this file refers (RESP2=26).
- The REQID, if any, does not match that of any successful STARTBR command (RESP2=36).
- The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than zero (RESP2=42).

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the file control operation (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error.

(Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTFND

occurs if an attempt to retrieve a record based on the search argument provided is unsuccessful (RESP2=80).

NOTFND can also occur if a generic RESETBR with keylength(0) specifies the EQUAL option.

Default action: terminate the task abnormally.

SYSIDERR

| occurs when the SYSID option specifies a name that is
| neither the local system nor a remote system (made
| known to CICS by defining a CONNECTION).
SYSIDERR also occurs when the link to the remote
system is closed (RESP2=130).

Default action: terminate the task abnormally.

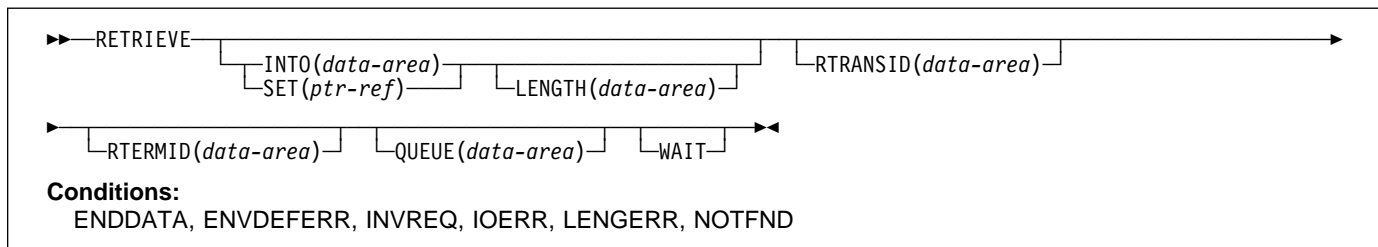
RETRIEVE

RETRIEVE

Function

Retrieve data stored for a task.

Command syntax



Note for dynamic transaction routing

Using RETRIEVE with WAIT could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

RETRIEVE retrieves data stored by expired START commands. It is the only method available for accessing such data.

A task that is not associated with a terminal can access only the single data record associated with the original START command; it does so by issuing a RETRIEVE command. The storage occupied by the data associated with the task is

Each data record is retrieved from temporary storage using the REQID of the original START command as the identification of the record in temporary storage.

When all expired data records have been retrieved, the ENDDATA condition occurs. The storage occupied by the single data record associated with a START command is released after the data has been retrieved by a RETRIEVE command; any storage occupied by data that has not been retrieved is released when the CICS system is terminated.

If the retrieved data contains FMHs (Function Management Headers), as specified by the FMH option on the associated START command, field EIBFMH in the EIB is set to X'FF'. If no FMH is present, EIBFMH is set to X'00'.

The following example shows how to retrieve data stored by a START command for the task, and store it in the user-supplied data area called DATAFLD.

```
EXEC CICS RETRIEVE
      INTO(DATAFLD)
      LENGTH(LENG)
```

The following example shows how to request retrieval of a data record stored for a task into a data area provided by CICS; the pointer reference (PREF) specified by the SET option is set to the address of the storage area reserved for the data record.

```
EXEC CICS RETRIEVE
      SET(PREF)
      LENGTH(LENG)
```

RETRIEVE options

Apar PN92143
Documentation for Apar PN92143 added 28/02/97

normally released on execution of the RETRIEVE command, or on termination of the task if no RETRIEVE command is executed prior to termination.

Apar PN92143
Documentation for Apar PN92143 added 28/02/97

If the START command specified ATTACH, the storage is
not released, the SET option must be used and the length of
the data is not returned because it is not known. The INTO
option in these circumstances causes the INVREQ condition
(ASSIGN STARTCODE returns 'U' rather than 'S' or 'SD').

A task that is associated with a terminal can access all data records associated with all expired START commands having the same transaction identifier and terminal identifier as this task, that is the task issuing the RETRIEVE command; it does so by issuing consecutive RETRIEVE commands. Expired data records are presented to the task on request in expiration-time sequence, starting with any data stored by the command that started the task, and including data from any commands that have expired since the task started.

INTO(data-area)

specifies the user data area into which retrieved data is to be written.

LENGTH(data-area)

specifies a halfword binary value to define the length of the data area the retrieved data is written into.

If you specify the INTO option, the argument must be a data area that specifies the maximum length of data that the program is prepared to handle. If the value specified is less than zero, zero is assumed. If the length of the data exceeds the value specified, the data is truncated to that value and the LENGERR condition occurs. On completion of the retrieval operation, the data area is set to the original length of the data.

If you specify the SET option, the argument must be a data area. On completion of the retrieval operation, the data area is set to the length of the data.

For a description of a safe upper limit, see "LENGTH options" on page 5.

QUEUE(data-area)

specifies the 8-character area for the temporary storage queue name that may be accessed by the transaction issuing the RETRIEVE command.

RTERMID(data-area)

specifies a 4-character area that can be used in the TERMID option of a START command that may be executed subsequently.

RTRANSID(data-area)

specifies a 4-character area that can be used in the TRANSID option of a START command that may be executed subsequently.

SET(ptr-ref)

specifies the pointer reference to be set to the address of the retrieved data.

If DATALOCATION(ANY) is associated with the application program, the address of the data may be above or below the 16MB line.

If DATALOCATION(BELOW) is associated with the application program, and the data resides above the 16MB line, the data is copied below the 16MB line, and the address of this copy is returned.

If TASKDATAKEY(USER) is specified for the running task, and storage protection is active, the data returned is in a user-key. If TASKDATAKEY(CICS) is specified and storage protection is active, the data returned is in a CICS-key.

If you use SET you must also include LENGTH.

WAIT

specifies that, if all expired data records have already been retrieved, the task is to be put into a wait state until further expired data records become available. Although this means that the ENDDATA condition is not raised at the time the RETRIEVE command is issued, it is raised

later if CICS enters shutdown or if the task is subject to deadlock time-out and it waits for longer than the deadlock time-out interval (see the DTIMOUT option of RDO DEFINE TRANSACTION).

An attempt to issue RETRIEVE WAIT during shutdown leads to an AICB abend if there is no data record already available to satisfy the request.

If you use WAIT, you must have at least one other option.

RETRIEVE

RETRIEVE conditions

ENDDATA

occurs in any of the following situations:

- No more data is stored for the task issuing a RETRIEVE command. It can be considered a normal end-of-file response when retrieving data records sequentially.
- The RETRIEVE command is issued by a task that is started by a START command that did not specify any of the data options FROM, RTRANSID, RTERMID, or QUEUE.
- The RETRIEVE command is issued by a nonterminal task that was not created as a result of a START command.
- WAIT was specified and the task was waiting for a data record but none became available before the deadlock time-out interval expired (see the DTIMOUT option of RDO DEFINE TRANSACTION).
- WAIT was specified and the task was waiting when CICS entered shutdown. An attempt to issue RETRIEVE WAIT during shutdown leads to an AICB abend if there is no data record already available to satisfy the request.
- A RETRIEVE command with the WAIT option is issued when no data is available; the task was initiated by a START command that specified an APPC connection or terminal in the TERMID option.

Default action: terminate the task abnormally.

ENVDEFERR

occurs when a RETRIEVE command specifies an option not specified by the corresponding START command.

Default action: terminate the task abnormally.

INVREQ

occurs if the RETRIEVE command is not valid for processing by CICS.

Default action: terminate the task abnormally.

IOERR

occurs if an input/output error occurs during a RETRIEVE operation. The operation can be retried by reissuing the RETRIEVE command.

Default action: terminate the task abnormally.

LENGERR

occurs if the length specified is less than the actual length of the stored data.

Default action: terminate the task abnormally.

NOTFND

occurs in any of the following situations:

- A RETRIEVE command is issued, but some prior task retrieved the data stored under the request identifier directly through temporary storage requests and then released the data.
- The request identifier associated with the START command is not unique, so when a RETRIEVE command is issued, CICS cannot find the data.

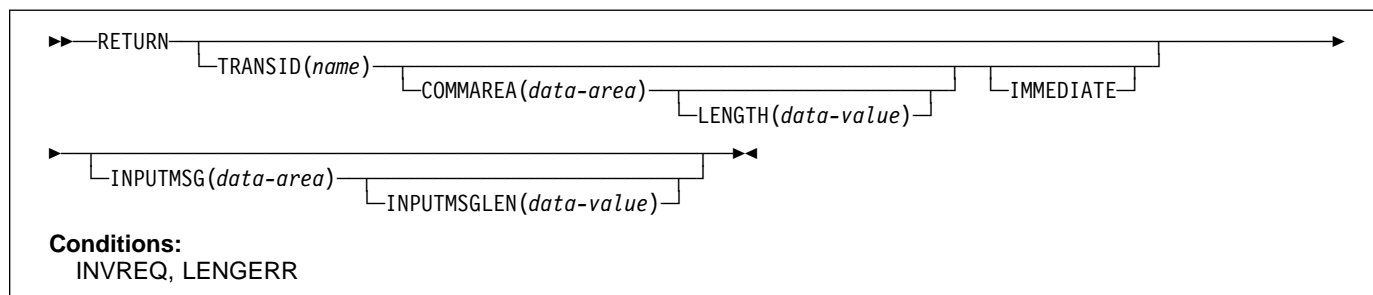
Default action: terminate the task abnormally.

RETURN

Function

Return program control.

Command syntax



RETURN returns control from an application program either to an application program at the next higher logical level, or to CICS.

The LENGTH option specifies the length of the data to be passed. The LENGTH value being passed must not be greater than the length of the data area specified in the COMMAREA option. If it is, the results are unpredictable and may result in a LENGERR condition, as described in the section about passing data to other programs in the *CICS/ESA Application Programming Guide*.

The valid range for the COMMAREA length is 0 through 32763 bytes. If the length provided is outside this range, the LENGERR condition occurs. The COMMAREA and IMMEDIATE options can be used only when the RETURN command is returning control to CICS; otherwise, the INVREQ condition occurs.

No resource security checking occurs on the RETURN TRANSID command. However, transaction security checking is still available when CICS attaches the returned transaction.

RETURN options

COMMAREA(data-area)

specifies a communication area that is to be made available to the next program that receives control. In a COBOL receiving program, you must give this data area the name DFHCOMMAREA. (See the *CICS/ESA Application Programming Guide*.) Because the data area is freed before the next program starts, a copy of the data area is created and a pointer to the copy is passed.

The communication area specified is passed to the next program that runs at the terminal. To ensure that the communication area is passed to the correct program, include the IMMEDIATE option.

This option is valid only on a RETURN command issued by a program at the highest logical level, that is, a program returning control to CICS.

IMMEDIATE

ensures that the transaction specified in the TRANSID option is attached as the next transaction regardless of any other transactions enqueued by ATI for this terminal. The next transaction starts immediately and appears to the operator as having been started by terminal data. If the terminal is using bracket protocol, the terminal is also held in bracket. This option is valid only on a RETURN command issued by a program at the highest logical level, that is a program returning control to CICS.

Note that in a multi region environment, using
IMMEDIATE does not affect the transaction definition as
this is still found in the terminal-owning region (TOR).

INPUTMSG(data-area)

specifies data to be passed either to another transaction, identified by the TRANSID option, or to a calling program in a multiprogram transaction. You can also use INPUTMSG when returning control to CICS from a user-written dynamic transaction routing program, when you might want to modify the initial input.

In all cases, the data in the INPUTMSG data area is passed to the first program to issue a RECEIVE command following the RETURN.

See the *CICS/ESA Application Programming Guide* for more information and illustrations about the use of INPUTMSG.

INPUTMSGLEN(data-value)

specifies a halfword binary value to be used with INPUTMSG.

LENGTH(data-value)

specifies a halfword binary value that is the length in bytes of the COMMAREA. For a description of a safe upper limit, see "LENGTH options" on page 5.

RETURN

TRANSID(name)

specifies the transaction identifier (1–4 characters) to be used with the next input message entered from the terminal with which the task that issued the RETURN command has been associated. The specified name must have been defined as a transaction to CICS.

Apar 60841

Documentation for Apar 60841 added 15 Nov 1994 (TUCKER)

If TRANSID is specified for a program running on a terminal that is defined with a permanent transaction ID, the terminal's permanent transaction is initiated next rather than the transaction specified on the RETURN.

If you specify a TRANSID of binary zeros, the transaction identifier for the next program to be associated with the terminal may be determined from subsequent input from the terminal. Issuing a RETURN with a TRANSID of binary zeros and a COMMAREA can cause unpredictable results if the next transaction is not coded to handle the COMMAREA or if it receives a COMMAREA not intended for it.

If you specify TRANSID on a program that is not at the highest level, and there is a subsequent error on COMMAREA or INPUTMSG on the final RETURN, the TRANSID is cleared.

The next transaction identifier is also cleared on an abnormal termination of the transaction.

If IMMEDIATE is specified with this option, control is passed to the transaction specified in the TRANSID option in preference to any transactions enqueued by ATI.

If IMMEDIATE is not specified with this option, an ATI initiated transaction of the same name enqueued to the terminal nullifies this option.

This option is not valid if the transaction issuing the RETURN command is not associated with a terminal, or is associated with an APPC logical unit.

RETURN conditions

INVREQ

occurs in any of the following situations:

- A RETURN command with the TRANSID option is issued in a program that is not associated with a terminal (RESP2=1).
- A RETURN command with the COMMAREA or IMMEDIATE option is issued by a program that is not at the highest logical level (RESP2=2).
- A RETURN command with the TRANSID option is issued in a program that is associated with an APPC logical unit (RESP2=4).
- A RETURN command with the INPUTMSG option is issued for a program that is not associated with a terminal, or that is associated with an APPC logical unit, or an IRC session (RESP2=8).
- PG domain not initialized. Parameters are not allowed on the EXEC RETURN statement in first stage PLT programs (RESP2=30).
- A RETURN command is issued with an INPUTMSG option by a program invoked by DPL (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

LENGERR occurs in the following conditions:

- The COMMAREA length is less than 0 or greater than 32763 (RESP2=11).
- The COMMAREA ADDRESS passed was zero, but the commarea length was non zero (RESP2=26).
- The INPUTMSG LENGTH was less than 0 or greater than 32767 (RESP2=27).

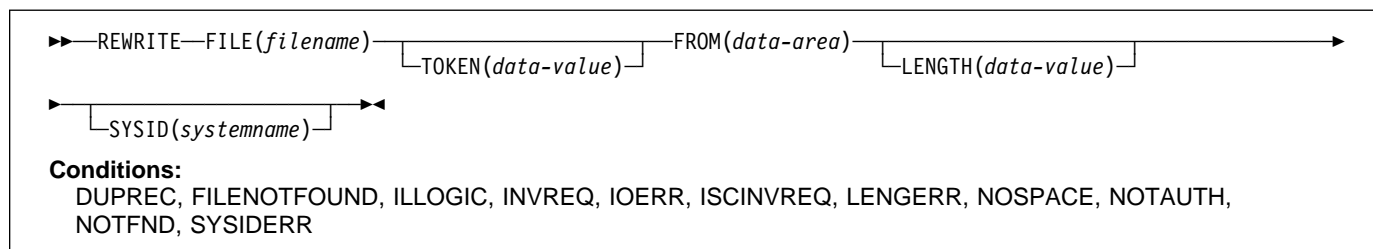
Default action: terminate the task abnormally.

REWRITE

Function

Update a record in a file.

Command syntax



REWRITE updates a record in a file on a local or a remote system. You must always precede this command with a READ UPDATE to read the record to be updated.

For example:

```
EXEC CICS REWRITE
      FROM(RECORD)
      FILE('MASTER')
```

For VSAM data sets, you must not change the key field in the record.

When this command is used to update a record in a CICS-maintained data table, the update is made to both the source VSAM KSDS and the in-memory data table. The details of the command for a CICS-maintained table are the same as for a VSAM KSDS.

| When this command is used to update a record in a
| user-maintained data table, the update is made to the
| in-memory data table.

REWRITE options

FILE(filename)

specifies the of the file to be accessed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

FROM(data-area)

specifies the record that is to be written to the data set referred to by this file.

LENGTH(data-value)

specifies the actual length, as a halfword binary value, of the record to be written.

This option must be specified if SYSID is specified.

This option must also be specified for a file defined as containing variable-length records; which includes user-maintained data tables. It need not be specified if the file contains fixed-length records. You are recommended to include it, however, because it causes a check to be made to ensure that the length of data being written is equal to that defined for the data set. If the lengths are not equal, the LENGERR condition occurs.

SYSID(systemname)

specifies the name of the system to which the request is directed.

TOKEN(data-value)

specifies as a fullword binary value a unique request identifier for a REWRITE, used to associate it with a previous READ UPDATE request.

REWRITE conditions

Note: RESP2 values are not set for files that are on remote systems.

DUPREC

occurs when an attempt is made to rewrite a record to a data set whose upgrade set has an alternate index with the UNIQUEKEY attribute, if the corresponding alternate key already exists in the alternate index (RESP2=150).

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if a file name referred to in the FILE option cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

REWRITE

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- A REWRITE command is issued without a token and no previous READ for UPDATE (also without a token) can be found (RESP2=30).
- A REWRITE command has attempted to change the length of a BDAM variable length record or block (RESP2=46).
- A REWRITE instruction includes a token whose value cannot be matched against any token in use for an existing READ for UPDATE request (RESP2=47).
- An attempt is made to function-ship a request which includes a TOKEN keyword (RESP2=48).

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the file control operation (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error.

(Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- The LENGTH option is not specified for a file with variable-length records, or for a BDAM file with undefined format records (RESP2=10).
- The length specified exceeds the maximum record size (of the source data set for a data-table); the record is truncated (RESP2=12).
- An incorrect length is specified for a file with fixed-length records (RESP2=14).

Default action: terminate the task abnormally.

NOTFND occurs when an attempt to REWRITE a record to a user-maintained data table has failed because the REWRITE is associated with a READ UPDATE request for a record that this transaction has deleted (using DELETE with RIDFLD) after it was read for update (RESP2=80).

This may be caused by a logic error in the application program.

Default action: terminate the task abnormally.

NOSPACE

occurs if no space is available on the direct access device for adding the updated record to the data set (RESP2=100). For user-maintained data tables, this condition occurs if CICS is unable to get sufficient storage in the CICS address space to store the updated data table entry (RESP2=103).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed (RESP2=130).

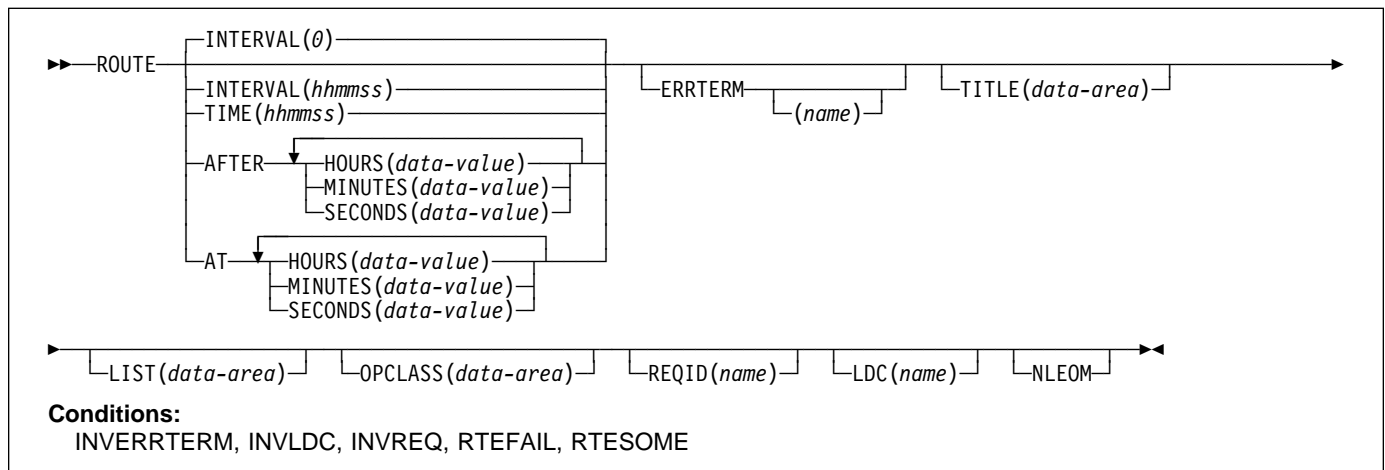
Default action: terminate the task abnormally.

ROUTE

Function

Route a BMS message. (This command is only available on full BMS. The *CICS/ESA Application Programming Guide* has further information about BMS.)

Command syntax



ROUTE routes a BMS logical message to one or more terminals or terminal operators.

This option is effective only if PRGDLY has been specified in the system initialization parameters.

| The default is INTERVAL(0), but for C the default is AFTER
 | HOURS(0) MINUTES(0) SECONDS(0).

| **HOURS(data-value)**
 specifies a fullword binary value in the range 0–99. This is a suboption of the AFTER and AT options. For its use and meaning, see the AFTER option.

ROUTE options

AFTER

specifies the amount of time to elapse before the route.

There are two ways to enter the time under AFTER and AT.

1. A combination of at least two of HOURS(0–99), MINUTES(0–59), and SECONDS(0–59). HOURS(1) SECONDS(3) would mean one hour and three seconds (the minutes default to zero).
2. As one of HOURS(0–99), MINUTES(0–5999), or SECONDS(0–359999). HOURS(1) means one hour. MINUTES(62) means one hour and two minutes. SECONDS(3723) means one hour, two minutes, and three seconds.

AT

specifies the time of the route. For the ways to enter the time, see the AFTER option.

ERRTERM(name)

specifies the name of the terminal to be notified if the message is deleted because it is undeliverable. The message number, title identification, and destination are indicated. If no name is specified, the originating terminal is assumed.

INTERVAL(hhmmss)

specifies the interval of time after which the data is to be transmitted to the terminals specified in the ROUTE command. The **mm** and **ss** are in the range 0–59.

| When using the C language, you are recommended to
 | use the AFTER/AT HOURS, MINUTES, and SECONDS
 | options as C does not provide a packed decimal data
 | type. You may use INTERVAL, but if the value specified
 | is **not** an integer constant, the application is responsible
 | for ensuring that the value passed to CICS is in packed
 | decimal format.

LDC(name) — logical units only

specifies a 2-character mnemonic to be used to determine the logical device code (LDC) to be transmitted in the FMH to the logical unit. The mnemonic identifies an LDC entry specified in the terminal control table TYPE=LDC.

When an LDC is specified, BMS uses the device type, the page size, and the page status associated with the LDC mnemonic to format the message. These values are taken from the extended local LDC table for the LU, if it has one. If the LU has only a local (unextended) LDC table, the values are taken from the system LDC table. The numeric value of the LDC is obtained from

ROUTE

the local LDC table, unless this is an unextended table and the value is not specified, in which case it is taken from the system table.

If the LDC option is omitted, the LDC mnemonic specified in DFHMSD is used; see “DFHMSD” on page 396 for further details. If the LDC option has also been omitted from DFHMSD, the action depends on the type of logical unit, as follows:

3601 LU

The first entry in the local or extended local LDC table is used, if there is one. If a default cannot be obtained in this way, a null LDC numeric value (X'00') is used. The page size used is the value that is specified in the terminal control table TYPE=TERMINAL, or (1,40) if such a value is not specified.

LUTYPE4 LU, batch LU, or batch data interchange LU

The local LDC table is not used to supply a default LDC; instead, the message is directed to the LU console. (Here, LU console means any medium on which the LU elects to receive such messages. For a batch data interchange LU, this does not imply sending an LDC in an FMH). The page size is obtained in the manner described for the 3601 LU.

For message routing, the LDC option of the ROUTE command takes precedence over all other sources. If this option is omitted and a route list is specified (LIST option), the LDC mnemonic in the route list is used; if the route list contains no LDC mnemonic, or no route list is specified, a default LDC is chosen as described above.

LIST(data-area)

specifies the data area that contains a list of terminals and operators to which data is to be directed. If this option is omitted, all terminals supported by BMS receive the data (unless the OPCLASS option is in effect). See the *CICS/ESA Application Programming Guide* for the format of the route list.

MINUTES(data-value)

specifies a fullword binary value in the range 0–59, when HOURS or SECONDS are also specified, or 0–5999 when MINUTES is the only option specified. This is a suboption of the AFTER and AT options. For its use and meaning, see the AFTER option.

NLEOM

specifies that data for a 3270 printer or a 3275 display with the printer adapter feature should be built with blanks and new-line (NL) characters, and that an end-of-message (EM) character should be placed at the end of the data. As the data is printed, each NL character causes printing to continue on the next line, and the EM character terminates printing.

The option is ignored if the device receiving the message (direct or routed) is not one of those mentioned above.

If this option is used, buffer updating and attribute modification of fields previously written into the buffer are not allowed. CICS includes the ERASE option with every write to the terminal.

The NL character occupies a buffer position. A number of buffer positions, equivalent to the value of the RDO options PAGESIZE or ALTPAGE or the PGESIZE or ALTPGE in the terminal control table for that terminal, are unavailable for data. This may cause data to wrap around in the buffer; if this occurs, the PGESIZE value must be reduced.

OPCLASS(data-area)

specifies the data area that contains a list of operator classes to which the data is to be routed. The classes are supplied in a 3-byte field, each bit position corresponding to one of the codes in the range 1 through 24 but in reverse order, that is, the first byte corresponds to codes 24 through 17, the second byte to codes 16 through 9, and the third byte to codes 8 through 1.

REQID(name)

specifies a prefix (2-character field) to be used as part of a temporary storage identifier for CICS message recovery. Only one prefix can be specified for each logical message. The default prefix is **.

BMS message recovery is provided for a logical message only if the PAGING option is specified in the BMS SEND commands, and if the syncpoint has been reached.

SECONDS(data-value)

specifies a fullword binary value in the range 0–59, when HOURS or MINUTES are also specified, or 0–359 999 when SECONDS is the only option specified. This is a suboption of the AFTER and AT options. For its use and meaning, see the AFTER option.

TIME(hhmmss)

specifies the time of day at which data is to be transmitted to the terminals specified in the ROUTE command.

When using the C language, you are recommended to use the AFTER/AT HOURS, MINUTES, and SECONDS options as C does not provide a packed decimal data type. You may use TIME, but if the value specified is **not** an integer constant, the application is responsible for ensuring that the value passed to CICS is in packed decimal format.

TITLE(data-area)

specifies the data area that contains the title to be used with a routing logical message. This title appears as part of the response to a page query command. See the *CICS/ESA Application Programming Guide* for the format of the title option.

ROUTE conditions**INVERRTERM**

occurs if the terminal identifier specified in the ERRTERM option is not valid or is assigned to a type of terminal not supported by BMS.

Default action: terminate the task abnormally.

INVLDC

occurs if the specified LDC mnemonic is not included in the LDC list for the logical unit.

Default action: terminate the task abnormally.

INVREQ

occurs if a request for BMS services is not valid for any of the following reasons:

- Bytes 10 through 15 of a route list entry do not contain blanks
- Hours out of range (RESP2=4)
- Minutes out of range (RESP2=5)
- Seconds out of range (RESP2=6)
- BMS commands are not supported for distributed program link (RESP2=200).

Default action: terminate the task abnormally.

RTEFAIL

occurs in any of the following situations:

- A ROUTE command would result in the message being sent only to the terminal that initiated the transaction.
- A ROUTE command is issued against a remote shippable terminal that is not yet installed in the application-owning region.

Default action: return control to the application program at the point immediately following the ROUTE command.

RTESOME

occurs if any of the terminals specified by the ROUTE command options do not receive the message.

Default action: return control to the application program at the point immediately following the ROUTE command.

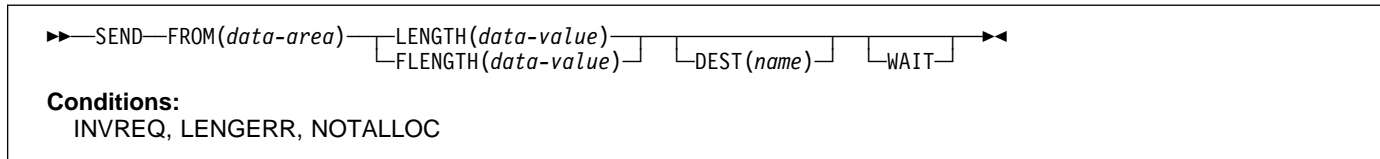
SEND (VTAM)

SEND (VTAM default)

Function

Write data to a standard CICS supported terminal (BTAM or TCAM).

Command syntax



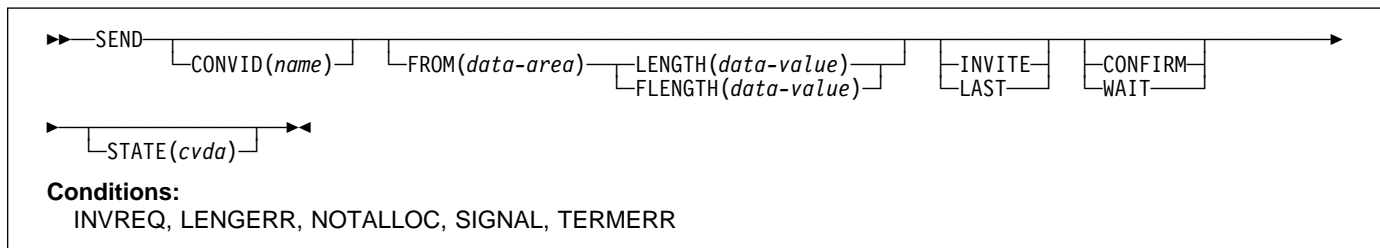
SEND writes data to a terminal. This form of the send command can be used by all CICS-supported terminals for which the other SEND descriptions are not appropriate.

SEND (APPC)

Function

Send data on an APPC mapped conversation.

Command syntax



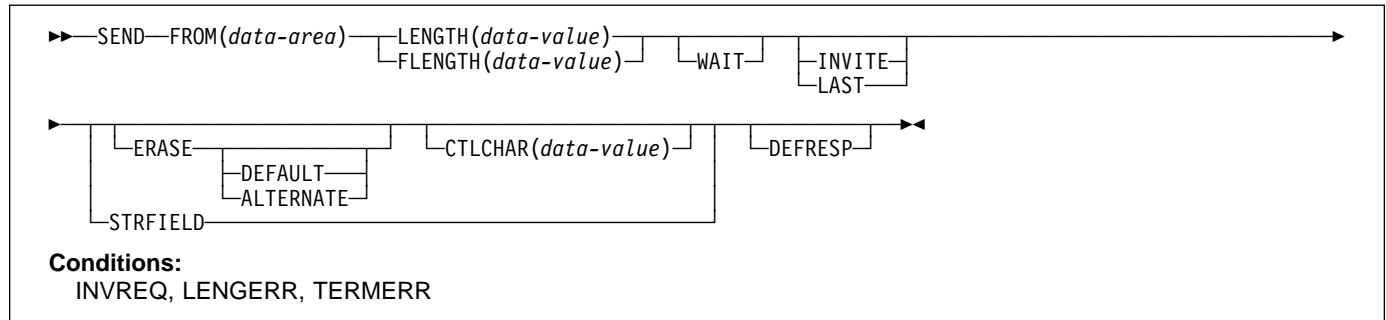
SEND sends data to a conversation partner on an APPC mapped conversation.

SEND (LUTYPE2/LUTYPE3)

Function

Write data to a 3270-display logical unit (LUTYPE2) or a 3270-printer logical unit (LUTYPE3).

Command syntax



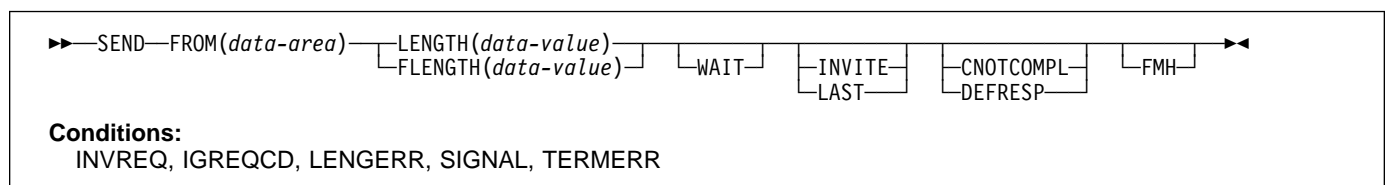
SEND writes data to a terminal.

SEND (LUTYPE4)

Function

Write data to a LUTYPE4 logical unit.

Command syntax



SEND writes data to a terminal.

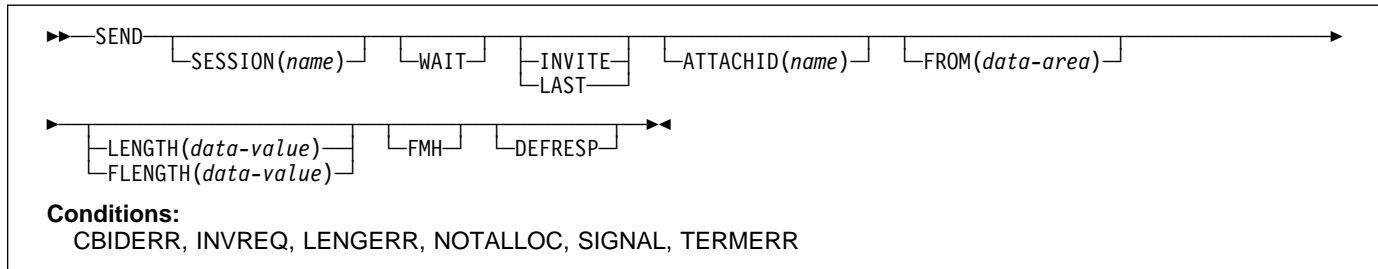
SEND (VTAM)

SEND (LUTYPE6.1)

Function

Send data on an LUTYPE6.1 conversation.

Command syntax



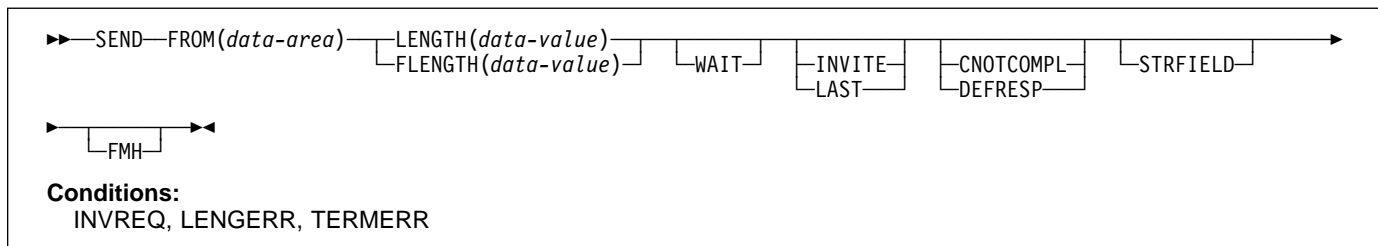
SEND sends data to a conversation partner on an LUTYPE6.1 conversation.

SEND (SCS)

Function

Write data to a 3270 SCS printer logical unit.

Command syntax



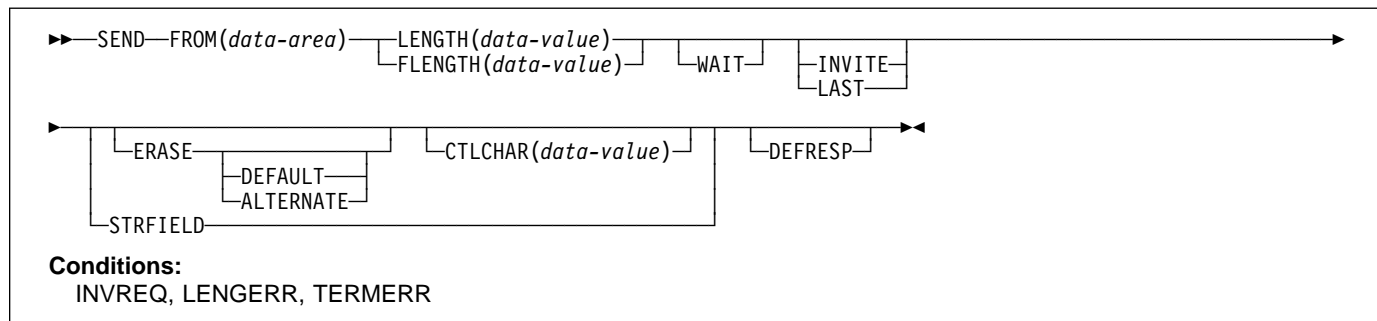
SEND writes data to a logical unit. The SCS printer logical unit accepts a character string as defined by Systems Network Architecture (SNA).

SEND (3270 logical)

Function

Write data to a 3270 logical unit.

Command syntax



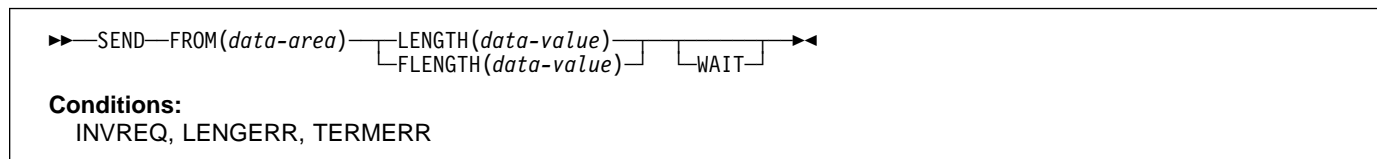
SEND writes data to a terminal.

SEND (3600 pipeline)

Function

Write data to a 3600 pipeline logical unit.

Command syntax



SEND writes data to a terminal.

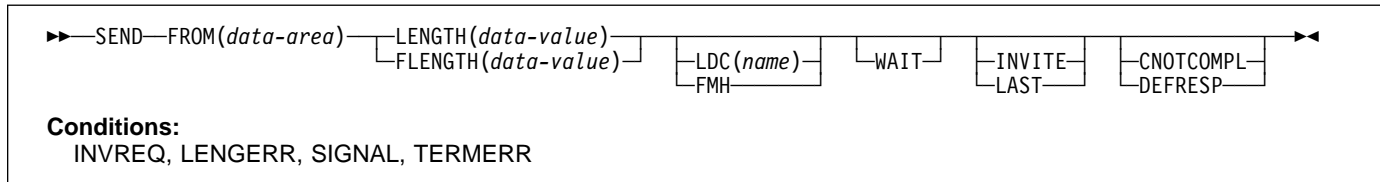
SEND (VTAM)

SEND (3600-3601)

Function

Write data to a 3600 (3601) logical unit.

Command syntax



SEND writes data to a terminal. This form of SEND also applies to the 4700 and the 3630 plant communication system.

A logical device code (LDC) is a code that can be included in an outbound FMH to specify the disposition of the data (for example, to which subsystem terminal it should be sent). Each code can be represented by a unique LDC mnemonic.

The installation can specify up to 256 2-character mnemonics for each TCTTE, and two or more TCTTEs can share a list of these mnemonics. Corresponding to each LDC mnemonic for each TCTTE is a numeric value (0 through 255).

A 3600 device and a logical page size are also associated with an LDC. "LDC" or "LDC value" is used in this book in reference to the code specified by the user. "LDC mnemonic" refers to the 2-character symbol that represents the LDC numeric value.

When the LDC option is specified, the numeric value associated with the mnemonic for the particular TCTTE, is inserted in the FMH. The numeric value associated with the LDC mnemonic is chosen by the installation, and is interpreted by the 3601 application program.

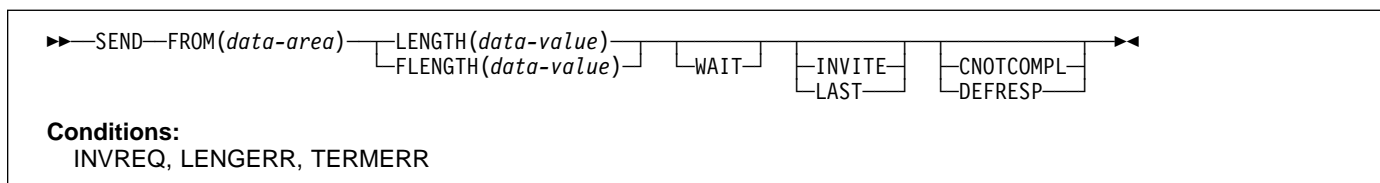
On output, the FMH can be built by the application program or by CICS. If your program supplies the FMH, you place it at the # front of your output data and specify the FMH option on your SEND command. If you omit the FMH option, CICS will provide # an FMH but you must reserve the first three bytes of the message for CICS to fill in.

SEND (3600-3614)

Function

Write data to a 3600 (3614) logical unit.

Command syntax



SEND writes data to a terminal. The data stream and communication format used between a CICS application program and a 3614 is determined by the 3614. The application program is therefore device dependent when handling 3614 communication.

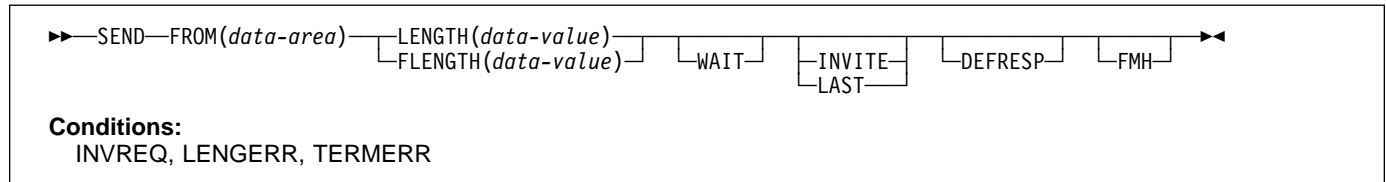
For further information about designing 3614 application programs for CICS, refer to the *CICS/OS/VS IBM 4700/3600/3630 Guide*.

SEND (3650 interpreter)

Function

Write data to a 3650 interpreter logical unit.

Command syntax



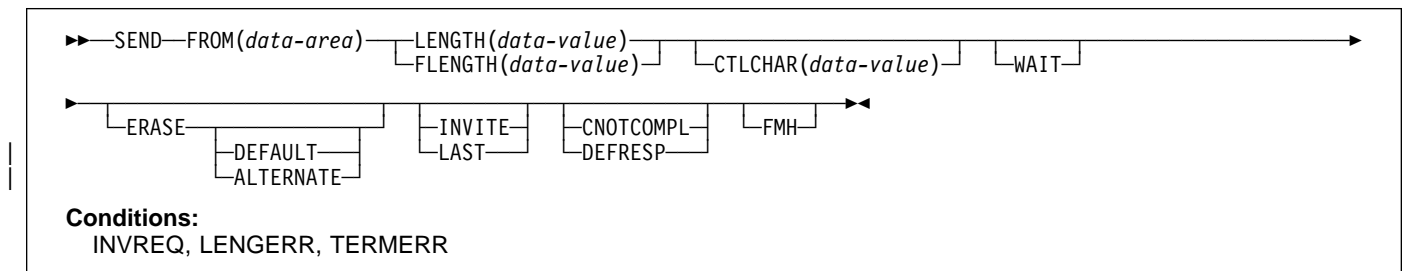
SEND writes data to a terminal.

SEND (3650-3270)

Function

Write data to a 3650 logical unit.

Command syntax



SEND writes data to a terminal.

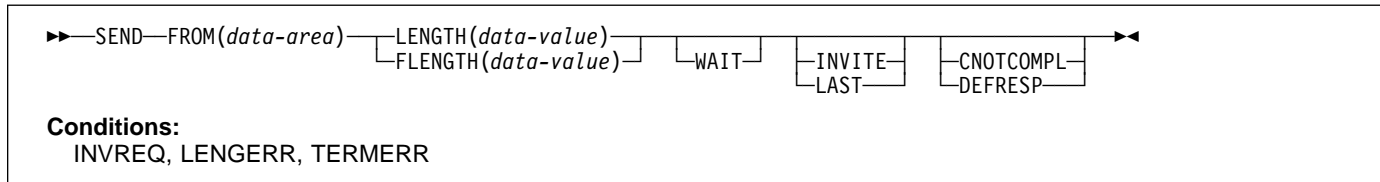
SEND (VTAM)

SEND (3650-3653)

Function

Write data to a 3650 (3653) logical unit.

Command syntax



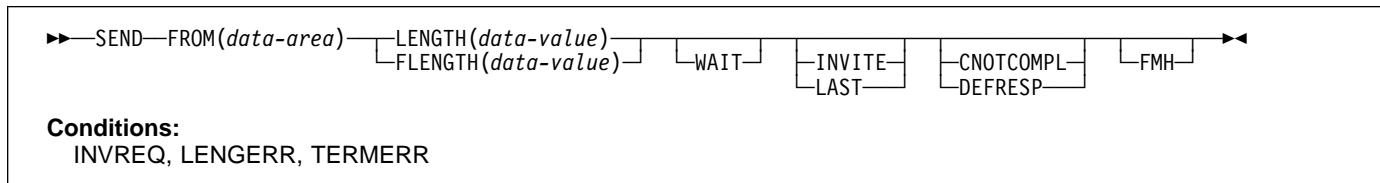
SEND writes data to a terminal.

SEND (3650-3680)

Function

Write data to a 3650 (3680) logical unit.

Command syntax



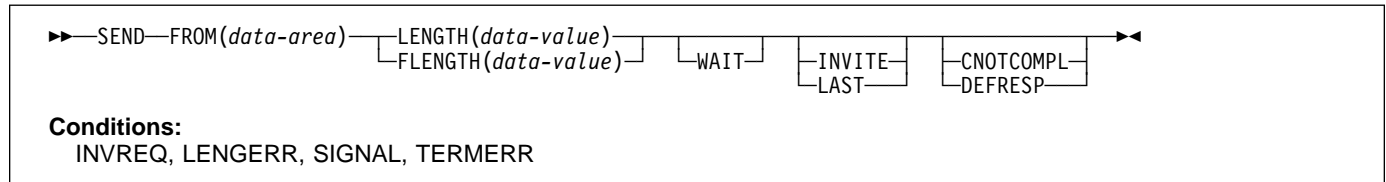
SEND writes data to a terminal.

SEND (3767)

Function

Write data to a 3767 interactive logical unit.

Command syntax



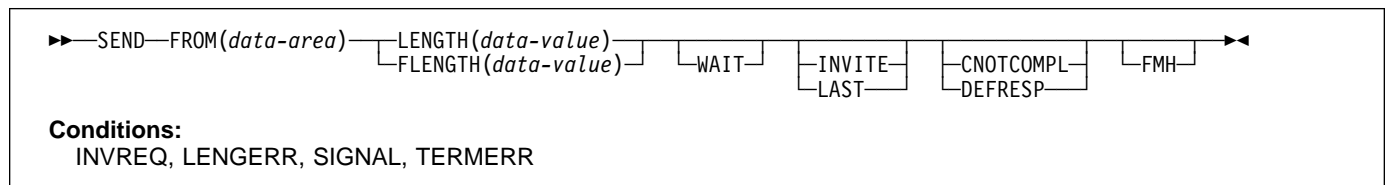
SEND writes data to a terminal. This form of SEND also applies to the 3770 interactive logical unit.

SEND (3770)

Function

Write data to a 3770 batch logical unit.

Command syntax



SEND writes data to a terminal.

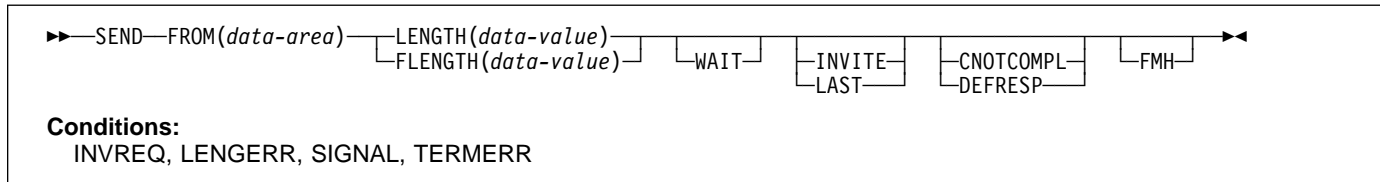
SEND (VTAM)

SEND (3790 full-function or inquiry)

Function

Write data to a 3790 full-function or inquiry logical unit.

Command syntax



SEND writes data to a terminal. This form of SEND also applies to the following devices:

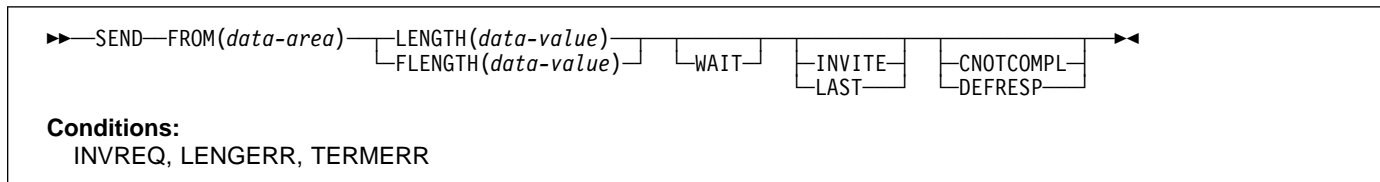
- 3650/3680 full-function logical unit
- 3770 full-function logical unit.

SEND (3790 SCS)

Function

Write data to a 3790 SCS printer logical unit.

Command syntax



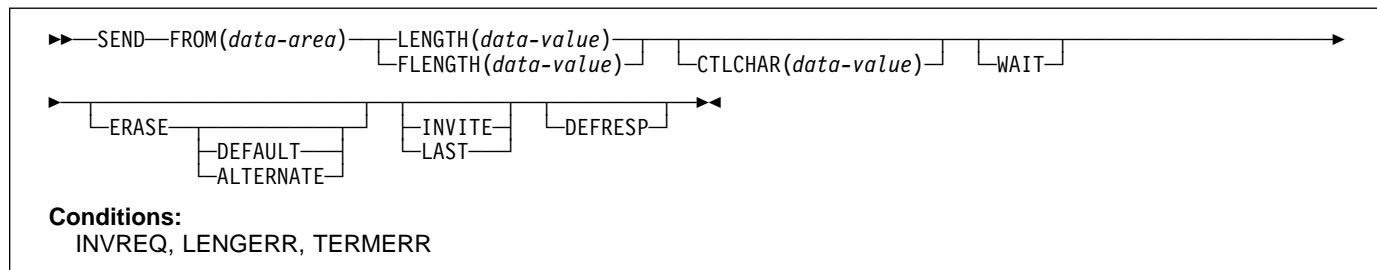
SEND writes data to a terminal.

SEND (3790 3270-display)

Function

Write data to a 3790 (3270-display) logical unit.

Command syntax



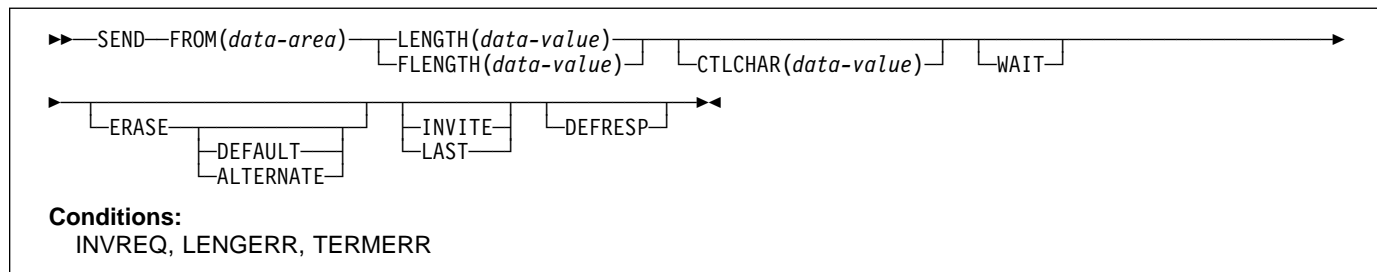
SEND writes data to a terminal.

SEND (3790 3270-printer)

Function

Write data to a 3790 (3270-printer) logical unit.

Command syntax



SEND writes data to a terminal.

SEND (VTAM)

SEND (VTAM) options

ALTERNATE

sets the terminal to use the ALTERNATE screen size.

ATTACHID(name)

specifies that an attach header (created by a BUILD ATTACH command) is to precede, and be concatenated with, the user data supplied in the FROM option. "name" (1–8 characters) identifies the attach header control block to be used in the local task.

CNOTCOMPL

indicates that the request/response unit (RU) sent as a result of this SEND command does not complete the chain. If this option is omitted and chain assembly has been specified, the RU terminates the chain.

CONFIRM

indicates that an application using a synchronization level 1 or 2 conversation requires a response from the remote application. A remote CICS application can respond positively by executing an ISSUE CONFIRMATION command, or negatively, by executing an ISSUE ERROR command, in which case the sending application has EIBERR and EIBERRCD set. CICS does not return control to the sending application until the response is received.

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

For compatibility with earlier releases, SESSION is accepted as a synonym for CONVID. New programs should use CONVID.

If this option is omitted, the principal facility is assumed.

CTLCHAR(data-value)

specifies a 1-byte write control character (WCC†) that controls a SEND command for a 3270. A COBOL user must specify a data area containing this character. If the option is omitted, all modified data tags are reset to zero and the keyboard is restored.

DEFAULT

sets the terminal to use the DEFAULT screen size.

DEFRESP

indicates that a definite response is required when the output operation has been completed.

DEST(name)

specifies the 4-byte symbolic name of the TCAM destination to which the message is to be sent. This

option is meaningful only for terminals defined using DFHTCT TYPE=SDSCI with DEVICE=TCAM.

If you use the DEST option, you must be aware of any restrictions placed on device-dependent data streams by the message control facility in use. See the programming information about the CICS-TCAM interface in the *CICS/ESA Customization Guide*.

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSIZE option in the profile definition.

FLENGTH(data-value)

A fullword alternative to LENGTH.

FMH

specifies that a function management header has been included in the data to be written. If the ATTACHID option is specified as well, the concatenated FMH flag is set in the attach FMH. The use of FMH is optional and is not supported for all terminal types. If not supplied, CICS takes no action, except for 3600/4700 terminals, where an FMH is mandatory. In this case, if FMH is not specified, CICS supplies one and places it in the first 3 bytes of the message, which you must reserve for this purpose.

FROM(data-area)

FROM(data-area)

specifies the data to be written to the logical unit, or a partner transaction.

INVITE

allows an application program to add control data to data already sent to a process in a connected APPC system. Control data is not transmitted by CICS until the subsequent execution of a WAIT or a SYNCPOINT command, unless CONFIRM or WAIT is also specified; or specifies that the next terminal control command to be executed for this facility is a RECEIVE, this allows optimal flows to occur.

LAST

specifies that this is the last SEND command for a transaction.

† Documented in the *IBM 3270 Data Stream Programmer's Reference*.

LDC(name)

specifies the 2-character mnemonic used to determine the appropriate logical device code (LDC) numeric value. The mnemonic represents an LDC entry in the terminal control table TYPE=LDC.

LENGTH(data-value)

specifies the length, as a halfword binary value, of the data to be written. For a description of a safe upper limit, see "LENGTH options" on page 5.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If this option is omitted, the principal facility for the task is used.

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
 CONFFREE
 CONFRECEIVE
 CONFSEND
 FREE
 PENDFREE
 PENDRECEIVE
 RECEIVE
 ROLLBACK
 SEND
 SYNCFREE
 SYNCRECEIVE
 SYNCSEND

STRFIELD

specifies that the data area specified in the FROM option contains structured fields. If this option is specified, the contents of all structured fields must be handled by the application program. The CONVERSE command, rather than a SEND command, must be used if the data area contains a read partition structured field. (Structured fields are described in the *CICS/ESA 3270 Data Stream Device Guide*.)

CTLCHAR and ERASE are mutually exclusive with STRFIELD, and their use with STRFIELD generates an error message.

WAIT

specifies that processing of the command must be completed before any subsequent processing is attempted.

If the WAIT option is not specified, control is returned to the application program when processing of the command has started. A subsequent input or output request (terminal control, BMS, or batch data interchange) to the terminal associated with the task causes the application program to wait until the previous request has been completed.

SEND (VTAM) conditions

Some of the following conditions may occur in combination. If more than one occurs, only the first is passed to the application program.

EIBRCODE, however, is set to indicate all the conditions that occurred.

CBIDERR

occurs if the requested attach header control block named in ATTACHID cannot be found.

Default action: terminate the task abnormally.

IGREQCD

occurs when an attempt is made to execute a SEND command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from the logical unit.

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- The CONFIRM option has been specified, but the APPC conversation is not sync level 1 or 2.
- The SEND command has been used on an APPC conversation that is not a mapped conversation or that is not using the EXEC CICS interface.
- A distributed program link server application attempted to send on its function-shipping session (its principal facility) (RESP2=200).
- For SEND (APPC), a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs if an out-of-range value is supplied in the LENGTH or FLENGTH option.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the CONVID value in the command does not relate to a conversation that is owned by the application, or if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

SIGNAL

occurs when an inbound SIGNAL data-flow control command has been received from a logical unit or session. EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

SEND (VTAM)

TERMERR

occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

SEND (non-VTAM)

SEND (System/3)

Function

Write data to a System/3.

Command syntax

```
▶▶—SEND—FROM(data-area)— $\left\{ \begin{array}{l} \text{LENGTH}(\textit{data-value}) \\ \text{FLENGTH}(\textit{data-value}) \end{array} \right\}$ — $\left\{ \begin{array}{l} \text{DEST}(\textit{name}) \\ \text{WAIT} \\ \text{ASIS} \\ \text{CNOTCOMPL} \end{array} \right\}$ ▶▶
```

Conditions:

INVREQ, LENGERR

SEND writes data to a terminal. This form of the SEND command also applies to the following devices:

- 2770 data communication system
- 2780 data transmission terminal
- 3660 supermarket scanning system
- 3780 communication terminal.

SEND (System/7)

Function

Write data to a System/7.

Command syntax

```
▶▶—SEND—FROM(data-area)— $\left\{ \begin{array}{l} \text{LENGTH}(\textit{data-value}) \\ \text{FLENGTH}(\textit{data-value}) \end{array} \right\}$ — $\left\{ \begin{array}{l} \text{DEST}(\textit{name}) \\ \text{WAIT} \\ \text{PSEUDOBIN} \\ \text{ASIS} \end{array} \right\}$ ▶▶
```

Conditions:

INVREQ, LENGERR

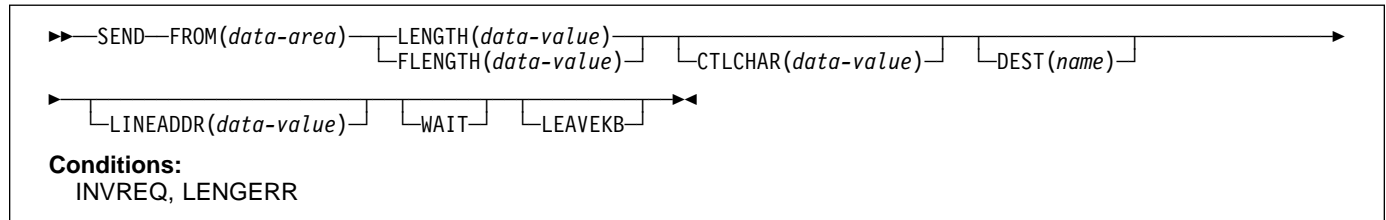
SEND writes data to a terminal. For a description of running with a System/7, see “CONVERSE (System/7)” on page 54.

SEND (2260)

Function

Write data to a 2260 or 2265 display station.

Command syntax



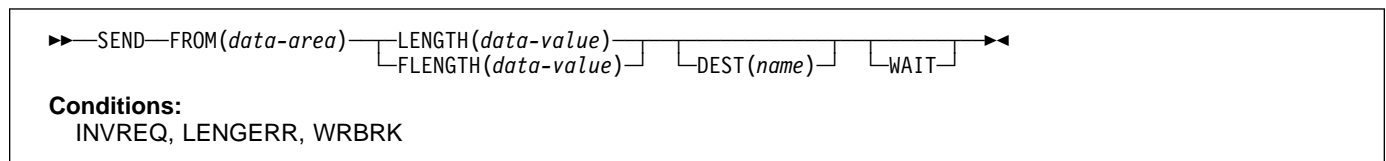
SEND writes data to a terminal.

SEND (2741)

Function

Write data to a 2741 communication terminal.

Command syntax



SEND writes data to a terminal. For more information about using a 2741 communication terminal, see "CONVERSE (2741)" on page 56.

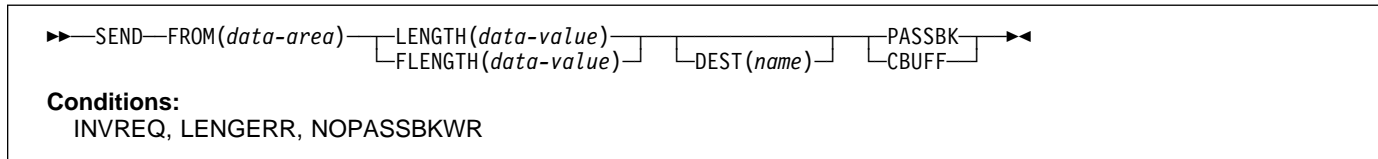
SEND (non-VTAM)

SEND (2980)

Function

Write data to a 2980 general banking terminal system.

Command syntax



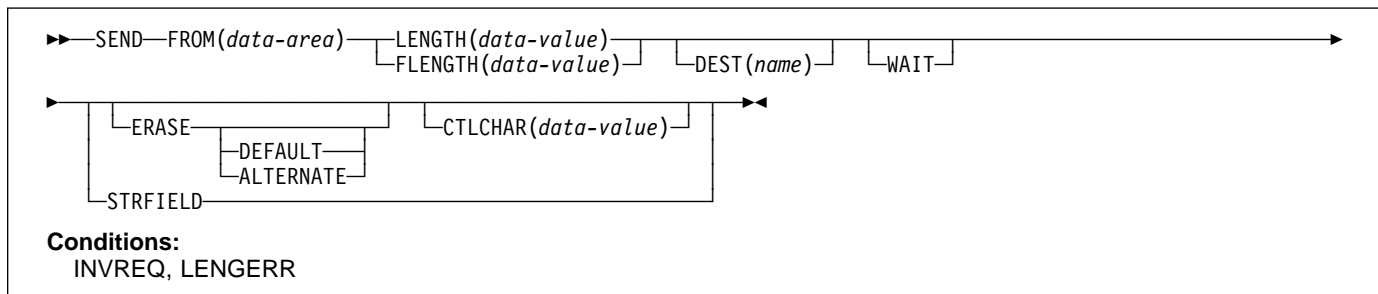
SEND writes data to a terminal. For more information about the 2980 general banking system, see "RECEIVE (2980)" on page 213.

SEND (3270 display)

Function

Write data to a 3270 information display system (BTAM or TCAM).

Command syntax



SEND writes data to a terminal.

SEND (3600 BTAM)

Function

| Write data to a 3600 or 4700 finance communication system (BTAM) - only valid for remote terminals. See "BTAM programmable terminals" on page 358.

Command syntax

```

▶▶—SEND—FROM(data-area)—LENGTH(data-value)—
└───┬───┬───┬───┬───┬───┬───┬───┬───┬───▶
      FLENGTH(data-value) WAIT ASIS

```

Conditions:
INVREQ, LENGERR

| SEND writes data to a terminal. This form of the SEND command also applies to the 4700 finance communication system.
| For more information about using the 3600 or 4700 finance communication system, see "CONVERSE (3600 BTAM)" on page 58.

SEND (3735)

Function

Write data to a 3735 Programmable Buffered Terminal.

Command syntax

```

▶▶—SEND—FROM(data-area)—LENGTH(data-value)—
└───┬───┬───┬───┬───┬───┬───┬───┬───┬───▶
      FLENGTH(data-value) WAIT ASIS

```

Conditions:
INVREQ, LENGERR

SEND writes data to a terminal. The 3735 Programmable Buffered Terminal may be serviced by CICS in response to terminal-initiated input (automatic answering), or as a result of an automatic (automatic calling) or time-initiated transaction.

For more information about the 3735, see "CONVERSE (3735)" on page 60.

SEND (non-VTAM)

SEND (3740)

Function

Write data to a 3740 data entry system.

Command syntax

```
▶—SEND—FROM(data-area)—LENGTH(data-value)—▶  
└──FLENGTH(data-value)└──WAIT└──ASIS└──▶
```

Conditions:

INVREQ, LENGERR

SEND writes data to a terminal. In batch mode, many files are exchanged between the 3740 and CICS in a single transmission. The transmission of an input batch must be complete before an output transmission can be started.

SEND (non-VTAM) options

ALTERNATE

sets the terminal to use the ALTERNATE screen size.

ASIS

indicates that output is to be sent in transparent mode (with no recognition of control characters and accepting any of the 256 possible combinations of eight bits as valid transmittable data).

Note: If you are using a katakana terminal, you might see some messages containing mixed English and katakana characters. That is because katakana terminals cannot display mixed-case output. Uppercase characters in the data stream appear as uppercase English characters, but lowercase characters appear as katakana characters. If this happens, ask your system programmer to specify MSGCASE=UPPER in the system initialization parameters so that messages contain uppercase characters only.

ATTACHID(name)

specifies that an attach header (created by a BUILD ATTACH command) is to precede, and be concatenated with, the user data supplied in the FROM option. "name" (1–8 characters) identifies the attach header control block to be used in the local task.

CBUFF

specifies that data is to be written to a common buffer in a 2972 control unit. The WAIT option is implied.

CNOTCOMPL

indicates that the request/response unit (RU) sent as a result of this SEND command does not complete the chain. If this option is omitted and chain assembly has been specified, the RU terminates the chain.

CTLCHAR(data-value)

specifies a 1-byte write control character (WCC†) that controls a SEND command for a 3270. A COBOL user must specify a data area containing this character. If the option is omitted, all modified data tags are reset to zero and the keyboard is restored.

DEFAULT

sets the terminal to use the DEFAULT screen size.

DEFRESP

indicates that a definite response is required when the output operation has been completed.

DEST(name)

specifies the 4-byte symbolic name of the TCAM destination to which the message is to be sent. This option is meaningful only for terminals defined using DFHTCT TYPE=SDSCI with DEVICE=TCAM.

If you use the DEST option, you must be aware of any restrictions placed on device-dependent data streams by the message control facility in use. See the programming information about the CICS-TCAM interface in the *CICS/ESA Customization Guide*

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSIZE option in the profile definition.

FLENGTH(data-value)

A fullword alternative to LENGTH.

FMH

specifies that a function management header has been included in the data to be written. If the ATTACHID option is specified as well, the concatenated FMH flag is set in the attach FMH.

FROM(data-area)

specifies the data to be written to the logical unit or terminal.

INVITE

specifies that the next terminal control command to be executed for this facility is a RECEIVE. This allows optimal flows to occur.

LAST

specifies that this is the last output operation for a transaction and therefore the end of a bracket.

LEAVEKB

specifies that the keyboard is to remain locked at the completion of the data transfer.

LENGTH(data-value)

specifies the length, as a halfword binary value, of the data to be written. For a description of a safe upper limit, see "LENGTH options" on page 5.

LINEADDR(data-value)

specifies that the writing is to begin on a specific line of a 2260/2265 screen. The data value is a halfword binary value in the range 1 through 12 for a 2260, or 1 through 15 for a 2265.

† Documented in the *IBM 3270 Data Stream Programmer's Reference*.

SEND (non-VTAM)

PASSBK

specifies that communication is with a passbook. The WAIT option is implied.

PSEUDOBIN (start-stop only)

specifies that the data being written is to be translated from System/7 hexadecimal to pseudobinary.

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If this option is omitted, the principal facility for the task is used.

STATE(cvda)

gets the state of the transaction program. The cvda values returned by CICS are:

- ALLOCATED
- FREE
- PENDFREE
- RECEIVE
- ROLLBACK
- SEND
- SYNCFREE
- SYNCRECEIVE
- SYNCSEND

STRFIELD

specifies that the data area specified in the FROM option contains structured fields. If this option is specified, the contents of all structured fields must be handled by the application program. The CONVERSE command, rather than a SEND command, must be used if the data area contains a read partition structured field. (Structured fields are described in the *CICS/ESA 3270 Data Stream Device Guide*.)

CTLCHAR and ERASE are mutually exclusive with STRFIELD, and their use with STRFIELD generates an error message.

WAIT

specifies that processing of the command must be completed before any subsequent processing is attempted.

If the WAIT option is not specified, control is returned to the application program when processing of the command has started. A subsequent input or output request (terminal control, BMS, or batch data interchange) to the terminal associated with the task causes the application program to wait until the previous request has been completed.

SEND (non-VTAM) conditions

CBIDERR

occurs if the requested attach header control block named in ATTACHID cannot be found.

Default action: terminate the task abnormally.

INVREQ

| occurs if a distributed program link server application
| attempted to send on its function-shipping session (its
| principal facility) (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

| occurs if an out-of-range value is supplied in the
| LENGTH or FLENGTH option.

Default action: terminate the task abnormally.

NOPASSBKWR

occurs if no passbook is present.

| Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

TERMERR

occurs for a terminal-related error, such as a session failure. This condition applies to VTAM-connected terminals only.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program (DFHZNAC) handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

WRBRK

occurs if the command is terminated by the attention key.

Default action: ignore the condition.

SEND CONTROL

SEND CONTROL

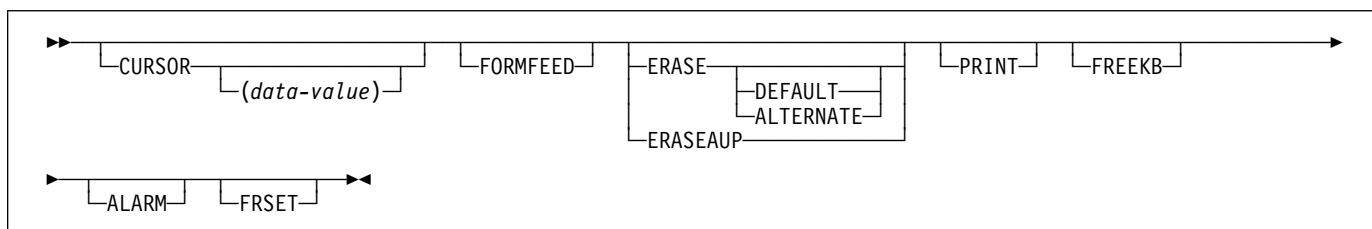
Function

Send device controls to a terminal without map or text data. The keywords are separated into those supported by minimum, standard, and full BMS. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

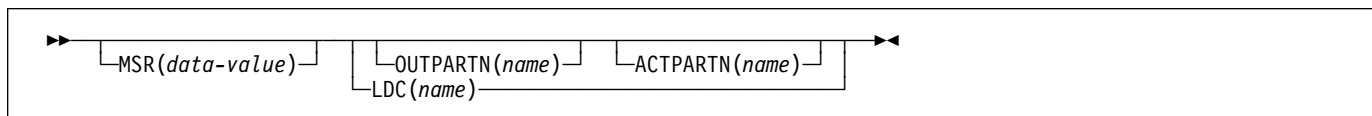
Command syntax

▶▶SEND CONTROL◀◀

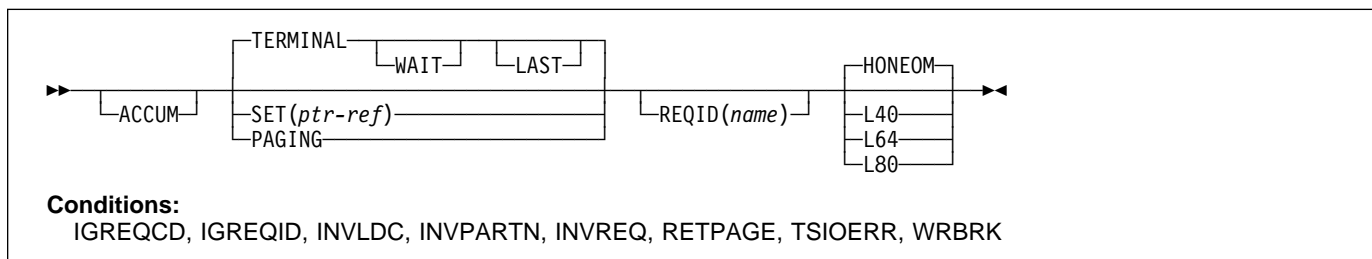
Minimum BMS:



Standard BMS:



Full BMS:



SEND CONTROL sends device controls to a terminal.

When using the SEND CONTROL command with any of the ALARM, FREEKB, FRSET, HONEOM, L40, L64, L80, or PRINT options, see CTRL on page 391 for a description of the option priority.

SEND CONTROL options

ACCUM

specifies that this command is one of a number of commands that are used to build a logical message. The logical message is completed by a SEND PAGE command, or deleted by a PURGE MESSAGE command.

ACTPARTN(name)

specifies the name (1–2 characters) of the partition to be activated. Activating a partition moves the cursor into the specified partition, and unlocks the keyboard for the specified partition.

This option is ignored if the target terminal does not support partitions, or if there is no application partition set.

ALARM

specifies that the 3270 audible alarm feature is to be activated. For logical units supporting FMHs (except interactive and batch logical units), ALARM instructs BMS to set the alarm flag in the FMH.

ALTERNATE

sets the terminal to use the ALTERNATE screen size.

CURSOR(data-value)

specifies the location to which the 3270 or 3604 cursor is to be returned on completion of a SEND CONTROL command.

The data value must be a halfword binary value that specifies the cursor position relative to zero; the range of values that can be specified depends on the size of the screen being used.

If ACCUM is being used, the most recent value of CURSOR specified is used to position the cursor.

The value specified in the CURSOR option must be positive. A negative value leads to unpredictable results.

If this option is omitted, the cursor is positioned at position zero of the screen.

DEFAULT

sets the terminal to use the DEFAULT screen size.

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSIZE option in the profile definition.

ERASEAUP

specifies that all unprotected character locations in the partition or the entire screen are to be erased. (This option applies only to the 3270 and 8775.)

FORMFEED

specifies that a new page is required. For 3270 printers and displays, the FORMFEED character is positioned at the start of the buffer. The application program must thus ensure that this buffer position is not overwritten by map or text data. It is ignored if the target terminal does not support FORMFEED (that is, the RDO option FORMFEED was not used, or the terminal control table TYPE=TERMINAL does not specify FF=YES).

FREEKB

specifies that the 3270 keyboard is to be unlocked. If FREEKB is omitted, the keyboard remains locked.

Note that the keyboard lock status is maintained separately for each partition on a terminal that supports partitions.

FRSET

specifies that the modified data tags (MDTs) of all fields currently in the 3270 (or partition) buffer are to be reset to the unmodified condition (that is, field reset).

This allows the ATTRB operand of DFHMDF for the next requested map to control the final status of fields written or rewritten in response to a BMS command, if no other attribute information has been written in the symbolic map.

HONEOM

specifies that the default printer line length is to be used. This length should be the same as that specified using the RDO options PAGESIZE or ALTPAGE, or PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL, and the same as the printer platen width; otherwise the data may not format correctly.

LAST

specifies that this is the last output operation for a transaction and, therefore, the end of a bracket. This option applies to logical units only.

LDC(name)

specifies a 2-character mnemonic to be used to determine the logical device code (LDC) to be transmitted in the FMH to the logical unit. The mnemonic identifies an LDC entry specified in the terminal control table TYPE=LDC.

When an LDC is specified, BMS uses the device type, the page size, and the page status associated with the LDC mnemonic to format the message. These values are taken from the extended local LDC table for the LU, if it has one. If the LU has only a local (unextended) LDC table, the values are taken from the system LDC table. The numeric value of the LDC is obtained from the local LDC table, unless this is an unextended table and the value is not specified, in which case it is taken from the system table.

L40, L64, or L80

specifies the line length for a 3270 printer; a carrier return and line feed are forced after 40, 64, or 80 characters have been printed on a line. Unacceptable results are likely if this differs from the page width specified by the RDO options PAGESIZE or ALTPAGE, or the PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL.

When using the options, refer to CTRL on page 391 for a description of the option priority.

MSR(data-value)

specifies the 4-byte data value that controls the 10/63 magnetic stripe reader attached to an 8775 or 3643 terminal. A set of constants is provided in DFHMSTRCA to assist in setting this 4-byte area. See "Magnetic slot reader (MSR) control value constants, DFHMSTRCA" on page 377 for a complete list. This option is ignored if the RDO option MSRCNTRL was not used, or if the

SEND CONTROL

terminal's control table does not specify
FEATURE=MSRCNTRL.

OUTPARTN(name)

specifies the name (1–2 characters) of the partition to which data is to be sent. This option is ignored if the terminal does not support partitions, or if there is no application partition set associated with the terminal. If there is an application partition set, and the OUTPARTN option is omitted, data is sent to the partition named by the PARTN operand of the DFHMSD (see “DFHMSD” on page 396) or the DFHMDI (see “DFHMDI” on page 390) map definition macros. If maps are not used, or if there is no PARTN operand, the output is sent to the first partition in the partition set.

PAGING

specifies that the output data is not to be sent immediately to the terminal, but is to be placed in temporary storage and displayed in response to paging commands entered by the terminal operator.

If PAGING is specified with a REQID that is defined as recoverable in the temporary storage table (TST), CICS provides message recovery for logical messages if the task has reached a syncpoint.

PRINT

specifies that a print operation is to be started at a 3270 printer or at a 3275 with the printer adapter feature, or that data on an LUTYPE2 (3274/76 or 3790) is to be printed on a printer allocated by the controller. If this option is omitted, the data is sent to the printer buffer but is not printed.

REQID(name)

specifies a 2-character prefix to be used as part of a temporary storage identifier for CICS message recovery. Only one prefix can be specified for each logical message. The default prefix is **.

BMS message recovery is provided for a logical message only if the PAGING option is specified in the BMS SEND commands, and if the syncpoint has been reached.

SET(ptr-ref)

specifies the pointer to be set to the address of the output data.

The SET option specifies that completed pages are to be returned to the application program. The pointer is set to the address of a list of completed pages. See the description of the SET option in the section about full-function BMS in the *CICS/ESA Application Programming Guide* for more guidance on using the SET option.

If TIOAPFX=YES is specified in the map definition, the pointer returned contains the address of the TIOA prefix. The user data starts at offset X'0C' from the start of the TIOA prefix.

TERMINAL

specifies that the output data is to be sent to the terminal that originated the transaction.

WAIT

specifies that control should not be returned to the application program until the output operation has been completed.

If WAIT is not specified, control returns to the application program when the output operation has started. A subsequent input or output command (terminal control, BMS, or batch data interchange) causes the application program to wait until the previous command has been completed.

SEND CONTROL conditions

IGREQCD

occurs when an attempt is made to execute a SEND CONTROL command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

IGREQID

occurs if the prefix specified in the REQID option is different from that established by a previous REQID option, or by default for this logical message—REQID (**).

Default action: terminate the task abnormally.

INVLDC

occurs if the specified LDC mnemonic is not included in the LDC list for the logical unit.

Default action: terminate the task abnormally.

INVPARTN

occurs if the specified partition is not defined in the partition set associated with the application program.

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- Control information is output to the same partition or LDC as mapped data while a BMS logical message is active. If neither partitions nor LDCs are in use, control information is output to the same device as mapped data.
- A distributed program link server application attempted to send on its function-shipping session (its principal facility) (RESP2=200).

Default action: terminate the task abnormally.

RETPAGE

occurs if the SET option is specified and a completed page is ready for return to the application program.

Default action: return control to the application program at the point immediately following the BMS SEND command.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

WRBRK

occurs if the command is interrupted by the terminal operator pressing the ATTN key. It applies only to the 2741 Communication Terminal, and only if write break is supported for CICS.

Default action: ignore the condition.

SEND MAP

SEND MAP

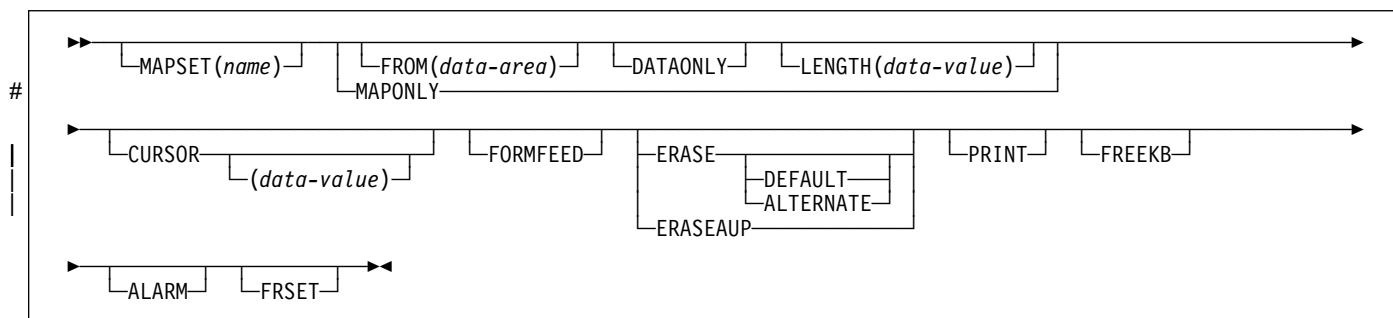
Function

Send mapped output data to a terminal. The keywords are separated into those supported by minimum, standard, and full BMS. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

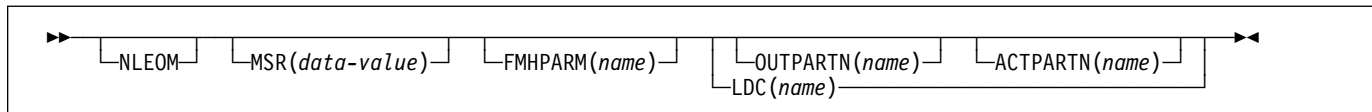
Command syntax

```
➔➔SEND MAP(name)➔➔
```

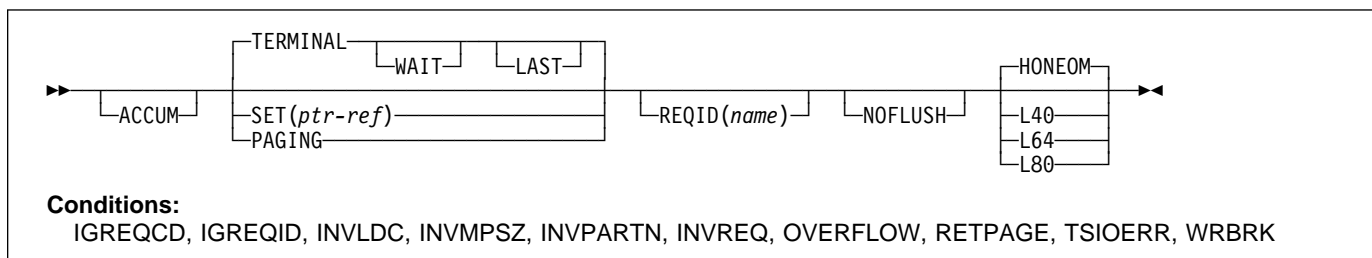
Minimum BMS:



Standard BMS:



Full BMS:



Apar 72253

Documentation for Apar 72253 added 16 Jun 1995 (TUCKER)

SEND MAP sends output data to a terminal.

When using the SEND MAP command with any of the ALARM, FREEKB, FRSET, HONEOM, L40, L64, L80, or PRINT options, see CTRL on page 391 for a description of the option priority.

See Appendix K, "BMS macro summary" on page 381 for map definition.

SEND MAP options

ACCUM

specifies that this command is one of a number of commands that are used to build a logical message. The logical message is completed by a SEND PAGE command, or deleted by a PURGE MESSAGE command.

ACTPARTN(name)

specifies the name (1–2 characters) of the partition to be activated. Activating a partition moves the cursor into the specified partition, and unlocks the keyboard for the specified partition.

This option is ignored if the target terminal does not support partitions, or if there is no application partition set.

ALARM

specifies that the 3270 audible alarm feature is to be activated. For logical units supporting FMHs (except interactive and batch logical units), ALARM instructs BMS to set the alarm flag in the FMH.

When using the ALARM option, refer to CTRL on page 391 for a description of the option priority.

ALTERNATE

sets the terminal to use the ALTERNATE screen size.

CURSOR(data-value)

specifies the location to which the 3270 or 3604 cursor is to be returned upon completion of a SEND MAP command.

The data value must be a halfword binary value that specifies the cursor position relative to zero; the range of values that can be specified depends on the size of the screen being used. If no data value is specified, symbolic cursor positioning is assumed. See the section about minimum BMS in the *CICS/ESA Application Programming Guide* for more information about symbolic cursor positioning.

This option overrides any IC option of the ATTRB operand of DFHMDF. If ACCUM is being used, the most recent value of CURSOR specified is used to position the cursor.

The value specified in the CURSOR option must be positive. A negative value leads to unpredictable results.

DATAONLY

specifies that only application program data is to be written. The attribute characters (3270 only) must be specified for each field in the supplied data. If the attribute byte in the user-supplied data is set to X'00', the attribute byte on the screen is unchanged. Any default data or attributes from the map are ignored.

DEFAULT

sets the terminal to use the DEFAULT screen size.

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures

that the correct screen size is selected, as defined for the transaction by the SCRNSIZE option in the profile definition.

ERASEAUP

specifies that before this page of output is displayed, all unprotected character locations in the partition or the entire screen are to be erased. (This option applies only to the 3270 and 8775.)

FMHPARM(name)

specifies the name (1–8 characters) of the outboard map to be used. (This option applies only to 3650 logical units with outboard formatting.)

FORMFEED

specifies that a new page is required. For 3270 printers and displays, the FORMFEED character is positioned at the start of the buffer. The application program must thus ensure that this buffer position is not overwritten by map or text data. It is ignored if the target terminal does not support FORMFEED (that is, the RDO option FORMFEED was not used, or the terminal control table TYPE=TERMINAL does not specify FF=YES).

FREEKB

specifies that the 3270 keyboard should be unlocked after the data is written. If FREEKB is omitted, the keyboard remains locked.

Note that the keyboard lock status is maintained separately for each partition on a terminal that supports partitions.

When using the FREEKB option, refer to CTRL on page 391 for a description of the option priority.

FROM(data-area)

specifies the data area containing the data to be processed. If this field is not specified, the name defaults to the name of the map suffixed with an O. This includes the 12-byte prefix generated by the TIOAPFX=YES option on the DFHMDF and DFHMDS BMS map definitions (see pages 395 and 401).

FRSET

specifies that the modified data tags (MDTs) of all fields currently in the 3270 (or partition) buffer are to be reset to the unmodified condition (that is, field reset) before any map data is written to the buffer.

This allows the ATTRB operand of DFHMDF for the requested map to control the final status of fields written or rewritten in response to a BMS command, if no other attribute information has been written in the symbolic map.

When using the FRSET option refer to CTRL on page 391 for a description of the option priority.

HONEOM

specifies that the default printer line length is to be used. This length should be the same as that specified using the RDO options PAGESIZE or ALTPAGE, or PGESIZE or ALTPGE in the terminal control table

SEND MAP

TYPE=TERMINAL, and the same as the printer platen width; otherwise the data may not format correctly.

When using the HONEOM option, refer to CTRL on page 391 for a description of the option priority.

LAST

specifies that this is the last output operation for a transaction and, therefore, the end of a bracket. This option applies to logical units only.

LDC(name)

specifies a 2-character mnemonic to be used to determine the logical device code (LDC) to be transmitted in the FMH to the logical unit. The mnemonic identifies an LDC entry specified in the terminal control table TYPE=LDC.

When an LDC is specified, BMS uses the device type, the page size, and the page status associated with the LDC mnemonic to format the message. These values are taken from the extended local LDC table for the logical unit, if it has one. If the logical unit has only a local (unextended) LDC table, the values are taken from the system LDC table. The numeric value of the LDC is obtained from the local LDC table, unless this is an unextended table and the value is not specified, in which case it is taken from the system table.

If the LDC option is omitted, the LDC mnemonic specified in the DFHMSD macro is used; see "DFHMSD" on page 396. If the LDC option has also been omitted from the DFHMSD macro, the action depends on the type of logical unit, as follows:

3601 logical unit

The first entry in the local or extended local LDC table is used, if there is one. If a default cannot be obtained in this way, a null LDC numeric value (X'00') is used. The page size used is the value that is specified in the terminal control table TYPE=TERMINAL, or (1,40) if such a value is not specified.

LUTYPE4 logical unit, batch logical unit, or batch data interchange logical unit

The local LDC table is not used to supply a default LDC; instead, the message is directed to the logical unit console (that is, to any medium that the logical unit elects to receive such messages). For a batch data interchange logical unit, this does not imply sending an LDC in an FMH. The page size is obtained in the manner described for the 3601 logical unit.

LENGTH(data-value)

specifies the length of the data to be formatted as a halfword binary value.

If the data area sending the map is longer than the data to be mapped, LENGTH should be specified. This should include the length of the 12-byte prefix generated

by the TIOAPFX=YES option on the DFHMDI and DFHMSD BMS map definitions (see pages 395 and 401). For a description of a safe upper limit, see "LENGTH options" on page 5.

L40, L64, or L80

specifies the line length for a 3270 printer; a carrier return and line feed are forced after 40, 64, or 80 characters have been printed on a line. Unacceptable results are likely if this differs from the page width specified by the RDO options PAGESIZE or ALTPAGE, or the PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL.

When using the options, refer to CTRL on page 391 for a description of the option priority.

MAP(name)

specifies the name (1–7 characters) of the map to be used.

MAPONLY

specifies that only default data from the map is to be written.

MAPSET(name)

specifies the unsuffixed name (1–7 characters) of the mapset to be used. The mapset must reside in the CICS program library. The mapset can be defined either by using RDO or by program autoinstall when the mapset is first used. If this option is not specified, the name given in the MAP option is assumed to be that of the mapset.

The number of maps per mapset is limited to a maximum of 9998.

MSR(data-value)

specifies the 4-byte data value that controls the 10/63 magnetic stripe reader attached to an 8775 or 3643 terminal. A set of constants is provided in DFHMSRCA to assist in setting this 4-byte area. See "Magnetic slot reader (MSR) control value constants, DFHMSRCA" on page 377 for a complete list. This option is ignored if the RDO option MSRCNTRL was not used, or if the terminal's control table does not specify FEATURE=MSRCNTRL.

NLEOM

specifies that data for a 3270 printer or a 3275 display with the printer adapter feature should be built with blanks and new-line (NL) characters, and that an end-of-message (EM) character should be placed at the end of the data. As the data is printed, each NL character causes printing to continue on the next line, and the EM character terminates printing.

This option must be specified in the first SEND MAP command used to build a logical message. The option is ignored if the device receiving the message (direct or routed) is not one of those mentioned above.

If this option is used, buffer updating and attribute modification of fields previously written into the buffer are

not allowed. CICS includes the ERASE option with every write to the terminal.

The NL character occupies a buffer position. A number of buffer positions, equivalent to the value of the RDO options PAGESIZE or ALTPAGE, or PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL for that terminal, is unavailable for data. This may cause data to wrap around in the buffer; if this occurs, the PGESIZE value must be reduced.

The NLEOM option overrides the ALARM option if the latter is present.

NOFLUSH

specifies that CICS does not clear pages on completion but returns control to the program (having set the OVERFLOW condition in EIBRESP).

OUTPARTN(name)

specifies the name (1–2 characters) of the partition to which data is to be sent. This option is ignored if the terminal does not support partitions, or if there is no application partition set associated with the terminal. If there is an application partition set, and the OUTPARTN option is omitted, data is sent to the partition named by the PARTN operand of the DFHMSD or DFHMDI map definitions. If maps are not used, or if there is no PARTN operand, the output is sent to the first partition in the partition set.

PAGING

specifies that the output data is not to be sent immediately to the terminal, but is to be placed in temporary storage and displayed in response to paging commands entered by the terminal operator.

If PAGING is specified with a REQID that is defined as recoverable in the temporary storage table (TST), CICS provides message recovery for logical messages if the task has reached a syncpoint.

PRINT

specifies that a print operation is to be started at a 3270 printer or at a 3275 with the printer adapter feature, or that data on an LUTYPE2 (3274/76 or 3790) is to be printed on a printer allocated by the controller. If this option is omitted, the data is sent to the printer buffer but is not printed.

When using the PRINT option, refer to CTRL on page 391 for a description of the option priority.

REQID(name)

specifies a 2-character prefix to be used as part of a temporary storage identifier for CICS message recovery. Only one prefix can be specified for each logical message. The default prefix is **.

BMS message recovery is provided for a logical message only if the PAGING option is specified in the BMS SEND commands and if the syncpoint has been reached.

SET(ptr-ref)

specifies the pointer to be set to the address of the input or output data.

The SET option specifies that completed pages are to be returned to the application program. The pointer is set to the address of a list of completed pages. See the description of the SET option in the section on full BMS in the *CICS/ESA Application Programming Guide* for more guidance about using the SET option.

The application program regains control either immediately following the SEND MAP command (if the current page is not yet completed), or at the label specified in a HANDLE CONDITION RETPAGE command, if the page has been completed.

If TIOAPFX=YES is specified in the map definition, the pointer returned contains the address of the TIOA prefix. The user data starts at offset X'0C' from the start of the TIOA prefix.

TERMINAL

specifies that the output data is to be sent to the terminal that originated the transaction.

WAIT

specifies that control should not be returned to the application program until the output operation has been completed.

If WAIT is not specified, control returns to the application program when the output operation has started. A subsequent input or output command (terminal control, BMS, or batch data interchange) causes the application program to wait until the previous command has been completed.

SEND MAP conditions

Some of the following conditions may occur in combination. If more than one occurs, only the first is passed to the application program.

IGREQCD

occurs when an attempt is made to execute a SEND MAP command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

IGREQID

occurs if the prefix specified in the REQID option on the SEND MAP command is different from that established by a previous REQID option, or by default for this logical message—REQID (**).

Default action: terminate the task abnormally.

INVLDC

occurs if the specified LDC mnemonic is not included in the LDC list for the logical unit.

Default action: terminate the task abnormally.

SEND MAP

INVMP SZ

occurs if the specified map is too wide for the terminal, or if a HANDLE CONDITION OVERFLOW command is active and the specified map is too long for the terminal.

Default action: terminate the task abnormally.

INVPARTN

occurs if the specified partition is not defined in the partition set associated with the application program.

Default action: terminate the task abnormally.

INVREQ

occurs if a request for BMS services is not valid for any of the following reasons:

- Text data is output to the same partition or LDC as mapped data while a BMS logical message is active. If neither partitions nor LDCs are in use, text data is output to the same logical message as mapped data.
- A separate SEND MAP command with the ACCUM option is issued to the terminal that originated the transaction while a routed logical message is being built.
- A SEND MAP command is issued for a map without field specifications by specifying the FROM option without the DATAONLY option.
- During overflow processing, data is sent to a different LDC from the LDC that caused page overflow.
- Partitions are in use, the OUTPARTN option has not been coded on the SEND MAP command, but the PARTN operand has been coded in the mapset definition. If the condition arises, it suggests that different versions of the mapset have different PARTN values, and that the suffix deduced for the partition is not the same as the suffix of the loaded mapset.
- Command not allowed for a distributed program link server program (RESP2=200).
- A SEND MAP command with the DATAONLY option is issued with a data area, supplied by the user, that resides above the 16MB line. But the length of this data area is not longer than the TIOA prefix.

Default action: terminate the task abnormally.

OVERFLOW

occurs if the mapped data does not fit on the current page. This condition is only raised if a HANDLE CONDITION OVERFLOW command is active.

Default action: ignore the condition.

RETPAGE

occurs if the SET option is specified and a completed page is ready for return to the application program.

Default action: return control to the application program at the point immediately following the BMS SEND MAP command.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

WRBRK

occurs if a SEND MAP command is interrupted by the terminal operator pressing the ATTN key. It applies only to the 2741 Communication Terminal, and only if write break is supported for CICS.

Default action: ignore the condition.

SEND MAP MAPPINGDEV

SEND MAP MAPPINGDEV

Function

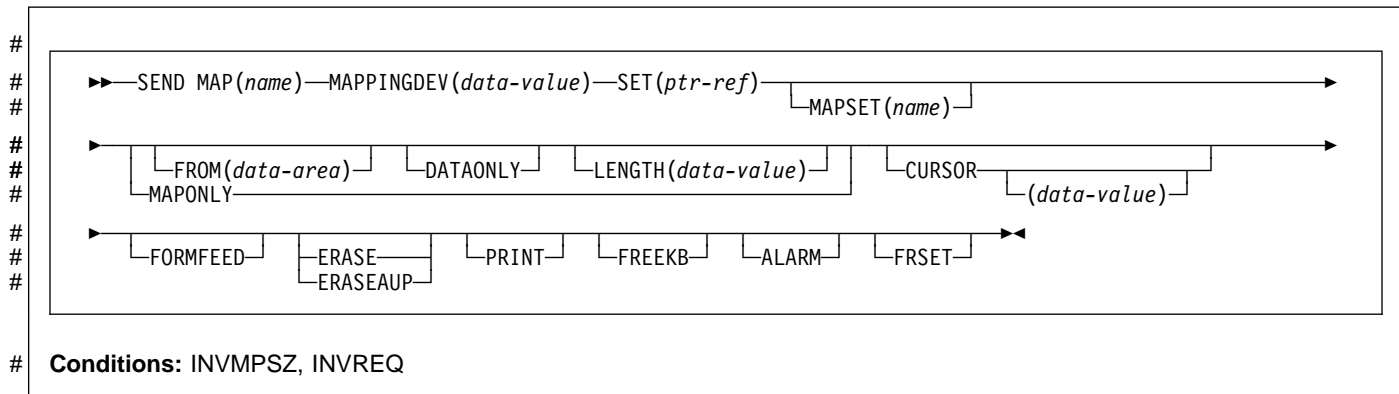
Create mapped output data formatted for the terminal described by MAPPINGDEV.

Apar PQ06521

SEND MAP MAPPINGDEV added by APAR PQ06521 12/08/97

Command syntax

Minimum BMS:



#

Usage

SEND MAP MAPPINGDEV creates mapped output data formatted for a terminal that may not be the principal facility of the transaction. The terminal characteristics to be used are defined by MAPPINGDEV.

The mapped data is not transmitted but is returned to the application in a buffer defined by the SET option.

Parameters

ALARM

specifies that the 3270 audible alarm feature is to be activated. For logical units supporting FMHs (except interactive and batch logical units), ALARM instructs BMS to set the alarm flag in the FMH.

When using the ALARM option, refer to CTRL on page 391 for a description of the option priority.

CURSOR(*data-value*)

specifies the location to which the 3270 cursor is to be returned upon completion of a SEND MAP MAPPINGDEV command.

The data value must be a halfword binary value that specifies the cursor position relative to zero; the range of values that can be specified depends on the size of the screen being used. If no data value is specified, symbolic cursor positioning is assumed. See the section

about minimum BMS in the information about symbolic cursor positioning.

This option overrides any IC option of the ATTRB operand of DFHMDF.

The value specified in the CURSOR option must be positive. A negative value leads to unpredictable results.

DATAONLY

specifies that only application program data is to be written. The attribute characters (3270 only) must be specified for each field in the supplied data. If the attribute byte in the user-supplied data is set to X'00', the attribute byte on the screen is unchanged. Any default data or attributes from the map are ignored.

ERASE

specifies that the screen printer buffer is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSIZE option in the RDO PROFILE definition.

ERASEAUP

specifies that before this page of output is displayed, all unprotected character locations in the entire screen are

```

# to be erased. (This option applies only to the 3270 and
# 8775.)

# FORMFEED
# specifies that a new page is required. For 3270 printers
# and displays, the FORMFEED character is positioned at
# the start of the buffer. The application program must
# thus ensure that this buffer position is not overwritten by
# map or text data. It is ignored if the target terminal does
# not support FORMFEED (that is, the RDO TYPETERM
# option FORMFEED was not used, or the terminal control
# table TYPE=TERMINAL does not specify FF=YES).

# FREEKB
# specifies that the 3270 keyboard should be unlocked
# after the data is written. If FREEKB is omitted, the
# keyboard remains locked.

# When using the FREEKB option, refer to CTRL on page
# 391 for a description of the option priority.

# FROM(data-area)
# specifies the data area containing the data to be
# processed. If this field is not specified, the name
# defaults to the name of the map suffixed with an O.
# This includes the 12-byte prefix generated by the
# TIOAPFX=YES option on the DFHMDI and DFHMSD
# BMS map definitions (see pages 395 and 401).

# FRSET
# specifies that the modified data tags (MDTs) of all fields
# currently in the 3270 buffer are to be reset to the
# unmodified condition (that is, field reset) before any map
# data is written to the buffer.

# This allows the ATTRB operand of DFHMDF for the
# requested map to control the final status of fields written
# or rewritten in response to a BMS command, if no other
# attribute information has been written in the symbolic
# map.

# When using the FRSET option refer to CTRL on page
# 391 for a description of the option priority.

# LENGTH(data-value)
# specifies the length of the data to be formatted as a
# halfword binary value.

# If the data area sending the map is longer than the data
# to be mapped, LENGTH should be specified. This
# should include the length of the 12-byte prefix generated
# by the TIOAPFX=YES option on the DFHMDI and
# DFHMSD BMS map definitions (see pages 395 and
# 401). For a description of a safe upper limit, see
# "LENGTH options" on page 5.

# MAP(name)
# specifies the name (1–7 characters) of the map to be
# used.

# MAPPINGDEV(data-value)
# specifies the name of a 3270 terminal whose BMS
# characteristics match those of the terminal to which the
# data will eventually be sent using a SEND TEXT
# MAPPED command or a terminal control SEND or
# CONVERSE.

# MAPONLY
# specifies that only default data from the map is to be
# written.

# MAPSET(name)
# specifies the unsuffixed name (1–7 characters) of the
# mapset to be used. The mapset must reside in the
# CICS program library. The mapset can be defined either
# by using RDO or by program autoinstall when the
# mapset is first used. If this option is not specified, the
# name given in the MAP option is assumed to be that of
# the mapset.

# The number of maps per mapset is limited to a
# maximum of 9998.

# PRINT
# specifies that a print operation is to be started at a 3270
# printer or at a 3275 with the printer adapter feature, or
# that data on an LUTYPE2 (3274/76 or 3790) is to be
# printed on a printer allocated by the controller. If this
# option is omitted, the data is sent to the printer buffer but
# is not printed.

# When using the PRINT option, refer to CTRL on page
# 391 for a description of the option priority.

# SET(ptr-ref)
# specifies the pointer to be set to the address of the
# mapped data.

# The storage area containing the mapped data has the
# same format as the page buffer returned when using the
# SET option in the full BMS SEND command. See the
# description of the MAPPINGDEV facility in the

# Error Conditions

# Some of the following conditions may occur in combination.
# If more than one occurs, only the first is passed to the
# application program.

# INVMPsz
# occurs if the specified map is too wide for the terminal
# specified by MAPPINGDEV or if a HANDLE CONDITION
# OVERFLOW command is active and the specified map
# is too long for the terminal specified by MAPPINGDEV.

# Default action: terminate the task abnormally.

# INVREQ
# occurs if the terminal specified by MAPPINGDEV does
# not exist, does not support BMS, or is not a 3270 printer
# or display.

# Default action: terminate the task abnormally.

```

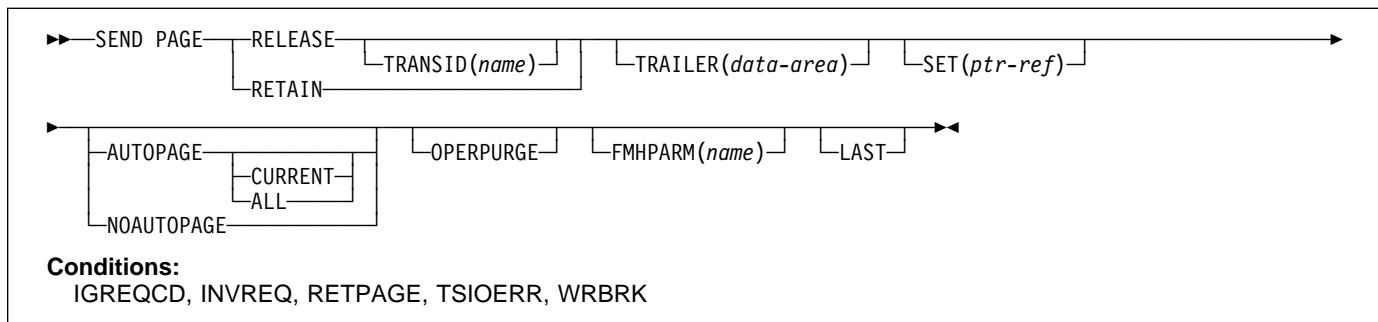
SEND PAGE

SEND PAGE

Function

Send last page of data. Only supplied by full BMS. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

Command syntax



SEND PAGE completes a BMS logical message. It causes BMS to generate a device-dependent data stream for the last (perhaps the only) page of data. Typically, this last page is only partially full.

Options can be included to specify how much control the terminal operator should have over the disposition of the logical message (AUTOPAGE, NOAUTOPAGE, and OPERPURGE), to determine whether control should return to the application program after transmission of the logical message (RELEASE or RETAIN), to add trailer data to a text logical message (TRAILER), and to return the device-dependent data stream for the last page of a logical message to the application program (SET). If this is a paging message, the last page of the logical message is transmitted to temporary storage and the terminal operator paging transaction is initiated. If it is a terminal logical message, the last page is transmitted to the terminal.

This is supported by full BMS only.

SEND PAGE options

ALL

specifies that if the ATTN key on a 2741 is pressed while a BMS logical message is being sent to the terminal, and the WRBRK condition is not active, transmission of the current page is to cease and no additional pages are to be transmitted. The logical message is deleted.

AUTOPAGE

specifies that each page of a BMS logical message is to be sent to the terminal as soon as it is available. If paging on request is specified for the terminal by the PGESTAT operand of the terminal control table TYPE=TERMINAL, or the RDO option AUTOPAGE(NO), AUTOPAGE overrides it for this logical message.

AUTOPAGE is assumed for 3270 printers; it does not apply to 3270 display terminals. If neither AUTOPAGE nor NOAUTOPAGE is specified, the terminal has the paging status specified for it using the terminal control table TYPE=TERMINAL or the RDO option AUTOPAGE.

CURRENT

specifies that if the ATTN key on a 2741 is pressed while a BMS logical message is being sent to the terminal, and the WRBRK condition is not active, transmission of the current page is to cease and transmission of the next page (if any) is to begin.

FMHPARM(name)

specifies the name (1–8 characters) of the outboard map to be used. This option applies only to 3650 logical units with outboard formatting.

LAST

specifies that this is the last output operation for a transaction and, therefore, the end of a bracket. If RELEASE is specified, LAST is assumed unless the SEND PAGE command is terminating a routing operation. This option applies to logical units only.

NOAUTOPAGE

specifies that pages of a BMS logical message are to be sent one at a time to the terminal. BMS sends the first page to the terminal when the terminal becomes available or on request of the terminal operator. Subsequent pages are sent to the terminal in response to requests from the terminal operator. (Refer to the *CICS/ESA CICS-Supplied Transactions* for more information about terminal operator paging commands.)

If automatic paging is specified for the terminal by the PGESTAT operand of the terminal control table TYPE=TERMINAL, NOAUTOPAGE overrides it for this logical message. For logical units, NOAUTOPAGE applies to all pages for all LDCs in the logical message.

NOAUTOPAGE does not apply to 3270 printers.

OPERPURGE

specifies that CICS is to delete the BMS logical message only when the terminal operator requests deletion. If the option is omitted, CICS deletes the message if the operator enters data that is not a paging command.

RELEASE

specifies that, after the SEND PAGE command, control is to be returned to CICS.

RETAIN

specifies that after the SEND PAGE command, control is returned to the application program when the operator has finished displaying the pages.

SET(ptr-ref)

specifies the pointer to be set to the address of the output data.

The SET option specifies that the last or only page is returned to the application program. The pointer is set to the address of the current page. A list of addresses is created and, if the ROUTE command is in operation, there is an address entry for each device. If the ROUTE command is not in operation, the list contains only the one entry. See the description of the SET option in the section about full BMS in the *CICS/ESA Application Programming Guide* for more guidance on using the SET option.

The application program regains control either immediately following the SEND PAGE command (if the current page is not yet completed), or at the label specified in a HANDLE CONDITION RETPAGE command if the page has been completed.

If TIOAPFX=YES is specified in the map definition, the pointer returned contains the address of the TIOA prefix. The user data starts at offset X'0C' from the start of the TIOA prefix.

TRAILER(data-area)

specifies the text data area that contains trailer data to be placed at the bottom of the last page only. The format of the trailer is:

2 bytes	Binary length of the data (n)
2 bytes	Binary zero
n bytes	Data.

See the *CICS/ESA Application Programming Guide* for more information.

TRANSID(name)

specifies the transaction identifier (1–4 alphanumeric characters) to be used with the next input message from the terminal the task is attached to. The identifier must have been defined using the appropriate RDO program definition. TRANSID is valid only if SEND PAGE RELEASE is specified.

If this option is specified in a program that is not at the highest logical level, the specified transaction identifier is used only if a new transaction identifier is not provided in another SEND PAGE command (or in a RETURN program control command) issued in a program at a higher logical level.

SEND PAGE conditions

IGREQCD

occurs when an attempt is made to execute a SEND PAGE command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

INVREQ

occurs if a request for BMS services is not valid for any of the following reasons:

- The disposition (TERMINAL, PAGING, or SET) of a BMS logical message is changed prior to its completion by the SEND PAGE command.
- Text data is output to the same partition or LDC as mapped data while a BMS logical message is active. If neither partitions nor LDCs are in use, text data is output to the same logical message as mapped data.
- The TRAILER option is specified when terminating a logical message built with SEND MAP commands only.
- During overflow processing data is sent to a different LDC from the LDC that caused page overflow.
- The length of the trailer is negative.
- Command not allowed for a distributed program link server program (RESP2=200).

Default action: terminate the task abnormally.

RETPAGE

occurs if the SET option is specified and the last or only completed page is ready for return to the application program.

Default action: return control to the application program at the point immediately following the BMS SEND PAGE command.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

WRBRK

occurs if the SEND PAGE command is interrupted by the terminal operator pressing the ATTN key. It applies only to the 2741 Communication Terminal, and only if write break is supported for CICS.

SEND PAGE

Default action: ignore the condition.

SEND PARTNSET

Function

This command is available on standard and full BMS only. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

Command syntax

```

▶—SEND PARTNSET—┬──▶
                  └─┬─(name)─┘
  
```

Conditions:

INVPARTNSET, INVREQ

SEND PARTNSET associates the partition set specified by the PARTNSET option with the application program. If the partition set name is omitted, the terminal is reset to the base (unpartitioned) state.

Note: A SEND PARTNSET command must not be followed immediately by a RECEIVE command. The two commands must be separated by a SEND MAP, SEND TEXT, or SEND CONTROL command, so that the partition set is sent to the terminal.

SEND PARTNSET conditions

The following conditions may occur together. If both occur, only the first one is passed to the application program.

INVPARTNSET

occurs if the partition set named in the SEND PARTNSET command is not a valid partition set (for example, it may be a mapset).

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- A SEND PARTNSET command is issued while a logical message is active.
- Command not allowed for a distributed program link server program (RESP2=200).

Default action: terminate the task abnormally.

SEND TEXT

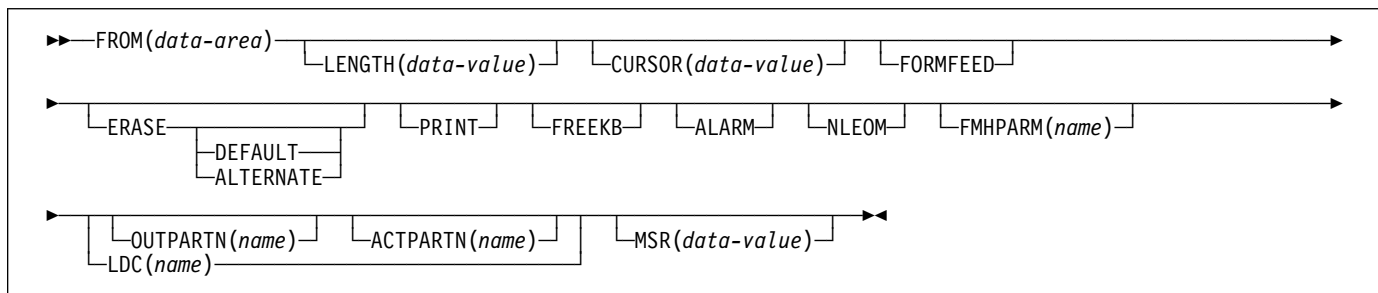
Function

Send data without mapping. The keywords are separated into those supported by standard and full BMS. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

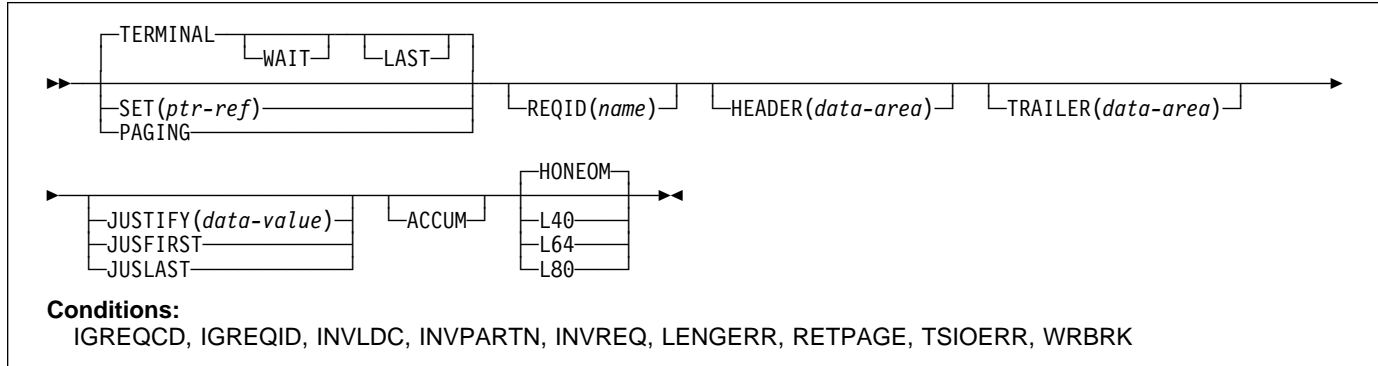
Command syntax

```
▶▶SEND TEXT▶▶
```

Standard BMS:



Full BMS:



SEND TEXT sends text data without mapping. The text is split into lines of the same width as the terminal, such that words are not broken across line boundaries. If the text exceeds a page, it is split into pages that fit on the terminal with application-defined headers and trailers.

When using the SEND TEXT command with any of the ALARM, FREEKB, FRSET, HONEOM, L40, L64, L80, or PRINT options, see CTRL on page 391 for a description of the option priority.

SEND TEXT options

ACCUM

specifies that this command is one of a number of commands that are used to build a logical message. The logical message is completed by a SEND PAGE command, or deleted by a PURGE MESSAGE command.

ACTPARTN(*name*)

specifies the name (1–2 characters) of the partition to be activated. Activating a partition moves the cursor into the specified partition, and unlocks the keyboard for the specified partition.

This option is ignored if the target terminal does not support partitions, or if there is no application partition set.

ALARM

specifies that the 3270 audible alarm feature is to be activated. For logical units supporting FMHs (except interactive and batch logical units), ALARM instructs BMS to set the alarm flag in the FMH.

ALTERNATE

sets the terminal to use the ALTERNATE screen size.

CURSOR(data-value)

specifies the location to which the 3270 or 3604 cursor is to be returned on completion of a SEND TEXT command.

The data value must be a halfword binary value that specifies the cursor position relative to zero; the range of values that can be specified depends on the size of the screen being used.

This option overrides any IC option of the ATTRB operand of DFHMDF. If ACCUM is being used, the most recent value of CURSOR specified is used to position the cursor.

The value specified in the CURSOR option must be positive. A negative value leads to unpredictable results.

DEFAULT

sets the terminal to use the DEFAULT screen size.

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSIZE option in the profile definition.

FMHPARM(name)

specifies the name (1–8 characters) of the outboard map to be used. (This option applies only to 3650 logical units with outboard formatting.)

FORMFEED

specifies that a new page is required. For 3270 printers and displays, the FORMFEED character is positioned at the start of the buffer. The application program must thus ensure that this buffer position is not overwritten by map or text data. It is ignored if the target terminal does not support FORMFEED (that is, the RDO option FORMFEED was not used, or the terminal control table TYPE=TERMINAL does not specify FF=YES).

The FORMFEED option can appear on any SEND TEXT ACCUM command. You need only specify it once within a physical page because it always forces a FORMFEED at the start of the physical page. To force a FORMFEED at the start of a particular SEND TEXT

ACCUM command, use the JUSFIRST option (see below) instead.

FREEKB

specifies that the 3270 keyboard should be unlocked after the data is written. If FREEKB is omitted, the keyboard remains locked.

When using the FREEKB option, refer to CTRL on page 391 for a description of the option priority.

Note that the keyboard lock status is maintained separately for each partition on a terminal that supports partitions.

FROM(data-area)

specifies the data area containing the data to be sent.

HEADER(data-area)

specifies the header data to be placed at the beginning of each page of text data. The format of the header is:

2 bytes	Binary length of the data (n)
1 byte	Page numbering required or not (blank)
1 byte	Reserved field
n bytes	Data

See the *CICS/ESA Application Programming Guide* for more information.

HONEOM

specifies that the default printer line length is to be used. This length should be the same as that specified using the RDO options PAGESIZE or ALTPAGE, or PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL, and the same as the printer platen width; otherwise the data may not format correctly.

When using the HONEOM option, refer to CTRL on page 391 for a description of the option priority.

JUSFIRST

specifies that the text data is to be placed at the top of the page. Any partially formatted page from previous requests is considered to be complete. If the HEADER option is specified, the header precedes the data. See also the description of the JUSTIFY option.

JUSLAST

specifies that the text data is to be positioned at the bottom of the page. The page is considered to be complete after the request has been processed. If the TRAILER option is specified, the trailer follows the data. See also the description of the JUSTIFY option.

JUSTIFY(data-value)

specifies the line of the page at which the text data is to be positioned. The data value must be a halfword binary value in the range 1 through 240. Although they may not be specified as constants, the special values –1 and –2 can be supplied dynamically to signify JUSFIRST or JUSLAST, respectively.

SEND TEXT

LAST

specifies that this is the last output operation for a transaction and, therefore, the end of a bracket. This option applies to logical units only.

LDC(name)

specifies a 2-character mnemonic to be used to determine the logical device code (LDC) to be transmitted in the FMH to the logical unit. The mnemonic identifies an LDC entry specified in the terminal control table TYPE=LDC.

When an LDC is specified, BMS uses the device type, the page size, and the page status associated with the LDC mnemonic to format the message. These values are taken from the extended local LDC table for the logical unit, if it has one. If the logical unit has only a local (unextended) LDC table, the values are taken from the system LDC table. The numeric value of the LDC is obtained from the local LDC table, unless this is an unextended table and the value is not specified, in which case it is taken from the system table.

LENGTH(data-value)

specifies the length of the data to be sent as a halfword binary value. For a description of a safe upper limit, see "LENGTH options" on page 5.

L40, L64, or L80

specifies the line length for a 3270 printer; a carrier return and line feed are forced after 40, 64, or 80 characters have been printed on a line. Unacceptable results are likely if this differs from the page width specified by the RDO options PAGESIZE or ALTPAGE, or the PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL.

When using the options, refer to CTRL on page 391 for a description of the option priority.

MSR(data-value)

specifies the 4-byte data value that controls the 10/63 magnetic stripe reader attached to an 8775 or 3643 terminal. A set of constants is provided in DFHMSRCA to assist in setting this 4-byte area. See "Magnetic slot reader (MSR) control value constants, DFHMSRCA" on page 377 for a complete list. This option is ignored if the RDO option MSRCNTRL was not used, or if the terminal's control table does not specify FEATURE=MSRCNTRL.

NLEOM

specifies that data for a 3270 printer or a 3275 display with the printer adapter feature should be built with blanks and new-line (NL) characters, and that an end-of-message (EM) character should be placed at the end of the data. As the data is printed, each NL character causes printing to continue on the next line, and the EM character terminates printing.

This option must be specified in the first SEND TEXT command used to build a logical message. The option

is ignored if the device receiving the message (direct or routed) is not one of those mentioned above.

If this option is used, buffer updating and attribute modification of fields previously written into the buffer are not allowed. CICS includes the ERASE option with every write to the terminal.

The NL character occupies a buffer position. A number of buffer positions, equivalent to the value of the RDO options PAGESIZE or ALTPAGE, or PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL for that terminal, is unavailable for data. This may cause data to wrap around in the buffer; if this occurs, the PGESIZE value must be reduced.

The NLEOM option overrides the ALARM option if the latter is present.

OUTPARTN(name)

specifies the name (1–2 characters) of the partition to which data is to be sent. This option is ignored if the terminal does not support partitions, or if there is no application partition set associated with the terminal. If there is an application partition set, and the OUTPARTN option is omitted, data is sent to the partition named by the PARTN operand of the DFHMSD or DFHMDI map definition. If maps are not used, or if there is no PARTN operand, the output is sent to the first partition in the partition set.

PAGING

specifies that the output data is not to be sent immediately to the terminal, but is to be placed in temporary storage and displayed in response to paging commands entered by the terminal operator.

If PAGING is specified with a REQID that is defined as recoverable in the temporary storage table (TST), CICS provides message recovery for logical messages if the task has reached a syncpoint.

PRINT

specifies that a print operation is to be started at a 3270 printer or at a 3275 with the printer adapter feature, or that data on an LUTYPE2 (3274/76 or 3790) is to be printed on a printer allocated by the controller. If this option is omitted, the data is sent to the printer buffer but is not printed.

When using the PRINT option, refer to CTRL on page 391 for a description of the option priority.

REQID(name)

specifies a 2-character prefix to be used as part of a temporary storage identifier for CICS message recovery. Only one prefix can be specified for each logical message. The default prefix is **.

BMS message recovery is provided for a logical message only if the PAGING option is specified in the BMS SEND commands and if the syncpoint has been reached.

SET(ptr-ref)

specifies the pointer to be set to the address of the data. It specifies that completed pages are to be returned to the application program. The pointer is set to the address of a list of completed pages. See the description of the SET option in the section full-function BMS in the *CICS/ESA Application Programming Guide* for more guidance about using the SET option.

The application program regains control either immediately following the BMS SEND command (if the current page is not yet completed), or at the label specified in a HANDLE CONDITION RETPAGE command if the page has been completed.

If TIOAPFX=YES is specified in the map definition, the pointer returned contains the address of the TIOA prefix. The user data starts at offset X'0C' from the start of the TIOA prefix.

TERMINAL

specifies that data is to be sent to the terminal that originated the transaction.

TRAILER(data-area)

specifies the text data area that contains trailer data to be placed at the bottom of each output page. The format of the trailer is:

2 bytes	Binary length of the data (n)
1 byte	Page numbering required or not (blank)
1 byte	Reserved field
n bytes	Data

See the *CICS/ESA Application Programming Guide* for more information.

WAIT

specifies that control should not be returned to the application program until the output operation has been completed.

If WAIT is not specified, control returns to the application program when the output operation has started. A subsequent input or output command (terminal control, BMS, or batch data interchange) causes the application program to wait until the previous command has been completed.

SEND TEXT conditions**IGREQCD**

occurs when an attempt is made to execute a SEND TEXT command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

IGREQID

occurs if the prefix specified in the REQID option on a BMS SEND command is different from that established by a previous REQID option, or by default for this logical message—REQID (**).

Default action: terminate the task abnormally.

INVLDC

occurs if the specified LDC mnemonic is not included in the LDC list for the logical unit.

Default action: terminate the task abnormally.

INVPARTN

occurs if the specified partition is not defined in the partition set associated with the application program.

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- Text data is output to the same partition or LDC as mapped data while a BMS logical message is active. If neither partitions nor LDCs are in use, text data is output to the same logical message as mapped data.
- During overflow processing, data is sent to a different LDC from the LDC that caused page overflow.
- The length of a header on a SEND TEXT command is negative.
- The length of a trailer on a SEND TEXT command is negative.
- Command not allowed for a distributed program link server program (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs if an out-of-range value is supplied in the LENGTH option.

Default action: terminate the task abnormally.

RETPAGE

occurs if the SET option is specified and a completed page is ready for return to the application program.

Default action: return control to the application program at the point immediately following the BMS SEND TEXT command.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

WRBRK

occurs if a SEND command is interrupted by the terminal operator pressing the ATTN key. It applies only to the 2741 Communication Terminal, and only if write break is supported for CICS.

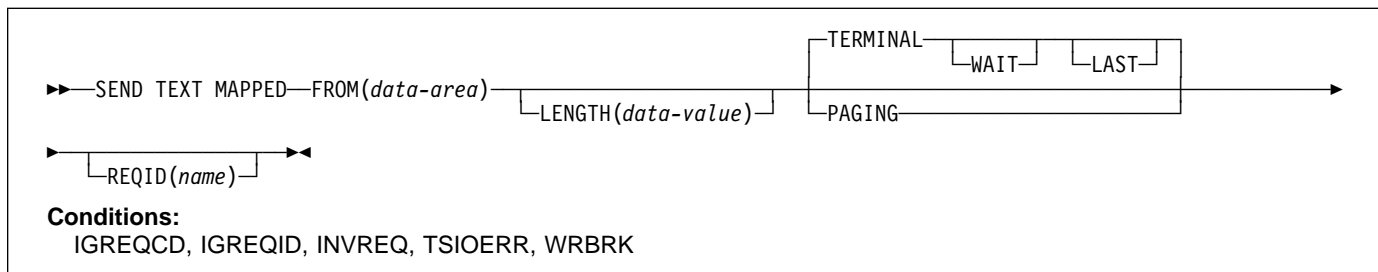
Default action: ignore the condition.

SEND TEXT MAPPED

Function

Send data with mapping. Only supplied by full BMS. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

Command syntax



SEND TEXT MAPPED sends a page of a device-dependent data stream previously built by BMS, and returned to the application program with the SET option.

It must only be used to send data previously generated by a BMS SEND command specifying the SET option. It references a 4-byte page control area (PGA) that BMS placed at the end of the device-dependent data stream.

The length of the device-dependent data stream set in the TIOATDL field of the page buffer returned by the SET option does not include the PGA. The LENGTH option of the SEND TEXT MAPPED command should be set from this TIOATDL, and hence does not include the PGA. However, if the application program copies the page buffer returned by the SET option, it should include the PGA in the copied data.

This command is only supported by full BMS.

SEND TEXT MAPPED options

FROM(data-area)

specifies the data area containing the data to be sent.

LAST

specifies that this is the last output operation for a transaction and, therefore, the end of a bracket. This option applies to logical units only.

LENGTH(data-value)

specifies the length of the data to be formatted as a halfword binary value. For a description of a safe upper limit, see "LENGTH options" on page 5.

PAGING

specifies that the output data is not to be sent immediately to the terminal, but is to be placed in temporary storage and displayed in response to paging commands entered by the terminal operator.

If PAGING is specified with a REQID that is defined as recoverable in the temporary storage table (TST), CICS provides message recovery for logical messages if the task has reached a syncpoint.

REQID(name)

specifies a 2-character prefix to be used as part of a temporary storage identifier for CICS message recovery. Only one prefix can be specified for each logical message. The default prefix is **.

BMS message recovery is provided for a logical message only if the PAGING option is specified in the BMS SEND commands and if the syncpoint has been reached.

TERMINAL

specifies that input data is to be sent to the terminal that originated the transaction.

WAIT

specifies that control should not be returned to the application program until the output operation has been completed.

If WAIT is not specified, control returns to the application program when the output operation has started. A subsequent input or output command (terminal control, BMS, or batch data interchange) causes the application program to wait until the previous command has been completed.

SEND TEXT MAPPED conditions

IGREQCD

occurs when an attempt is made to execute a SEND TEXT command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

IGREQID

occurs if the prefix specified in the REQID option on a BMS SEND command is different from that established by a previous REQID option, or by default for this logical message—REQID (**).

Default action: terminate the task abnormally.

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

WRBRK

occurs if a SEND command is interrupted by the terminal operator pressing the ATTN key. It applies only to the 2741 Communication Terminal, and only if write break is supported for CICS.

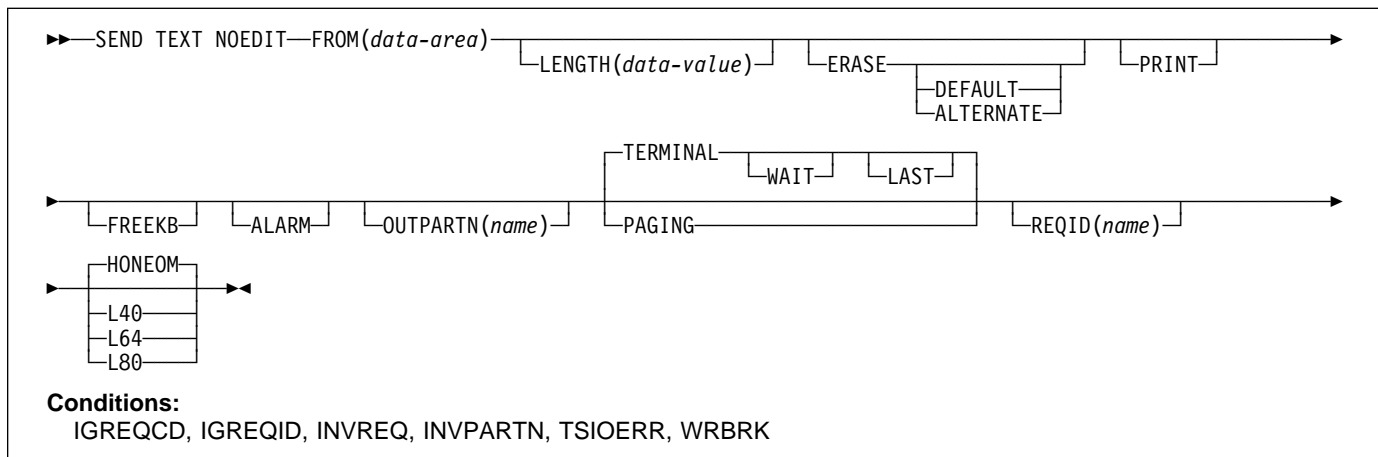
Default action: ignore the condition.

SEND TEXT NOEDIT

Function

Send a page. Only supplied by full BMS. For further information about BMS, see the *CICS/ESA Application Programming Guide*.

Command syntax



SEND TEXT NOEDIT sends a page of a device-dependent data stream built by the application program. The data stream cannot contain structured fields. This command differs from a terminal control SEND, because the data stream may be written to temporary storage and interfaced to the terminal operator paging transaction (using the PAGING option). Also the device-dependent data stream can be sent to a partition (using the OUTPARTN option).

If the OUTPARTN option is specified, the data stream is sent to the specified partition. This command is used to output a user-generated data stream. It differs from a terminal control SEND in that data may be output to temporary storage (using the PAGING option), or routed like any other BMS data.

When using the SEND TEXT NOEDIT command with any of the ALARM, FREEKB, FRSET, HONEOM, L40, L64, L80, or PRINT options, see CTRL on page 391 for a description of the option priority.

This command is supported on full BMS only.

SEND TEXT NOEDIT options

ALARM

specifies that the 3270 audible alarm feature is to be activated. For logical units supporting FMHs (except interactive and batch logical units), ALARM instructs BMS to set the alarm flag in the FMH.

ALTERNATE

sets the terminal to use the ALTERNATE screen size.

DEFAULT

sets the terminal to use the DEFAULT screen size.

ERASE

specifies that the screen printer buffer or partition is to be erased and the cursor returned to the upper left corner of the screen. (This option applies only to the 3270, or 8775, and to the 3604 Keyboard Display.)

The first output operation in any transaction, or in a series of pseudoconversational transactions, should always specify ERASE. For transactions attached to 3270 screens or printers, unless explicitly overridden by the DEFAULT or ALTERNATE option, this also ensures that the correct screen size is selected, as defined for the transaction by the SCRNSIZE option in the profile definition.

FREEKB

specifies that the 3270 keyboard should be unlocked after the data is written. If FREEKB is omitted, the keyboard remains locked.

Note that the keyboard lock status is maintained separately for each partition on a terminal that supports partitions.

When using the FREEKB option, refer to CTRL on page 391 for a description of the option priority.

FROM(data-area)

specifies the data area containing the data to be sent.

HONEOM

specifies that the default printer line length is to be used. This length should be the same as that specified using

the RDO options PAGESIZE or ALTPAGE, or PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL, and the same as the printer platen width; otherwise the data may not format correctly.

When using the HONEOM option, refer to CTRL on page 391 for a description of the option priority.

LAST

specifies that this is the last output operation for a transaction and, therefore, the end of a bracket. This option applies to logical units only.

LENGTH(data-value)

specifies the length of the data to be sent as a halfword binary value. For a description of a safe upper limit, see "LENGTH options" on page 5.

L40, L64, or L80

specifies the line length for a 3270 printer; a carrier return and line feed are forced after 40, 64, or 80 characters have been printed on a line. Unacceptable results are likely if this differs from the page width specified by the RDO options PAGESIZE or ALTPAGE, or the PGESIZE or ALTPGE in the terminal control table TYPE=TERMINAL.

When using the options, refer to CTRL on page 391 for a description of the option priority.

OUTPARTN(name)

specifies the name (1–2 characters) of the partition to which data is to be sent. This option is ignored if the terminal does not support partitions, or if there is no application partition set associated with the terminal. If there is an application partition set, and the OUTPARTN option is omitted, data is sent to the partition named by the PARTN operand of the DFHMSD or DFHMDI map definition. If maps are not used, or if there is no PARTN operand, the output is sent to the first partition in the partition set.

PAGING

specifies that the output data is not to be sent immediately to the terminal, but is to be placed in temporary storage and displayed in response to paging commands entered by the terminal operator.

If PAGING is specified with a REQID that is defined as recoverable in the temporary storage table (TST), CICS provides message recovery for logical messages if the task has reached a syncpoint.

PRINT

specifies that a print operation is to be started at a 3270 printer or at a 3275 with the printer adapter feature, or that data on an LUTYPE2 (3274/76 or 3790) is to be printed on a printer allocated by the controller. If this option is omitted, the data is sent to the printer buffer but is not printed.

When using the PRINT option, refer to CTRL on page 391 for a description of the option priority.

REQID(name)

specifies a 2-character prefix to be used as part of a temporary storage identifier for CICS message recovery. Only one prefix can be specified for each logical message. The default prefix is **.

TERMINAL

specifies that the data is to be sent to the terminal that originated the transaction.

WAIT

specifies that control should not be returned to the application program until the output operation has been completed.

If WAIT is not specified, control returns to the application program when the output operation has started. A subsequent input or output command (terminal control, BMS, or batch data interchange) causes the application program to wait until the previous command has been completed.

SEND TEXT NOEDIT conditions**IGREQCD**

occurs when an attempt is made to execute a SEND TEXT command after a SIGNAL data-flow control command with a request change direction (RCD) code has been received from an LUTYPE4 logical unit.

Default action: terminate the task abnormally.

IGREQID

occurs if the prefix specified in the REQID option on a BMS SEND command is different from that established by a previous REQID option, or by default for this logical message—REQID (**).

Default action: terminate the task abnormally.

INVPARTN

occurs if the specified partition is not defined in the partition set associated with the application program.

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- Text data is output to the same partition or LDC as mapped data while a BMS logical message is active. If neither partitions nor LDCs are in use, text data is output to the same logical message as mapped data.
- During overflow processing, data is sent to a different LDC from the LDC that caused page overflow.
- The length of a header on a SEND TEXT command is negative.
- The length of a trailer on a SEND TEXT command is negative.
- Command not allowed for a distributed program link server program (RESP2=200).

SEND TEXT NOEDIT

Default action: terminate the task abnormally.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

WRBRK

occurs if a SEND command is interrupted by the terminal operator pressing the ATTN key. It applies only to the 2741 Communication Terminal, and only if write break is supported for CICS.

Default action: ignore the condition.

SIGNOFF

Function

Sign off from a terminal.

Command syntax

▶—SIGNOFF—◀

Conditions:
INVREQ

SIGNOFF enables you to sign off from the terminal or principal facility that you previously signed on to. When signoff is complete, the terminal reverts to the security capabilities and operator characteristics associated with the default user for this CICS region. The national language reverts to the national language of the default user, if defined, or the national language associated with the definition of the terminal.

Other resource managers

When this command is executed, CICS immediately recognizes the signoff and establishes the default attributes for the terminal. The transaction (and any associated task-related user exits, function shipping, or distributed transaction processing) may have invoked other resource managers (RMs), for example, IMS, DB2, or VSAM. **It is unpredictable whether these other RMs recognize the signoff before the transaction terminates.**

The default attributes apply for all RMs invoked by subsequent transactions at the terminal.

SIGNOFF condition

INVREQ

occurs in any of the following situations:

- No user is currently signed on (RESP2=1)
- There is no terminal with this task (RESP2=2)
- This task's terminal has preset security (RESP2=3)
- Signoff is attempted using transaction routing without using the CRTE transaction (RESP2=4).
- The CICS ESM interface is not initialized (RESP2=18).
- Command not allowed for a distributed program link server program (RESP2=200).

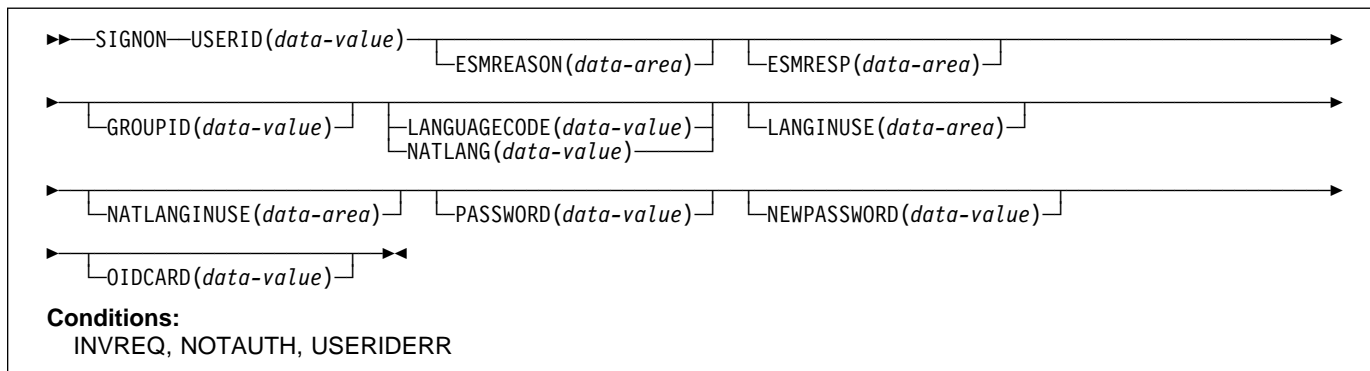
Default action: terminate the task abnormally.

SIGNON

Function

Sign on to a terminal.

Command syntax



SIGNON associates the security capabilities and operator characteristics of the specified user with the terminal. It allows all types of terminal, with the exception of APPC, to be signed on to. SIGNON signs on to the terminal or principal facility associated with the issuing transaction.

There is no implied signoff with the SIGNON command. If you want to sign on a user at a terminal to which a user is already signed on, you must first issue a SIGNOFF command. Note that there is no default value for the USERID option.

| PASSWORD is used as a parameter which means that if
| there is a dump it will become visible. You should therefore
| clear the field as soon as possible after using PASSWORD.

Other resource managers

When this command is executed, CICS immediately recognizes the signon and establishes the specified user's security and operating attributes for the terminal. The transaction (and any associated task-related user exits, function shipping, or distributed transaction processing) may have invoked other resource managers (RMs), for example, IMS, DB2, or VSAM. **It is unpredictable whether these other RMs recognize the signon before the transaction terminates.**

The new user attributes apply for all RMs invoked by subsequent transactions at the terminal.

SIGNON options

| If an optional input field contains all blanks, it is ignored.

ESMREASON(*data-area*)

| returns the reason code, in a fullword binary field, that
| CICS receives from the external security manager.

| If the ESM is RACF, this field is the RACF reason code.

ESMRESP(*data-area*)

| returns the response code, in a fullword binary field, that
| CICS receives from the external security manager.

| If the ESM is RACF, this field is the RACF return code.

GROUPID(*data-value*)

| assigns the user that is being signed on to a RACF user
| group. This overrides, for this session only, the default
| group name specified for the user in the RACF
| database.

#

Apar 15449

#

Following paragraph added by Apar 15449 04/06/98

#

If specified, this value, and the USERID, may be propagated to remote systems. See the *CICS/ESA CICS-RACF Security Guide* for information on implementing LU6.2 security for remote users.

LANGUAGECODE(*data-value*)

| specifies the national language that the user being
| signed on wants CICS to use. You specify the language
| as a standard three-character IBM code. This is an
| alternative to the one-character code that you specify on
| the NATLANG option.

| See Appendix I, "National language codes" on page 373
| for possible values of the code.

LANGINUSE(data-area)

the LANGINUSE option allows an application program to receive the national language chosen by the signon process. The language is identified as a standard three-character IBM code, instead of the one-character code used by NATLANGINUSE. It is an alternative to the existing NATLANGINUSE option.

See Appendix I, “National language codes” on page 373 for possible values of the code.

NATLANG(data-value)

specifies as 1 character field identifying the national language the user wants to use during the signed-on session.

See Appendix I, “National language codes” on page 373 for possible values of the code.

NATLANGINUSE(data-area)

specifies as 1 character the national language used during the signed-on session. NATLANGINUSE corresponds to the following (in order of decreasing priority):

- The language supplied in the NATLANG option of the SIGNON command.
- The language associated with the user. This is specified in the ESM language segment.
- The language associated with the definition of the terminal.
- The language associated with the default USERID for the CICS region.
- The default language specified in the system initialization parameters.

See Appendix I, “National language codes” on page 373 for possible values of the code.

NEWPASSWORD(data-value)

specifies an optional 8-byte field defining a new password. This option is only valid if PASSWORD is also specified.

OIDCARD(data-value)

specifies an optional 65-byte field containing further security data from a magnetic strip reader (MSR) on 32xx devices.

PASSWORD(data-value)

specifies an 8-byte password required by the external security manager (ESM).

USERID(data-value)

specifies the 8-byte signon USERID.

SIGNON conditions**INVREQ**

occurs in any of the following situations:

- The terminal is already signed on (RESP2=9)

- No terminal is associated with this task (RESP2=10)
 - This task’s terminal has preset security (RESP2=11)
 - The response from CICS security modules is unrecognized (RESP2=12)
 - There is an unknown return code in ESMRESP from the external security manager; or the external security manager (ESM) is not active, or has failed in an unexpected way (RESP2=13)
 - The required national language is not available (RESP2=14)
 - Signon was attempted using transaction routing without using the CRTE transaction (RESP2=15)
 - The CICS ESM interface is not initialized (RESP2=18)
 - The terminal is of an invalid type (RESP2=25)
 - An error occurred during SNSCOPE checking. The limit of MVS ENQ requests was reached (RESP2=26)
 - The external security manager (ESM) is not active (RESP2=27)
 - The required national language is invalid (RESP2=28)
 - Command not allowed for a distributed program link server program (RESP2=200)
 - The user is already signed on. This relates to the signon scope checking (RESP2=29).
- Default action: terminate the task abnormally.

NOTAUTH

occurs in any of the following situations:

- A password is required (RESP2=1)
- The supplied password is wrong (RESP2=2)
- A new password is required (RESP2=3)
- The new password is not acceptable (RESP2=4)
- An OIDCARD is required (RESP2=5)
- The supplied OIDCARD is wrong (RESP2=6)
- The USERID is not authorized to use this terminal (RESP2=16)
- The USERID is not authorized to use the application (RESP2=17)
- The USERID is revoked (RESP2=19)
- The USERID’s access to the specified group has been revoked (RESP2=20)
- The signon failed during SECLABEL checking (RESP2=21)
- The signon failed because the ESM is not currently accepting signon (RESP2=22)
- The GROUPID is not known to the ESM (RESP2=23)

SIGNON

- | • The USERID is not contained in the GROUPID
| (RESP2=24).
- | Default action: terminate the task abnormally.

USERIDERR

occurs in any of the following situations:

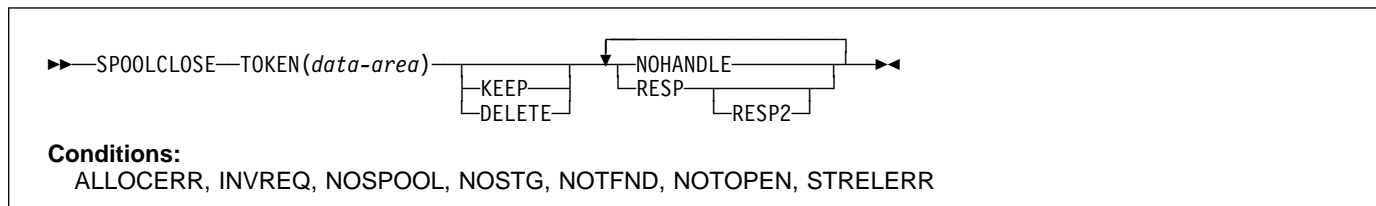
- | • The USERID is not known to the external security
| manager (RESP2=8).
- | • The USERID is all blanks or nulls (RESP2=30)
- | Default action: terminate the task abnormally.

SPOOLCLOSE

Function

Close a spool report.

Command syntax



Conditions:

ALLOCERR, INVREQ, NOSPOOL, NOSTG, NOTFND, NOTOPEN, STRELERR

The SPOOLCLOSE command closes a CICS spool report and, optionally, changes its retention characteristics. If more than one transaction is trying to read reports from JES, SPOOLCLOSE should **not** be immediately followed by SPOOLOPEN. It should be followed by a WAIT, so that other transactions can use the interface.

A default disposition is taken if both KEEP and DELETE are omitted from the SPOOLCLOSE command, or if the report is closed implicitly by a SYNCPOINT or RETURN command:

- When an INPUT report is explicitly closed by a SPOOLCLOSE command, the default disposition is DELETE.
- In all other cases, the default disposition is KEEP.

SPOOLCLOSE options

DELETE

For an INPUT report, DELETE specifies that the **next** report is to be read on the subsequent OPEN INPUT.

For an OUTPUT report, DELETE specifies that the report is to be purged.

Note: When a JCL job is submitted using the internal reader (INTRDR) with the DELETE option specified, the job is sometimes run before the output is deleted.

KEEP

For an INPUT report, KEEP specifies that the report is to be read again when SPOOLOPEN INPUT is next issued.

For an OUTPUT report, KEEP specifies that the report is to be sent to its destination node.

TOKEN(*data-area*)

specifies the 8-character CICS-allocated token used to identify a report.

SPOOLCLOSE conditions

Note: There are no default actions.

Condition	RESP2 Meaning
ALLOCERR	- MVS dynamic allocation has rejected a request to allocate an input data set. RESP2 gives the dynamic allocation response code that denotes this error. The first two characters are the information reason code (S99INFO), and the second two are the error reason code (S99ERROR), as defined in the <i>MVS/ESA Application Development Guide: Authorized Assembler Language Programs, GC28-1645</i> .
INVREQ	4 Unsupported language.
INVREQ	8 Unsupported function.
INVREQ	40 Subsystem interface already enabled. Note: Errors 1024 and over are internal, and should not occur. If one of these error codes is returned, contact your IBM support center.
NOSPOOL	4 No subsystem present.
NOSPOOL	8 Interface being disabled; CICS is quiescing.
NOSPOOL	12 Interface has been stopped.
NOSTG	- An MVS GETMAIN has failed within the JES interface subtask (DFHSPSS). RESP2 gives the GETMAIN register 15 return code.
NOTFND	1024 Input or output function has been corrupted, and SPOOLCLOSE could not complete.
NOTOPEN	8 Data set has not been opened.
STRELERR	- An MVS FREEMAIN has failed within the JES interface subtask (DFHSPSS). RESP2 gives the FREEMAIN register 15 return code.

SPOOLOPEN INPUT

Function

Open a spool report.

Command syntax

```

▶—SPOOLOPEN INPUT—TOKEN(data-area)—USERID(data-value)—
└─CLASS(data-value)┘
└─NOHANDLE┘
└─RESP┘
└─RESP2┘
▶

```

Conditions:

ALLOCERR, ILOGIC, INVREQ, NOSPOOL, NOSTG, NOTAUTH, NOTFND, NOTOPEN, OPENERR, SPOLBUSY, SPOLERR, STRELERR

The SPOOLOPEN INPUT command opens a spool report for input from the system spooler to CICS.

It prepares to get (read) an existing spool data set directly using external writer name (USERID) and specified class.

Another task could have allocated a spool file for input. In this case, you should retry after a suitable time interval.

When this command has been successfully executed, you should read the report and proceed to CLOSE as soon as possible, in order to permit other users to use the JES single thread. If SPOOLCLOSE is not issued before transaction end or SYNCPOINT, CICS performs an implicit SPOOLCLOSE KEEP, and writes a message to CSMT to alert the system programmer to the possible unnecessary retention of resources. You should not SPOOLOPEN a data set using this command until you are prepared to process it completely.

This command, if successful, returns a token, which is used later to identify the report in SPOOLREAD and SPOOLCLOSE commands.

SPOOLOPEN INPUT options

CLASS(*data-value*)

specifies a 1-character class designation. The CLASS operand can be used as a selection parameter for input reports. If it is omitted, the first report for the specified USERID is obtained, regardless of its class.

TOKEN(*data-area*)

specifies the 8-character CICS-allocated token used to identify a report.

USERID(*data-value*)

specifies the 8-character user identifier. It must begin with the same 4 characters as the CICS generic applid, so that CICS can check that users are not attempting to access data sets not intended for their CICS system.

SPOOLOPEN INPUT conditions

Note: There are no default actions.

Condition	RESP2 Meaning
ALLOCERR	- MVS dynamic allocation has rejected a request to allocate an input data set. RESP2 gives the dynamic allocation response code that denotes this error. The first two characters are the information reason code (S99INFO), and the second two are the error reason code (S99ERROR), as defined in the <i>MVS/ESA Application Development Guide: Authorized Assembler Language Programs, GC28-1645</i> .
ILOGIC	3 Invalid CLASS value specified.
INVREQ	4 Unsupported language.
INVREQ	8 Unsupported function.
INVREQ	16 USERID missing.
INVREQ	36 INPUT OUTPUT missing.
INVREQ	40 Subsystem interface already enabled. Note: Errors 1024 and over are internal, and should not occur. If one of these error codes is returned, contact your IBM Support Center.
NOSPOOL	4 No subsystem present.
NOSPOOL	8 Interface being disabled; CICS is quiescing.
NOSPOOL	12 Interface has been stopped.
NOSTG	- An MVS GETMAIN has failed within the JES interface subtask (DFHSPSS). RESP2 gives the GETMAIN register 15 return code.
NOTAUTH	- An application has issued a SPOOLOPEN INPUT command with an unauthorized USERID. For the USERID to be authorized, its first four characters must match the first four characters of the current CICS applid id.

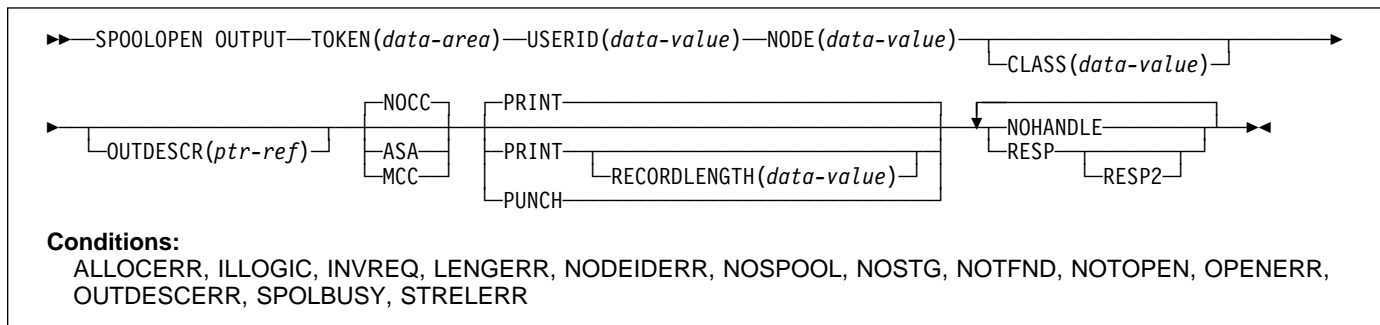
Condition	RESP2	Meaning
NOTFND	4	No data sets could be located for retrieval for the specified external writer name.
#		Can also be returned if the CICS region
#		USERID does not have ALTER access to the
#		appropriate PROFILE in the JESSPOOL
#		class. See the <i>CICS/ESA CICS-RACF</i>
#		<i>Security Guide</i> for more information about
#		RACF authorization of JES.
NOTFND	1024	Input or output function has been corrupted, and SPOOLCLOSE could not complete.
NOTOPEN	8	Data set has not been opened or a task which has not issued the SPOOLOPEN for a spool data set has attempted to access it.
NOTOPEN	1024	Subtask OPEN macro failure.
OPENERR	-	An internal error occurred during SPOOLOPEN processing that has forced the request to fail.
OPENERR	4	A VSAM SHOWCB macro failed to return the lengths of the VSAM control blocks used to access the JES spool file.
SPOLBUSY	-	The JES/input single thread within the JES interface was not available.
SPOLBUSY	4	Interface already in use by another task.
SPOLBUSY	8	Interface already in use by current task.
SPOLERR	-	The MVS subsystem interface macro (IEFSSREQ) has failed. No input data set name was selected.
		RESP2 gives the 'IEFSSREQ' response code.
STRELERR	-	An MVS FREEMAIN has failed within the JES interface subtask (DFHPSPSS).
		RESP2 gives the FREEMAIN register 15 return code.

SPOOLOPEN OUTPUT

Function

Open a spool report for output.

Command syntax



The SPOOLOPEN OUTPUT command opens a spool report for output from CICS to the system spooler and defines its characteristics.

It results in a dynamic allocation of the output file using the nodeid to specify the remote destination and the userid to specify the remote user. As this is a multithread output request, requesters of this service could interleave. This SPOOLOPEN enables users to acquire the token for a report that it expects to create (write). This token is used to identify the report in later SPOOLWRITE and SPOOLCLOSE commands.

When printing on a local device, use the NOCC|ASA|MCC options to control output formatting. If you do not specify a format, the default value of NOCC is used.

Apar PN13451

Documentation for Apar PN13451 added 13/02/95

NODE and USERID can be used to write the data set # directly to the local spool file only if specified with a value of # ' * '.

If you do not issue SPOOLCLOSE before the end of the transaction, CICS performs an implicit SPOOLCLOSE DELETE and writes a message to CSMT to alert you to the possible unnecessary retention of resources.

Note: If you retrieve a formatted data set, the system spooler could have changed the data set format. For example, the system spooler could have converted an MCC format data set to ASA format during data set creation. This does not affect the final printed output.

SPOOLOPEN OUTPUT options

ASA

specifies that the report has each record prefixed with an ASA carriage-control character, and this character must be used by the operating system to control formatting when the report is printed.

CLASS(data-value)

specifies a 1-character class designation. If it is omitted, class A is assumed.

MCC

specifies that the report has each record prefixed with an IBM machine command code carriage-control character, and this character must be used by the operating system to control formatting when the report is printed.

NOCC

specifies that the report has no internal formatting controls. When the report is printed, the operating system prefixes each record with a carriage-control character that causes page skipping according to the default operating system lines-per-page value.

NODE(data-value)

specifies the 8-character identifier of a destination node that the system spooler uses to route the file. It is a sender field.

OUTDESCR(ptr-ref)

(MVS/SP—JES2 Version 3, or JES3 Version 4.2.1 only, or a later upward-compatible release) specifies a pointer variable to be set to the address of a field that contains the address of a string of parameters to the OUTPUT statement of MVS JCL. This is called double indirect addressing. The user must set up the pointer, the address field, and the string. This means that the OUTDESCR option cannot be used from within CECI. The format of the string is:

Offset Length Contents
 0 4 Length (n) of following text string
 4 n OUTPUT statement parameters

The parameters use the same keywords and values as the OUTPUT statement but the syntax varies slightly. The following is the format of the OUTDESCR parameter string:

```
keyword1(value1) [keyword2(value2)]
[keyword3(value3,value4)] ...
```

This corresponds to the following OUTPUT statement parameter string:

```
keyword1=value1 [keyword2=value2]
[keyword3=(value3,value4)] ...
```

:

For details of valid keywords and values, see the *TSO/E Command Reference* manual (SC28-1991-0).

The OUTDESCR operand:

- Can override the NODE and USERID operands only if they are specified with a value of '*'.
 • Cannot override the CLASS operand, even if it is omitted and defaults to class A.

Use this operand to set additional attributes for the spool data set.

PRINT

allows large records (maximum 32760 bytes) to be written to the spool. This is the default setting. This is included for compatibility with the spool support provided with CICS/DOS/VS and CICS/VSE.

PUNCH

must be specified if the CLASS parameter for the output data set implies punch, and the data set is destined for a VM/RSCS node. This ensures that the record length indicator is set to 80, which is a requirement of VM/RSCS for punch files.

RECORDLENGTH(data-value)

specifies, as a halfword binary variable, the maximum length of record to write to a print data set. The default value is 32760.

TOKEN(data-area)

specifies the 8-character CICS-allocated token used to identify a report.

USERID(data-value)

specifies the 8-character identifier of the destination userid that processes the report. The report carries this identifier, which is used to select the report at its destination. It is a sender field and must be declared with a length of 8 characters.

Apar PQ07753
 # Documentation for Apar PQ07753 added 06/10/97

Validity checking is performed for USERID. Checks are made for blanks (X'40'), and nulls (X'00').

Sending the internal reader buffer directly to JES:

Instead of waiting for the buffer in your address space to fill up, send the contents of the internal reader buffer directly to JES by coding as your last record:

```
/*EOF
```

This control statement delimits the job in the data set, and makes it eligible for immediate processing.

For more information about using the internal reader, and about other /* control statements, see the *esajcluc..*

SPOOLOPEN OUTPUT conditions

Note: There are no default actions.

Condition	RESP2	Meaning
ALLOCERR	-	MVS dynamic allocation has rejected a request to allocate an input data set. RESP2 gives the dynamic allocation response code that denotes this error. The first two characters are the information reason code (S99INFO), and the second two are the error reason code (S99ERROR), as defined in the <i>MVS/ESA Application Development Guide: Authorized Assembler Language Programs, GC28-1645.</i>
ILLOGIC	3	Invalid CLASS value specified.
INVREQ	4	Unsupported language.
INVREQ	8	Unsupported function.
INVREQ	16	USERID missing.
INVREQ	20	NODE missing.
INVREQ	36	INPUT OUTPUT missing.
INVREQ	40	Subsystem interface already enabled. Note: Errors 1024 and over are internal, and should not occur. If one of these error codes is returned, contact your IBM support center.
INVREQ	44	Error in the OUTDESCR string.
INVREQ	48	OUTDESCR specified but function not available (wrong level of MVS or JES).
INVREQ	52	OUTDESCR specified but bad pointer found on keyword or in OUTDESCR condition.
LENGERR		RECORDLENGTH not in the range 0 through 32760. RESP2 shows the incorrect value.
NODEIDERR		JES cannot identify the NODE/USERID combination specified on SPOOLOPEN OUTPUT. RESP2 gives the dynamic allocation response code that denotes this error. The first two characters are the information reason code (S99INFO), and the second two are the error reason code (S99ERROR), as defined in the <i>MVS/ESA Application Development Guide: Authorized Assembler Language Programs, GC28-1645.</i>
NOSPOOL	4	No subsystem present.
NOSPOOL	8	Interface being disabled; CICS is quiescing.

SPOOLOPEN OUTPUT

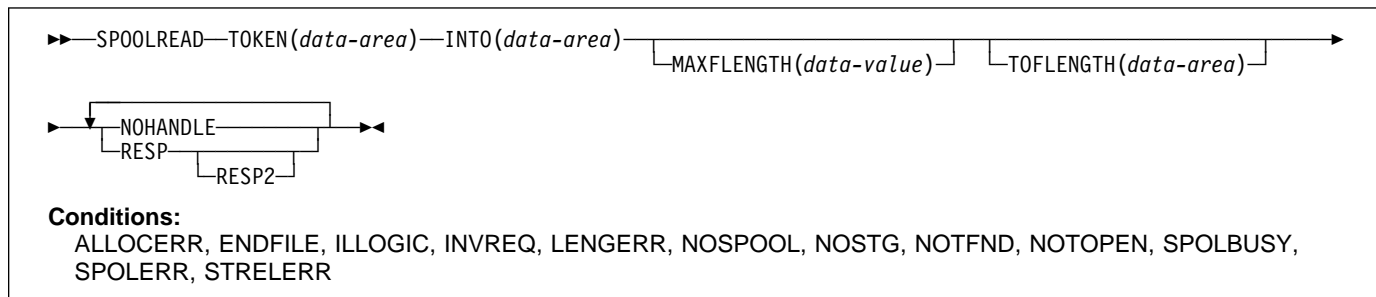
Condition	RESP2	Meaning
NOSPOOL	12	Interface has been stopped.
NOSTG		An MVS GETMAIN has failed within the JES interface subtask(DFHPSPSS). RESP2 gives the GETMAIN register 15 return code.
NOTFND	4	No data sets could be located for retrieval for the specified external writer name.
NOTOPEN	8	Data set has not been opened.
NOTOPEN	1024	Subtask OPEN macro failure.
OPENERR	-	An internal error occurred during SPOOLOPEN processing that has forced the request to fail.
OPENERR	4	A VSAM SHOWCB macro failed to return the lengths of the VSAM control blocks used to access the JES spool file.
OUTDESCRERR		The MVS macro OUTADD or OUTDEL (invoked as a result of the OUTDESCR specification) failed. RESP2 gives the reason code from the OUTADD or OUTDEL macro. See the <i>MVS/ESA Application Development Guide: Authorized Assembler Language Programs, GC28-1645</i> . for descriptions of codes.
SPOLBUSY	-	The JES/input single thread within the JES interface was not available.
SPOLBUSY	4	Interface already in use by another task.
SPOLBUSY	8	Interface already in use by current task.
STRELERR	-	An MVS FREEMAIN has failed within the JES interface subtask (DFHPSPSS). RESP2 gives the FREEMAIN register 15 return code.

SPOOLREAD

Function

The SPOOLREAD command obtains the next record from the system spooler.

Command syntax



SPOOLREAD options

INTO(*data-area*)

specifies the data area for the variable-length data. It is a receiver field.

MAXLENGTH(*data-value*)

specifies, as a fullword binary variable, the maximum length of data transferred. This is set by the user on input. The limit of **length** is 32760 bytes. This is the maximum size of the CICS buffer used to read a record.

TOLENGTH(*data-area*)

specifies, as a fullword binary variable, the length of the data that is transferred. This is set by CICS on input. It is optional and, if it is omitted, you are not notified of the actual length of the data received.

TOKEN(*data-area*)

specifies the 8-character CICS-allocated token used to identify a report.

Condition	RESP2 Meaning
ENDFILE	- All data for the current spool file being read has been retrieved. You should proceed to issue a SPOOLCLOSE command as soon as possible, to release the lock on the JES single thread, and to terminate current SYSOUT data set processing.
ILLOGIC	3 Invalid CLASS value specified.
INVREQ	4 Unsupported language.
INVREQ	8 Unsupported function.
INVREQ	12 Read attempt after end of file.
INVREQ	24 INTO missing.
INVREQ	40 Subsystem interface already enabled.
	Note: Errors 1024 and over are internal, and should not occur. If one of these error codes is returned, contact your IBM support center.

SPOOLREAD conditions

Note: There are no default actions.

Condition	RESP2 Meaning
ALLOCERR	- MVS dynamic allocation has rejected a request to allocate an input data set. RESP2 gives the dynamic allocation response code that denotes this error. The first two characters are the information reason code (S99INFO), and the second two are the error reason code (S99ERROR), as defined in the <i>MVS/ESA Application Development Guide: Authorized Assembler Language Programs, GC28-1645</i> .
NOSPOOL	4 No subsystem present.
NOSPOOL	8 Interface being disabled; CICS is quiescing.
NOSPOOL	12 Interface has been stopped.

SPOOLREAD

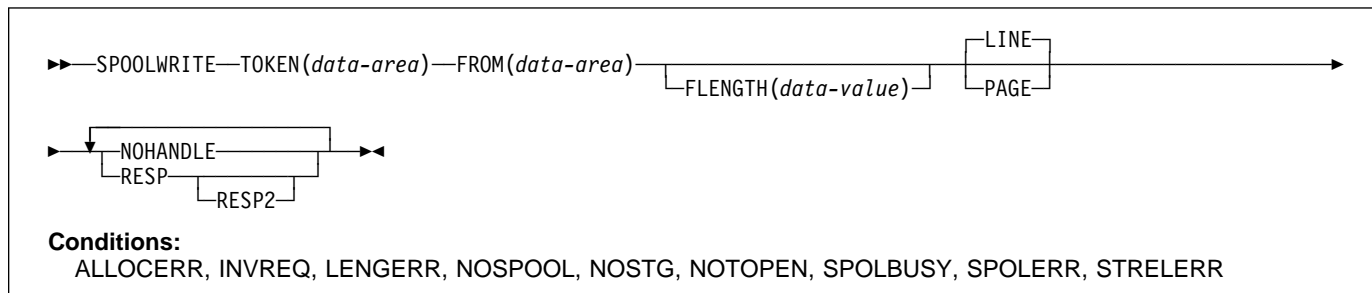
Condition	RESP2	Meaning
NOSTG	-	An MVS GETMAIN has failed within the JES interface subtask (DFHPSPSS). RESP2 gives the GETMAIN register 15 return code.
NOTFND	4	No data sets could be located for retrieval for the specified external writer name.
NOTOPEN	8	Data set has not been opened.
NOTOPEN	12	Attempt to read an output file.
NOTOPEN	1024	Subtask OPEN macro failure.
SPOLBUSY	-	The JES/input single thread within the JES interface was not available.
SPOLBUSY	4	Interface already in use by another task.
SPOLBUSY	8	Interface already in use by current task.
SPOLEERR	-	The MVS subsystem interface macro (IEFSSREQ) has failed. No input data set name was selected. RESP2 gives the 'IEFSSREQ' response code.
STRELERR	-	An MVS FREEMAIN has failed within the JES interface subtask (DFHPSPSS). RESP2 gives the FREEMAIN register 15 return code.

SPOOLWRITE

Function

The SPOOLWRITE command writes data to a spool report.

Command syntax



SPOOLWRITE options

FLENGTH(data-value)

specifies the fullword binary variable that is to be set to the length of the data that is transferred. This is set by the user on output. It is optional and, if it is omitted, CICS uses the length of the data area.

For OS/VS COBOL users, FLENGTH is mandatory.

FROM(data-area)

specifies the data area from which to take the variable length data. The data itself is not altered in any way by CICS. FROM is a sender field.

LINE|PAGE

specifies the format of the data to be sent. The default action is LINE.

The PAGE option must be used to correctly format information for the advanced function printer (AFP) page printing devices. If a customer is creating MIXED mode type data, that is LINE records and X'5A' (AFPDs or MODCA) pagemode records, the LINE or PAGE operand must match the type record being written to spool.

TOKEN(data-area)

specifies the 8-character CICS-allocated token used to identify a report. It is a receiver on SPOOLOPEN and a sender on all other commands.

SPOOLWRITE conditions

Note: There are no default actions.

Condition	RESP2 Meaning
ALLOCERR	- MVS dynamic allocation has rejected a request to allocate an input data set. RESP2 gives the dynamic allocation response code that denotes this error. The first two characters are the information reason code (S99INFO), and the second two are the error reason code (S99ERROR), as defined in the <i>MVS/ESA Application Development Guide: Authorized Assembler Language Programs, GC28-1645</i> .
INVREQ	4 Unsupported language.
INVREQ	8 Unsupported function.
INVREQ	28 FROM missing.
INVREQ	40 Subsystem interface already enabled.
LENGERR	- The value specified in the FLENGTH parameter on a SPOOLWRITE command is not in the valid range 1 to RECORDLENGTH value specified or defaulted at the SPOOLOPEN data set. If the buffer space is too small, it receives as much data as possible. RESP2 contains the difference between FLENGTH and RECORDLENGTH, or zero if FLENGTH is negative or greater than 32760.
NOSPOOL	4 No subsystem present.
NOSPOOL	8 Interface being disabled; CICS is quiescing.
NOSPOOL	12 Interface has been stopped.
NOSTG	- An MVS GETMAIN has failed within the JES interface subtask (DFHPSRSS). RESP2 gives the GETMAIN register 15 return code.
NOTOPEN	8 Spool report has not been opened.
NOTOPEN	16 Attempt to write an input file.

SPOOLWRITE

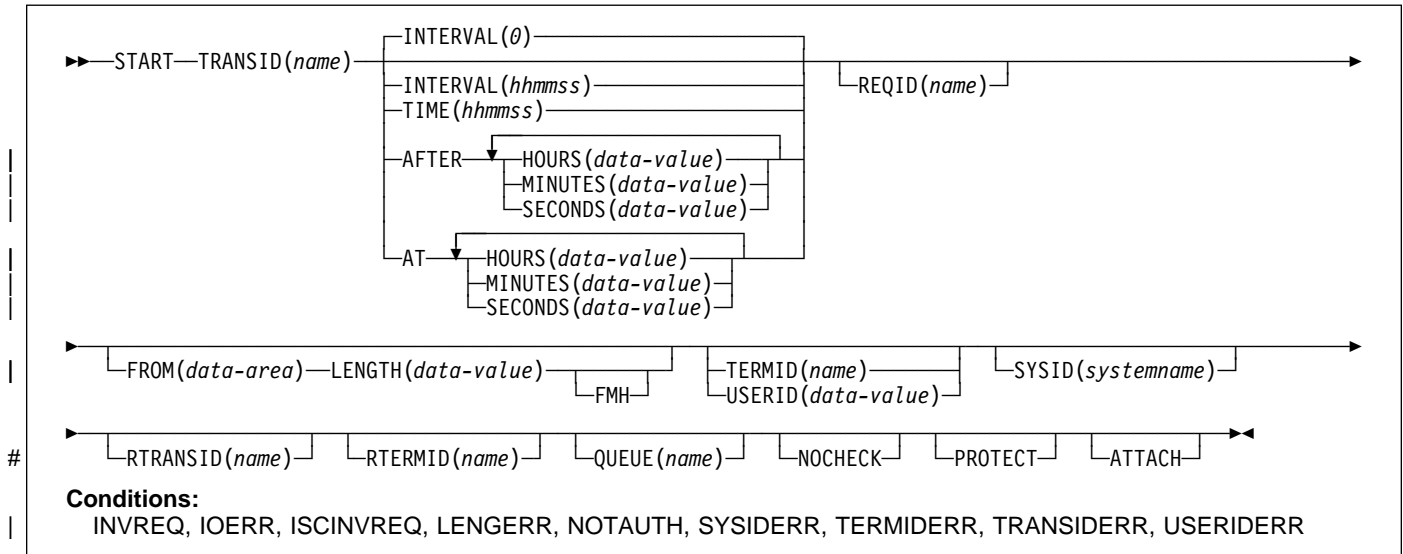
Condition	RESP2	Meaning
NOTOPEN	1024	Subtask OPEN macro failure.
SPOLBUSY	-	The JES/input single thread within the JES interface was not available.
SPOLBUSY	4	Interface already in use by another task.
SPOLBUSY	8	Interface already in use by current task.
SPOLEERR	-	The MVS subsystem interface macro (IEFSSREQ) has failed. No input data set name was selected. RESP2 gives the 'IEFSSREQ' response code.
STRELEERR	-	An MVS FREEMAIN has failed within the JES interface subtask (DFHPSPSS). RESP2 gives the FREEMAIN register 15 return code.

START

Function

Start task at a specified time.

Command syntax



Apar PN92143

Documentation for Apar PN92143 added 28/02/97

Note for dynamic transaction routing

Using START if later CANCELED by another task, or if the started transaction uses RETRIEVE WAIT, could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

START starts a task, on a local or remote system, at a # specified time. The time is specified by INTERVAL, AFTER, # AT or TIME. See the section about expiration times in the # *CICS/ESA Application Programming Guide*.

The default is INTERVAL(0), but for C the default is AFTER HOURS(0) MINUTES(0) SECONDS(0).

The starting task may pass data to the started task and may also specify a terminal to be used by the started task as its principal facility.

Note that **CEDF** is an exception to the START command and is not valid as a TRANSID name. You should therefore not attempt to start CEDF in this way.

You can use the RTRANSID, RTERMINID, and QUEUE options to pass further data to the started task. These options can contain arbitrary data values whose meanings depend on what you have specified in the started and starting tasks. One possible way of using them is in the

following situation. One task can start a second task, passing it a transaction name and a terminal name to be used when the second task starts a third task. The first task may also pass the name of a queue to be accessed by the second task.

One or more constraints have to be satisfied before the transaction to be executed can be started, as follows:

- The specified interval must have elapsed or the specified expiration time must have been reached. (For more information, see the *CICS/ESA Application Programming Guide*.) The INTERVAL or AFTER options should be specified when a transaction is to be executed on a remote system; this avoids complications arising when the local and remote systems are in different time zones.
- If the TERMINID option is specified, the named terminal must exist and be available. If the named terminal does not exist when the time interval expires, the START is discarded.

START

- If the PROTECT option is specified, the starting task must have taken a successful syncpoint. This option, coupled to extensions to system tables, reduces the exposure to lost or duplicated data caused by failure of a starting task.
- If the transaction to be executed is on a remote system, the format of the data must be declared to be the same as that at the local system. This is done using the RDO options DATASTREAM and RECORDFORMAT, or DATASTR and RECFM on the terminal control table TYPE=SYSTEM. For CICS-CICS, these are always the default values. For CICS-IMS/VS, care should be taken to specify the correct values.

Execution of a START command naming a transaction in the local system cancels any outstanding POST commands executed by the starting task.

| START commands are queued by means of the
| TRANSACTION definition as described in the *CICS/ESA
| Resource Definition Guide*.

Passing data by interval control

If data is to be passed by interval control (using the FROM option), it is queued on a temporary storage queue. The REQID option allows you to specify the name of the temporary storage queue to be used. This identifier may be recoverable (in temporary storage terms) or nonrecoverable. The difference between recoverable and nonrecoverable temporary storage is described in the *CICS/ESA Application Programming Guide*. The *CICS/ESA Resource Definition Guide* describes how to define recoverable temporary storage queues.

| If you also specify the PROTECT option, the temporary
| storage queue identified by the REQID option should be
| defined as recoverable. If you do not specify the PROTECT
| option, the temporary storage queue should not be defined
| as recoverable. Unpredictable results can occur if these
| rules are not followed.

If you specify the FROM and not the REQID option, CICS
will generate a queue name for FROM data passed on the
START. If PROTECT is not specified, the queue name has
a prefix 'DF'. If PROTECT is specified, the queue name has
a prefix of the (unprintable) character X'FC'. CICS treats all
FC-prefixed queues as recoverable if there is a TST, and
non-recoverable if there is no TST.

Error checking and performance considerations

The NOCHECK option specifies that no response (to execution of the START command) is expected by the starting transaction. For START commands naming tasks to be started on a local system, error conditions are returned; error conditions are not returned for tasks to be started on a remote system. The NOCHECK option allows CICS to

improve performance when the START command has to be shipped to a remote system; it is also a prerequisite if the shipping of the START command is queued pending the establishing of links to the remote system.

Starting tasks without terminals

If the task to be started is not associated with a terminal, each START command results in a separate task being started. This happens regardless of whether or not data is passed to the started task. The following examples show how to start a specified task, not associated with a terminal, in one hour:

```
EXEC CICS START
      TRANSID('TRNL')
      INTERVAL(10000)
      REQID('NONGL')
```

```
EXEC CICS START
      TRANSID('TRNL')
      AFTER HOURS(1)
      REQID('NONGL')
```

Starting tasks with terminals but without data

Only one task is started if several START commands, each specifying the same transaction and terminal, expire at the same time or before the terminal is available.

The following examples show how to request initiation of a task associated with a terminal. Because no request identifier is specified in this example, CICS assigns one and returns it to the application program in the EIBREQID field in the EXEC interface block.

```
EXEC CICS START
      TRANSID('TRN1')
      TIME(185000)
      TERMID('STA5')
```

```
EXEC CICS START
      TRANSID('TRN1')
      AT HOURS(18) MINUTES(50)
      TERMID('STA5')
```

Starting tasks with terminals and data

Data is passed to a started task if one or more of the FROM, RTRANSID, RTERMID, and QUEUE options is specified. Such data is accessed by the started task by using a RETRIEVE command.

It is possible to pass many data records to a new task by issuing several START commands, each specifying the same transaction and terminal.

Execution of the first START command ultimately causes the new task to be started and allows it to retrieve the data specified on the command. The new task is also able to retrieve data specified on subsequently executed START

| commands that expire before the new task is terminated. If
 | the transaction has been defined as restartable (by defining
 | the transaction using the RDO option RESTART(YES)) and
 | such data has not been retrieved before the new task is
 | terminated, another new task is started and is able to retrieve
 | the outstanding data. If the subsequent new task fails to
 | retrieve the outstanding data, a third task will be started and
 | so on, up to a maximum of five times, after which the data is
 | discarded. If the transaction has not been defined as
 | restartable, no new task is initiated and the data is discarded.

The following examples show how to start a task associated with a terminal and pass data to it:

```
EXEC CICS START
      TRANSID('TRN2')
      TIME(173000)
      TERMID('STA3')
      REQID(DATAREC)
      FROM(DATAFLD)
      LENGTH(100)

EXEC CICS START
      TRANSID('TRN2')
      AT HOURS(17) MINUTES(30)
      TERMID('STA3')
      REQID(DATAREC)
      FROM(DATAFLD)
      LENGTH(100)
```

There are two ways to enter the time under AFTER and AT.

1. A combination of at least two of HOURS(0–99), MINUTES(0–59), and SECONDS(0–59). HOURS(1) SECONDS(3) would mean one hour and three seconds (the minutes default to zero).
2. As one of HOURS(0–99), MINUTES(0–5999), or SECONDS(0–359999). HOURS(1) means one hour. MINUTES(62) means one hour and two minutes. SECONDS(3723) means one hour, two minutes, and three seconds.

START failures without exception conditions

There are some circumstances in which a START command is executed without error, but the started task never takes place:

- When the transaction or its initial program is disabled at the time CICS attempts to create the task.
- When the START specifies a terminal and an expiration time, and the terminal is not defined (and cannot be located by the XICTENF or XALTENF exits) at expiration time.
- When the START specifies a terminal that is not defined (and cannot be located by the XICTENF or XALTENF exits) at the time CICS attempts to create the task.

These exposures result from the delay between the execution of the START and the time of task creation. Even when the START is immediate, CICS may delay creating the task, either because the required terminal is not free or because of other system constraints.

You can use INQUIRE commands to ensure that the transaction and program are enabled at the time of the START command, but either may become disabled before task creation.

You get a TERMIDERR condition if the requested terminal does not exist at the time of the START, but if it is deleted

START

subsequently, as occurs if the user logs off, your START request is discarded with the terminal definition.

START options

AFTER

specifies the amount of time to elapse before starting.

AT specifies the time of starting.

```
# _____ Apar PN92143 _____
# |
# | Documentation for Apar PN92143 added 28/02/97
# |
```

ATTACH

```
# specifies that a non-terminal task is to be started at once
# in the local system. The only other options that may be
# used with ATTACH are FROM and LENGTH. If these
# are specified, the data is not written to temporary
# storage. The attached task retrieves the data in the
# normal way but in effect only its address is passed. The
# task issuing the start must ensure that the data is valid
# when it is retrieved either by synchronizing its execution
# with the attached task, or by placing the data in shared
# storage.
```

```
# The attached task has a STARTCODE of U and cannot
# be cancelled, so EIBREQID is set to nulls.
```

```
# ATTACH allows a START issued in a PLTPI program to
# take effect before initialization has completed.
```

FMH

specifies that the user data to be passed to the started task contains function management headers.

FROM(data-area)

specifies the data to be stored for a task that is to be started at some future time.

| HOURS(data-value)

specifies a fullword binary value in the range 0–99.

INTERVAL(hhmmss)

specifies the expiration time as an interval of time that is to elapse from the time at which the START command is issued. The **mm** and **ss** are each in the range 0–59. The time specified is added to the current clock time by CICS when the command is executed, to calculate the expiration time.

```
| When using the C language, you are recommended to
| use the AFTER/AT HOURS, MINUTES, and SECONDS
| options as C does not provide a packed decimal data
| type. You may use INTERVAL, but if the value specified
| is not an integer constant, the application is responsible
| for ensuring that the value passed to CICS is in packed
| decimal format.
```

LENGTH(data-value)

specifies a halfword binary data value that is the length of the data to be stored for the new task.

| MINUTES(data-value)

```
| specifies a fullword binary value in the range 0–59, when
| HOURS or SECONDS are also specified, or 0–5999
| when MINUTES is the only option specified.
```

NOCHECK

specifies that, for a remote system, CICS should improve performance of the START command by providing less error checking and slightly less function. For more information, see the section about improving the performance of intersystem START requests in the *CICS/ESA Intercommunication Guide*.

PROTECT

specifies that the new task is not started until the starting task has taken a syncpoint. If the starting task abends before the syncpoint is taken, the request to start the new task is canceled. If the REQID option is also specified, the request identifier should be a name defined as recoverable to temporary storage. If the started transaction is remote, PROTECT specifies that it must not be scheduled until the local transaction has successfully completed a syncpoint. For more information about the PROTECT option with remote transactions, see the *CICS/ESA Intercommunication Guide*.

QUEUE(name)

```
# specifies a name (1–8 characters) that is passed to the
# started task. If this name represents a temporary storage
# queue, the queue must be local to the started task. The
# contents of the queue are not passed.
```

If you are also specifying REQID, make sure that the name of the REQID and the name of the QUEUE are not the same.

REQID(name)

```
| specifies a name (1–8 characters), which should be
| unique, to identify a command. This option can be used
| when another task is to be provided with the capability of
| canceling an unexpired command.
```

```
| If this option is omitted, CICS generates a unique
| request identifier in the EIBREQID field of the EXEC
| interface block, unless the NOCHECK option is
| specified, in which case field EIBREQID is set to nulls
| and cannot be used subsequently to cancel the START
| command.
```

```
| If you include any of the data options (FROM,
| RTERMID, RTRANSID or QUEUE), the data is stored in
| temporary storage using the REQID name specified (or
| CICS generated) as the identifier. The temporary
| storage record thus identified must be local to the CICS
| system where the START command is processed. The
| START command is processed on the system identified
| by the SYSID option or, if the SYSID option is omitted,
| on the system associated with the TRANSID option.
```

```
# If the same REQID name is specified on more than one
# START command, the behavior of subsequent
# RETRIEVE and CANCEL requests is unpredictable. In
```

particular, the correspondence between each START
and its data is lost.

RTERMID(name)

specifies a value (1–4 characters), for example a terminal name, that may be retrieved when the transaction, specified in the TRANSID option in the START command, is started.

When retrieved, the value may be used in the TERMID option of a subsequent START command.

RTRANSID(name)

specifies a value (1–4 characters), for example a transaction name, that may be retrieved when the transaction, specified in the TRANSID option in the START command, is started.

When retrieved, the value may be used in the TRANSID option of a subsequent START command.

| **SECONDS(data-value)**

| specifies a fullword binary value in the range 0–59, when
| HOURS or MINUTES are also specified, or 0–359 999
| when SECONDS is the only option specified.

SYSID(systemname)

specifies the name of the system to which the request is directed.

TERMID(name)

specifies the symbolic identifier (1–4 alphanumeric characters) of the principal facility associated with a transaction to be started as a result of a START command. This principal facility can be either a terminal (the usual case) or an APPC session. Where an APPC session is specified, the connection (or modeset) name is used instead of a terminal identifier. This option is required when the transaction to be started must communicate with a terminal; it should be omitted otherwise.

The terminal identifier must be defined as either a local or a remote terminal on the system in which the START command is executed, **when the start of the transaction takes effect.**

TIME(hhmmss)

specifies the time when a new task should be started.

| When using the C language, you are recommended to
| use the AFTER/AT HOURS, MINUTES, and SECONDS
| options as C does not provide a packed decimal data
| type. You may use TIME, but if the value specified is
| **not** an integer constant, the application is responsible for
| ensuring that the value passed to CICS is in packed
| decimal format.

TRANSID(name)

specifies the symbolic identifier (1–4 characters) of the transaction to be executed by a task started as the result of a START command.

| If SYSID is specified, and names a remote system, the
| transaction is assumed to be on that system irrespective

of whether or not the transaction definition is defined as remote in the PCT. Otherwise the transaction definition is used to find out whether the transaction is on a local or a remote system.

| **USERID(data-value)**

| Specifies the userid under whose authority the started
| transaction is to run, if the started transaction is not
| associated with a terminal (that is, when TERMID is not
| specified). This is referred to as *userid1*.

| If you omit both TERMID and USERID, CICS uses
| instead the userid under which the transaction that
| issues the START command is running. This is referred
| to as *userid2*.

| By using either *userid1* or *userid2* CICS ensures that a
| started transaction always runs under a valid userid,
| which must be authorized to all the resources referenced
| by the started transaction.

| CICS performs a surrogate security check against
| *userid2* to verify that this user is authorized to *userid1*.
| If *userid2* is not authorized, CICS returns a NOTAUTH
| condition. The surrogate check is not done here if
| USERID is omitted.

START conditions

INVREQ

occurs in any of the following situations:

Apar PQ19047

Documentation for Apar PQ19047 added 27/10/98,
modifies RCF 11734 added 1/12/97

- # • The START command is not valid for processing by CICS
- # • Values specified in the INTERVAL option are out of range
- # • The value specified in HOURS, for AFTER and AT options, is out of range (RESP2=4)
- # • The value specified in MINUTES, for AFTER and AT options, is out of range (RESP2=5)
- # • The value specified in SECONDS, for AFTER and AT options, is out of range (RESP2=6)
- # • A START operation specifies a REQID name that corresponds to an existing user temporary storage queue
- # • An attempt was made to ship a START request with the ATTACH option (RESP2=11)
- # • A START request with the ATTACH option has failed (RESP2=12).
- # • A USERID is specified and the CICS external security manager interface is not initialized (RESP2=18)

Apar PN92143

Documentation for Apar PN92143 added 28/02/97
RESP2=11 and 12 added

START

Default action: terminate the task abnormally.

IOERR

occurs in any of the following situations:

- | • An input/output error occurred during a START operation.
- | • A START operation attempts to write to temporary storage when the temporary storage data set is full.
- | •
- # • Uses a REQID that corresponds to an existing 'recoverable' temporary storage queue. This condition only occurs when the FROM option is also used.

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LENGERR

occurs if LENGTH is not greater than zero.

Default action: terminate the task abnormally.

NOTAUTH

occurs in any of the following situations:

- | • A resource security check fails on TRANSID (name) (RESP2=7).
- | • A surrogate user security check fails on USERID (name) (RESP2=9).
- | The security access capabilities of the transaction that issued the command do not allow the command to be performed with the value specified in the USERID option. The security access capabilities of the transaction have been established by the external security manager according to user security, and whether link security or the execution diagnostic facility (EDF) have been in use.

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION).

SYSIDERR also occurs when the link to the remote system is known but unavailable. This condition may not be raised if the user exit XISLCLQ is enabled (see the *CICS/ESA Customization Guide* for programming information).

Default action: terminate the task abnormally.

TERMIDERR

occurs if the terminal identifier in a START command cannot be found in the terminal control table.

Default action: terminate the task abnormally.

TRANSIDERR

occurs if the transaction identifier specified in a START command cannot be found in the program control table.

Default action: terminate the task abnormally.

USERIDERR

occurs in any of the following situations:

- | • The specified USERID is not known to the external security manager (RESP2=8).
- | • The external security manager is in a state such that CICS cannot determine whether a specified USERID is valid (RESP2=10).

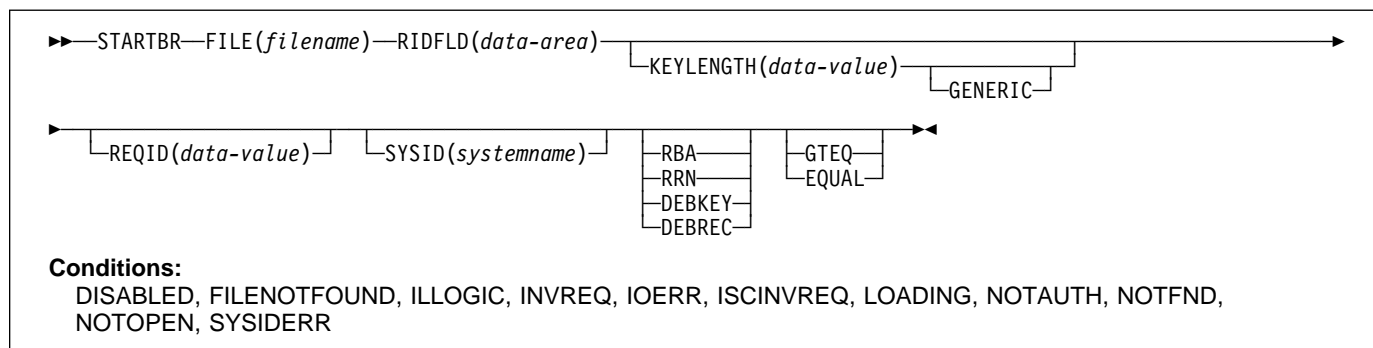
Default action: Terminate the task abnormally.

STARTBR

Function

Start browse of a file.

Command syntax



STARTBR specifies the record in a file, or in a data table, on a local or a remote system, where you want the browse to start. No records are read until a READNEXT command (or, for VSAM and CICS-maintained tables, a READPREV command) is executed.

A browse operation may be:

- A direct browse of a key sequenced data set (KSDS) by record key.
- A direct browse of an entry sequenced data set (ESDS) by relative byte address (RBA).
- A direct browse of a relative record data set (RRDS) by relative record number (RRN).
- A browse of a key sequenced data set (KSDS) using an alternate index path.
- A browse of an entry sequenced data set (ESDS) using an alternate index path. In this case, an ESDA is browsed by key in the same way as a KSDS. Some of the options that are not valid for a direct ESDS browse are valid for an alternate index browse.

The options specified on the STARTBR command define the characteristics that apply throughout the subsequent browse operation. Specifically, if GENERIC or GTEQ are specified, they are used not only when determining the starting point of the browse, but also whenever the value of RIDFLD is changed before issuing a READNEXT command.

If you specify the RBA option, it applies to every READNEXT or READPREV command in the browse, and causes CICS to return the relative byte address of each retrieved record.

None of these options can be changed during a browse, except by means of the RESETBR command. For a CICS-maintained data table, STARTBR refers to the source VSAM KSDS.

STARTBR options

DEBKEY (blocked BDAM)

specifies that deblocking is to occur by key. If neither DEBREC nor DEBKEY is specified, deblocking does not occur.

DEBREC (blocked BDAM)

specifies that deblocking is to occur by relative record (relative to zero). If neither DEBREC nor DEBKEY is specified, deblocking does not occur.

EQUAL (VSAM and data table)

specifies that the search is satisfied only by a record having the same key (complete or generic) as that specified in the RIDFLD option.

This option is the default field for a direct ESDS browse.

FILE(filename)

specifies the of the file to be accessed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

GENERIC (VSAM KSDS, path or data table)

specifies that the search key is a generic key whose length is specified in the KEYLENGTH option. The search for a record is satisfied when a record is found that has the same starting characters (generic key) as those specified.

GTEQ (VSAM or data table)

specifies that, if the search for a record having the same key (complete or generic) as that specified in the RIDFLD option is unsuccessful, the first record having a greater key satisfies the search.

STARTBR

This option is the default for directly browsing through a KSDS or an RRDS. It is not valid for directly browsing an ESDS, although it is valid for browsing through an ESDS using a path.

KEYLENGTH(data-value)

specifies the length (halfword binary) of the key that has been specified in the RIDFLD option, except when RBA or RRN is specified, in which case KEYLENGTH is not valid.

This option must be specified if GENERIC is specified, and it can be specified whenever a key is specified. If the length specified is different from the length defined for the data set and the operation is not generic, the INVREQ condition occurs.

The INVREQ condition also occurs if a STARTBR command specifies GENERIC, and the KEYLENGTH is not less than that specified in the VSAM definition.

If KEYLENGTH(0) is used with the object of positioning on the first record in the data set, the GTEQ option must also be specified. If EQUAL is specified either explicitly or by default with KEYLENGTH(0), the results of the STARTBR is unpredictable.

RBA (VSAM ESDS, KSDS, or CMT)

(base data sets only, not paths)
specifies that the record identification field specified in the RIDFLD option contains a relative byte address.
Use this option only when browsing an ESDS base, or a KSDS base when using relative byte addresses instead of keys to identify the records.

| You cannot use RBA for user-maintained data tables.

REQID(data-value)

specifies as a halfword binary value a unique request identifier for a browse, used to control multiple browse operations on the same or different data sets. If this option is not specified, a default value of zero is assumed.

RIDFLD(data-area)

specifies the record identification field. When combined with RBA or RRN this is a 4-character field. The contents can be a key, a relative byte address, or relative record number (for VSAM data sets), or a block reference, physical key, and deblocking argument (for BDAM data sets). For a relative byte address or a relative record number, the format of this field must be fullword binary. For a relative byte address, the RIDFLD can be greater than or equal to zero. For a relative record number, the RIDFLD can be greater than or equal to 1.

For VSAM, a full record id of X'FF's indicates that the browse is to be positioned at the end of the data set in preparation for a backwards browse using READPREV commands.

RRN (VSAM RRDS)

specifies that the record identification field specified in the RIDFLD option contains a relative record number.

This option should only be used with files referencing relative record data sets.

SYSID(systemname)

specifies the name of the system to which the request is directed.

If you specify SYSID, and omit both RBA and RRN, you must also specify KEYLENGTH.

STARTBR conditions

Note: RESP2 values are not set for files that are on remote systems.

DISABLED

occurs if a file is disabled (RESP2=50). A file may be disabled because:

- It was initially defined as disabled and has not since been enabled.
- It has been disabled by a SET FILE or a CEMT SET FILE command.

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if a file name referred to in the FILE option cannot be found in the FCT and SYSID has not been specified (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- Browse operations are not allowed according to the file entry specification in the FCT (RESP2=20).
- The KEYLENGTH and GENERIC options are specified, and the length defined for the data set to which this file specified in the KEYLENGTH option is greater than or equal to the length of a full key (RESP2=25).
- The KEYLENGTH option is specified (but the GENERIC option is not specified), and the specified length does not equal the length defined for the data set to which this file refers (RESP2=26).
- An attempt is made to start a browse with a REQID already in use for another browse (RESP2=33).
- The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than zero (RESP2=42).

- The specified file is a user-maintained data table (RESP2=44). Browse commands are not in the subset of the file control API that may be used for user-maintained data tables.

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the file control operation (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error.

(Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

LOADING

occurs if a STARTBR is issued to a user-maintained data table that is currently being loaded (RESP2=104). A user-maintained data table cannot be modified during loading.

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTFND

occurs if an attempt to position on a record based on the search argument provided is unsuccessful (RESP2=80).

NOTFND can also occur if a generic STARTBR with KEYLENGTH(0) specifies the EQUAL option.

Default action: terminate the task abnormally.

NOTOPEN

occurs in any of the following situations (RESP2=60):

- The requested file is CLOSED and UNENABLED. The CLOSED, UNENABLED state is reached after a CLOSE request has been received against an OPEN ENABLED file and the file is no longer in use. This state can also be specified as the initial state by means of the FILSTAT parameter of the file control table TYPE=FILE, or by defining a user-maintained data table using the RDO options STATUS=UNENABLED and OPENTIME=FIRSTREF.
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.

This condition does not occur if the request is made either to a CLOSED, ENABLED file or to a CLOSED, DISABLED file. In the first case, the file is opened as part of executing the request. In the second case, the DISABLED condition occurs.

Default action: terminate the task abnormally.

STARTBR

SYSIDERR

| occurs when the SYSID option specifies a name that is
| neither the local system nor a remote system (made
| known to CICS by defining a CONNECTION).
SYSIDERR also occurs when the link to the remote
system is closed (RESP2=130).

Default action: terminate the task abnormally.

SUSPEND

Function

Suspend a task.

Command syntax

`▶▶SUSPEND▶▶`

SUSPEND relinquishes control to a task of higher or equal dispatching priority. Control is returned to the task issuing the command as soon as no other task of a higher or equal priority is ready to be processed.

SYNCPOINT

Function

Establish a syncpoint.

Command syntax

▶—SYNCPOINT—▶

Conditions:

INVREQ, ROLLEDBACK

SYNCPOINT divides a task (usually a long-running one) into smaller LUWs. It specifies that all changes to recoverable resources made by the task since its last syncpoint are to be committed.

SYNCPOINT conditions

INVREQ

occurs if SYNCPOINT was in a program that is linked to from a remote system that has not specified the SYNCONRETURN option, or if it has been linked to locally and is defined with EXECUTIONSET=DPLSUBSET (RESP2=200).

Default action: terminate the task abnormally.

ROLLEDBACK

occurs when a SYNCPOINT command is driven into rollback by a remote system that is unable to commit the syncpoint. All changes made to recoverable resources in the current LUW are backed out.

Default action: terminate the task abnormally.

SYNCPOINT ROLLBACK

Function

Back out to last syncpoint.

Command syntax

▶—SYNCPOINT—ROLLBACK—▶

Conditions:
INVREQ

SYNCPOINT ROLLBACK option

ROLLBACK

specifies that all changes to recoverable resources made by the task since its last syncpoint are to be backed out.

This option can be used, for example, to tidy up in a HANDLE ABEND routine, or to revoke database changes after the application program finds irrecoverable errors in its input data.

If the LUW updates remote recoverable resources using an MRO or APPC session, the ROLLBACK option is propagated to the back-end transaction.

When a distributed transaction processing conversation is in use, the remote application program has the EIB fields EIBSYNRB, EIBERR, and EIBERRCD set. For the conversation to continue, the remote application program should execute a SYNCPOINT ROLLBACK command.

When the mirror transaction is involved in the LUW using an MRO or APPC session, the mirror honors the rollback request, revokes changes, and then terminates normally.

This option is not supported across LUTYPE6.1 VTAM sessions to the mirror or back-end transactions. In these cases, the front-end transactions could be abended to cause the back-end transactions to back out.

SYNCPOINT ROLLBACK condition

INVREQ

occurs if SYNCPOINT ROLLBACK was in a program that is linked to from a remote system that has not specified the SYNCONRETURN option, or if it has been linked to locally and is defined with EXECUTIONSET=DPLSUBSET (RESP2=200).

Default action: terminate the task abnormally.

UNLOCK

UNLOCK

Function

Release exclusive control.

Command syntax

```
UNLOCK FILE(filename) [TOKEN(data-value)] [SYSID(systemname)]
```

Conditions:

DISABLED, FILENOTFOUND, ILLOGIC, INVREQ, IOERR, ISCINVREQ, NOTAUTH, NOTOPEN, SYSIDERR

UNLOCK releases the exclusive control established in response to a READ command with the UPDATE option. You use it if you retrieve a record for update, and then decide that you do not want to update the record after all. However, for a data set for which the system programmer has specified automatic logging, or for a user-maintained data table that has been defined as recoverable, the resource remains under the task control enqueue until either a synpoint command is executed or the task is terminated. The record can be in a data set, or in a CICS or user maintained data table, on a local or a remote system.

Use this command to terminate a VSAM WRITE MASSINSERT operation against a VSAM file.

UNLOCK options

FILE(filename)

specifies the of the file to be released.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

SYSID(systemname)

specifies the name of the system to which the request is directed.

TOKEN(data-value)

specifies as a fullword binary value a unique request identifier for an UNLOCK, used to associate it with a previous READ for UPDATE request.

UNLOCK conditions

Note: RESP2 values are not set for files that are on remote systems.

DISABLED

occurs if a file is disabled (RESP2=50). A file may be disabled because:

- It was initially defined as disabled and has not since been enabled.

- It has been disabled by a SET FILE or a CEMT SET FILE command.

This condition cannot occur when the UNLOCK follows a successful read for update or a VSAM WRITE MASSINSERT.

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if a file name referred to in the FILE option cannot be found in the FCT and SYSID has not been specified (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM and CICS-maintained data tables)

occurs if a VSAM error occurs that does not fall within one of the other CICS response categories (RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- An unlock includes a token whose value cannot be matched against any token in use for an existing READ for UPDATE request (RESP2=47).
- An attempt is made to function-ship a request which includes a TOKEN keyword (RESP2=48).

If an UNLOCK command does not have a token, an attempt is made to match it to a READ for UPDATE which also does not have a token. If this is not found, no action is taken and a NORMAL response is returned.

Default action: terminate the task abnormally.

IOERR

occurs if there is an I/O error during the file control operation (RESP2=120). An I/O error is any unusual event that is not covered by a CICS condition.

For VSAM files, IOERR normally indicates a hardware error.

(Further information is available in the EXEC interface block; refer to Appendix A, "EXEC interface block" on page 343 for details.)

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTOPEN

occurs in any of the following situations (RESP2=60):

- The requested file is CLOSED and UNENABLED. The CLOSED, UNENABLED state is reached after a CLOSE request has been received against an OPEN ENABLED file and the file is no longer in use. This state can also be specified as the initial state by means of the FILSTAT parameter of the file control table TYPE=FILE, or by defining a file using the RDO options STATUS=UNENABLED and OPENTIME=FIRSTREF.
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.

This condition does not occur if the request is made to either a CLOSED, ENABLED file or a CLOSED, DISABLED file. In the first case, the file is opened as part of executing the request. In the second case, the DISABLED condition occurs.

It also cannot occur when the UNLOCK follows a successful READ for update or a WRITE MASSINSERT operation.

Default action: terminate the task abnormally.

SYSIDERR

| occurs when the SYSID option specifies a name that is
 | neither the local system nor a remote system (made
 | known to CICS by defining a CONNECTION).
 SYSIDERR also occurs when the link to the remote
 system is closed (RESP2=130).

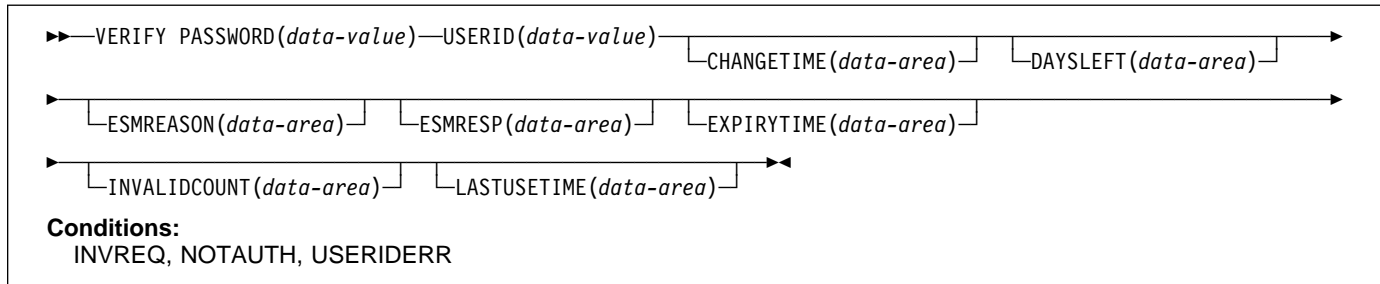
Default action: terminate the task abnormally.

VERIFY PASSWORD

Function

Allows an application to check that a password matches the password recorded by an external security manager (ESM) for a userid, and return values recorded by the external security manager for the password.

Command syntax



Unlike the SIGNON command, VERIFY PASSWORD does not depend upon the principal facility, so it can be issued when the facility is an APPC session.

When the external security manager is RACF, the CHANGETIME and EXPIRYTIME outputs always show as midnight.

If a VERIFY PASSWORD request is successful, you should not infer that a signon would also be successful. For example, the USERID may be revoked in one or more RACF group connections, or may not be able to signon in this CICS region.

Warning: Clear password fields after use

You should clear the password fields on the EXEC CICS commands that have a password option as soon as possible after use. This is to ensure that passwords are not revealed in system or transaction dumps.

VERIFY PASSWORD options

CHANGETIME(data-area)

returns the date and time the password was last changed, in ABSTIME units.

When the external security manager is RACF, the time is shown as midnight.

DAYSLEFT(data-area)

returns the number of days from now, in a halfword binary field, until the password expires. If the password is non-expiring, -1 is returned.

ESMREASON(data-area)

returns the reason code, in a fullword binary field, that CICS receives from the external security manager.

If the ESM is RACF, this field is the RACF reason code.

ESMRESP(data-area)

returns the response code, in a fullword binary field, that CICS receives from the external security manager.

If the external security manager is RACF, this field is the RACF return code.

EXPIRYTIME(data-area)

returns the date and time the password will expire, in ABSTIME units.

When the external security manager is RACF, the time is shown as midnight.

INVALIDCOUNT(data-area)

returns the number of times an invalid password was entered for this user.

LASTUSETIME(data-area)

returns the data and time this userid was last accessed, in ABSTIME units.

PASSWORD(data-value)

specifies the password, 8 characters, that you want the external security manager to check for the specified userid. The other data is not returned if the password is not valid.

USERID(data-value)

specifies the userid, 8 characters, of the user whose password is to be checked.

Note: In the CHANGETIME, LASTUSETIME, and EXPIRYTIME options, the time value returned is in the same format as the ASKTIME command, so it can be reformatted as a date and time, in a format specified by the caller, by using the FORMATTIME command.

Unlike the SIGNON command, the VERIFY PASSWORD command does not depend upon the principal facility, so can be issued when the facility is an APPC session.

If a user has a never-expiring password that was established with the RACF PASSWORD USER(userid) NOINTERVAL

| command, the outputs DAYSLEFT and EXPIRYTIME have
| little meaning and are shown as -1.

| **VERIFY PASSWORD conditions**

| **INVREQ**

| occurs in any of the following situations:

- | • There is an unknown return code in ESMRESP from
| the external security manager (RESP2=13).
- | • The CICS external security manager interface is not
| initialized (RESP2=18).
- | • The external security manager is not responding
| (RESP2=29).

#

Apar 69349

#

Documentation for Apar 69349 added 31 May
1995 (TUCKER)

- # • The userid field contains a blank character in an
invalid position (RESP2=32).

| Default action: terminate the task abnormally.

| **NOTAUTH**

| occurs in any of the following situations:

- | • The supplied password is wrong (RESP2=2). If the
| external security manager is RACF, the revoke
| count maintained by RACF is incremented.
- | • A new password is required (RESP2=3).
- | • The USERID is revoked (RESP2=19).

| Default action: terminate the task abnormally.

| **USERIDERR**

| occurs if the USERID is not known to the external
| security manager (RESP2=8).

| Default action: terminate the task abnormally.

WAIT CONVID (APPC)

Function

Ensure accumulated data is transmitted on an APPC mapped conversation.

Command syntax



WAIT CONVID allows an application program to ensure that any accumulated application data and control indicators from a SEND command, or the results of a CONNECT PROCESS command, are transmitted to the partner transaction.

NOTALLOC

occurs if the CONVID value in the command does not relate to a conversation that is owned by the application.

Default action: terminate the task abnormally.

WAIT CONVID options

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

STATE(cvda)

gets the state of the current conversation. The cvda values returned by CICS are:

ALLOCATED
CONFFREE
CONFRECEIVE
CONFSEND
FREE
PENDFREE
PENDRECEIVE
RECEIVE
ROLLBACK
SEND
SYNCFREE
SYNCRECEIVE
SYNCSEND

WAIT CONVID conditions

INVREQ

occurs in any of the following situations:

- The command is used on a conversation that is not using the EXEC CICS interface or that is not a mapped conversation
- Command not supported for distributed program link when it refers to the principal facility (RESP2=200).

Default action: terminate the task abnormally.

WAIT EVENT

Function

Wait for an event to occur.

Command syntax

```
▶—WAIT EVENT—ECADDR(ptr-value)—NAME(name)—▶
```

Conditions:
INVREQ

Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

WAIT EVENT synchronizes a task with the completion of an event initiated by the same task or by another task. The event would normally be the posting, at the expiration time, of a timer-event control area provided in response to a POST command, as described in “POST” on page 178. The WAIT EVENT command provides a method of directly relinquishing control to some other task until the event being waited on is completed.

CICS includes the addresses of all ECBs passed by WAIT EVENT commands of current tasks in the ECBLIST passed by CICS to the MVS WAIT facility when it runs out of work. Such ECBs can be posted using the MVS POST facility or by hand posting. Hand posting could, for example, be done by moving an appropriate value into the ECB.

A given ECB may not be waited on by more than one task at the same time. If this rule is not followed and the ECBLIST passed by CICS on the MVS WAIT contains duplicate ECB addresses, MVS abends CICS.

The following example shows you how to suspend the processing of a task until the specified event control area is posted:

```
EXEC CICS WAIT EVENT ECADDR(PVALUE)
```

WAIT EVENT options

ECADDR(*ptr-value*)

specifies the address of the timer-event control area that must be posted before task activity can be resumed.

NAME(*name*)

specifies the symbolic name, 1–8 alphanumeric characters, that is returned in SUSPENDVALUE or HVALUE, when a task issues WAIT EVENT and is the subject of an INQ TASK command or a CEMT INQ TASK.

WAIT EVENT condition

INVREQ

occurs in any of the following situations:

- The value of the pointer is zero (X'00000000') (RESP2=2).
- The specified event control area address is above the 16MB line for programs executing in 24-bit mode (RESP2=3).
- The event control area address is not aligned on a fullword boundary (RESP2=4).
- The timer-event control area specified on a WAIT EVENT is in user-key task-lifetime storage, and is inaccessible to another transaction. This condition can only occur if the storage for the timer-event control area is obtained other than by a POST command, and is for posting as an ECB by some other task on completion of an event (RESP2=6).

Note: CICS obtains storage for a timer-event control area in response to a POST command (and which can be used in conjunction with the WAIT EVENT command) from a shared subpool in user-key storage. This ensures that timer-event control areas are in shared storage and, when referenced by a subsequent WAIT EVENT command, do not fail with an INVREQ.

Default action: terminate the task abnormally.

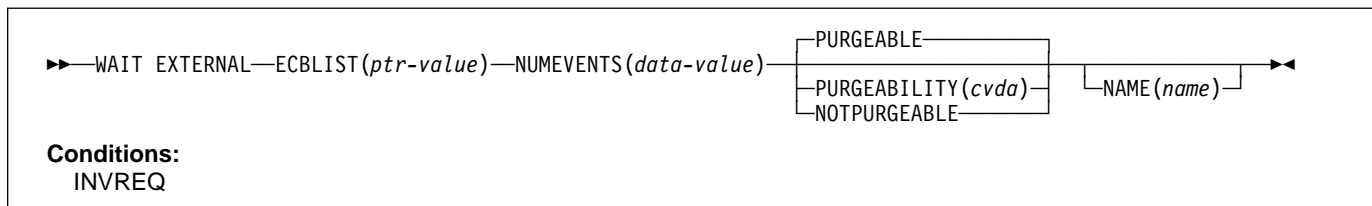
WAIT EXTERNAL

WAIT EXTERNAL

Function

Synchronize events.

Command syntax



Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

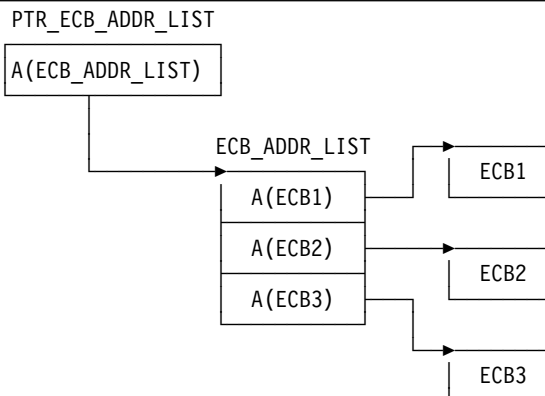
WAIT EXTERNAL waits for events that post MVS-format ECBs. The command causes the issuing task to be suspended until one of the ECBs has been posted, that is until one of the events has occurred. The task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted. You must ensure that each ECB is cleared (set to binary zeros) no later than the earliest time it could be posted. CICS cannot do this for you. If you wait on an ECB that has been previously posted and not subsequently cleared, your task is not suspended and continues to run as though the WAIT EXTERNAL had not been issued.

CICS uses extended ECBs and the MVS POST exit mechanism for ECBs passed by WAIT EXTERNAL; therefore do not use WAIT EXTERNAL unless you are sure that the ECBs are not posted by any method other than the MVS POST service or the standard 'optimized post' logic using a compare-and-swap (CS) instruction. Note that the standard 'optimized post' logic is only applicable when the ECB is not waiting, that is when the wait bit X'80' is not on.

If a WAIT EXTERNAL ECB is hand posted, for example by another task moving a value into the ECB, unpredictable errors occur. If there is any possibility of hand posting, use the WAITCICS command. Use WAIT EXTERNAL whenever possible, because it usually has less overhead.

A given ECB must not be waited on by more than one task at the same time. If this rule is not followed, the second task to wait on the ECB gets an INVREQ condition.

Figure 5 shows how to use the ECBLIST parameter to point to a list of ECB addresses that in turn point to individual ECBs. Note that the ECBLIST variable is a pointer pointing to the first address of the list.



```
DCL
| ECB1      FIXED BIN(31),      /* actual ecb */
| ECB2      FIXED BIN(31),      /* actual ecb */
| ECB3      FIXED BIN(31);     /* actual ecb */
| DCL                               /* list of ecb addresses */
| 1 ECB_ADDR_LIST,
| 2 ECB_ADDR(3) PTR;
| DCL                               /* ptr to each addr list */
| PTR_ECB_ADDR_LIST PTR;
ECB_ADDR(1) = ADDR(ECB1);
ECB_ADDR(2) = ADDR(ECB2);
ECB_ADDR(3) = ADDR(ECB3);
/* set up pointer */
PTR_ECB_ADDR_LIST = ADDR(ECB_ADDR_LIST);
/* PTR_ECB_ADDR_LIST = ADDR(ECB_ADDR(1));
   (alternative) */
EXEC CICS WAIT EXTERNAL
        ECBLIST(PTR_ECB_ADDR_LIST)
        NUMEVENTS(3)
        PURGEABLE
```

Figure 5. ECBLIST option, EXEC CICS WAIT EXTERNAL

WAIT EXTERNAL options

ECBLIST(ptr-value)

is a pointer to a list of addresses of MVS-format ECBs representing events. Both the ECBLIST and the ECBs can be above the 16MB line, that is they can be 31-bit addresses. Each ECB must be fullword aligned. Null (X'00000000') ECB addresses are ignored.

NAME(name)

specifies the symbolic name, 1–8 alphanumeric characters, that is returned in SUSPENDVALUE or HVALUE, when a task issues WAIT EXTERNAL and is the subject of an INQ TASK command or a CEMT INQ TASK.

NUMEVENTS(data-value)

is the number of such events, corresponding to the number of addresses in the ECBLIST. The field is fullword binary. When NUMEVENTS is specified as 1, ECBLIST must still be an address that points to a list containing just one ECB address.

PURGEABILITY(cvda)

determines the outcome of:

- An attempt to perform a deadlock time-out
- A SET TASK PURGE|FORCEPURGE command
- A CEMT SET TASK PURGE|FORCEPURGE.

on the issuing task while it is waiting. The values passed to CICS are PURGEABLE (the default value), or NOTPURGEABLE. The outcome is:

Function	PURGEABLE	NOTPURGEABLE
DTIMOUT expired	Abend AEXY	No effect
CEMT SET TASK PURGE EXEC CICS SET TASK PURGE	Abend AEXY	No effect
CEMT SET TASK FORCEPURGE EXEC CICS SET TASK FORCEPURGE	Abend AEXY	Abend AEXY

See the *CICS/ESA Recovery and Restart Guide* for information about DTIMOUT and SET TASK PURGE|FORCEPURGE.

WAIT EXTERNAL condition

INVREQ

occurs in any of the following situations:

- An ECB is not valid, for example the ECB is not fullword aligned (RESP2=1).
- An ECB is already being waited on (RESP2=2).
- NUMEVENTS is not a positive number (RESP2=3).
- PURGEABILITY is specified with an incorrect CVDA (RESP2=4).
- No valid ECBs have been found in the list, because either the ECBLIST address is not valid or all the ECB addresses are not valid (RESP2=5).

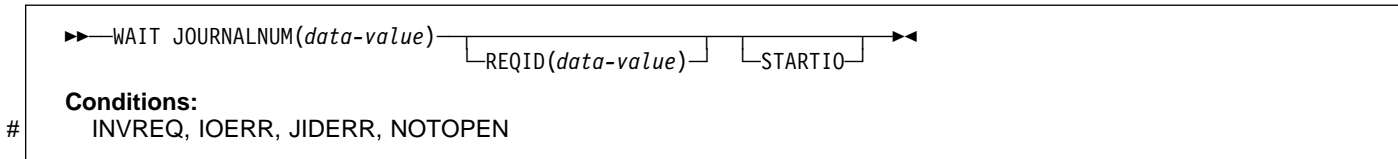
Default action: terminate the task abnormally.

WAIT JOURNALNUM

Function

Synchronize with journal output.

Command syntax



WAIT JOURNALNUM synchronizes the task with the output of one or more journal records that have been created but whose output has been deferred; that is, with asynchronous journal output requests.

The journal records in the journal buffer area may already be written out to auxiliary storage, or the journal record output operation may be in progress. If the output operation has already been completed, control returns immediately to the requesting task; if not, the requesting task waits until the operation has been completed. If STARTIO is specified, output is initiated immediately.

If the requesting program has made a succession of successful asynchronous output requests to the same journal, it is necessary to synchronize on only the last of these requests to ensure that all of the journal records have reached auxiliary storage. This may be done either by issuing a stand-alone WAIT JOURNALNUM command, or by making the last output command itself synchronous (by specifying the WAIT option in the WRITE JOURNALNUM command).

The following example shows how to request synchronization with the output of a journal record:

```
EXEC CICS WAIT JOURNALNUM(4)
      REQID(ENTRYID)
```

WAIT JOURNALNUM options

JOURNALNUM(*data-value*)

specifies a halfword numeric value in the range 1–99 to be taken as the journal identifier. The value 1 specifies that the system log is the journal for this operation.

REQID(*data-value*)

specifies a fullword binary variable set to a number that refers to the journal record that has been created but possibly not yet written out.

If REQID is not specified, the task is synchronized with the output of the last record created for the journal specified by JOURNALNUM.

STARTIO

specifies that output of the journal record is to be initiated immediately.

WAIT JOURNALNUM conditions

INVREQ

occurs if a WAIT JOURNALNUM command is issued before any WRITE JOURNALNUM command has been issued in the same task.

Default action: terminate the task abnormally.

IOERR

occurs if the physical output of a journal record was not accomplished because of an irrecoverable I/O error.

Default action: terminate the task abnormally.

JIDERR

occurs if the specified journal identifier does not exist in the journal control table (JCT).

Default action: terminate the task abnormally.

NOTOPEN

occurs if the WAIT JOURNALNUM command cannot be satisfied because the specified journal is not open.

Default action: terminate the task abnormally.

WAIT SIGNAL

Function

Suspend task on a logical unit.

Command syntax

<pre> ▶←WAIT SIGNAL→←▶ Conditions: NOTALLOC, SIGNAL, TERMERR </pre>
--

WAIT SIGNAL, for a principal facility only, suspends a task until a SIGNAL condition occurs. Some logical units can interrupt the normal flow of data to the application program by a SIGNAL data-flow control command to CICS, signaling an attention, that in turn causes the SIGNAL condition to occur.

The HANDLE CONDITION SIGNAL command causes a branch to a user routine when an attention is received.

The logical units for which you can code a WAIT SIGNAL command are:

- LUTYPE4
- LUTYPE6.1
- 3600 (3601)
- 3767 interactive
- 3770 batch
- 3790 full-function.

WAIT SIGNAL conditions

NOTALLOC

- # occurs if the facility specified in the command is not owned by the application.
- # Default action: terminate the task abnormally.

SIGNAL

occurs when the data-flow control command has been received from the principal facility.

EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

TERMERR

occurs for a terminal-related error.

A CANCEL TASK request by a user node error program (NEP) may cause this condition if the task has an outstanding terminal control request active when the node abnormal condition program (CSNE) handles the session error.

Default action: terminate the task abnormally with abend code ATNI.

WAIT TERMINAL

WAIT TERMINAL

Function

Ensure terminal operation has completed on an LUTYPE6.1 logical unit.

Command syntax



WAIT TERMINAL ensures that terminal operation has completed.

WAIT TERMINAL options

CONVID(name)

identifies the conversation to which the command relates. The 4-character name identifies either the token returned by a previously executed ALLOCATE command in EIBRSRCE in the EIB, or the token representing the principal session (returned by a previously executed ASSIGN command).

SESSION(name)

specifies the symbolic identifier (1–4 characters) of a session TCTTE. This option specifies the alternate facility to be used. If both this option and CONVID are omitted, the principal facility for the task is used.

WAIT TERMINAL conditions

INVREQ

occurs if a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option (RESP2=200).

Default action: terminate the task abnormally.

NOTALLOC

occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

SIGNAL

occurs when an inbound SIGNAL data-flow control command is received from a logical unit or session. EIBSIG is always set when an inbound signal is received.

Default action: ignore the condition.

WAITCICS

Function

Synchronize events.

Command syntax

```

  ▶▶ WAITCICS—ECBLIST(ptr-value)—NUMEVENTS(data-value)
  ┌── PURGEABLE ───┐
  ┌── PURGEABILITY(cvda) ───┐
  └── NOTPURGEABLE ───┘
  └── NAME(name) ───▶

```

Conditions:
INVREQ

Note for dynamic transaction routing

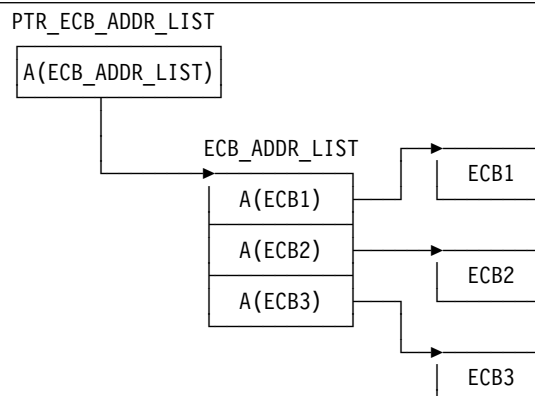
Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

WAITCICS waits for events that post MVS-format ECBs. The command causes the issuing task to be suspended until one of the ECBs has been posted, that is until one of the events has occurred. The task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted. You must ensure that each ECB is cleared, set to binary zeros, no later than the earliest time it could be posted. CICS cannot do this for you. If you wait on an ECB that has been previously posted and not subsequently cleared, your task is not suspended and continues to run as though the WAITCICS had not been issued.

CICS includes the addresses of all ECBs passed by WAITCICS commands of current tasks in the ECBLIST passed by CICS to the MVS WAIT facility when it runs out of work. Such ECBs can be posted using the MVS POST facility or by hand posting. Hand posting could, for example, be done by moving an appropriate value into the ECB. If hand posting is definitely not going to be used, it is preferable to use WAIT EXTERNAL.

A given ECB may not be waited on by more than one task at the same time. If this rule is not followed and the ECBLIST passed by CICS on the MVS WAIT contains duplicate ECB addresses, MVS abends CICS.

Figure 6 shows how to use the ECBLIST parameter to point to a list of ECB addresses that in turn point to individual ECBs. Note that the ECBLIST variable is a pointer pointing to the first address of the list.



```

DCL
| ECB1      FIXED BIN(31),    /* actual ecb */
| ECB2      FIXED BIN(31),    /* actual ecb */
| ECB3      FIXED BIN(31);    /* actual ecb */
| DCL                               /* list of ecb addresses */
|   1 ECB_ADDR_LIST,
|   2 ECB_ADDR(3) PTR;
| DCL                               /* ptr to each addr list */
|   PTR_ECB_ADDR_LIST PTR;
ECB_ADDR(1) = ADDR(ECB1);
ECB_ADDR(2) = ADDR(ECB2);
ECB_ADDR(3) = ADDR(ECB3);
                                   /* set up pointer */
PTR_ECB_ADDR_LIST = ADDR(ECB_ADDR_LIST);
/* PTR_ECB_ADDR_LIST = ADDR(ECB_ADDR(1));
   (alternative) */

EXEC CICS WAITCICS
        ECBLIST(PTR_ECB_ADDR_LIST)
        NUMEVENTS(3)
        PURGEABLE

```

Figure 6. ECBLIST option, EXEC CICS WAITCICS

WAITCICS

WAITCICS options

ECBLIST(ptr-value)

is a pointer to a list of addresses of MVS-format ECBs representing events. Both the ECBLIST and the ECBs can be above the 16MB line, that is they can be 31-bit addresses. Each ECB must be fullword aligned. Null (X'00000000' and X'FF000000') ECB addresses are ignored.

NAME(name)

specifies the symbolic name, 1–8 alphanumeric characters, as the reason for the wait. The value you specify is returned in the SUSPENDVALUE or HVALUE option respectively of the INQ TASK or CEMT INQ TASK commands.

NUMEVENTS(data-value)

is the number of such events, corresponding to the number of addresses in the ECBLIST. The field is fullword binary. When NUMEVENTS is specified as one, ECBLIST must still be an address that points to a list containing just one ECB address.

PURGEABILITY(cvda)

causes the issuing task to be suspended until one of the ECBs has been posted; that is, until one of the events has occurred. The values passed to CICS are PURGEABLE (the default value), or NOTPURGEABLE. The field is fullword binary. If while this task is waiting another function attempts to purge it, the result is as follows:

Function	PURGEABLE	NOTPURGEABLE
DTIMOUT expired	Abend AEXY	No effect
CEMT SET TASK PURGE EXEC CICS SET TASK PURGE	Abend AEXY	No effect
CEMT SET TASK FORCEPURGE EXEC CICS SET TASK FORCEPURGE	Abend AEXY	Abend AEXY

See the *CICS/ESA Recovery and Restart Guide* for information about DTIMOUT and SET TASK PURGE|FORCEPURGE.

WAITCICS condition

INVREQ

occurs in any of the following situations:

- An ECB is not valid, for example the ECB is not fullword aligned, (RESP2=1).
- NUMEVENTS is not a positive number (RESP2=3).
- PURGEABILITY is specified with an incorrect CVDA (RESP2=4).
- No valid ECBs have been found in the list, because either the ECBLIST address is not valid, or all the ECB addresses are not valid (RESP2=5).

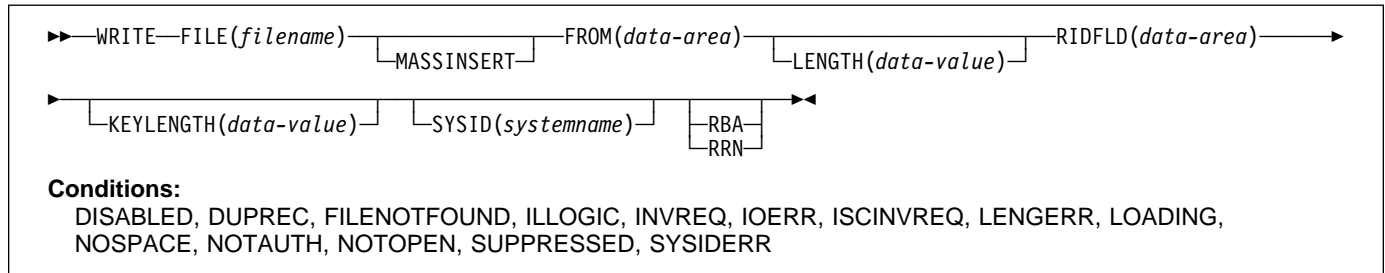
| Default action: terminate the task abnormally.

WRITE

Function

Write a record.

Command syntax



WRITE writes a new record to a file on a local or a remote system. For example:

```
EXEC CICS WRITE
      FROM(RECORD)
      LENGTH(DATLEN)
      FILE('MASTER')
      RIDFLD(KEYFLD)
```

When this command is used to write a record to a CICS-maintained data table, the update is made to both the source VSAM KSDS and the in-memory data table, unless the XDTAD user exit rejects the record from the table. The details of the command for a CICS-maintained table are the same as for a VSAM KSDS.

When this command is used to write a record to a user-maintained data table, the update is made to the in-memory data table.

For a VSAM ESDS, the record is always added at the end of the data set. VSAM does not use the identification field specified in RIDFLD when calculating the RBA of the new record, but the new RBA is returned to the application in the record identification field specified in the RIDFLD option.

For a VSAM KSDS, the record is added in the location specified by the associated key; this location may be anywhere in the data set. For VSAM data sets, the key in the record and the key in the RIDFLD identification field must be the same.

Records for ESDS and KSDS can be either fixed-length or variable-length. Those for a relative record data set must be fixed-length. MASSINSERT operations must proceed with ascending keys, and must be terminated by an UNLOCK before any other request to the same data set.

WRITE options

FILE(filename)

specifies the name of the file to be accessed.

If SYSID is specified, the data set to which this file refers is assumed to be on a remote system irrespective of whether the name is defined in the FCT. Otherwise, the FCT entry is used to find out whether the data set is on a local or a remote system.

FROM(data-area)

specifies the record that is to be written to the data set referred to by this file.

For user-maintained data tables, there is no need to include the RIDFLD option if the UMT has not been pre-loaded with a VSAM file.

KEYLENGTH(data-value)

specifies the length (halfword binary) of the key that has been specified in the RIDFLD option, except when RBA or RRN is specified, in which case KEYLENGTH is not valid. You must code KEYLENGTH if you are also using SYSID (unless you are also using RBA or RRN). If the length specified is different from the length defined for the data set, the INVREQ condition occurs.

LENGTH(data-value)

specifies the length, as a halfword binary value, of the record to be written.

You must specify this option if you are using SYSID.

You must also specify this option for a file with variable-length records (including user-maintained data tables). It need not be specified for fixed-length records, but its inclusion is recommended because it causes a check to be made that the record being written is not longer than that defined for the data set.

WRITE

MASSINSERT (VSAM)

specifies that the WRITE command is part of a mass-insert operation, that is, a series of WRITES each specifying MASSINSERT.

| You cannot use MASSINSERT for user-maintained data
| tables.

RBA (VSAM ESDS base)

specifies that the record identification field specified in
the RIDFLD option contains a relative byte address.
Use this option only when writing to an ESDS base.

| You cannot use RBA for user-maintained data tables.

RIDFLD(data-area)

specifies the record identification field. When combined
with RBA or RRN this is a 4-character field. The
contents can be a key, a relative byte address, or
relative record number (for VSAM data sets), or a block
reference, a physical key, and a deblocking argument
(for BDAM data sets). For a relative byte address or a
relative record number, the format of this field must be
fullword binary. If RBA is specified, RIDFLD contains
the relative byte address (greater than or equal to zero)
of the record that was written. If RRN is specified,
RIDFLD contains the relative record number (greater
than or equal to 1) of the record that was written.

When adding records to a keyed data set, the field must
contain the complete key.

RRN (VSAM RRDS)

specifies that the record identification field specified in
the RIDFLD option contains a relative record number.

SYSID(systemname)

specifies the name of the system to which the request is
directed.

If you specify SYSID, and omit both RBA and RRN, you
must also specify LENGTH and KEYLENGTH; they
cannot be found in the FCT.

LENGTH must either be specified explicitly or must be
capable of being defaulted from the FROM option using
the length attribute reference in assembler language, or
STG and CSTG in PL/I. LENGTH must be specified
explicitly in OS/VS COBOL or C.

WRITE conditions

Note: RESP2 values are not set for files that are on remote
systems.

DISABLED

occurs if a file is disabled (RESP2=50). A file may be
disabled because:

- It was initially defined as disabled and has not since
been enabled.
- It has been disabled by a SET FILE or a CEMT
SET FILE command.

Default action: terminate the task abnormally.

DUPREC

occurs if an attempt is made to add a record to a data
set, by referring to a file, or a path over a file (with the
UNIQUEKEY attribute), in which the same key already
exists (RESP2=150).

Default action: terminate the task abnormally.

FILENOTFOUND

occurs if a file name referred to in the FILE option
cannot be found in the FCT (RESP2=1).

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

occurs if a VSAM error occurs that does not fall within
one of the other CICS response categories
(RESP2=110).

(See EIBRCODE in the EXEC interface block; refer to
Appendix A, "EXEC interface block" on page 343 for
details.)

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- Add operations are not allowed according to the file
entry specification in the FCT (RESP2=20).
- When writing records containing embedded keys,
the key in the record area (FROM option) and the
key in RIDFLD do not match (RESP2=23).
- The KEYLENGTH option is specified, and the
specified length does not equal the length defined
for the data set that this file refers to (RESP2=26).
- A WRITE with the MASSINSERT option is issued
against a BDAM file (RESP2=38).
- A BDAM key conversion error occurred
(RESP2=40).
- The WRITE command does not conform to the
format of WRITE for a user-maintained data table
(RESP2=44).

Default action: terminate the task abnormally.

IOERR

occurs in any of the following situations:

- There is an I/O error during the file control operation
(RESP2=120). An I/O error is any unusual event
that is not covered by a CICS condition. (Further
information is available in the EXEC interface block;
refer to Appendix A, "EXEC interface block" on
page 343 for details.)
For VSAM files, IOERR normally indicates a
hardware error.
- You are trying to write to a BDAM track address that
is not defined for the data set (RESP2=120).

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition (RESP2=70).

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- The length specified for the write operation exceeds the maximum record size; the record is truncated (RESP2=12).
- An incorrect length is specified for a write operation involving fixed-length records (RESP2=14).
- LENGTH is omitted for a WRITE to a file with variable-length records or to a BDAM file with undefined-format records. (RESP2=10).

LOADING

occurs if a WRITE is issued to a user-maintained data table that is currently being loaded (RESP2=104). A user-maintained data table cannot be modified during loading.

Default action: terminate the task abnormally.

NOSPACE

occurs in the following situations:

- No space is available on the direct access device for adding records to a data set (RESP2=100).
- The maximum number of table entries specified for the user-maintained table has already been reached (RESP2=102).
- CICS is unable to get sufficient storage in the CICS address space to create an in-memory table entry for the record being written (RESP2=103).

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on FILE(filename) (RESP2=101).

Default action: terminate the task abnormally.

NOTOPEN

occurs in any of the following situations (RESP2=60):

- The requested file is CLOSED and UNENABLED. The CLOSED, UNENABLED state is reached after a CLOSE request has been received against an OPEN ENABLED file and the file is no longer in use. This state can also be specified as the initial state by means of the FILSTAT parameter of the file control table TYPE=FILE, or by defining a file using the RDO options STATUS=UNENABLED and OPENTIME=FIRSTREF.
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.

This condition does not occur if the request is made to either a CLOSED, ENABLED file or a CLOSED, DISABLED file. In the first case, the file is opened as part of executing the request. In the second case, the DISABLED condition occurs.

SUPPRESSED

occurs if a user exit program that is invoked at the XDTAD exit point decides not to add the record to the user-maintained data table (RESP2=105).

Default action: terminate the task abnormally.

SYSIDERR

occurs when the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed (RESP2=130).

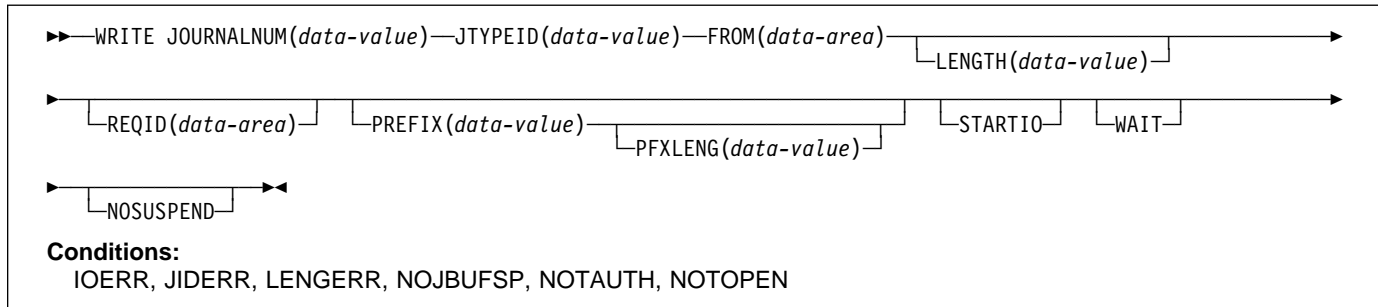
Default action: terminate the task abnormally.

WRITE JOURNALNUM

Function

Create a journal record.

Command syntax



`WRITE JOURNALNUM` creates a journal record. The request can be for synchronous or asynchronous output; definitions of these terms, and detailed information regarding the synchronization of journal output, are contained in the *CICS/ESA Application Programming Guide*.

The following example shows how to request synchronous journal output and wait for the output operation to be completed:

```
EXEC CICS WRITE
      JOURNALNUM(2)
      JTYPEID('XX')
      FROM(KEYDATA)
      LENGTH(8)
      PREFIX(PROGNAME)
      PFXLENG(6)
      WAIT
```

In this example, `STARTIO` is not specified, so the task waits until the journal buffer is full or until output is initiated by a `STARTIO` request in another task. CICS limits the wait to one second.

The following example shows how to request deferred (asynchronous) journal output:

```
EXEC CICS WRITE
      JOURNALNUM(1)
      JTYPEID('SD')
      FROM(COMDATA)
      LENGTH(10)
      REQID(ENTRYID)
```

WRITE JOURNALNUM options

FROM(data-area)

specifies the user data to be built into the journal record.

JOURNALNUM(data-value)

specifies a halfword numeric value in the range 1 through 99 to be taken as the journal identifier. `JOURNALNUM(1)` specifies the system log.

JTYPEID(data-value)

specifies a 2-character identifier to be placed in the journal record to identify its origin.

LENGTH(data-value)

specifies the length (halfword binary value) in bytes of the user data to be built into the journal record. The minimum value is 0 and the maximum value is (buffer size-76) minus `PFXLENG`.

NOSUSPEND

specifies that an application program is not to be suspended for the `NOJBUFSP` condition.

PFXLENG(data-value)

specifies the length (halfword binary value) in bytes of the user prefix data to be included in the journal record. The minimum value is 0 and the maximum value is (buffer size-72) minus `LENGTH`.

PREFIX(data-value)

specifies the user prefix data to be included in the journal record. A data area must be provided in COBOL programs.

REQID(data-area)

specifies a data area that identifies the journal record. The data area is a fullword binary variable. CICS sets the variable to a number that depends upon the position in the data set of the record being created. `REQID` applies to asynchronous output (`WAIT` option not specified).

STARTIO

specifies that output of the journal record is to be initiated immediately. If WAIT is specified for a journal with a low utilization, STARTIO should be specified also to prevent the requesting task waiting for the journal buffer to be filled. Very high utilization ensures that the buffer is cleared quickly, so that STARTIO is unnecessary.

WAIT

specifies that synchronous journal output is required. The requesting task waits until the record has been written.

WRITE JOURNALNUM conditions**IOERR**

occurs if the physical output of a journal record was not accomplished because of an irrecoverable I/O error.

Default action: terminate the task abnormally.

JIDERR

occurs if the specified journal identifier does not exist in the journal control table (JCT).

Default action: terminate the task abnormally.

LENGERR

occurs if the computed length for the journal record exceeds the total buffer space allocated for the journal, as specified in the JCT entry for the journal, or if the length specified for the prefix or for the data is negative.

Default action: terminate the task abnormally.

NOJBUFSP

occurs if the journal buffers are logically full (that is, the current buffer is full, and I/O is in progress on the alternate buffer) or the remaining space in the current buffer is insufficient for this journal record.

Default action: suspend task activity until the journal request can be satisfied. CICS ensures that both buffers are written out to auxiliary storage, thus freeing them for new records. (The default can be overridden by the NOSUSPEND option.)

NOTAUTH

occurs when a resource security check has failed on JOURNALNUM(data-value).

Default action: terminate the task abnormally.

NOTOPEN

occurs if the journal command cannot be satisfied because the specified journal is not open.

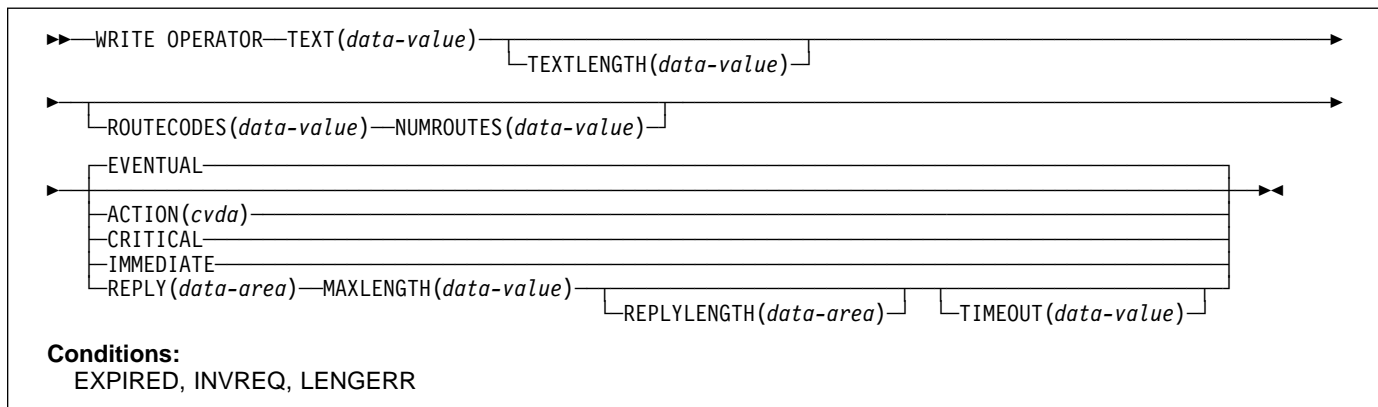
Default action: terminate the task abnormally.

WRITE OPERATOR

Function

Write a message on the system console.

Command syntax



`WRITE OPERATOR` enables an application to write a message to one or more system consoles and, if necessary, wait for a reply. The command can specify route codes. This is of particular use to application packages that need to issue their own operator messages.

As a result of a change in the way CICS handles messages sent to the console, text lengths of greater than 113 characters are split into two lines.

None of the variables below can be defined as PL/I variable character strings.

Note: If `ACTION` (or one of the equivalent `CVDA` values below) is specified, the message is retained by MVS until the console operator explicitly deletes it or CICS terminates.

The action code is identical with the MVS descriptor code to be associated with the message. Only one of the descriptor codes 2, 3, or 11 may be specified for this parameter.

If `ACTION` is not specified, no descriptor code is associated with the message. The descriptor codes have the following meanings:

- 2 Immediate action
- 3 Eventual action
- 11 Critical eventual action.

The `CRITICAL` option is equivalent to a specification of `ACTION(11)`. The `EVENTUAL` option is equivalent to a specification of `ACTION(3)`. The `IMMEDIATE` option is equivalent to a specification of `ACTION(2)`.

Messages retained by MVS can be handled by the console operator in a variety of ways (see the *MVS/ESA Operations: System Commands* manual). Refer to your system

programmer for information about how this command affects the appearance of the console screen to the operator.

WRITE OPERATOR options

Note: CICS/VSE, unlike CICS/ESA, does not support the options `ACTION`, `ROUTECODES`, and `NUMROUTES`. If a program containing any of these three options is translated on VSE, warning message DFH7117 is produced and the option ignored.

`ACTION(cvda)`

specifies an action code to be associated with this message. The `CVDA` values are:

- CRITICAL** specifies that the message requires eventual action by the operator and has enough critical importance to remain on the console screen. The message remains on the screen until it is deleted by the operator.
- EVENTUAL** specifies that the operator should take action when there is time. The message is rolled off when other messages fill up the screen, but is still retained by the operating system until the operator explicitly deletes it.
- IMMEDIATE** specifies that the operator should take action immediately. The message remains on the console screen until it is deleted by the operator.

MAXLENGTH(data-value)

is a fullword binary field that contains the length of the reply area (in the range 1–119 bytes). You must specify MAXLENGTH if you specify REPLY.

NUMROUTES(data-value)

is a fullword binary field that defines the number of routing codes.

REPLY(data-area)

provides a data area for receiving the operator's reply. If you specify this option, your application pauses until either a reply is received or the TIMEOUT period expires.

REPLYLENGTH(data-area)

specifies the actual length (fullword binary value) of the operator's reply.

ROUTECODES(data-value)

is a variable-length field. Each code is one byte and contains a binary number in the range 1–28. The default is a single code, set to 2. In COBOL programs only, you must use a data-area that contains the 1-byte values rather than a data-value.

TEXT(data-value)

is a data value containing the text to be sent.

```
# For COBOL programs to be compiled with a Language
# Environment-conforming compiler, and translated with
# the COBOL3 translator option, there is a restriction in
# the length of data-value, which cannot exceed 160
# bytes. If you are using an earlier compiler, such as VS
# COBOL II, that does not support the COBOL3 translator
# option, you must use a data-area that contains the text
# to be sent to the operator, and not a data-value.
```

TEXTLENGTH(data-value)

specifies the length (fullword binary value) of the text.

```
| If the REPLY option is specified, the length is in the
| range 0–121 bytes.
```

If the REPLY option is not specified, the length is in the range 0–690 bytes.

If the length of the text is greater than 113, CICS formats the message in a multiline write to operator (WTO); each line has 69 bytes with a maximum of ten lines.

```
# — Apar 85038 —
```

```
# Documentation for Apar 85038 added 11/07/96
```

```
# The output is edited in such a way that each line is
# broken, if possible, at a space character. The next line
# starts with a non-space character. If there is not room to
# reformat the data in this way within the overall limit of
# 690 bytes of ten lines of 69 bytes, the output is not
# reformatted.
```

TIMEOUT(data-value)

is a fullword binary field that contains the maximum time (in seconds) that CICS waits for a reply before returning

control to this transaction. This must be in the range 0–86 400 (24 hours). The system default value is specified in the OPERTIM system initialization table. You can only specify TIMEOUT if you have also specified REPLY.

WRITE OPERATOR conditions**EXPIRED**

TIMEOUT has occurred before the operator's reply was received (RESP2=7).

Default action: terminate the task abnormally.

INVREQ

occurs in any of the following situations:

- The TEXTLENGTH value is not valid (RESP2=1)
- The NUMROUTES value is not valid (RESP2=2)
- The ROUTECODE value is not valid (RESP2=3)
- The MAXLENGTH value is not valid (RESP2=4)
- The TIMEOUT value is not valid (RESP2=5)
- The ACTION value is not valid (RESP2=6).

Default action: terminate the task abnormally.

LENGERR

occurs when the reply was longer than MAXLENGTH, and has been truncated (RESP2=8).

Default action: terminate the task abnormally.

WRITEQ TD

Function

Write data to transient data queue.

Command syntax

```
▶—WRITEQ TD—QUEUE(name)—FROM(data-area)—┐LENGTH(data-value)—┐SYSID(systemname)—◀
```

Conditions:

DISABLED, INVREQ, IOERR, ISCINVREQ, LENGERR, NOSPACE, NOTAUTH, NOTOPEN, QIDERR, SYSIDERR

WRITEQ TD writes transient data to a predefined symbolic destination.

The following example shows how to write data to a predefined symbolic destination; in this case, the control system message log (CSML):

```
EXEC CICS WRITEQ TD
      QUEUE('CSML')
      FROM(MESSAGE)
      LENGTH(LENG)
```

WRITEQ TD options

FROM(*data-area*)

specifies the data that is to be written to the transient data queue.

LENGTH(*data-value*)

specifies the length (halfword binary value) of the data to be written.

QUEUE(*name*)

specifies the symbolic name (1–4 alphanumeric characters) of the queue to be written to. The name must have been defined in the DCT by the system programmer.

SYSID(*systemname*) — remote systems only

specifies the name (1–4 characters) of the system to which the request is directed.

If SYSID is specified, the data set is assumed to be on a remote system irrespective of whether or not the name is defined as remote in the DCT. Otherwise the entry in the DCT is used to find out whether the data set is on a local or a remote system.

WRITEQ TD conditions

DISABLED

occurs when the queue has been disabled.

Default action: terminate the task abnormally.

INVREQ

occurs if WRITEQ names an extrapartition queue that has been opened for input.

Note: This condition cannot be raised for intrapartition queues.

Default action: terminate the task abnormally.

IOERR

occurs when an input/output error occurs and the data record in error is skipped.

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LENGERR

occurs in any of the following situations:

- WRITEQ names an extrapartition queue and does not specify a length consistent with the RECFORM and RECSIZE options specified in the destination control table TYPE=SDSCI that defined the queue. The check is made after the XTDOU exit has been invoked; this exit may change the length of the data to be passed to the access method.
- WRITEQ names an intrapartition queue and does not specify a length consistent with the control interval defined for the intrapartition data set. Again, the check is made after the XTDOU exit has been invoked.

Default action: terminate the task abnormally.

NOSPACE

occurs if no more space exists on the intrapartition or extrapartition queue.

#

Apar PQ07370

#

The following paragraph was added by APAR PQ
07370 6/7/1998

#

It also occurs for an intrapartition queue that has been
defined with a size greater than 2 gigabytes, and the
RBA (relative byte address) of the entry being written
exceeds 2 gigabytes. When this happens, no more data
should be written to the queue because it may be lost.

Default action: terminate the task abnormally.

NOTAUTH

occurs when a resource security check has failed on
QUEUE(name).

Default action: terminate the task abnormally.

NOTOPEN

occurs if the destination is closed.

Note: This condition cannot be raised for intrapartition
queues.

Default action: terminate the task abnormally.

QIDERR

occurs if the symbolic destination to be used with a
transient data control command cannot be found.

Default action: terminate the task abnormally.

SYSIDERR

| occurs when the SYSID option specifies a name that is
| neither the local system nor a remote system (made
| known to CICS by defining a CONNECTION).
| SYSIDERR also occurs when the link to the remote
| system is closed.

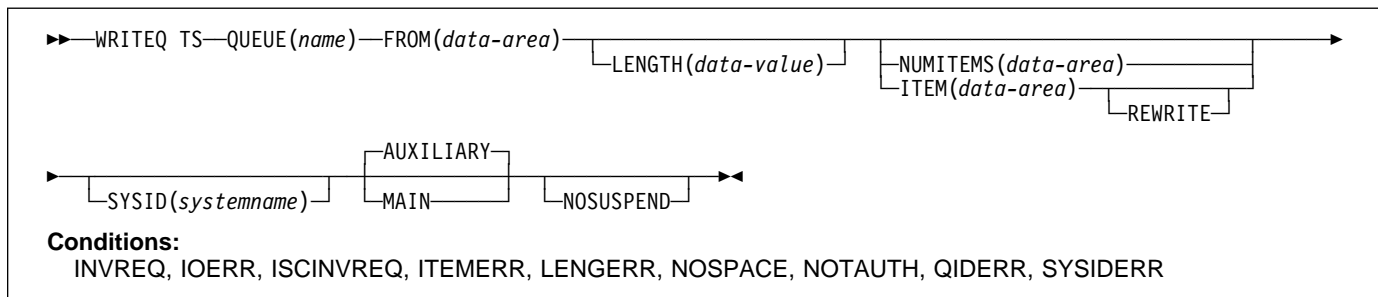
Default action: terminate the task abnormally.

WRITEQ TS

Function

Write data to a temporary storage queue.

Command syntax



Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS/ESA Application Programming Guide* for more information about transaction affinities.

WRITEQ TS stores temporary data (records) in a temporary storage queue in main or auxiliary storage.

If a queue has been defined as recoverable, the program must not issue a WRITEQ TS if a DELETEQ TS has previously been issued within the same logical unit of work. In other words, following a DELETEQ TS, no WRITEQ TS can be issued until after a syncpoint has occurred.

If there is insufficient space available in the temporary storage data set to satisfy the WRITEQ TS request, the task is suspended until space does become available. (Space may be released by other tasks in the system.) However, if space is not available in the temporary storage data set, and the NOSUSPEND option has been specified, the NOSPSPACE condition is raised.

WRITEQ TS options

AUXILIARY

specifies that the temporary storage queue is on a direct access storage device in auxiliary storage. This option is ignored for an existing queue.

This is the default value for the first write.

FROM(data-area)

specifies the data to be written to temporary storage.

ITEM(data-area)

specifies, as a halfword binary value, the item number of the logical record to be replaced in the queue (REWRITE option also specified).

#

#

#

#

#

#

#

#

#

Apar 66387

Documentation for Apar 66387 added 13 Feb 1995 (TUCKER)

ITEM can be both an input and output field to CICS. As such, programmers should ensure that the ITEM field is not defined within protected storage when issuing a WRITEQ command. If the ITEM value were a literal (say), command checking (CMDPROT=YES) would result in an AEYD abend occurring.

Note: In earlier releases, ITEM on a WRITEQ TS without REWRITE would perform a similar function to NUMITEMS. This function is retained for compatibility.

LENGTH(data-value)

specifies the length, as a halfword binary value, of the data to be written.

You must specify this option if you are using SYSID.

The maximum length is 32763. For a description of a safe upper limit, see "LENGTH options" on page 5.

MAIN

specifies that the temporary storage queue is in main storage. This option is ignored for an existing queue.

If you use the MAIN option to write data to a temporary storage queue on a remote system, the data is stored in main storage if the remote system is accessed by the CICS multiregion operation (MRO) facility. If these conditions are not met, the data is stored in auxiliary storage.

| If the system is MRO and MAIN is specified, the queue
| is not recoverable and SYNCPOINT ROLLBACK does
| not function.

NOSUSPEND

| specifies that the application program is not to be
| suspended if there is insufficient space in the temporary
| storage data set to satisfy the WRITEQ TS request.
| Instead, the NOSPACE condition is raised.

This does not apply to the MAIN queue.

NUMITEMS(data-area)

specifies a halfword binary field into which CICS stores a number indicating how many items there are now in the queue, after the WRITEQ TS command is executed.

If the record starts a new queue, the item number assigned is 1; subsequent item numbers follow on sequentially. NUMITEMS is not valid if REWRITE is specified.

QUEUE(name)

| specifies the symbolic name (1-8 characters) of the
| queue to be written to. If the queue name appears in
| the TST, and the entry is marked as remote, the request
| is shipped to a remote system. The name must be
| unique within the CICS system. Do not use X'FA'
| through X'FF', or **, or \$\$, or DF, as the first character
| of the name; these characters are reserved for CICS
| use. The name cannot consist solely of binary zeros.

REWRITE

specifies that the existing record in the queue is to be overwritten with the data provided. If the REWRITE option is specified, the ITEM option must also be specified. If the specified queue does not exist, the QIDERR condition occurs. If the correct item within an existing queue cannot be found, the ITEMERR condition occurs and the data is not stored.

SYSID(systemname) — remote systems only

specifies the name (1–4 characters) of the system to which the request is directed.

WRITEQ TS conditions**INVREQ**

occurs in any of the following situations:

- A WRITEQ TS command specifies a queue name that consists solely of binary zeros
- A WRITEQ TS command specifies a queue that is locked and awaiting ISC session recovery
- The queue was created by CICS internal code

Default action: terminate the task abnormally.

IOERR

occurs when there is an irrecoverable input/output error.

Default action: terminate the task abnormally.

ISCINVREQ

occurs when the remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

WRITEQ TS

ITEMERR

occurs in any of the following situations:

- | • The item number specified in a WRITEQ TS command with the REWRITE option, is not valid (that is, it is outside the range of entry numbers assigned for the queue).
- The maximum number of items (32 767) is exceeded.

Default action: terminate the task abnormally.

LENGERR

| occurs in any of the following situations:

- | • The length of the stored data is zero or negative.
- # • The length of the stored data is greater than 32763.
- | • A destructive overlay has occurred.

Default action: terminate the task abnormally.

NOSPACE

occurs when insufficient space is available in the temporary storage data set to contain the data, and the NOSUSPEND option is specified.

Default action: ignore the condition.

NOTAUTH

occurs when a resource security check has failed on QUEUE(name).

Default action: terminate the task abnormally.

QIDERR

occurs when the queue specified by a WRITEQ TS command with the REWRITE option cannot be found, either in main or in auxiliary storage.

Default action: terminate the task abnormally.

SYSIDERR

| occurs when the SYSID option specifies a name that is
| neither the local system nor a remote system (made
| known to CICS by defining a CONNECTION).
SYSIDERR also occurs when the link to the remote
system is closed.

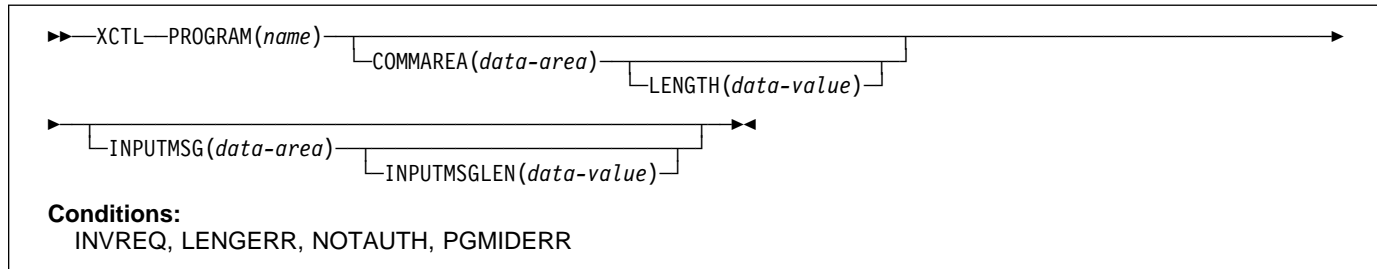
Default action: terminate the task abnormally.

XCTL

Function

Transfer program control.

Command syntax



XCTL transfers control from one application program to another at the same logical level. The program from which control is transferred is released. If the program to which control is transferred is not already in main storage, it is loaded.

The following example shows how to request a transfer of control to an application program called PROG2:

```
EXEC CICS XCTL PROGRAM('PROG2')
```

XCTL options

COMMAREA(data-area)

specifies a communication area to be made available to the invoked program. In this option a pointer to the data area is passed. In COBOL, you must give this data area the name DFHCOMMAREA in the receiving program. (See the section about passing data to other programs in the *CICS/ESA Application Programming Guide*.)

INPUTMSG(data-area)

specifies data to be passed to the invoked program when it first issues a RECEIVE command. If the invoked program passes control to another program by a LINK command, a linked chain is created, as described under the INPUTMSG option of the LINK command. The INPUTMSG data remains available until a RECEIVE command is issued or until control returns to CICS.

INPUTMSGLEN(data-value)

specifies a halfword binary value that specifies the length of the data passed by INPUTMSG.

LENGTH(data-value)

specifies the length (halfword binary data value) in bytes of the communication area. For a description of a safe upper limit, see "LENGTH options" on page 5.

PROGRAM(name)

specifies the identifier (1–8 alphanumeric characters) of the program to which control is to be passed unconditionally. The specified name must have been

defined as a program to CICS, though if AUTOINSTALL is active a definition is autoinstalled.

XCTL conditions

INVREQ

occurs in the following circumstances:

- An XCTL command with the INPUTMSG option is issued for a program that is not associated with a terminal, or that is associated with an APPC logical unit, or an IRC session (RESP2=8).
- EXEC XCTL is not allowed in a GLUE or TRUE (RESP2=29).
- The program manager domain has not yet been initialized. This is probably due to a XCTL request having been made in a first stage PLT (RESP2=30).
- An XCTL command with the INPUTMSG option is issued in a program invoked by DPL (RESP2=200).

Default action: terminate the task abnormally.

LENGERR

occurs in the following conditions:

- LENGTH is less than 0 or greater than 32763 (RESP2=11).
- The COMMAREA address passed was zero, but LENGTH was non-zero (RESP2=26).
- INPUTMSGLEN was less than 0 or greater than 32767 (RESP2=27).
- LENGTH or INPUTMSGLEN is greater than the length of the data area specified in the COMMAREA or INPUTMSG options, and while that data was being copied a destructive overlap occurred because of the incorrect length (RESP2=28).

Default action: terminate the task abnormally.

XCTL

NOTAUTH

occurs when a resource security check has failed on PROGRAM(name) (RESP2=101).

Default action: terminate the task abnormally.

PGMIDERR

occurs in any of the following situations:

- A program has no entry in the PPT and either program autoinstall was switched off, or the program autoinstall control program indicated that the program should not be autoinstalled (RESP2=1).
- The program is disabled (RESP2=2).
- A program could not be loaded because
 - This was the first load of the program and the program load failed, usually because the load module could not be found.
 - This was a subsequent load of the program, but the first load failed.

In order to reset the load status the load module must be in the DFHRPL concatenation, and a SET PROGRAM NEWCOPY will be required (RESP2=3).

- The installed program definition is for a remote program (RESP2=9).
- The program autoinstall control program failed either because the program autoinstall control program is incorrect, incorrectly defined, or as a result of an abend in the program autoinstall control program. Program autoinstall is disabled and message DFHPG0202 or DFHPG0203 written to the CSPL (RESP2=21).
- The model returned by the program autoinstall control program was not defined in the PPT table, or was not enabled (RESP2=22).
- The program autoinstall control program returned invalid data (RESP2=23).
- Define for the program failed due to autoinstall returning an invalid program name or definition (RESP2=24).

Default action: terminate the task abnormally.

Appendix A. EXEC interface block

This appendix contains a description of the EXEC interface block (EIB). An application program can read all the fields in the EIB of the associated task by name, but must not directly change the contents of any of them.

For each field, the contents and format (for each of the application programming languages COBOL, C, PL/I, and ASM) are given. All fields contain zeros in the absence of meaningful information. Fields are listed in alphabetical order.

EIB fields

EIBAID

contains the attention identifier (AID) associated with the last terminal control or basic mapping support (BMS) input operation from a display device such as the 3270.

```
COBOL: PIC X(1).
C:      unsigned char eibaid;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBATT

indicates that the RU contains attach header data (X'FF').

```
COBOL: PIC X(1).
C:      unsigned char eibatt;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBCALEN

contains the length of the communication area that has been passed to the application program from the last program, using the COMMAREA and LENGTH options. If no communication area is passed, this field contains zeros.

```
COBOL: PIC S9(4) COMP.
C:      short int eibcalen;
PL/I:   FIXED BIN(15)
ASM:    H
```

EIBCOMPL

indicates, on a terminal control RECEIVE command, whether the data is complete (X'FF'). If the NOTRUNCATE option has been used on the RECEIVE command, CICS retains data in excess of the amount requested via the LENGTH or MAXLENGTH option. EIBRECV is set indicating that further RECEIVE commands are required. EIBCOMPL is not set until the last of the data has been retrieved.

EIBCOMPL is always set when a RECEIVE command without the NOTRUNCATE option is executed.

```
COBOL: PIC X(1).
C:      unsigned char eibcompl;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBCONF

indicates that a CONFIRM request has been received (X'FF') for an APPC conversation.

```
COBOL: PIC X(1).
C:      unsigned char eibconf;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBCPOSN

contains the cursor address (position) associated with the last terminal control or basic mapping support (BMS) input operation from a display device such as the 3270.

```
COBOL: PIC S9(4) COMP.
C:      short int eibcposn;
PL/I:   FIXED BIN(15)
ASM:    H
```

EIBDATE

contains the date the task is started; this field is updated by the ASKTIME command. The date is in packed decimal form (0CYYDDD+) where C shows the century with values 0 for the 1900s and 1 for the 2000s. For example, the dates 31 December 1999 and 1 January 2000 have EIBDATE values of 0099365 and 0100001 respectively.

```
COBOL: PIC S9(7) COMP-3.
C:      char eibdate [4];
PL/I:   FIXED DEC(7,0)
ASM:    PL4
```

EIBDS

contains the symbolic identifier of the last data set referred to in a file control request.

```
COBOL: PIC X(8).
C:      char eibds [8];
PL/I:   CHAR(8)
ASM:    CL8
```

EIBEOC

indicates that an end-of-chain indicator appears in the RU just received (X'FF').

```
COBOL: PIC X(1).
C:      unsigned char eibeoc;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBERR • EIBFN

EIBERR

indicates that an error has been received (X'FF') on an APPC conversation.

```
COBOL: PIC X(1).
C:      unsigned char eiberr;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBERRCD

when EIBERR is set, contains the error code that has been received. The following values can be returned in the first two bytes of EIBERRCD:

```
X'0889' Conversation error detected
X'0824' SYNCPOINT ROLLBACK requested.
```

```
COBOL: PIC X(4).
C:      char eiberrcd [4];
PL/I:   CHAR(4)
ASM:    CL4
```

See the *CICS/ESA Distributed Transaction Programming Guide* for information about other EIBERRCD values that can occur.

EIBFMH

indicates that the user data just received or retrieved contains an FMH (X'FF').

```
COBOL: PIC X(1).
C:      unsigned char eibfmh;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBFN

contains a code that identifies the last CICS command issued by the task.

```
COBOL: PIC X(2).
C:      char eibfn [2];
PL/I:   CHAR(2)
ASM:    CL2
```

Code Command

```
0202 ADDRESS
0204 HANDLE CONDITION
0206 HANDLE AID
0208 ASSIGN
020A IGNORE CONDITION
020C PUSH
020E POP
0210 ADDRESS SET

0402 RECEIVE
0404 SEND
0406 CONVERSE
0408 ISSUE EODS
040A ISSUE COPY
040C WAIT TERMINAL
040E ISSUE LOAD
0410 WAIT SIGNAL
0412 ISSUE RESET
0414 ISSUE DISCONNECT
0416 ISSUE ENDOUTPUT
0418 ISSUE ERASEAUP
```

```
041A ISSUE ENDFILE
041C ISSUE PRINT
041E ISSUE SIGNAL
0420 ALLOCATE
0422 FREE
0424 POINT
0426 BUILD ATTACH
0428 EXTRACT ATTACH
042A EXTRACT TCT
042C WAIT CONVID
042E EXTRACT PROCESS
0430 ISSUE ABEND
0432 CONNECT PROCESS
0434 ISSUE CONFIRMATION
0436 ISSUE ERROR
0438 ISSUE PREPARE
043A ISSUE PASS
043C EXTRACT LOGONMSG
043E EXTRACT ATTRIBUTES

0602 READ
0604 WRITE
0606 REWRITE
0608 DELETE
060A UNLOCK
060C STARTBR
060E READNEXT
0610 READPREV
0612 ENDBR
0614 RESETBR

0802 WRITEQ TD
0804 READQ TD
0806 DELETEQ TD

0A02 WRITEQ TS
0A04 READQ TS
0A06 DELETEQ TS

0C02 GETMAIN
0C04 FREEMAIN

0E02 LINK
0E04 XCTL
0E06 LOAD
0E08 RETURN
0E0A RELEASE
0E0C ABEND
0E0E HANDLE ABEND

1002 ASKTIME
1004 DELAY
1006 POST
1008 START
100A RETRIEVE
100C CANCEL

1202 WAIT EVENT
1204 ENQ
1206 DEQ
1208 SUSPEND

1402 WRITE JOURNALNUM
1404 WAIT JOURNALNUM
```


1602 SYNCPOINT

1802 RECEIVE MAP
 1804 SEND MAP
 1806 SEND TEXT
 1808 SEND PAGE
 180A PURGE MESSAGE
 180C ROUTE
 180E RECEIVE PARTN
 1810 SEND PARTNSET
 1812 SEND CONTROL

1C02 DUMP

1E02 ISSUE ADD
 1E04 ISSUE ERASE
 1E06 ISSUE REPLACE
 1E08 ISSUE ABORT
 1E0A ISSUE QUERY
 1E0C ISSUE END
 1E0E ISSUE RECEIVE
 1E10 ISSUE NOTE
 1E12 ISSUE WAIT
 1E14 ISSUE SEND

2002 BIF DEEDIT

4802 ENTER TRACENUM
 4804 MONITOR

4A02 ASKTIME ABSTIME
 4A04 FORMATTIME

|
 | 5602 SPOOLOPEN
 | 5604 SPOOLREAD
 | 5606 SPOOLWRITE
 | 5610 SPOOLCLOSE

5E06 CHANGE TASK
 5E22 WAIT EXTERNAL
 | 5E32 WAITCICS

6A02 QUERY SECURITY
 6C02 WRITE OPERATOR
 6C12 ISSUE DFHWTO

7402 SIGNON
 7404 SIGNOFF
 | 7406 VERIFY PASSWORD
 | 7408 CHANGE PASSWORD

7E02 DUMP TRANSACTION

Codes 82nn and 84nn are used by the CICS/ESA Front End Programming Interface. See the *CICS/ESA Front End Programming Interface User's Guide* for a description.

EIBFREE

indicates that the application program cannot continue using the facility. The application program should either free the facility or should terminate so that the facility is freed by CICS (X'FF').

COBOL: PIC X(1).
 C: unsigned char eibfree;
 PL/I: CHAR(1)
 ASM: CL1

EIBNODAT

indicates that no data has been sent by the remote application (X'FF'). A message has been received from the remote system that conveyed only control information. For example, if the remote application executed a SEND command with the WAIT option, any data would be sent across the link. If the remote application then executed a SEND INVITE command without using the FROM option to transmit data at the same time, it would be necessary to send the INVITE instruction across the link by itself. In this case, the receiving application finds EIBNODAT set. The use of this field is restricted to application programs holding conversations across APPC links only.

COBOL: PIC X(1).
 C: unsigned char eibnodat;
 PL/I: CHAR(1)
 ASM: CL1

EIBRCODE

contains the CICS response code returned after the function requested by the last CICS command to be issued by the task has been completed.

Note: For new commands where EIBRESP and EIBRESP2 are the strategic means of interrogating the resulting condition of an executed command, byte 3 of EIBRCODE has the same value as EIBRESP. Any further information is in EIBRESP2 rather than EIBRCODE. For a normal response, this field contains hexadecimal zeros (6 X'00').

Almost all of the information in this field can be used within application programs by the HANDLE CONDITION command.

COBOL: PIC X(6).
 C: char eibrancode [6];
 PL/I: CHAR(6)
 ASM: CL6

The following list contains the values of the bytes together with the names of the conditions associated with the return codes.

See the notes at the end of the list of values for explanations of the numbers following some of the conditions.

EIBRCODE

EIBFN	EIBRCODE	Condition	EIBFN	EIBRCODE	Condition
02 ..	E0	INVREQ	0A ..	D1	ISCINVREQ
04 ..	04	EOF	0A ..	D6	NOTAUTH
04 ..	10	EODS	0A ..	E1	LENGERR
04 ..	C1	EOF	0C ..	E1	LENGERR
04 ..	C2	ENDINPT	0C ..	E2	NOSTG
04 ..	D0	SYSIDERR (see note 1)	0E ..	01	PGMIDERR
04 ..	D2	SESSIONERR (see note 2)	0E ..	D0	SYSIDERR (see note 1)
04 ..	D3	SYSBUSY (see note 3)	0E ..	D6	NOTAUTH
04 ..	D4	SESSBUSY	0E ..	E0	INVREQ
04 ..	D5	NOTALLOC	0E ..	F1	TERMERR
04 ..	E0	INVREQ (see note 4)	0E ..	E1	LENGERR
04 ..	E1	LENGERR (see note 5)	10 ..	01	ENDDATA
04 ..	E3	WRBRK	10 ..	04	IOERR
04 ..	E4	RDATT	10 ..	11	TRANSIDERR
04 ..	E5	SIGNAL	10 ..	12	TERMIDERR
04 ..	E6	TERMIDERR	10 ..	20	EXPIRED
04 ..	E7	NOPASSBKRD	10 ..	81	NOTFND
04 ..	E8	NOPASSBKWR	10 ..	D0	SYSIDERR (see note 1)
04 ..	EA	IGREQCD	10 ..	D1	ISCINVREQ
04 ..	EB	CBIDERR	10 ..	D6	NOTAUTH
04 ..	EC	PARTNERIDERR	10 ..	D8	USERIDERR
04 ..	ED	NETNAMEIDERR	10 ..	E1	LENGERR
04 ..	F1	TERMERR	10 ..	E9	ENVDEFERR
04 20	EOC	10 ..	FF	INVREQ
04 40	INBFMH	12 ..	32	ENQBUSY
04 F6	NOSTART	12 ..	E0	INVREQ
04 F7	NONVAL	12 ..	E1	LENGERR
06 ..	01	FILENOTFOUND	14 ..	01	JIDERR
06 ..	02	ILLOGIC (see note 6)	14 ..	02	INVREQ
06 ..	08	INVREQ	14 ..	05	NOTOPEN
06 ..	0C	NOTOPEN	14 ..	06	LENGERR
06 ..	0D	DISABLED	14 ..	07	IOERR
06 ..	0F	ENDFILE	14 ..	09	NOJBUFSP
06 ..	80	IOERR (see note 6)	14 ..	D6	NOTAUTH
06 ..	81	NOTFND	16 ..	01	ROLLEDBACK
06 ..	82	DUPREC	18 ..	01	INVREQ
06 ..	83	NOSPACE	18 ..	02	RETPAGE
06 ..	84	DUPKEY	18 ..	04	MAPFAIL
06 ..	85	SUPPRESSED	18 ..	08	INVMPSZ (see note 7)
06 ..	86	LOADING	18 ..	20	INVERRTERM
06 ..	D0	SYSIDERR (see note 1)	18 ..	40	RTESOME
06 ..	D1	ISCINVREQ	18 ..	80	RTEFAIL
06 ..	D6	NOTAUTH	18 ..	E1	LENGERR
06 ..	E1	LENGERR	18 ..	E3	WRBRK
08 ..	01	QZERO	18 ..	E4	RDATT
08 ..	02	QIDERR	18 02	PARTNFAIL
08 ..	04	IOERR	18 04	INVPARTN
08 ..	08	NOTOPEN	18 08	INVPARTNSET
08 ..	10	NOSPACE	18 10	INVLDC
08 ..	C0	QBUSY	18 20	UNEXPIN
08 ..	D0	SYSIDERR (see note 1)	18 40	IGREQCD
08 ..	D1	ISCINVREQ	18 80	TSIOERR
08 ..	D6	NOTAUTH	18 01	OVERFLOW
08 ..	D7	DISABLED	18 04	EODS
08 ..	E0	INVREQ	18 08	EOC
08 ..	E1	LENGERR	18 10	IGREQID
0A ..	01	ITEMERR	1A ..	E0	INVREQ
0A ..	02	QIDERR	1A ..	04	DSSTAT
0A ..	04	IOERR			
0A ..	08	NOSPACE			
0A ..	20	INVREQ			
0A ..	D0	SYSIDERR (see note 1)			

EIBFN	EIBRCODE	Condition
1A ..	08	FUNCERR
1A ..	0C	SELNERR
1A ..	10	UNEXPIN
1A ..	E1	LENGERR
1A 11	EODS
1A 2B	IGREQCD
1A 20	EOC
22 ..	80	INVEXITREQ
4A 01	INVREQ
56 0D	NOTFND
56 10	INVREQ
56 13	NOTOPEN
56 14	ENDFILE
56 15	ILLOGIC
56 16	LENGERR
56 2A	NOSTG
56 46	NOTAUTH
56 50	NOSPOOL
56 55	ALLOCERR
56 56	STRELERR
56 57	OPENERR
56 58	SPOLBUSY
56 59	SPOLEERR
56 5A	NODEIDERR

Notes:

1. When SYSIDERR occurs, further information is provided in bytes 1 and 2 of EIBRCODE, as follows:

```

.. 04 00 .. .. .. request was for a function
                    that is not valid.
.. 04 04 .. .. .. no session available and
                    NOQUEUE.
.. 04 08 .. .. .. modename not found2.
.. 04 0C .. .. .. modename not valid2.
.. 04 10 .. .. .. task canceled or timed
                    out during allocation2.
.. 04 14 .. .. .. mode group is out of
                    service2.
.. 04 18 .. .. .. close - DRAIN=ALL2.
.. 08 .. .. .. sysid is not available.
.. 08 00 .. .. .. no session available,
                    all sessions are out
                    of service, or released,
                    or being quiesced.
.. 08 04 .. .. .. no session available,
                    request to queue rejected
                    by XZIQUE global user
                    exit program.
.. 08 08 .. .. .. no session available;
                    request rejected by XZIQUE
                    global user exit program.
.. 0C xx .. .. .. sysid definition error.
.. 0C 00 .. .. .. name not that of TCTSE.
.. 0C 04 .. .. .. name not that of remote
                    TCTSE.
.. 0C 08 .. .. .. mode name not found.
.. 0C 0C .. .. .. profile not found

```

Further information about SYSIDERR can be found in the *CICS/ESA Intercommunication Guide*.

2. When SESSIONERR occurs, further information is provided in bytes 1 and 2 of EIBRCODE, as follows:

```

.. 08 .. .. .. session out of service
.. 0C xx .. .. session definition error
.. 0C 00 .. .. name not found
.. 0C 0C .. .. profile not found.

```

Further information about SESSIONERR can be found in the *CICS/ESA Intercommunication Guide*.

² For APPC only.

3. If SYSBUSY occurs on an ALLOCATE command that attempts to acquire a session to an APPC terminal or system, byte 3 of the EIBRCODE indicates where the error condition was detected:

- 00 the request was for a session to a connected terminal or system.
- 01 the request was for a session to a remotely connected terminal or system, and the error occurred in the terminal-owning region (TOR) or an intermediate system.
- 02 the request was for a session to a remotely connected terminal or system, and the error occurred in the application-owning region (AOR).

Further information about SYSBUSY can be found in the *CICS/ESA Intercommunication Guide*.

4. When INVREQ occurs during terminal control operations, further information is provided in bytes 1 or 3 of EIBRCODE as follows:

- .. 24 ISSUE PREPARE command - STATE error.
- 04 ALLOCATE command - TCTTE already allocated.
- 08 FREE command - TCTTE in wrong state.
- 0C CONNECT PROCESS command - SYNCLVL 2 requested, but cannot be supported on the session in use.
- 10 EXTRACT ATTACH command - incorrect data.
- 14 SEND command - CONFIRM option specified, but conversation not SYNCLVL 1.
- 18 EXTRACT TCT command - incorrect netname.
- 1C an incorrect command has been issued for the terminal or logical unit in use.
- 20 an incorrect command has been issued for the LUTYPE6.2 conversation type in use.
- 28 GETMAIN failure on ISSUE PASS command.

5. When LENGERR occurs during terminal control operations, further information is provided in byte 1 of EIBRCODE, as follows:

- .. 00 input data is overlong and has been truncated.
- .. 04 on output commands, an incorrect (FROM)LENGTH has been specified, either less than zero or greater than 32 767.
- .. 08 on input commands, an incorrect (TO)LENGTH has been specified, greater than 32 767.
- .. 0C length error has occurred on ISSUE PASS command.

Note: This field is not used exclusively for the above and may take other values.

6. When ILLOGIC or IOERR occurs during file control operations, further information is provided in field EIBRCODE, as follows:

- .. xx xx xx xx .. BDAM response.
- .. xx VSAM return code.
- xx VSAM error code.

Details of these response codes are given in the *DFSMS/MVS Macro Instructions for Data Sets* manual for VSAM, and the *Data Facility Product Customization Guide* for BDAM.

7. When INVMPsz occurs during BMS operations, byte 3 of field EIBRCODE contains the terminal code:

- xx terminal code.

These are the same as the mapset suffixes shown in Table 27 on page 400.

8. Codes 82nn and 84nn are used by the CICS/ESA Front End Programming Interface. See the *CICS/ESA Front End Programming Interface User's Guide* for a description.

EIBRECV

indicates that the application program is to continue receiving data from the facility by executing RECEIVE commands (X'FF').

```
COBOL: PIC X(1).
C:      unsigned char eibrecv;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBREQID

contains the request identifier assigned to an interval control command by CICS; this field is not used when a request identifier is specified in the application program.

```
COBOL: PIC X(8).
C:     char eibreqid [8];
PL/I:  CHAR(8)
ASM:   CL8
```

EIBRESP

contains a number corresponding to the RESP condition that occurred. These numbers are listed below (in decimal) for the conditions that can occur during execution of the commands described in this manual.

COBOL: PIC S9(8) COMP
 C: long int eibresp;
 PL/I: FIXED BIN(31)
 ASM: F

No.	Condition	No.	Condition
00	NORMAL	44	QIDERR
		45	NOJBUFSP
01	ERROR	46	DSSTAT
02	RDATT	47	SELNERR
03	WRBRK	48	FUNCERR
04	EOF	49	UNEXPIN
05	EODS		
06	EOC	50	NOPASSBKRD
07	INBFMH	51	NOPASSBKWR
08	ENDINPT	53	SYSIDERR
09	NONVAL	54	ISCINVREQ
		55	ENQBUSY
10	NOSTART	56	ENVDEFERR
11	TERMIDERR	57	IGREQCD
12	FILENOTFOUND	58	SESSIONERR
13	NOTFND	59	SYSBUSY
14	DUPREC		
15	DUPKEY	60	SESSBUSY
16	INVREQ	61	NOTALLOC
17	IOERR	62	CBIDERR
18	NOSPACE	63	INVEXITREQ
19	NOTOPEN	64	INVPARTNSET
		65	INVPARTN
20	ENDFILE	66	PARTNFAIL
21	ILLOGIC	69	USERIDERR
22	LENGERR		
23	QZERO	70	NOTAUTH
24	SIGNAL	72	SUPPRESSED
25	QBUSY		
26	ITEMERR	80	NOSPOOL
27	PGMIDERR	81	TERMERR
28	TRANSIDERR	82	ROLLEDBACK
29	ENDDATA	84	DISABLED
		85	ALLOCERR
31	EXPIRED	86	STRELERR
32	RETPAGE	87	OPENERR
33	RTEFAIL	88	SPOLBUSY
34	RTESOME	89	SPOLEERR
35	TSIOERR	90	NODEIDERR
36	MAPFAIL	91	TASKIDERR
37	INVERRTERM	92	TCIDERR
38	INVMPSTZ	93	DSNNOTFOUND
39	IGREQID	94	LOADING
		95	MODELIDERR
40	OVERFLOW	96	OUTDESCERR
41	INVLDC	97	PARTNERIDERR
42	NOSTG	98	PROFILEIDERR
43	JIDERR	99	NETNAMEIDERR

EIBRESP2

contains more detailed information that may help explain why the RESP condition occurred. This field contains meaningful values, as documented with each command

to which it applies. For requests to remote files, EIBRESP2 contains zeros.

For programs written in C, any value passed via the *exit* or *return* function is saved in EIBRESP2.

COBOL: PIC S9(8) COMP.
 C: long int eibresp2;
 PL/I: FIXED BIN(31)
 ASM: F

EIBRLDBK

indicates rollback.

COBOL: PIC X(1).
 C: unsigned char eibrldbkb;
 PL/I: CHAR(1)
 ASM: CL1

EIBRSRCE

contains the symbolic identifier of the resource being accessed by the latest executed command:

Type of command	Resource	Number of characters (H=halfword binary)
BMS	Map name	7
File control	File name	8
Interval control	Transaction name	4
Journal control	Journal number	H
Program control	Program name	8
# Temporary storage control	TS queue name	8
Terminal control	Terminal or logical unit identifier or LU6.1 session or APPC conversation identifier	4
Transient data control	TD queue name	4

Identifiers less than eight characters in length are padded on the right with blanks.

COBOL: PIC X(8).
 C: char eibrsrce [8];
 PL/I: CHAR(8)
 ASM: CL8

EIBSIG

indicates that SIGNAL has been received (X'FF').

COBOL: PIC X(1).
 C: unsigned char eibsig;
 PL/I: CHAR(1)
 ASM: CL1

EIBSYNC • EIBTRNID

EIBSYNC

indicates that the application program must take a syncpoint or terminate. Before either is done, the application program must ensure that any other facilities, owned by it, are put into the send state, or are freed (X'FF').

```
COBOL: PIC X(1).  
C:      unsigned char eibsync;  
PL/I:   CHAR(1)  
ASM:    CL1
```

EIBSYNRB

indicates that the application program should issue a SYNCPOINT ROLLBACK command (X'FF'). This field is only set in application programs holding a conversation on an APPC or MRO link.

```
COBOL: PIC X(1).  
C:      unsigned char eibsynrb;  
PL/I:   CHAR(1)  
ASM:    CL1
```

EIBTASKN

contains the task number assigned to the task by CICS. This number appears in trace table entries generated while the task is in control. The format of the field is packed decimal.

```
COBOL: PIC S9(7) COMP-3.  
C:      char eibtaskn [4];  
PL/I:   FIXED DEC(7,0)  
ASM:    PL4
```

EIBTIME

contains the time at which the task is started (this field is
updated by the ASKTIME command). The time is in
packed decimal form (0HHMMSS+), and can contain a
value in the range 0000000+ to 0240000+. Both
0000000+ and 0240000+ are valid.

```
COBOL: PIC S9(7) COMP-3.  
C:      char eibtime [4];  
PL/I:   FIXED DEC(7,0)  
ASM:    PL4
```

EIBTRMID

contains the symbolic terminal identifier of the principal facility (terminal or logical unit) associated with the task.

```
COBOL: PIC X(4).  
C:      char eibtrmid [4];  
PL/I:   CHAR(4)  
ASM:    CL4
```

EIBTRNID

contains the symbolic transaction identifier of the task.

```
COBOL: PIC X(4).  
C:      char eibtrnid [4];  
PL/I:   CHAR(4)  
ASM:    CL4
```

Appendix B. Codes returned by ASSIGN

This appendix describes the codes returned by the ASSIGN command.

ASSIGN TERMCODE

This section gives the meanings of the terminal type codes in the first byte of the data area returned by the TERMCODE option of the ASSIGN command.

The codes are derived from the DEVICE attribute of the TYPETERM RDO resource. The second byte of the data area contains a model number in character form, as set by the TERMMODEL RDO attribute of TYPETERM. TYPETERM is described in the *CICS/ESA Resource Definition Guide*.

The codes are listed here as both bit patterns and hexadecimal values.

Code	Meaning	
.... ...1	X'01'	7770
.... ..1.	X'02'	System 7
.... 1...	X'08'	Console
...1 ..1.	X'12'	Sequential disk
...1 .1..	X'14'	Magnetic tape
...1 1...	X'18'	Card reader or line printer
...1 1..1	X'19'	Spooling system printer
...1 1.1.	X'1A'	Spooling internal reader
..1.	X'20'	Hard-copy terminals
..1. ...1	X'21'	Model 33/35 TWX
..1. ..1.	X'22'	Teletypewriter
..1. .1..	X'24'	1050
..1. 1...	X'28'	2740
..1. 1.1.	X'2A'	2741 Correspondence
..1. 1.11	X'2B'	2741 EBCDIC
.1..	X'40'	Video terminals
.1.. ...1	X'41'	2260 local
.1.. 1...	X'48'	2260 remote
.1.. 1.1.	X'4A'	1053
.1.. 11..	X'4C'	2265
.1.1	X'50'	TCAM
1...	X'80'	Bisynchronous
1... ..1.	X'82'	2770
1... .1..	X'84'	2780
1... .1.1	X'85'	3780
1... .11.	X'86'	2980
1... 1...	X'88'	3735
1... 1..1	X'89'	3740
1... 1.1.	X'8A'	3600 bisynchronous
1.1 ...1	X'91'	3277 remote
1.1 ..1.	X'92'	3275 remote
1.1 ...11	X'93'	BTAM 3284 remote and VTAM 3270P
1.1 .1..	X'94'	BTAM 3286 remote
1.1 1..1	X'99'	3277 local
1.1 1.11	X'9B'	BTAM 3284 local
1.1 11..	X'9C'	BTAM 3286 local
1.1.	X'A0'	Bisynchronous - programmable
1.1. ...1	X'A1'	System/3
1.1. .1..	X'A4'	System/370
1.1. .11.	X'A6'	System/7 with BSCA
1.11	X'B0'	SDLC device class

Code	Meaning	
1.11 ...1	X'B1'	3601
1.11 ..1.	X'B2'	3614
1.11 .1..	X'B4'	3790
1.11 .1.1	X'B5'	3790 User program
1.11 .11.	X'B6'	3790 SCS printer
1.11 1...	X'B8'	3650 Pipeline
1.11 1..1	X'B9'	3653 Host conversational
1.11 1.1.	X'BA'	3650 Attached 3270 HC
1.11 1.11	X'BB'	3650 User program
1.11 11.1	X'BD'	Contention logical unit
1.11 111.	X'BE'	Interactive logical unit
1.11 1111	X'BF'	Batch logical unit
11..	X'C0'	LUTYPE 6

Note: An ASSIGN TERMCODE for an ISC session returns a X'C0' for LUTYPE 6. An INQUIRE CONNECTION then determines whether this ISC connection is using LUTYPE6.1 or APPC protocols.

11.. ...1	X'C1'	LUTYPE 4
11.1 ...1	X'D1'	ISC MM conversation
11.1 ..1.	X'D2'	LUC mode group entry
11.1 ..11	X'D3'	LUC session

Note: X'D3' is not used.

ASSIGN FCI

This section gives the meanings of the facility control indicator codes in the data area returned by the FCI option of the ASSIGN command.

Code	Meaning	
.... ...1	X'01'	TERMINAL FACILITY MASK
.... ..1.	X'02'	K C P MACRO FILE MASK
.... .1..	X'04'	NONTERMINAL FACILITY MASK
.... 1...	X'08'	DESTINATION CONTROL TABLE
...1	X'10'	AID FACILITY MASK
111.	X'E0'	reserved

ASSIGN

Appendix C. Translated code for CICS commands

Application programs can be written in COBOL, C, PL/I, or assembler language, and contain CICS commands. CICS translates these programs and creates an equivalent source program where each command is now translated into a call macro or statement in the language of the original source program.

COBOL

For a COBOL application program, each command is replaced by one or more MOVE statements followed by a COBOL CALL statement.

The purpose of the MOVE statements is to assign constants to COBOL data variables; this enables constants and names to be specified as arguments to options in the commands.

For example, for OS/VS COBOL and VS COBOL II, a # command such as:

```
EXEC CICS RECEIVE MAP('A') END-EXEC.
```

may be translated to:

```
MOVE ' ' TO DFHEIV0.
MOVE 'A' TO DFHC0070.
CALL 'DFHEI1' USING DFHEIV0 DFHC0070 AI.
```

Declarations for the generated variables DFHEIV0 and DFHC0070 are included automatically in the working-storage section; their names are reserved. The string within the quotation marks moved to DFHEIV0 consists of characters some of which may be unprintable. Do not use EXEC, CICS, DLI, END-EXEC, or names starting with DFH, as names for user variables.

With the COBOL3 translator option the command:

```
# EXEC CICS RECEIVE MAP('A') END-EXEC.
```

may be translated to:

```
# Call 'DFHEI1' using by content x'1802c0000700000000004090000000
# - '20f0f0f0f0f9404040' by content 'A ' by reference AI
# end-call.
```

The translator modifies the linkage section by inserting the EIB structure as the first parameter, and inserts declarations of the temporary variables that it requires into the working-storage section. It also inserts a DFHCOMMAREA if the first item in the linkage section is not a DFHCOMMAREA.

C

For a C application program, each command is replaced by reassignment statements followed by a dfhexec statement that passes the parameters.

PL/I

For a PL/I application program, each command is always replaced by a DO statement, a declaration of a generated entry name, a CALL statement, and an END statement. The ENTRY declaration ensures that the appropriate conversions for argument values take place.

If a PL/I on-unit consists of a single EXEC CICS command, the command should be inside a BEGIN block, for example:

```
ON ERROR BEGIN;
    EXEC CICS RETURN;
END;
```

In a similar way, if an EXEC CICS command is associated with a PL/I condition prefix, the command should be inside a BEGIN block, for example:

```
(NOZERODIVIDE): BEGIN;
    EXEC CICS GETMAIN
    SET(ptr-ref)
    LENGTH(data-value);
END;
```

If OPTIONS(MAIN) is specified, the translator modifies the parameter list by inserting the EIB structure pointer as the first parameter. If OPTIONS(MAIN) is not specified (that is, if the program is to be link-edited to the main module), the parameter list is not modified, and it is the application programmer's responsibility to address the EIB structure in the link-edited program if access to it is required. In either case, where a program commences with a valid PL/I PROCEDURE statement, the translator inserts the declaration of the EIB structure.

Assembler language

The invocation of a CICS assembler-language application program obeys system standards, which means that on entry to the application program, registers 1, 15, 14, and 13 contain the following:

- Register 1 contains the address of the parameter list; there are at least two entries in this list, as follows:
 - Address of the EIB (EXEC interface block)
 - Address of the COMMAREA; if no COMMAREA, entry is X'00000000'.
- Register 15 contains the address of the entry point
- Register 14 contains the address of the return point
- Register 13 contains the address of the save area.

All other registers are undefined.

Translated code

DFHECALL macro

For an assembler-language application program, each command is replaced by an invocation of the DFHECALL macro.

This macro expands to a system-standard call sequence using registers 15, 14, 0, and 1, whose contents are:

- Register 15 contains the address of the entry point in the EXEC interface program
- Register 14 contains the address of the return point in your application program
- Register 0 is undefined
- Register 1 contains the address of the parameter list.

The entry point held in register 15 is resolved in the EXEC interface processor (DFHEAI) that must be link-edited with your application program.

You can specify the exit from the application program by a RETURN command in your source program. Alternatively, you can let the translator-inserted macro DFHEIRET, which has been inserted before the END statement, do it. This macro only restores the registers and returns control to the address in register 14. Note that this can be used to return from a top-level program but is not advisable from a lower-level program.

During assembly, the DFHECALL macro builds an argument list in dynamic storage, so that the application program is reentrant, and then invokes the EXEC interface program (DFHEIP). DFHEIP also obeys system standards, as described above.

In addition to the invocation of the DFHECALL macro, the translator also inserts the following macros into your source program:

DFHEIGBL

This macro sets globals if you are using EXEC DLI in either a batch or an online CICS application program. Within DFHEIGBL, if DFHEIDL is set to 1, this means that the program contains EXEC DLI commands. If DFHEIDB is set to 1, this means that the program is batch DL/I. If you are not using DL/I, it is commented and set to 0.

DFHEIENT

This macro is inserted after the first CSECT or START instruction. It performs prolog code; that is, it:

- Saves registers
- Gets an initial allocation of the storage defined by DFHEISTG)
- Sets up a base register (default register 3)
- Sets up a dynamic storage register (default register 13)
- Sets up a register to address the EIB (default register 11).

DFHEIRET

This macro performs epilog code; that is, it:

- Restores registers
DFHEIRET RCREG=nn, where *nn* (any register number other than 13) contains the return code to be placed in register 15 after the registers are restored.
- Returns control to the address in register 14.

DFHEISTG and DFHEIEND

These macros define dynamic storage; that is, they:

- Define the storage required for the parameter list
- Define a save area.

A copybook, DFHEIBLK, containing a DSECT that describes the EIB, is also included automatically.

Note that the program must have an END statement because the translator does not otherwise insert the default macros.

The example in Figure 7 shows a simple assembler-language application program that uses the BMS command SEND MAP to send a map to a terminal. The lower part of the figure shows the output after program INSTRUCT has been translated.

Source program	
INSTRUCT CSECT	
EXEC CICS SEND MAP('DFH\$AGA') MAPONLY ERASE	
END	
The above source program is translated to:	
DFHEIGBL ,	INSERTED BY TRANSLATOR
INSTRUCT CSECT	
DFHEIENT	INSERTED BY TRANSLATOR
* EXEC CICS SEND MAP('DFH\$AGA') MAPONLY ERASE	
DFHECALL =X'1804C0000800000000046204000020',	
(CHA7,=CL7'DFH\$AGA*'),(____RF,DFHEIV00)	
DFHEIRET	INSERTED BY TRANSLATOR
DFHEISTG	INSERTED BY TRANSLATOR
DFHEIEND	INSERTED BY TRANSLATOR
END	

Figure 7. Translated code for a CICS command

Extensions to dynamic storage

You can extend dynamic storage to provide extra storage for user variables.

You do this by defining these variables in your source program in a DSECT called DFHEISTG. The maximum amount of dynamic storage obtainable using the DFHEISTG DSECT is 65264 bytes. (Note that DFHEISTG is a reserved name.) This storage is initialized to X'00'. At translation, the translator inserts the DFHEISTG macro immediately following your DFHEISTG DSECT instruction. In this way the DSECT describes dynamic storage needed for the parameter list, for the command-level interface, and for any user variables.

The example in Figure 8 shows a simple assembler-language application program that uses such variables in dynamic storage.

Source program	
DFHEISTG DSECT	
COPY DFH\$AGA	INPUT MAP DSECT
COPY DFH\$AGB	OUTPUT MAP DSECT
MESSAGE DS CL39	
INQUIRY CSECT	
EXEC CICS RECEIVE MAP('DFH\$AGA')	
MVC NUMBO,KEYI	
MVC MESSAGE,=CL(L'MESSAGE)'THIS IS A MESSAGE'	
EXEC CICS SEND MAP('DFH\$AGB') ERASE	
END	
The above source program is translated to:	
DFHEIGBL ,	INSERTED BY TRANSLATOR
DFHEISTG DSECT	
DFHEISTG	INSERTED BY TRANSLATOR
COPY DFH\$AGA	INPUT MAP DSECT
COPY DFH\$AGB	OUTPUT MAP DSECT
MESSAGE DS CL39	
INQUIRY CSECT	
DFHEIENT	INSERTED BY TRANSLATOR
* EXEC CICS RECEIVE MAP('DFH\$AGA')	
DFHECALL =X'1802C0000800000000040900000020',	
(CHA7,=CL7'DFH\$AGA*'),(____RF,DFH\$AGAI)	
MVC NUMBO,KEYI	
MVC MESSAGE,=CL(L'MESSAGE)'THIS IS A MESSAGE'	
* EXEC CICS SEND MAP('DFH\$AGB') ERASE	
DFHECALL =X'1804C000080000000004E204000020',	
(CHA7,=CL7'DFH\$AGB*'),(____RF,DFH\$AGBO)	
DFHEIRET	INSERTED BY TRANSLATOR
DFHEISTG	INSERTED BY TRANSLATOR
DFHEIEND	INSERTED BY TRANSLATOR
END	

Figure 8. Translated code for user variables

Multiple base registers

The values provided by the automatic insertion of DFHEIENT may be inadequate for application programs that produce a translated output greater than 4095 bytes.

For example, the translator by default only sets up one base register (register 3), or, when the DLI translator option has been specified, the literals produced by the translator initializing the DIB could fall outside the range of that single base register.

To overcome this problem, you can prevent the translator from automatically inserting its version of the DFHEIENT macro by specifying the translator option NOPROLOG. This enables you to provide your own DFHEIENT macro with the CODEREG operand so that you can specify more than one base register. You must code your own version of the DFHEIENT macro, which can have up to three operands, in place of the first CSECT or START instruction in your source program. The three operands are as follows:

- CODEREG - base registers
- DATAREG - dynamic-storage registers
- EIBREG - register to address the EIB.

Translated code

For example, the source code shown in Figure 7 would become:

```
| INSTRUCT DFHEIENT CODEREG=(2,3,4),  
    DATAREG=(13,5),  
    EIBREG=6  
    EXEC CICS SEND  
    MAP('DFH$AGA')  
    MAPONLY ERASE  
    END
```

The symbolic register DFHEIPLR is equated to the first DATAREG either explicitly specified or obtained by default. It is recommended that, because register 13 points to the save area defined in dynamic storage by DFHEISTG, you use register 13 as the first dynamic-storage register.

DFHEIPLR is assumed by the expansion of a CICS command to contain the value set up by DFHEIENT. You should either dedicate this register or ensure that it is restored before each CICS command.

Assembler-language programs, translated with the DLI option, have a DLI initialization call inserted after each CSECT statement. Assembler-language programs larger than 4095 bytes, that do not use the CODEREG operand of the DFHEIENT macro to establish multiple base registers, must include an LTORG statement to ensure that the literals, generated by either DFHEIENT or a DLI initialization call, fall within the range of the base register.

Note that, in general, an LTORG statement is needed for every CSECT that exceeds 4095 bytes in length.

Appendix D. Terminal control

This appendix contains general information that applies to all terminals and logical units. For more detail, see the command descriptions.

Commands and options for terminals and logical units

This section describes the commands and options that apply to terminals and logical units.

Fullword lengths

For all terminal control commands, fullword length options can be used instead of halfword length options. In particular, where the following options are used in CONVERSE, RECEIVE, or SEND, the corresponding alternative can be specified instead (except for those noted):

Option	Alternative
LENGTH	FLENGTH
TOLENGTH	TOFLENGTH
FROMLENGTH	FROMFLENGTH
MAXLENGTH	MAXFLENGTH

Application programs should be consistent in their use of fullword and halfword options on terminal control commands. The maximum value that can be specified as a parameter on any length keyword is 32767. See the *CICS/ESA Application Programming Guide* for more information.

Read from terminal or logical unit (RECEIVE)

The RECEIVE command is used to read data from a terminal or logical unit. The INTO option is used to specify the area into which the data is to be placed. Alternatively, a pointer reference can be specified in the SET option. CICS acquires an area large enough to hold the data and sets the pointer reference to the address of that data.

The contents of this area are available to the task until the next terminal I/O command. However, the area does not belong to the task and is released by CICS while processing the next request. Therefore, this area cannot be passed back to CICS for further processing.

The application can use MAXLENGTH to specify the maximum length of data that the program accepts. If the MAXLENGTH option is omitted on a RECEIVE command for which the INTO option is specified, the maximum length of data the program accepts can be specified in the LENGTH option. If the MAXLENGTH option is omitted on a RECEIVE command for which the SET option is specified, CICS acquires enough storage to hold all the available data.

If the data exceeds the specified maximum length and the NOTRUNCATE option is specified, the remaining data is made available to satisfy subsequent RECEIVE commands. If NOTRUNCATE is not specified, the data is truncated and the LENGERR condition occurs. In this event, if the LENGTH option is specified, the named data area is set to the actual data length (before truncation occurs) when data has been received. The first RECEIVE command in a task started by a terminal does not issue a terminal control read but simply copies the input buffer, even if the data length is zero. A second RECEIVE command must be issued to cause a terminal control read.

When a PA key is defined as a print key by the system initialization parameter PRINT, and that key is pressed in response to a RECEIVE command, it has no effect on the application program. The RECEIVE command is satisfied, and the application allowed to continue, when another attention (that is, one of the other PA keys, any of the PF keys, the ENTER key, or the light pen) is made at the keyboard.

Write to terminal or logical unit (SEND)

The SEND command is used to write data to a terminal or logical unit. The options FROM and LENGTH specify respectively the data area from which the data is to be taken and the length (in bytes) of the data. For a transaction started by automatic transaction initiation (ATI), a SEND command should always precede the first RECEIVE in a transaction.

WAIT option of SEND command: Unless the WAIT option is specified also, the transmission of the data associated with the SEND command is deferred until a later event, such as a syncpoint, occurs. This deferred transmission reduces the flows of data by allowing data-flow controls to be transmitted with the data.

Transmission is not deferred for distributed transaction processing when interregion communication (IRC) is in use.

Synchronize terminal I/O for a transaction (WAIT TERMINAL)

This command is used to ensure that a terminal operation has completed before further processing occurs in a task under which more than one terminal or logical unit operation is performed. Alternatively, the WAIT option can be specified in a SEND command. (A wait is always carried out for a RECEIVE command.) Either method may cause execution of a task to be suspended. If suspension is necessary, control is returned to CICS. Execution of the task is resumed when the operation is completed.

Terminal control

Even if the WAIT option is not specified in a SEND command, the EXEC interface program ensures that the operation is completed before issuing a subsequent RECEIVE or SEND command.

Converse with terminal or logical unit (CONVERSE)

For most terminals or logical unit types, a conversational mode of communication can be used. The CONVERSE command is used for this purpose and means that the 3650 application program communicates with the host processor. If this option is not specified, the 3650 application program cannot communicate with the host processor. In general, the CONVERSE command can be considered as a combination of a SEND command followed immediately by a WAIT TERMINAL command and then by a RECEIVE command. However, not all options of the SEND and RECEIVE commands are valid for the CONVERSE command; specific rules are given in the syntax descriptions for different devices. The TOLENGTH option is equivalent to the LENGTH option of the RECEIVE command, and the FROMLENGTH option is equivalent to the LENGTH option of the SEND command.

Send an asynchronous interrupt (ISSUE SIGNAL)

This command is used, in a transaction in receive mode, to signal to the sending transaction that a mode change is needed. The execution of the command raises the SIGNAL condition on the next SEND or RECEIVE command executed in the sending transaction, and a previously executed HANDLE CONDITION command for this condition can be used either to action the request or to ignore it.

Relinquish a telecommunication line (ISSUE RESET)

This command is used to relinquish use of a telecommunication line. The command applies only to binary synchronous devices using BTAM. The next BTAM operation is a read or write initial.

Disconnect a switched line (ISSUE DISCONNECT)

This command is used to break a line connection between a terminal and the processor, or to break a session between TCAM or VTAM logical units, when the transaction is completed. If the terminal is a buffered device, the data in the buffers is lost.

When used with a VTAM terminal, ISSUE DISCONNECT, which does not become effective until the task completes, signs off the terminal, frees the COMMAREA, clears the next TRANID, stops any BMS paging, and, if autoinstall is in effect, deletes the terminal definition.

TCAM-supported terminals and logical units

Because TCAM permits many applications to share a single network, the CICS-TCAM interface supports data streams rather than specific terminals or logical units.

Operations for terminals supported by TCAM use the same options as the terminals supported by other access methods. With the exception of the BUFFER option for the 3270, all options applicable for input operations are supported by CICS-TCAM. However, the conditions ENDINPT and EOF do not occur.

All output requests are the same for TCAM as for other CICS-supported terminals, except that:

- The ISSUE RESET command cannot be used
- The ISSUE COPY and ISSUE PRINT commands for the 3270 cannot be used
- The DEST option is available on CONVERSE and SEND commands, in addition to other appropriate options.

With the exception of 3650 logical units, operations for logical units supported by TCAM use the same options as logical units supported by VTAM.

BTAM programmable terminals

CICS/ESA 4.1 does not support local BTAM terminals. However, it does support transaction routing from BTAM terminals attached to a remote system.

When BTAM is used by CICS for programmable binary synchronous telecommunication line management, CICS initializes the telecommunication line with a BTAM communication line management; CICS initializes the communication read initial (TI); the terminal response must be a write initial (TI) or the equivalent. If an application program makes an input request, CICS issues a read continue (TT) to that line; if the application program makes an output request, CICS issues a read interrupt (RVI) to that line. If end of transmission (EOT) is not received on the RVI, CICS issues a read continue (TT) until the EOT is received.

When TCAM is used, all of this line control is handled by the message control facility rather than by CICS.

The programmable terminal response to a read interrupt must be "end of transmission" (EOT). The EOT response may, however, be preceded by writes, in order to exhaust the contents of output buffers (if the input buffer size is not exceeded by this data). The input buffer size is specified by the system programmer during preparation of the TCT. CICS issues a read continue until it receives an EOT, or until the input message is greater than the input buffer (an error condition).

After receiving an EOT, CICS issues a write initial (TI) or the equivalent (depending on the type of line). The programmable terminal response must be a read initial (TI) or the equivalent.

If the application program makes another output request, CICS issues a write continue (TT) to that line. If the application program makes an input request after it has made an output request, CICS turns the line around with a write reset (TR). (CICS does not recognize a read interrupt.)

To ensure that binary synchronous terminals (for example, System/370, 1130, 2780) remain coordinated, CICS processes the data collection or data transmission transaction on any line to completion, before polling other terminals on that line.

The programmable terminal actions required for the above activity, with the corresponding user application program commands and CICS actions, are summarized in Table 15.

Table 15. BTAM programmable terminal programming		
User Application Program Command	CICS ¹ Actions	Programmable Terminal Action
RECEIVE	Read initial (TI)	Write initial (TI)
	Read continue (TT)	Write continue (TT)
SEND	Read interrupt (RVI) ²	Write reset (TR) or
	Read continue (TT) ³	Write continue Write reset
SEND	Write initial (TI)	Read initial (TI)
	Write continue (TT)	Read continue (TT)
RECEIVE	Write reset (TR) ⁴	Read continue (TT)
	Read initial (TI)	Write initial (TI)

1. CICS issues the macro shown, or, if the line is switched, the equivalent. The user-written programmable terminal program must issue the equivalent of the BTAM operation shown.
2. An RVI sequence is indicated by the DECFLAGS field of the data event control block (DECB) being set to X'02' and a completion code of X'7F' being returned to the event control block (ECB).
3. The read continue is issued only if the EOT character is not received on the read interrupt.
4. Write reset is issued only for point-to-point terminals.

Automatically initiated transactions attached to a device cause the message:

```
DFH2503 AUTO OUTPUT HAS BEEN REQ,
      PLEASE PREPARE TO RECEIVE
```

to be sent to the device, which must be prepared to receive it.

Input data is deblocked to ETX, ETB, RS, and US characters. These characters are moved with the data but are not included in the data length. Characters such as NL (new line), CR (carriage return), LF (line feed), and EM (end of message) are included as data in a CICS application program.

Teletypewriter programming

The Teletypewriter (World Trade only) uses two different control characters for print formatting, as follows:

< carriage return (X'22' in ITA2 code or X'15' in EBCDIC)

≡ line feed (X'28' in ITA2 code or X'25' in EBCDIC)

The character < should always be used first; otherwise following characters (data) may be printed while the type bar is moving to the left.

Message format

Message begin: To start a message on a new line at the left margin, the message text must begin with X'1517' (EBCDIC). CICS recognizes the X'17' and changes it to X'25' (X'17' is an IDLE character).

Message body: To write several lines with a single transmission, the lines must be separated by X'1525', or if multiple blank lines are required, by X'152525...25'.

Message end before next input: To allow input of the next message on a new line at the left margin, the preceding message must end with X'1517'. CICS recognizes X'15' and changes the character following it to X'25'.

Message end before next output: In the case of two or more successive output messages, the "message begin" and the "message end" look the same; that is X'1517', except for the last message (see above). To make the "message end" of the preceding message distinguishable from the "message begin" of the next message, the next to last character of the "message end" must not be X'15'.

Message length

Messages for teletypewriter terminals should not exceed a length of about 3000 bytes or approximately 300 words.

Connection through VTAM

Both the TWX Model 33/35 Common Carrier Teletypewriter Exchange and the WTTY Teletypewriter (World Trade only) can be connected to CICS through BTAM, or through VTAM using NTO.

If a device is connected through VTAM using NTO, the protocols used are the same as for the 3767 logical unit, and the application program can make use of these protocols (for example, HANDLE CONDITION SIGNAL). However, the data stream is not translated to a 3767 data stream but remains as that for a TWX/WTTY.

Display device operations

In addition to the standard terminal control commands for sending and receiving data, several commands and lists are provided for use with display devices such as the 3270.

The commands are:

- Print displayed information (ISSUE PRINT)
- Copy displayed information (ISSUE COPY)
- Erase all unprotected fields (ISSUE ERASEAUP)
- Handle input without data (RECEIVE)
- Handle attention identifiers (HANDLE AID).

The lists are:

- Standard attention identifier list (DFHAID)
- Standard attribute and printer control character list (DFHBMSCA).

For devices with switchable screen sizes, the size of the screen that can be used, and the size to be used for a given transaction, are defined by CICS table generation. These values can be obtained by means of the ASSIGN command, described in "ASSIGN" on page 22.

The ERASE option should always be included in the first SEND command, to clear the screen and format it according to the transmitted data. This first SEND with ERASE also selects the screen size to be used, as specified using the RDO option SCRNSIZE, or in the TCT. If ERASE is omitted, the screen size is the same as its previous setting, which may be incorrect.

Use the CLEAR key outside of a transaction to set the screen to its default size.

Print displayed information (ISSUE PRINT)

ISSUE PRINT prints displayed data on the first available printer that is eligible to respond to a print request.

For a BTAM-supported 3270, this is a printer on the same control unit.

For a 3270 logical unit or a 3650 host-conversational (3270) logical unit, it is a printer defined by PRINTTO or ALTPRT in the terminal control table TYPE=TERMINAL, by RDO, or by a printer supplied by the autoinstall user program.

For a 3270-display logical unit with the PTRADAPT feature (LUTYPE2 specified in TRMTYPE and PTRADAPT specified in FEATURE in the terminal control table TYPE=TERMINAL) used with a 3274 or 3276, it is a printer allocated by the printer authorization matrix. See *An Introduction to the IBM 3270 Information Display System*.

For a 3790 (3270-display) logical unit, it is a printer allocated by the 3790.

For a printer to be available, it must be in service and not currently attached to a task.

For a BTAM printer to be eligible, it must be attached to the same control unit as the display, must have a buffer capacity equal to or greater than that of the display, and must have FEATURE=PRINT specified in the associated terminal control table TYPE=TERMINAL.

For a 3270 logical unit to be eligible, it must have been specified by PRINTTO or ALTPRT in the terminal control table TERMINAL=TYPE, by RDO, or by a printer supplied by the autoinstall user program, and it must have the correct buffer capacity; FEATURE=PRINT is not necessary. If COPY is specified with ALTPRT or PRINTTO, the printer must be on the same control unit.

If an ISSUE PRINT command is executed, the printer involved must be owned by the same CICS system that owns the terminal that is running the transaction.

For some 3270 displays, it is possible also to print the displayed information without using CICS. See *An Introduction to the IBM 3270 Information Display System* manual.

Copy displayed information (ISSUE COPY)

The ISSUE COPY command is used to copy the format and data contained in the buffer of a specified terminal into the buffer of the terminal that started the transaction. This command cannot be used for an LUTYPE2 connection. Both terminals must be attached to the same remote control unit. The terminal whose buffer is to be copied is identified in the TERMID option. If the terminal identifier is not valid, that is, it does not exist in the TCT, then the TERMIDERR condition occurs. The copy function to be performed is defined by the copy control character (CCC) specified in the CTLCHAR option of the ISSUE COPY command.

The WAIT option of the ISSUE COPY command ensures that the operation has been completed before control is returned to the application program.

Erase all unprotected fields (ISSUE ERASEAUP)

The ISSUE ERASEAUP command is used to erase all unprotected fields of a 3270 buffer, by the following actions:

1. All unprotected fields are cleared to nulls (X'00').
2. The modified data tags (MDTs) in each unprotected field are reset to zero.
3. The cursor is positioned to the first unprotected field.
4. The keyboard is restored.

The WAIT option of the ISSUE ERASEAUP command ensures that the operation has been completed before control is returned to the application program.

Handle input without data (RECEIVE)

The RECEIVE command with no options causes input to take place and the EIB to be updated. However, data received by CICS is not passed on to the application program and is lost. A wait is implied. Two of the fields in the EIB that are updated are described below.

Cursor position (EIBCPOSN): For every terminal control (or BMS) input operation associated with a display device, the screen cursor address (position) is placed in the EIBCPOSN field in the EIB. The cursor address is in the form of a halfword binary value and remains until updated by a new input operation.

Attention identifier (EIBAID): For every terminal control (or BMS) input operation associated with a display device, an attention identifier (AID) is placed in field EIBAID in the EIB. The AID indicates which method the terminal operator has used to initiate the transfer of information from the device to CICS; for example, the ENTER key, a program function key, the light pen, and so on. The field contents remain unaltered until updated by a new input operation.

Field EIBAID can be tested after each terminal control (or BMS) input operation to determine further processing, and a standard attention identifier list (DFHAID) is provided for this purpose. Alternatively, the HANDLE AID command can be used to pass control to specified labels when the AIDs are received.

EIBAID and EIBCPOSN are also updated at task initiation for non-ATI tasks and after each terminal control and BMS input.

Appendix E. SAA Resource Recovery

SAA Resource Recovery is the recovery element of the Systems Application Architecture (SAA) Common Programming Interface (CPI).

SAA Resource Recovery provides that architecture's alternative application programming interface (API) to EXEC CICS SYNCPOINT and EXEC CICS SYNCPOINT ROLLBACK functions in CICS/ESA. (See the *SAA Common Programming Interface-Resource Recovery Reference*, SC31-6821, for more details.)

CICS/ESA 3.2.1 and above supports only those SAA Resource Recovery return codes that match existing EXEC CICS commands. This leaves only two return codes: RR_OK and RR_BACKED_OUT.

SRRCMT

Commit call (equivalent to EXEC CICS SYNCPOINT). The return codes are:

- RR_OK
- RR_COMMITTED_OUTCOME_PENDING
- RR_COMMITTED_OUTCOME_MIXED
- RR_PROGRAM_STATE_CHECK
- RR_BACKED_OUT
- RR_BACKED_OUT_OUTCOME_PENDING
- RR_BACKED_OUT_OUTCOME MIXED

Because of the restriction, these are replaced by:

- RR_COMMITTED_OUTCOME_PENDING, RR_OK
- RR_COMMITTED_OUTCOME_MIXED, RR_OK
- RR_PROGRAM_STATE_CHECK, shown as abend code ASP2
- RR_BACKED_OUT_OUTCOME_PENDING, RR_BACKED_OUT
- RR_BACKED_OUT_OUTCOME MIXED, RR_BACKED_OUT

SRRBACK

Backout call (equivalent to EXEC CICS SYNCPOINT ROLLBACK). The return codes are:

- RR_OK
- RR_COMMITTED_OUTCOME_PENDING
- RR_COMMITTED_OUTCOME_MIXED

Because of the restriction, all these are replaced by RR_OK.

Appendix F. Common Programming Interface Communications (CPI Communications)

Common Programming Interface Communications (CPI Communications) is the communication element of the Systems Applications Architecture (SAA) Common Programming Interface (CPI).

CPI Communications in CICS provides an alternative application programming interface (API) to existing CICS Advanced Program-to-Program Communications (APPC) support. CPI Communications provides distributed transaction processing (DTP) on APPC sessions and can be used in assembler language, COBOL, PL/I, or C.

CPI Communications defines an API that can be used in APPC networks that include multiple system platforms, where the consistency of a common API is seen to be of benefit.

The CPI Communications interface can converse with applications on any system that provides an APPC API. This includes applications on CICS platforms. You may use APPC API commands on one end of a conversation and CPI Communications commands on the other. CPI Communications requires specific information (side information) to begin a conversation with a partner program. CICS implementation of side information is achieved using the partner resource which your system programmer is responsible for maintaining.

The application's calls to the CPI Communications interface are resolved by link-editing it with the CICS CPI Communications link-edit stub (DFHCPLC). You can find further guidance information in the *CICS/ESA System Definition Guide*.

CPI Communications language interfaces

The CPI Communications API is defined as a general call interface. The interface is described in the *SAA Common Programming Interface Communications Reference* manual.

Appendix G. API restrictions for distributed program link

This appendix lists the API commands, indicating whether or not they are supported in a program running in a resource region in response to a distributed program link command.

Summary of the restricted API commands

<i>Table 16. Restricted API commands</i>	
ADDRESS ALLOCATE ASSIGN CONNECT PROCESS CONVERSE EXTRACT PROCESS FREE CONVID HANDLE AID	ISSUE PURGE MESSAGE RECEIVE ROUTE SEND SIGNOFF SIGNON WAIT TERMINAL

List of API commands

The following table summarizes the CICS API commands by functional area, indicating whether or not they are supported in a program invoked by a distributed program link command. Generally, if the program issues a command that is not supported, CICS returns an INVREQ condition, with a RESP2 value of 200.

<i>Table 17 (Page 1 of 4). Summary of the CICS API by functional area</i>		
<i>Functional area</i>	<i>Command</i>	<i>Supported?</i>
Abend support	ABEND ASSIGN ABCODE ASRAINTRPT ASRAPSW ASRAREGS ORGABCODE HANDLE ABEND	YES
APPC mapped communication	ALLOCATE(APPC) CONNECT PROCESS CONVERSE EXTRACT PROCESS FREE CONVID ISSUE ABEND CONFIRMATION ERROR PREPARE SIGNAL RECEIVE SEND WAIT CONVID	NO
Note: The above APPC commands are restricted only when they refer to the principal facility.		
Signon	SIGNON SIGNOFF	NO
Batch data interchange commands	ISSUE ABORT QUERY ADD RECEIVE END REPLACE ERASE SEND NOTE WAIT	NO

API restrictions

<i>Table 17 (Page 4 of 4). Summary of the CICS API by functional area</i>		
Functional area	Command	Supported?
Program control	LINK LOAD RELEASE RETURN XCTL	YES
Note: LINK, RETURN and XCTL do not support INPUTMSG.		
Security	QUERY SECURITY	YES
Storage control	FREEMAIN GETMAIN	YES
Syncpoint	SYNCPOINT	YES
Task control	ASSIGN TASKPRIORITY CHANGE TASK DEQ ENQ SUSPEND WAIT EXTERNAL	YES
Temporary storage	DELETEQ TS READQ TS WRITEQ TS	YES
Terminal control	ASSIGN FACILITY CONVERSE HANDLE AID RECEIVE SEND WAIT TERMINAL	NO
Transient data	DELETEQ TD READQ TD WRITEQ TD	YES

Appendix H. CVDA numeric values

This appendix lists the CVDA values and their numeric equivalents for the relevant EXEC CICS commands in this book.

Table 18. CVDAs and numeric values in alphabetic sequence

cvda	value
ALLOCATED	81
ALTERABLE	52
BASESPACE	664
CICSEXECKEY	381
CONFFREE	82
CONFRECEIVE	83
CONFSEND	84
CRITICAL	11
CTRLABLE	56
EVENTUAL	3
FREE	85
IMMEDIATE	2
LOG	54
LUW	246
NOLOG	55
NONCICS	661
NOTALTERABLE	53
NOTAPPLIC	1
NOTCTRLABLE	57
NOTPURGEABLE	161
NOTREADABLE	36
NOTUPDATABLE	38
PENDFREE	86
PENDRECEIVE	87
PURGEABLE	160
READABLE	35
RECEIVE	88
ROLLBACK	89
SEND	90
SUBSPACE	663
SYNCFREE	91
SYNCRECEIVE	92
SYNCSEND	93
TASK	233
UPDATABLE	37
USER	382

Table 19. CVDAs and numeric values in numeric sequence

value	cvda
1	NOTAPPLIC
2	IMMEDIATE
3	EVENTUAL
11	CRITICAL
35	READABLE
36	NOTREADABLE
37	UPDATABLE
38	NOTUPDATABLE
52	ALTERABLE
53	NOTALTERABLE
54	LOG
55	NOLOG
56	CTRLABLE
57	NOTCTRLABLE
81	ALLOCATED
82	CONFFREE
83	CONFRECEIVE
84	CONFSEND
85	FREE
86	PENDFREE
87	PENDRECEIVE
88	RECEIVE
89	ROLLBACK
90	SEND
91	SYNCFREE
92	SYNCRECEIVE
93	SYNCSEND
160	PURGEABLE
161	NOTPURGEABLE
233	TASK
246	LUW
381	CICSEXECKEY
382	USER
661	NONCICS
663	SUBSPACE
664	BASESPACE

Appendix I. National language codes

Language codes are held as one character for NATLANG and NATLANGINUSE, and three characters for LANGUAGECODE and LANGINUSE (see under **Suffix** and **IBM code** below).

Table 20. CICS language suffixes

Suffix	IBM Code	Language name
A	ENG	UK English
B	PTB	Brazilian Portuguese
C	CHS	Simplified Chinese
D	DAN	Danish
E	ENU	US English
F	FRA	French
G	DEU	German
H	KOR	Korean
I	ITA	Italian
J	ISL	Icelandic
K	JPN	Japanese
L	BGR	Bulgarian
M	MKD	Macedonian
N	NOR	Norwegian
O	ELL	Greek
P	PTG	Portuguese
Q	ARA	Arabic
R	RUS	Russian
S	ESP	Spanish
T	CHT	Traditional Chinese
U	UKR	Ukrainian
V	SVE	Swedish
W	FIN	Finnish
X	HEB	Hebrew
Y	SHC	Serbo-Croatian (Cyrillic)
Z	THA	Thai
1	BEL	Byelorussian
2	CSY	Czech
3	HRV	Croatian
4	HUN	Hungarian
5	PLK	Polish
6	ROM	Romanian
7	SHL	Serbo-Croatian (Latin)
8	TRK	Turkish
9	NLD	Dutch

There are other IBM codes not supported by CICS.

Table 21. Other IBM language codes

IBM Code	Language name
AFR	Afrikaans
CAT	Catalan
DES	Swiss German
ENA	Australian English
ENP	English Upper Case
FRB	Belgian French
FRC	Canadian French
FRS	Swiss French
GAE	Irish Gaelic
ITS	Swiss Italian
NLB	Belgian Dutch - Flemish
NON	Norwegian - Nynorsk
RMS	Rhaeto-Romanic
SKY	Slovakian
SLO	Slovenian
SRL	Serbian (Latin)
SRB	Serbian (Cyrillic)
SQI	Albanian
URD	Urdu

National language codes

Appendix J. BMS-related constants

This appendix contains the BMS-related standard attribute and printer control characters, a bit map for attributes, MSR control value constants, and attention identifier constants.

The standard list DFHBMSCA makes it simpler to provide field attributes and printer control characters. Table 22 lists the symbolic names for the various combinations of attributes and control characters. If you need combinations other than the ones shown, you must generate them separately. To help you do this, see Table 23 on page 377 for a bit map of attributes. To find the value of an attribute constant, see the *3274 Control Unit Reference Summary*.

You can get the standard attribute and printer character control list by copying copybook DFHBMSCA into your application.

- For COBOL users, it consists of a set of 01 statements that can be copied into the working storage section.
- For C users, it is included in applications as follows:


```
#include "dfhbmsca.h"
```
- For PL/I users, it consists of DECLARE statements defining elementary character variables.
- For assembler-language users, the list consists of a set of EQU statements.

| You must use the symbolic name DFHDFT in the application structure to override a map attribute with the default. You can | use a high value, such as X'FF', to reset the COLOR, HIGHLIGHT, OUTLINE, PS, SOSI, or VALIDN attributes to their default | values. On the other hand, to specify default values in a set attribute (SA) sequence in text build, you should use the symbolic names DFHDFCOL, DFHBASE, or DFHDFHI.

Table 22 (Page 1 of 2). Standard attribute and printer control character list, DFHBMSCA

Constant	Meaning
DFHBMPPEM	Printer end-of-message
DFHBMPNL	Printer new-line
DFHBMPFF	Printer form feed
DFHBMPCR	Printer carriage return
DFHBMASK	Autoskip
DFHBMUNP	Unprotected
DFHBMUNN	Unprotected and numeric
DFHBMPRO	Protected
DFHBMBRY	Bright
DFHBMNDAR	Dark
DFHBMFSE	MDT set
DFHBMPRF	Protected and MDT set
DFHBMASF	Autoskip and MDT set
DFHBMASB	Autoskip and bright
DFHBMPSO	shift-out value X'0E'.
DFHBMPSI	shift-in value X'0F'.
DFHBMEOF	Field erased
DFHBMCUR	Field containing cursor flagged
DFHBMEC	Erased field containing cursor (COBOL only)
DFHBMFLG	Flags (COBOL only)
DFHBMDET	Field detected
DFHSA ¹	Set attribute (SA) order
DFHERROR	Error code
DFHCOLOR ¹	Color
DFHPS ¹	Programmed symbols
DFHHLT ¹	Highlight
DFH3270 ¹	Base 3270 field attribute
DFHVAL	Validation
DFHOUTLN	Field outlining attribute code
DFHBKTRN	Background transparency attribute code
DFHALL ¹	Reset all to defaults
DFHDFT	Default
DFHDFCOL ¹	Default color
DFHBLUE	Blue
DFHRED	Red
DFHPINK	Pink
DFHGREEN	Green
DFHTURQ	Turquoise
DFHYELLO	Yellow
DFHNEUTR	Neutral
DFHBASE ¹	Base programmed symbols
DFHDFHI ¹	Normal

Table 22 (Page 2 of 2). Standard attribute and printer control character list, DFHBMSCA

Constant	Meaning
DFHBLINK	Blink
DFHREVRS	Reverse video
DFHUNDLN	Underscore
DFHMFIL ²	Mandatory fill
DFHMENT ²	Mandatory enter
DFHMFEE	Mandatory fill and mandatory enter
DFHMT	Trigger
DFHMFT	Mandatory fill and trigger
DFHMET	Mandatory enter and trigger
DFHMFET	Mandatory fill and mandatory enter and trigger
DFHUNNOD	Unprotected, nondisplay, nonprint, nondetectable, MDT
DFHUNIMD	Unprotected, intensify, light-pen detectable, MDT
DFHUNNUM	Unprotected, numeric, MDT
#	Apar 67669
#	Documentation for Apar 67669 added 7 Feb 1995 (TUCKER)
#	DFHUNNUB Unprotected, numeric, intensify, intensify, light-pen detectable
	DFHUNINT Unprotected, numeric, intensify, light-pen detectable, MDT
	DFHUNNON Unprotected, numeric, nondisplay, nonprint, nondetectable, MDT
	DFHPROTI Protected, intensify, light-pen detectable
	DFHPROTN Protected, nondisplay, nonprint, nondetectable
	DFHDFFR Default outline
	DFHUNDER Underline
	DFHRIGHT Right vertical line
	DFHOVER Overline
	DFHLEFT Left vertical line
	DFHBOX Underline and right vertical and overline and left vertical
	DFHSOSI SOSI=yes
	DFHTRANS Background transparency
	DFHOPAQ No background transparency
	Note: 1 For text processing only. Use for constructing embedded set attribute orders in user text.
	2 Cannot be used in set attribute orders.

Table 23. Bit map for attributes

prot	a/n	hi	spd	ndp	mdt	ebcd	ascii	char
U						40	20	b (blank)
U					Y	C1	41	A
U			Y			C4	44	D
U			Y		Y	C5	45	E
U		H	Y			C8	48	H
U		H	Y		Y	C9	49	I
U				Y		4C	3C	<
U				Y	Y	4D	28	(
U	N					50	26	&
U	N				Y	D1	4A	J
U	N		Y			D4	4D	M
U	N		Y		Y	D5	4E	N
U	N	H	Y			D8	51	Q
U	N	H	Y		Y	D9	52	R
U	N			Y		5C	2A	*
U	N			Y	Y	5D	29)
P						60	2D	- (hyphen)
P					Y	61	2F	/
P			Y			E4	55	U
P			Y		Y	E5	56	V
P		H	Y			E8	59	Y
P		H	Y		Y	E9	5A	Z
P				Y		6C	25	%
P				Y	Y	6D	5F	_ (underscore)
P	S					F0	30	0
P	S				Y	F1	31	1
P	S		Y			F4	34	4
P	S		Y		Y	F5	35	5
P	S	H	Y			F8	38	8
P	S	H	Y		Y	F9	39	9
P	S			Y		7C	40	@
P	S			Y	Y	7D	27	'

The attributes in the headings are:

prot = protected
 a/n = automatic skip or numeric
 hi = high intensity
 spd = selector pen detectable
 ndp = nondisplay print
 mdt = modified data tag

The hexadecimal codes in the headings are:

ebcd = extended binary-coded decimal interchange code
 ascii = American National Standard Code for Information Interchange
 char = graphic character equivalent to hex code

The characters in the body of the above table mean the following:

H = High
 P = Protected
 N = Numeric
 U = Unprotected
 S = Automatic skip
 Y = Yes

Magnetic slot reader (MSR) control value constants, DFHMSRCA

A selection of MSR control value constants has been created for CICS and stored in copybook DFHMSRCA. The patterns are stored as named constants that can be loaded by simple application program commands. Provision of such constants saves the programmer from having to build a commonly used bit pattern whenever it is required.

MSR control byte values

A selection of MSR control byte values has been created for CICS and stored in the copybook DFHMSRCA. Table 24 on page 378 shows the meaning of each bit. The constants supplied in DFHMSRCA are listed in Table 25 on page 378.

Table 24. MSR control byte values

Byte (purpose)	Bit	Value	Meaning
1 (STATE MASK) If a bit is on in the STATE MASK byte, the state it represents is adopted by the device if the corresponding bit is also on in the STATE VALUE byte.	0	USER	User mode. Turn the yellow light on if the same bit is on in STATE VALUE.
	1	LOCK	Locked/Unlocked. If locked, MSR input is inhibited.
	2	AUTO	Autoenter on/off. If set on, any card read by the MSR causes an ENTER operation. If off, only a secure card causes an ENTER.
	3	Ai1S	Suppress audible alarm 1.
	4	Ai2S	Suppress audible alarm 2.
2 (STATE VALUE) Modifies state to on or off if the corresponding bit is set on in STATE MASK.			
3 (INDICATOR MASK) Performs a similar function to STATE MASK, but for indicators.	0		Light 1 (Green)
	1		Light 2 (Yellow)
	2		Light 3 (Red)
	3		Audible Alarm 1 (Long buzz)
	4		Audible Alarm 2 (Short buzz)
4 (INDICATOR VALUE) Performs a similar function to STATE VALUE.			

Table 25. Standard list DFHMSRCA

Constant	Meaning
DFHMSRST	MSR reset. All lights and buzzers off. MSR available for input.
DFHMSSCON	Transaction ready for more input. Green and yellow on; emit short buzz; IN PROCESS (user) mode set.
DFHMSSFIN	Input complete. Green on; emit short buzz; IN PROCESS mode reset.
DFHMSSALR	Operator alert. Green, yellow, and red on; emit long buzz; IN PROCESS mode reset.
DFHMSSALS	Operator alert. Green, yellow, and red on; emit long buzz; IN PROCESS mode set.
DFHMSSIPY	IN PROCESS state set. Yellow on.
DFHMSSIPN	IN PROCESS state reset.
DFHMSSLKY	MSR operation inhibited. Yellow on.
DFHMSSLKN	MSR input allowed. Green on. Yellow on.
DFHMSSAEY	MSR autoenter on. Yellow on.
DFHMSSAEN	MSR autoenter off. Yellow on.
DFHMSSLBN	Long buzzer suppressed. Yellow on.
DFHMSSLBY	Long buzzer permitted. Yellow on.
DFHMSSBN	Short buzzer suppressed. Yellow on.
DFHMSSBY	Short buzzer permitted. Yellow on.
DFHMSSNOP	Leave all MSR settings unchanged.

Attention identifier constants, DFHAID

The standard attention identifier list, DFHAID, simplifies testing the contents of the EIBAID field. Table 26 shows the symbolic name for the attention identifier (AID) and the corresponding 3270 function.

You can get a copy of the list by copying DFHAID into your application program. For COBOL users, it consists of a set of 01 statements that must be copied into the working-storage section. For C users, it consists of a series of defined constants. For PL/I users, it consists of DECLARE statements defining elementary character variables.

<i>Table 26. Standard list DFHAID</i>	
Constant	Meaning
DFHENTER	ENTER key
DFHCLEAR	CLEAR key
DFHPA1–DFHPA3	PA1–PA3 keys
DFHPPF1–DFHPPF24	PF1–PF24 keys
DFHOPID	OPERID or MSR
DFHMSRE	Extended (standard) MSR
DFHTRIG	Trigger field
DFHPEN	SELECTOR PEN or CURSOR SELECT key
DFHCLRP	CLEAR PARTITION key
DFHSTRF	Structured field pseudo-AID.
Note: DFHCLRP and DFHSTRF do not apply to minimum function BMS.	

Appendix K. BMS macro summary

This appendix contains the syntax of each BMS macro, separating the various operands and options into those appropriate to minimum, standard, and full BMS.

| When coding, you should have the title in column 1, the
| macro in column 10, continuation lines should have * in
| column 72 and continue on column 16 on the next line.

For more information about BMS, see the *CICS/ESA Application Programming Guide*.

Mapset, map, and field definition

You should ensure that the names of maps, and names of fields within a mapset (or within multiple mapsets that are copied into one application program) are unique. Maps and mapsets must not have the same name.

Before CICS can load a physical map, you must define a physical map using an RDO transaction with the MAPSET attribute.

| An alternative to defining maps using RDO is to use the
| program autoinstall exit to create the definition when the
| mapset is first used. (For programming information about the
| autoinstall user program, see the *CICS/ESA Customization Guide*.)

You assemble a BMS mapset definition to generate either a symbolic description map or a physical map. The *CICS/ESA System Definition Guide* tells you how to assemble and catalog the maps.

Mapset definition macro (DFHMSD)

The DFHMSD macro defines a mapset.

Map definition macro (DFHMDI)

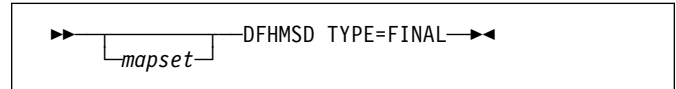
The DFHMDI macro defines a map within the mapset defined by the previous DFHMSD macro. A map contains zero or more fields.

Field definition macro (DFHMDF)

The DFHMDF macro defines a field within a map defined by the previous DFHMDI macro.

Ending a mapset definition

A mapset definition ends with a macro of the form:



“mapset” is optional, but if used it must be the same as that on the DFHMSD macro that began the mapset definition.

Partition set definition

Partitions are defined by coding the macros DFHPSD (partition set definition) and DFHPDI (partition definition). Each partition definition must be part of a partition set definition.

Partition set definition macro (DFHPSD)

Each partition set definition contains a single DFHPSD macro followed by one or more DFHPDI macros, and ending with a partition set definition TYPE=FINAL.

| Before CICS can load a physical map, you must define a
| physical map using an RDO transaction with the MAPSET
| attribute.

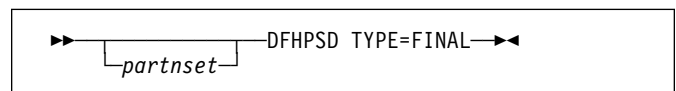
| An alternative to defining maps using RDO is to use the
| program autoinstall exit to create the definition when the
| mapset is first used. (For programming information about the
| autoinstall user program, see the *CICS/ESA Customization Guide*.)

Partition definition macro (DFHPDI)

A partition set contains one or more partitions. Each partition is defined by coding a partition definition macro.

Ending a partition set definition

A partition set definition ends with a macro of the form:



The PARTNSET name (if specified) must match that specified on the DFHPSD macro that started the partition set definition.

BMS macro summary

Field groups

Very often, an output data display field has to contain several subfields, all sharing the same display attributes, and each of which might have to be modified separately. At output, subfields that have not been modified by the program can adopt default data values from the output map. For example, a display can include a date field of a “day” subfield, “month” subfield, and “year” subfield. The contents of the year subfield remain constant over a relatively long period; its value can safely be taken from a map. However, the day value and month value must be updated regularly. Similarly, on input the terminal operator can enter data in each subfield separately.

You use the GRPNAME operand to define a group of subfields that combine to produce a field. The start of the group is indicated by a DFHMDF macro with the GRPNAME operand. This operand defines the first subfield, and specifies the attributes and name of the group. It is followed by other DFHMDF macros, one for each of the other subfields. Each of these must specify the group name, but cannot specify attribute values. The definition of the group is terminated by a DFHMDF macro that specifies a different group name, by one that specifies no group name, or by a DFHMDF or DFHMDF macro.

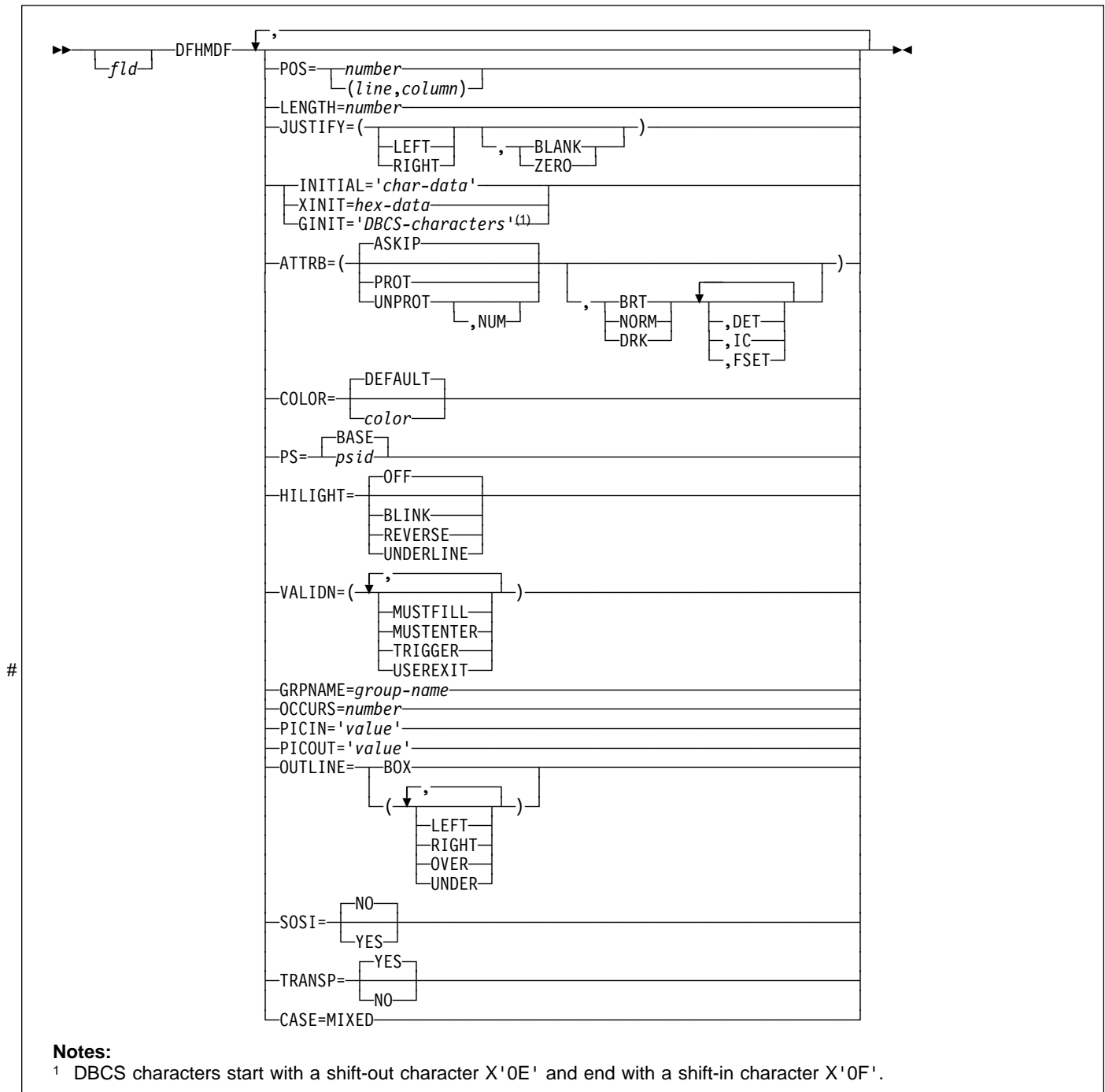
Briefly, a group of fields in a map would appear as follows in the map definition:

```
MAPSET DFHMDF . . . .
      .
      .
MAP   DFHMDF . . . .
      .
      .
DD    DFHMDF GRPNAME=DATE, POS=40,
      LENGTH=2, ATTRB= . . .
      .
MM    DFHMDF GRPNAME=DATE, POS=46,
      LENGTH=2
      .
YY    DFHMDF GRPNAME=DATE, POS=52,
      LENGTH=2
      .
FIELD DFHMDF LENGTH=5, COLOR=GREEN, . . .
      DFHMDF TYPE=FINAL
```

The POS operand specifies the position of the attribute byte of the field even though subfields of a group, other than the first, do not have attributes. If the subfields are positioned contiguously with no intervening blanks, the POS of the second and succeeding subfields must specify the position of the last character of the previous subfield.

DFHMDF

Command syntax



The DFHMDF macro defines a field within a map defined by the previous DFHMDI macro. A map contains zero or more fields.

| "fld" is the name (1–30 characters) of the field. You should, however, refer to your compiler manual to make sure that there are no other restrictions on the length.

For more information about defining field names, see the # CICS/ESA Application Programming Guide. If "fld" is

DFHMDF

omitted, application programs cannot access the field to change its attributes or alter its contents. For an output map, omitting the field name may be appropriate when the INITIAL operand is used to specify the contents of a field. If a field name is specified and the map that includes the field is used in a mapping operation, nonnull data supplied by the user overlays data supplied by initialization (unless default data only is being written).

The performance of input mapping operations is optimized if DFHMDF macros are arranged in numerical order of the POS operand.

You cannot define more than 1023 named fields for a COBOL, C, or PL/I input/output map.

You should ensure that the names of maps, and names of fields within a mapset (or within multiple mapsets that are copied into one application program) are unique. Maps and mapsets must not have the same name.

Before CICS can load a physical map, you must define a physical map using an RDO transaction with the MAPSET attribute.

DFHMDF operands

ATTRB

This operand applies only to 3270 data stream devices; it is ignored for other devices, except that ATTRB=DRK is honored for the SCS Printer Logical Unit. It is also ignored (except for ATTRB=DRK) if the NLEOM option is specified on the SEND MAP command for transmission to a 3270 printer. In particular, ATTRB=DRK should not be used as a method of protecting secure data on output on non-3270, non-SCS printer terminals. Refer to *An Introduction to the IBM 3270 Information Display System* for further information.

If ATTRB is specified within a group of fields, it must be specified in the first field entry. A group of fields appears as one field to the 3270. Therefore, the ATTRB specification refers to all of the fields in a group as one field rather than as individual fields. It specifies device-dependent characteristics and attributes, such as the capability of a field to receive data, or the intensity to be used when the field is output. It could however, be used for making an input field nondisplay for secure entry of a password from a screen. For input map fields, DET and NUM are the only valid options; all others are ignored.

ASKIP is the default and specifies that data cannot be keyed into the field and causes the cursor to skip over the field.

BRT specifies that a high-intensity display of the field is required. Because of the 3270 attribute character bit assignments, a field specified as BRT is also potentially detectable. However, for the field to be

recognized as detectable by BMS, DET must also be specified.

DET

specifies that the field is potentially detectable.

The first character of a 3270 detectable field must be one of the following:

? > & blank

If ? or >, the field is a selection field; if & or blank, the field is an attention field. (See *An Introduction to the IBM 3270 Information Display System* for further details about detectable fields.)

A field for which BRT is specified is potentially detectable to the 3270, because of the 3270 attribute character bit assignments, but is not recognized as such by BMS unless DET is also specified.

DET and DRK are mutually exclusive.

If DET is specified for a field on a map with MODE=IN, only one data byte is reserved for each input field. This byte is set to X'00', and remains unchanged if the field is not selected. If the field is selected, the byte is set to X'FF'.

No other data is supplied, even if the field is a selection field and the ENTER key has been pressed.

If the data in a detectable field is required, all of the following conditions must be fulfilled:

1. The field must begin with one of the following characters:
? > & blank
and DET must be specified in the output map.
2. The ENTER key (or some other attention key) must be pressed after the field has been selected, although the ENTER key is not required for detectable fields beginning with & or a blank.
3. DET must not be specified for the field in the input map. DET must, however, be specified in the output map. For more information about BMS support of the light pen, see the *CICS/ESA Application Programming Guide*.

DRK

specifies that the field is nonprint/nondisplay. DRK cannot be specified if DET is specified.

FSET

specifies that the modified data tag (MDT) for this field should be set when the field is sent to a terminal.

Specification of FSET causes the 3270 to treat the field as though it has been modified. On a subsequent read from the terminal, this field is read, whether or not it has been modified. The MDT remains set until the field is rewritten without ATTRB=FSET, or until an output mapping request causes the MDT to be reset.

Either of two sets of defaults may apply when a field to be displayed on a 3270 is being defined but not all parameters are specified. If no ATTRB parameters are specified, ASKIP and NORM are assumed. If any parameter is specified, UNPROT and NORM are assumed for that field unless overridden by a specified parameter.

IC specifies that the cursor is to be placed in the first position of the field. The IC attribute for the last field for which it is specified in a map is the one that takes effect. If not specified for any fields in a map, the default location is zero. Specifying IC with ASKIP or PROT causes the cursor to be placed in an unkeyable field.

This option can be overridden by the CURSOR option of the SEND MAP command that causes the write operation.

NORM specifies that the field intensity is to be normal.

NUM ensures that the data entry keyboard is set to numeric shift for this field unless the operator presses the alpha shift key, and prevents entry of nonnumeric data if the Keyboard Numeric Lock feature is installed.

PROT specifies that data cannot be keyed into the field.

If data is to be copied from one device to another attached to the same 3270 control unit, the first position (address 0) in the buffer of the device to be copied from must not contain an attribute byte for a protected field. Therefore, when preparing maps for 3270s, ensure that the first map of any page does not contain a protected field starting at position 0.

UNPROT specifies that data can be keyed into the field.

CASE

specifies that the field contains both uppercase and lowercase data that is to be converted to uppercase if FEATURE=KATAKANA has been included in the terminal definition.

This should be specified if a field is known to contain lowercase Latin characters but may be displayed on a

katakana display. It should not be specified if the field may contain valid katakana characters.

COLOR

indicates the individual color, or the default color for the mapset (where applicable).

The valid colors are blue, red, pink, green, turquoise, yellow, and neutral.

The COLOR operand is ignored unless the terminal supports color, as indicated by FEATURE in the terminal control table TYPE=TERMINAL, or by the RDO option COLOR.

GINIT

specifies constant or default data for an output field. GINIT is used to specify data in DBCS character strings, which must be enclosed by SO (shift out, X'0E') and SI (shift in, X'0F') characters. When GINIT is specified, the length must be even and is the number of bytes in the string (that is, not the number of DBCS characters).

#

Apar 69549

#

Documentation for Apar 69549 added 13 Apr 1995 (TUCKER)

#

#

If a graphic data type (PS=X'F8') is used, and the language is COBOL2, a PIC G is generated. Only one of GINIT, INITIAL, or XINIT may be specified.

#

GRPNAME

is the name used to generate symbolic storage definitions and to combine specific fields under one group name. The same group name must be specified for each field that is to belong to the group. The length of the name is up to 30 characters though you should refer to your compiler manual to make sure that there are no other restrictions on the length.

#

Apar PQ27428

#

Documentation for Apar PQ27428 added 24/06/99

#

For more information on defining the group name, see the *CICS/ESA Application Programming Guide*. The same rules apply to group names.

#

If this operand is specified, the OCCURS operand cannot be specified.

#

The fields in a group must follow on; there can be gaps between them, but not other fields from outside the group. A field name must be specified for every field that belongs to the group, and the POS operand must also be specified to ensure that the fields follow each other. All the DFHMDF macros defining the fields of a group must be placed together, and in the correct order (ascending numeric order of the POS value).

For example, the first 20 columns of the first six lines of a map can be defined as a group of six fields, as long as the remaining columns on the first five lines are not defined as fields.

DFHMDF

The ATTRB operand specified on the first field of the group applies to all of the fields within the group.

HIGHLIGHT

specifies the default highlighting attribute for all fields in all maps in a mapset.

OFF is the default and indicates that no highlighting is used.

BLINK specifies that the field must blink.

REVERSE specifies that the character or field is displayed in reverse video, for example, on a 3278, black characters on a green background.

UNDERLINE

specifies that a field is underlined.

The HIGHLIGHT operand is ignored unless the terminal supports highlighting, as indicated by the appropriate RDO program definition or by FEATURE in the terminal control table TYPE=TERMINAL.

INITIAL (or XINIT)

specifies constant or default data for an output field. INITIAL is used to specify data in character form; XINIT is used to specify data in hexadecimal form.

For fields with the DET attribute, initial data that begins with one of the following characters:

? > & blank

should be supplied.

The number of characters that can be specified in the INITIAL operand is restricted to the continuation limitation of the assembler to be used or to the value specified in the LENGTH operand (whichever is the smaller).

Hexadecimal data is written as an even number of hexadecimal digits, for example, XINIT=C1C2. If the number of valid characters is smaller than the field length, the data is padded on the right with blanks. For example, if LENGTH=3, XINIT=C1C2 results in an initial field of 'AB'.

If hexadecimal data is specified that corresponds with line or format control characters, the results are unpredictable. The XINIT operand should therefore be used with care. Only one of GINIT, INITIAL, or XINIT may be specified.

JUSTIFY

specifies the field justifications for input operations. This operand is ignored for TCAM-supported 3600 and 3790, and for VTAM-supported 3600, 3650, and 3790 terminals, because input mapping is not available.

LEFT specifies that data in the input field is left-adjusted.

RIGHT specifies that data in the input field is right-adjusted.

BLANK specifies that blanks are to be inserted in any unfilled positions in an input field.

ZERO specifies that zeros are to be inserted in any unfilled positions in an input field.

LEFT and RIGHT are mutually exclusive, as are BLANK and ZERO. If certain arguments are supplied but others are not, assumptions are made as follows:

Specified	Assumed
LEFT	BLANK
RIGHT	ZERO
BLANK	LEFT
ZERO	RIGHT

If JUSTIFY is omitted, but the NUM attribute is specified, RIGHT and ZERO are assumed. If JUSTIFY is omitted, but attributes other than NUM are specified, LEFT and BLANK are assumed.

Note: If a field is initialized by an output map or contains data from any other source, data that is typed as input overwrites only the equivalent length of the existing data; surplus existing data remains in the field and could cause unexpected interpretation of the new data.

LENGTH

specifies the length (1–256 bytes) of the field or group of fields. This length should be the maximum length required for application program data to be entered into the field; it should not include the one-byte attribute indicator appended to the field by CICS for use in subsequent processing. The length of each individual subfield within a group must not exceed 256 bytes. LENGTH can be omitted if PICIN or PICOUT is specified, but is required otherwise. You can specify a length of zero only if you omit the label (field name) from the DFHMDF macro. That is, the field is not part of the application data structure and the application program cannot modify the attributes of the field. You can use a field with zero length to delimit an input field on a map.

The map dimensions specified in the SIZE operand of the DFHMDF macro defining a map can be smaller than the actual page size or screen size defined for the terminal.

If the LENGTH specification in a DFHMDF macro causes the map-defined boundary on the same line to be exceeded, the field on the output screen is continued by wrapping.

OCCURS

specifies that the indicated number of entries for the field are to be generated in a map, and that the map definition is to be generated in such a way that the fields are addressable as entries in a matrix or an array. This permits several data fields to be addressed by the same name (subscripted) without generating a unique name for each field.

OCCURS and GRPNAME are mutually exclusive; that is, OCCURS cannot be used when fields have been defined under a group name. If this operand is omitted, a value of OCCURS=1 is assumed.

OUTLINE

allows lines to be included above, below, to the left, or to the right of a field. You can use these lines in any combination to construct boxes around fields or groups of fields.

PICIN (COBOL and PL/I only)

specifies a picture to be applied to an input field in an IN or INOUT map; this picture serves as an editing specification that is passed to the application program, thus permitting the user to exploit the editing capabilities of COBOL or PL/I. BMS checks that the specified characters are valid picture specifications for the language of the map.

However, the validity of the input data is not checked by BMS or the high-level language when the map is used, so any desired checking must be performed by the application program. The length of the data associated with "value" should be the same as that specified in the LENGTH operand if LENGTH is specified. If both PICIN and PICOUT (see below) are used, an error message is produced if their calculated lengths do not agree; the shorter of the two lengths is used. If PICIN or PICOUT is not coded for the field definition, a character definition of the field is automatically generated regardless of other operands that are coded, such as ATTRB=NUM.

As an example, assume the following map definition is created for reference by a COBOL application program:

```
MAPX DFHMSD TYPE=DSECT,
      LANG=COBOL,
      MODE=INOUT
MAP DFHMDI LINE=1,COLUMN=1,
      SIZE=(1,80)
F1 DFHMDF POS=0,LENGTH=30
F2 DFHMDF POS=40,LENGTH=10,
      PICOUT='$$$,$$0.00'
F3 DFHMDF POS=60,LENGTH=6,
      PICIN='9999V99',
      PICOUT='ZZ9.99'
      DFHMSD TYPE=FINAL
```

This generates the following DSECT:

```
01 MAPI.
02 F1L PIC S9(4) COMP.
02 F1A PIC X.
02 FILLER REDEFINES F1A.
03 F1F PIC X.
02 F1I PIC X(30).
02 FILLER PIC X.
02 F2L PIC S9(4) COMP.
02 F2A PIC X.
02 FILLER REDEFINES F2A.
03 F2F PIC X.
02 F2I PIC X(10).
02 FILLER PIC X.
02 F3L PIC S9(4) COMP.
02 F3A PIC X.
02 FILLER REDEFINES F3A.
03 F3F PIC X.
02 F3I PIC 9999V99.
02 FILLER PIC X.

01 MAPO REDEFINES MAPI.
02 FILLER PIC X(3).
02 F10 PIC X(30).
02 FILLER PIC X.
02 FILLER PIC X(3).
02 F20 PIC $$$,$$0.00.
02 FILLER PIC X.
02 FILLER PIC X(3).
02 F30 PIC ZZ9.99.
02 FILLER PIC X.
```

The valid picture values for COBOL input maps are:

A P S V X 9 / and (

The valid picture values for PL/I input maps are:

A B E F G H I K M P R S T V

X Y and Z

1 2 3 6 7 8 9 / + - , . *

\$ and (

Apar PQ14850
 # Documentation for Apar PQ14850 added 09/07/98

For PL/I, a currency symbol can be used as a picture
 # character. The symbol can be any sequence of
 # characters enclosed in < and >, for example <DM>.

Refer to the appropriate language reference manual for
 the correct syntax of the PICTURE attribute.

PICOUT (COBOL and PL/I only)

is similar to PICIN, except that a picture to be applied to
 an output field in the OUT or INOUT map is generated.

Note: The valid picture values for COBOL output maps
 are:

A B E P S V X Z 0 9 , . + - \$

CR DB / and (

The valid picture values for PL/I output maps are:

A B E F G H I K M P R S T V

X Y and Z

1 2 3 6 7 8 9 / + - , . * \$

CR DB and (

Apar PQ14850
 # Documentation for Apar PQ14850 added 09/07/98

For PL/I, a currency symbol can be used as a picture
 # character. The symbol can be any sequence of
 # characters enclosed in < and >, for example <DM>.

Refer to the appropriate language reference manual for
 the correct syntax of the PICTURE attribute.

Note:

Apar PQ21323
 # Documentation for Apar PQ21323 added
 # 11/12/98

COBOL supports multiple currency signs and
 # multi-character currency signs in PICTURE
 # specifications.

The default currency picture symbol is the dollar
 # sign (\$), which represents the national currency
 # symbol; for example the dollar (\$), the pound (£),
 # or the yen (¥).

The default currency picture symbol may be
 # replaced by a different currency picture symbol
 # that is defined in the SPECIAL NAMES clause.

The currency sign represented by the picture
 symbol is defined in the same clause. For
 example:

SPECIAL NAMES.

CURRENCY SIGN IS '\$' WITH PICTURE SYMBOL '\$'.

CURRENCY SIGN IS '£' WITH PICTURE SYMBOL '£'.

CURRENCY SIGN IS 'EUR' WITH PICTURE SYMBOL '#'.

WORKING STORAGE SECTION.

01 USPRICE PIC \$99.99.

01 UKPRICE PIC £99.99.

01 ECPRIE PIC #99.99.

LENGTH must be specified when PICOUT
 specifies a COBOL picture containing a currency
 symbol that will be replaced by a currency sign
 of length greater than 1.

POS

specifies the location of a field. This operand specifies
 the individually addressable character location in a map
 at which the attribute byte that precedes the field is
 positioned.

number specifies the displacement (relative to zero)
 from the beginning of the map being
 defined.

(line,column)

specify lines and columns (relative to one)
 within the map being defined.

The location of data on the output medium
 is also dependent on DFHMDFI operands.

The first position of a field is reserved for an
 attribute byte. When supplying data for
 input mapping from non-3270 devices, the
 input data must allow space for this attribute
 byte. Input data must not start in column 1
 but may start in column 2.

The POS operand always contains the
 location of the first position in a field, which
 is normally the attribute byte when
 communicating with the 3270. For the
 second and subsequent fields of a group,
 the POS operand points to an assumed
 attribute-byte position, ahead of the start of
 the data, even though no actual attribute
 byte is necessary. If the fields follow on
 immediately from one another, the POS
 operand should point to the last character
 position in the previous field in the group.

When a position number is specified that
 represents the last character position in the
 3270, two special rules apply:

- ATTRIB=IC should not be coded. The
 cursor can be set to location zero by
 using the CURSOR option of a SEND
 MAP, SEND CONTROL, or SEND
 TEXT command.

- If the field is to be used in an output mapping operation with MAP=DATAONLY on the SEND MAP command, an attribute byte for that field must be supplied in the symbolic map data structure by the application program.

PS specifies that programmed symbols are to be used. This overrides any PS operand set by the DFHMDFI macro or the DFHMDF macro.

BASE is the default and specifies that the base symbol set is to be used.

psid specifies a single EBCDIC character, or a hexadecimal code of the form X'nn', that identifies the set of programmed symbols to be used.

The PS operand is ignored unless the terminal supports programmed symbols, as indicated by PROGSYMBOLS(YES) on the RDO TYPETERM definition, or by FEATURE in the terminal control table TYPE=TERMINAL.

SOSI

indicates that the field may contain a mixture of EBCDIC and DBCS data. The DBCS subfields within an EBCDIC field are delimited by SO (shift out) and SI (shift in) characters. SO and SI both occupy a single screen position (normally displayed as a blank). They can be included in any non-DBCS field on output, if they are correctly paired. The terminal user can transmit them inbound if they are already present in the field, but can add them to an EBCDIC field only if the field has the SOSI attribute.

TRANSP

determines whether the background of an alphanumeric field is transparent or opaque, that is, whether an underlying (graphic) presentation space is visible between the characters.

VALIDN

specifies:

- # • the validation that is to be used on an 8775 terminal
- # • that this field is to be processed by the BMS global
- # user exits

This overrides any VALIDN operand on the DFHMDFI macro or the DFHMDF macro.

MUSTFILL specifies that the field must be filled completely with data. An attempt to move the cursor from the field before it has been filled, or to transmit data from an incomplete field, raises the INHIBIT INPUT condition

MUSTENTER

specifies that data must be entered into the field, though need not fill it. An attempt to

move the cursor from an empty field raises the INHIBIT INPUT condition

TRIGGER specifies that this field is a trigger field. Trigger fields are discussed in the *CICS/ESA Application Programming Guide*.

Apar PQ12071

Documentation for Apar PQ12071 added 12/03/98

USEREXIT

specifies that this field is to be processed by the BMS global user exits, XBMIN and XBMOU, if this field is received or transmitted in a 3270 datastream when the respective exit is enabled. For further information on the use of the BMS global user exits, refer to the *CICS/ESA Customization Guide*.

The MUSTFILL, MUSTENTER and TRIGGER specifications are valid only for terminals that support the field validation extended attribute, otherwise they are ignored. The USEREXIT specification applies to all 3270 devices.

Note: The USEREXIT specification is totally unconnected with the field validation extended attribute as defined in the 3270 datastream architecture.

The VALIDN operand is ignored unless the terminal supports validation, as indicated by VALIDATION(YES) option on the RDO TYPETERM definition, or by FEATURE in the terminal control table TYPE=TERMINAL.

XINIT

see INITIAL, earlier in the list. Only one of GINIT, INITIAL, or XINIT may be specified.

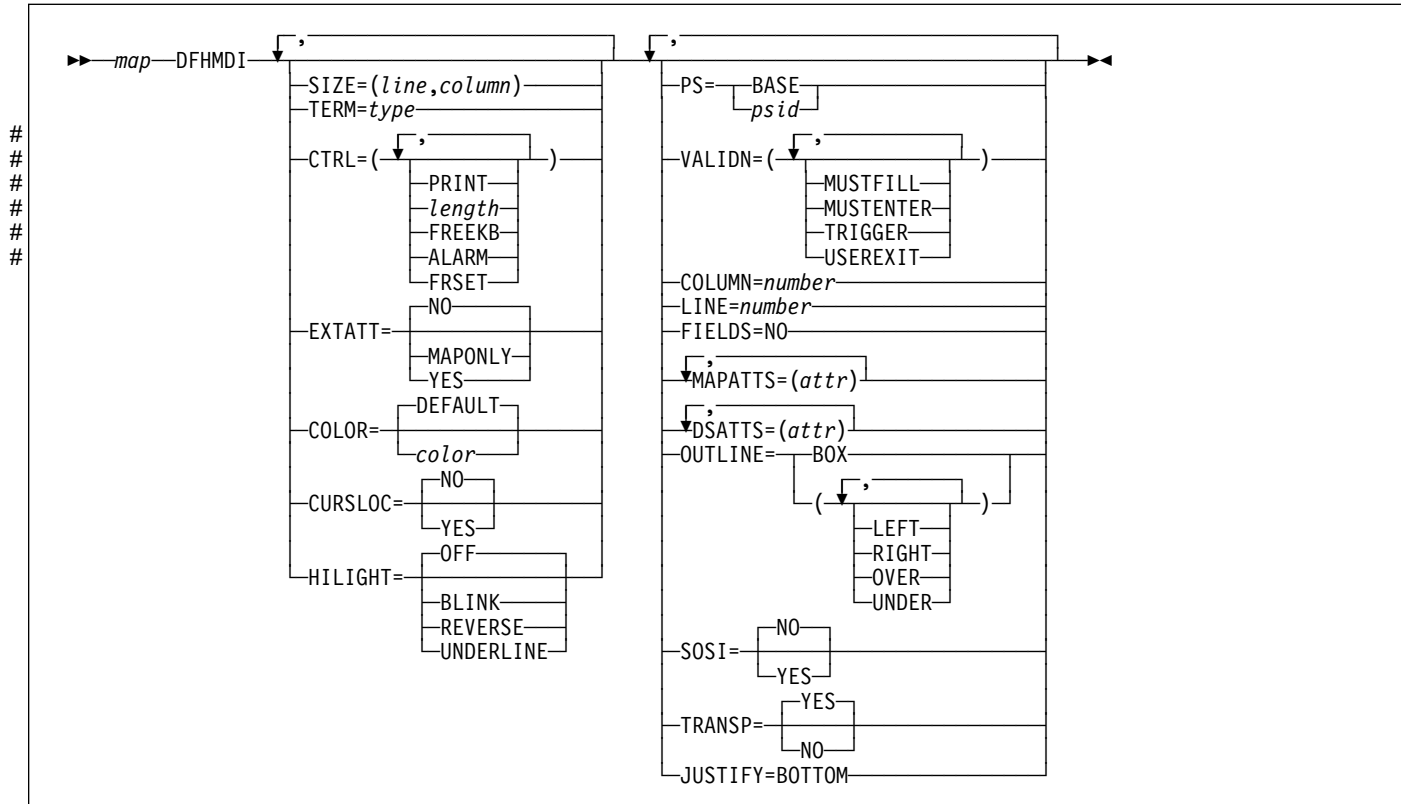
DFHMDI

Function

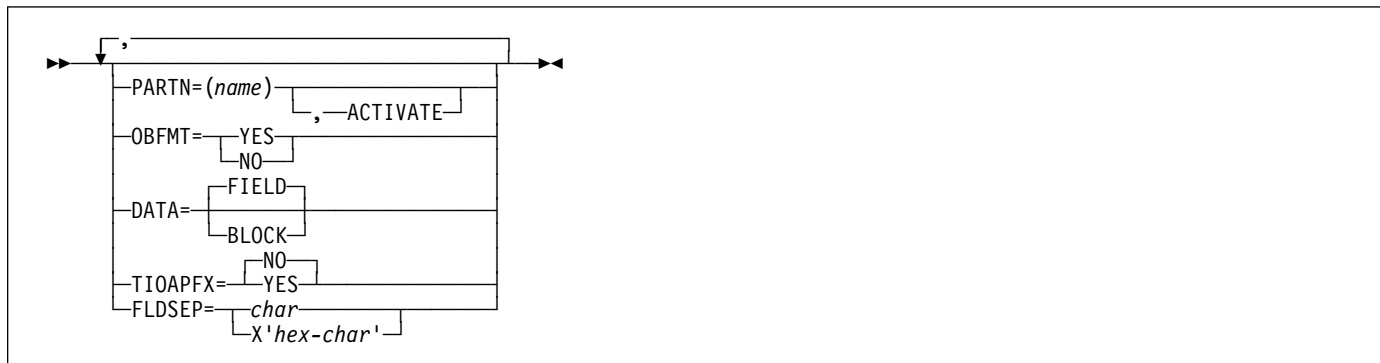
Map definition.

Command syntax

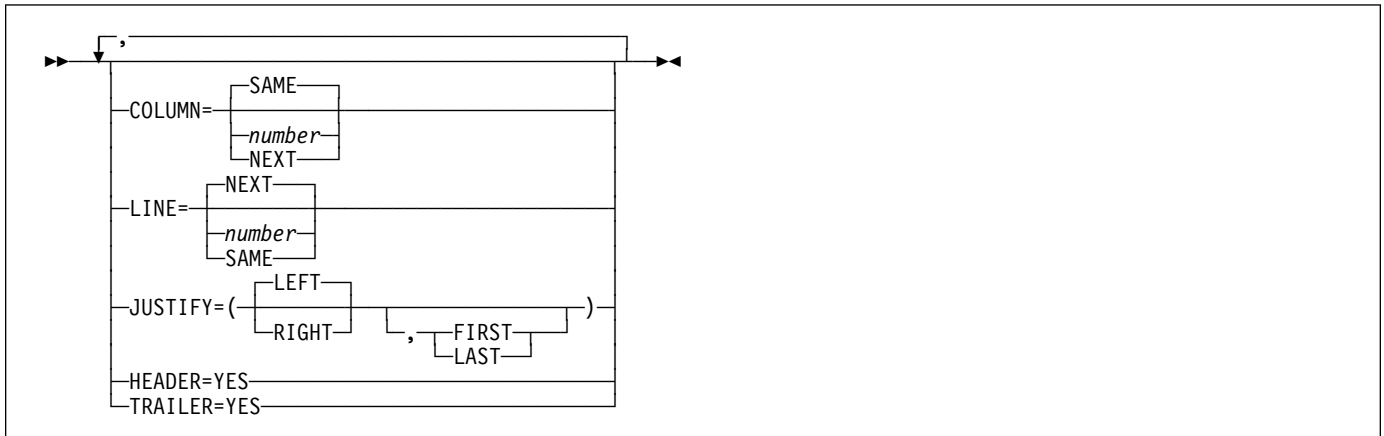
Minimum BMS:



Standard BMS:



Full BMS:



The DFHMDI macro defines a map within the mapset defined by a previous DFHMSD macro. A map contains zero or more fields.

“map” is the name (1–7 characters) of the map.

You should ensure that the names of maps, and names of fields within a mapset (or within multiple mapsets that are copied into one application program) are unique. Maps and mapsets must not have the same name.

Before CICS can load a physical map, you must define a physical map using an RDO transaction with the MAPSET attribute.

Note for COBOL users: If the maps are for use in a COBOL program, and STORAGE=AUTO has not been specified in the DFHMSD macro, they must be specified in descending size sequence. (Size refers to the generated 01-level data areas and not to the size of the map on the screen.)

DFHMDI operands

COLOR

indicates the individual color, or the default color for the mapset (where applicable). This is overridden by the COLOR operand of the DFHMDF macro.

The valid colors are blue, red, pink, green, turquoise, yellow, and neutral.

The COLOR operand is ignored unless the terminal supports color, as indicated by FEATURE in the terminal control table TYPE=TERMINAL, or with the RDO TYPETERM definition option COLOR.

COLUMN

specifies the column in a line at which the map is to be placed, that is, it establishes the left or right map margin. The JUSTIFY operand of the DFHMDI macro controls whether map and page margin selection and column counting are to be from the left or right side of the page. The columns between the specified map margin and the

page margin are not available for subsequent use on the page for any lines included in the map.

NUMBER is the column from the left or right page margin where the left or right map margin is to be established.

NEXT indicates that the left or right map margin is to be placed in the next available column from the left or right on the current line.

SAME indicates that the left or right map margin is to be established in the same column as the last nonheader or nontrailer map used that specified COLUMN=number and the same JUSTIFY operands as this macro.

For input operations, the map is positioned at the extreme left-hand or right-hand side, depending on whether JUSTIFY=LEFT or JUSTIFY=RIGHT has been specified.

CTRL

defines characteristics of IBM 3270 terminals. Use of **any** of the control options in the SEND MAP command overrides **all** control options in the DFHMDI macro, which in turn overrides **all** control options in the DFHMSD macro.

If CTRL is used with cumulative BMS paging (that is, the ACCUM option is used on the BMS SEND MAP commands), it must be specified on the last (or only) map of a page, unless it is overridden by the ALARM, FREEKB and so on, options on the SEND MAP or accumulated SEND CONTROL command.

PRINT must be specified if the printer is to be started; if omitted, the data is sent to the printer buffer but is not printed. This operand is ignored if the mapset is used with 3270 displays without the Printer Adapter feature.

LENGTH indicates the line length on the printer; length can be specified as L40, L64, L80, or HONEOM. L40, L64, and L80 force a new line after 40, 64, or 80 characters,

DFHMDI

respectively. HONEOM causes the default printer line length to be used. If this option is omitted, BMS sets the line length from the TCT page size.

FREEKB causes the keyboard to be unlocked after the map is written. If FREEKB is not specified, the keyboard remains locked; data entry from the keyboard is inhibited until this status is changed.

ALARM activates the 3270 audible alarm. For non-3270 VTAM terminals it sets the alarm flag in the FMH. (This feature is not supported by interactive and batch logical units.)

FRSET specifies that the modified data tags (MDTs) of all fields currently in the 3270 buffer are to be reset to an unmodified condition (that is, field reset) before map data is written to the buffer. This allows the DFHMDF macro with the ATTRB operand to control the final status of any fields written or rewritten in response to a BMS command.

CURSLOC

indicates that for all RECEIVE MAP operations using this map on 3270 terminals, BMS sets a flag in the application data structure element for the field where the cursor is located.

The flag may be tested by DFHBMCUR (see copybook DFHBMSCA in Appendix J, "BMS-related constants" on page 375).

To test the flag (COBOL example):

```
(DFHBMSCA)
...
02 DFHBMEOF PIC X VALUE X'80'.
02 DFHBMCUR PIC X VALUE X'02'.
02 DFHBMEC PIC X VALUE X'82'.
02 DFHBMFLG PIC X.
    88 DFHERASE VALUES ARE X'80', X'82'.
    88 DFHCURSR VALUES ARE X'02', X'82'.
MOVE FLD1F TO DFHBMFLG.
IF DFHERASE THEN ...
    ELSE ...
IF DFHCURSR THEN ...
    ELSE ...
```

Notes:

1. If CURSLOC=YES is specified for the MAP definitions, and there is no data for any field of the application data structure, but the cursor lies within a field known to the application data structure, BMS sets the cursor flag for the appropriate field, but the data for all fields in the application data structure is null, and the MAPFAIL condition does not occur. The unmapped data stream is not available to the application program unless it is a RECEIVE DATA FROM request.
2. A valid CURSLOC definition in DFHMDI overrides the definition in DFHMDS.

DATA

specifies the format of the data.

FIELD specifies that the data is passed as contiguous fields, each field having the format:

LL	A	data field
----	---	------------

"LL" is two bytes specifying the length of the data as input from the terminal (ignored in output processing). "A" is a byte into which the programmer can place an attribute to override that specified in the map used to process this data (see copybook DFHBMSCA in Appendix J, "BMS-related constants" on page 375).

BLOCK specifies that the data is passed as a continuous stream in the following format:

A	data field	space
---	------------	-------

This stream is processed as line segments of the length specified in the map used to process the data. The data is in the form in which it appears at the terminal; that is, it contains data fields and interspersed blanks corresponding to any spaces that are to appear between the fields on output. You cannot use DSATTS=YES if you specify DATA=BLOCK.

Block data is further discussed in the *CICS/ESA Application Programming Guide*.

DSATTS

specifies the attribute types to be included in the symbolic description map. These types can be one or more of the following: COLOR, HILIGHT, OUTLINE, PS, SOSI, TRANSP, and VALIDN. Any type included in DSATTS should also be included in MAPATTS.

EXTATT

this operand is supported for compatibility with previous releases. Each of the extended attributes can be defined individually. For new maps, the operands DSATTS and MAPATTS should be used instead.

NO is equivalent to specifying neither the DSATTS operand nor the MAPATTS operand.

YES is equivalent to:
MAPATTS=(COLOR,HILIGHT,PS,VALIDN)
DSATTS=(COLOR,HILIGHT,PS,VALIDN)

MAPONLY is equivalent to:
MAPATTS=(COLOR,HILIGHT,PS,VALIDN)

FIELDS

specifies whether or not the map contains fields. If you specify FIELDS=NO, you create a null map that defines a "hole" in BMS's view of the screen. BMS cannot change the contents of such a hole after it has created it by sending a null map.

FLDSEP

specifies the field separator sequence (1–4 characters) for input from non-3270 devices. Input from non-3270 devices can be entered as a single string of data with the field separator sequence delimiting fields. The data between the field separators is moved to the input fields in the map in order.

HEADER

allows the map to be used during page building without terminating the OVERFLOW condition. This operand may be specified for more than one map in a mapset.

HILIGHT

specifies the default highlighting attribute for all fields in all maps in a mapset. This is overridden by the HILIGHT operand of the DFHMDF.

OFF is the default and indicates that no highlighting is used.

BLINK specifies that the field must blink.

REVERSE specifies that the character or field is displayed in reverse video, for example, on a 3278, black characters on a green background.

UNDERLINE

specifies that a field is underlined.

The HILIGHT operand is ignored unless the terminal supports highlighting, as indicated by HILIGHT(YES) on the RDO TYPETERM definition, or by FEATURE in the terminal control table TYPE=TERMINAL.

JUSTIFY

specifies the position of the map on the page.

LEFT specifies that the map is to be positioned starting at the specified column from the left margin on the specified line.

RIGHT specifies that the map is to be positioned starting at the specified column from the right margin on the specified line.

FIRST specifies that the map is to be positioned as the first map on a new page. Any partially formatted page from preceding BMS commands is considered to be complete. This operand can be specified for only one map per page.

LAST indicates that the map is to be positioned at the bottom of the current page. This operand can be specified for multiple maps to be placed on one page. However, maps other than the first map for which it is specified must be able to be positioned horizontally without requiring that more lines be used.

BOTTOM for a SEND MAP ACCUM command has the same effect as LAST, above. For a SEND MAP command (without ACCUM) and a RECEIVE MAP command, JUSTIFY=BOTTOM positions the map at the bottom of the screen if the number of lines in the map is specified in the SIZE operand. No account is taken of trailer maps in the mapset. JUSTIFY=BOTTOM is equivalent to specifying

LINE=(screendepth-mapdepth+1)

on the map definition, but it allows the same map to be used for different screen sizes. JUSTIFY=BOTTOM is ignored if the number of lines is not also specified. If JUSTIFY=BOTTOM and LINE are both specified, the value specified in LINE is ignored.

LEFT and RIGHT are mutually exclusive, as are FIRST and LAST. If neither FIRST nor LAST is specified, the data is mapped at the next available position as

DFHMDI

determined by other parameters of the map definition and the current mapping operation. FIRST or LAST is ignored unless ACCUM is specified on SEND MAP commands; otherwise only one map is placed on each page.

Note: If a field is initialized by an output map or contains data from any other source, data that is keyed as input overwrites only the equivalent length of the existing data; surplus existing data remains in the field and could cause unexpected interpretation of the new data.

LINE

specifies the starting line on a page in which data for a map is to be formatted.

NUMBER is a value in the range 1–240, specifying a starting line number. A request to map, on a line and column, data that has been formatted in response to a preceding BMS command, causes the current page to be treated as though complete. The new data is formatted at the requested line and column on a new page.

NEXT specifies that formatting of data is to begin on the next available completely empty line. If LINE=NEXT is specified in the DFHMDI macro, it is ignored for input operations and LINE=1 is assumed.

SAME specifies that formatting of data is to begin on the same line as that used for a preceding BMS command. If COLUMN=NEXT is specified, it is ignored for input operations and COLUMN=1 is assumed. If the data does not fit on the same line, it is placed on the next available completely-empty line.

MAPATTS

specifies the attribute types to be included in the physical map. These types can be one or more of the following: COLOR, HILIGHT, OUTLINE, PS, SOSI, TRANSP, and VALIDN. This list must include all the attribute types to be specified for individual fields in the map (DFHMDF macro).

Where possible these values are deduced from operands already specified in the DFHMDI and DFHMDS macros. For example, if COLOR=BLUE has been specified, MAPATTS=COLOR is assumed.

OBFMT

specifies whether outboard formatting is to be used. This operand is available only for 3650 logical units, or for an 8100 series processor running DPS Release 2 and defined to CICS as an LUTYPE2 logical unit. For more information, see the *CICS/ESA Application Programming Guide*.

The OBFMT operand overrides the OBFMT operand on the DFHMDS macro.

YES specifies that this map definition can be used in outboard formatting.

NO specifies that this map definition cannot be used in outboard formatting.

OUTLINE

allows lines to be included above, below, to the left, or to the right of a field. You can use these lines in any combination to construct boxes around fields or groups of fields.

PARTN

specifies the default partition to be associated with maps in this mapset. If the ACTIVATE option is specified, the specified partition is also activated when maps in this mapset are output to a terminal that supports partitions.

This option overrides the PARTN option of the DFHMDS macro and is overridden by any OUTPARTN or ACTPARTN option on the SEND MAP command, or the INPARTN option on a RECEIVE MAP command.

The PARTN option is ignored if the target terminal does not support partitions, or if there is no partition set associated with the transaction.

PS specifies that programmed symbols are to be used. This overrides the PS operand of the DFHMDS macro and is overridden by the PS operand of the DFHMDF macro.

BASE specifies that the base symbol set is to be used.

psid specifies a single EBCDIC character, or a hexadecimal code of the form X'nn', that identifies the set of programmed symbols to be used.

The PS operand is ignored unless the terminal supports programmed symbols, as indicated by the PROGSYMBOLS(YES) on the RDO TYPETERM definition, or by FEATURE in the terminal control table TYPE=TERMINAL.

SIZE

specifies the size of a map.

line is a value in the range 1–240, specifying the depth of a map as a number of lines.

column is a value in the range 1–240, specifying the width of a map as a number of columns.

This operand is required in the following cases:

- An associated DFHMDF macro with the POS operand is used.
- The map is to be referred to in a SEND MAP command with the ACCUM option.
- The map is to be used when referring to input data from other than a 3270 terminal in a RECEIVE MAP command.

SOSI

indicates that the field may contain a mixture of EBCDIC and DBCS data. The DBCS subfields within an EBCDIC field are delimited by SO (shift out) and SI (shift in) characters. SO and SI both occupy a single screen position (normally displayed as a blank). They can be included in any non-DBCS field on output, if they are correctly paired. The terminal user can transmit them inbound if they are already present in the field, but can add them to an EBCDIC field only if the field has the SOSI attribute.

TERM

kept for compatibility with previous releases.

TIOAPFX

specifies whether BMS should include a filler in the symbolic description maps to allow for the unused TIOA prefix. This operand overrides the TIOAPFX operand specified for the DFHMSD macro.

YES specifies that the filler should be included in the symbolic description maps. If TIOAPFX=YES is specified, all maps within the mapset have the filler. TIOAPFX=YES should **always** be used for command-level application programs.

NO is the default and specifies that the filler is not to be included.

TRAILER

allows the map to be used during page building without terminating the OVERFLOW condition. This operand may be specified for more than one map in a mapset. If a trailer map is used other than in the overflow environment, the space normally reserved for overflow trailer maps is not reserved while mapping the trailer map.

TRANSP

determines whether the background of an alphanumeric field is transparent or opaque, that is, whether an underlying (graphic) presentation space is visible between the characters.

VALIDN

- # specifies:
- # • the validation that is to be used on an 8775 terminal
 - # • that this field is to be processed by the BMS global user exits

This is overridden by the VALIDN operand of the DFHMDF macro, and overrides the VALIDN operand of the DFHMSD macro.

MUSTFILL specifies that the field must be filled completely with data. An attempt to move the cursor from the field before it has been filled, or to transmit data from an incomplete field, raises the INHIBIT INPUT condition

MUSTENTER

specifies that data must be entered into the field, though need not fill it. An attempt to move the cursor from an empty field raises the INHIBIT INPUT condition

TRIGGER

specifies that this field is a trigger field. Trigger fields are discussed in the *CICS/ESA Application Programming Guide*.

Apar PQ12071

Documentation for Apar PQ12071
added 12/03/98

USEREXIT

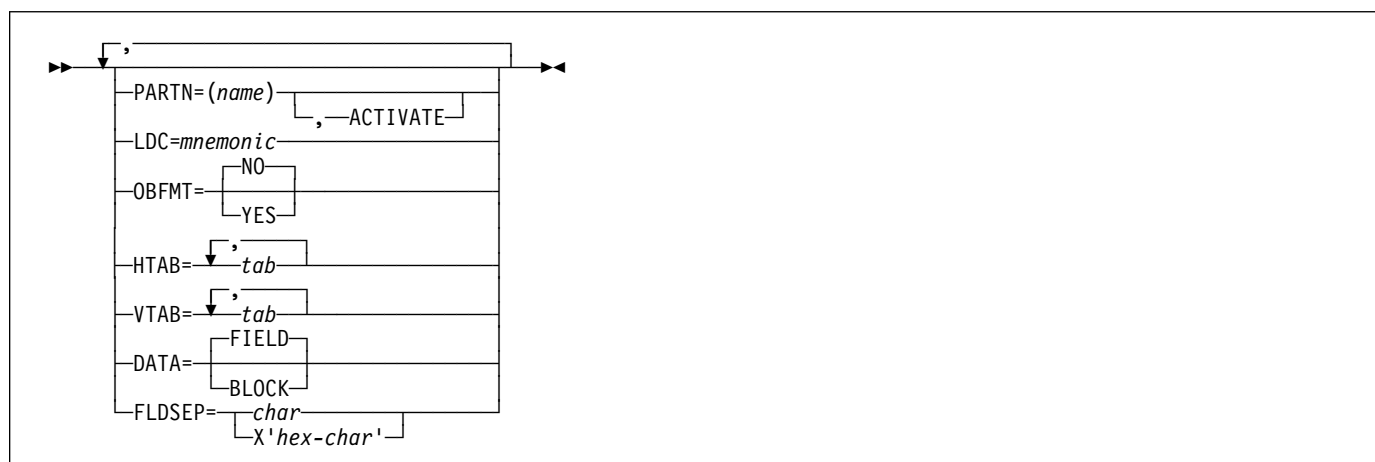
specifies that this field is to be processed by the BMS global user exits, XBMIN and XBMOU, if this field is received or transmitted in a 3270 datastream when the respective exit is enabled. For further information on the use of the BMS global user exits, refer to the *CICS/ESA Customization Guide*.

The MUSTFILL, MUSTENTER and TRIGGER specifications are valid only for terminals that support the field validation extended attribute, otherwise they are ignored. The USEREXIT specification applies to all 3270 devices.

Note: The USEREXIT specification is totally unconnected with the field validation extended attribute as defined in the 3270 datastream architecture.

The VALIDN operand is ignored unless the terminal supports validation, as indicated by VALIDATION(YES) on the RDO TYPETERM definition, or by FEATURE in the terminal control table TYPE=TERMINAL.

Standard BMS:



A DFHMSD macro defines a mapset; it begins:

```
DFHMSD TYPE=MAP    (or TYPE=DSECT)
```

and ends:

```
DFHMSD TYPE=FINAL
```

“mapset” is the name (1–7 characters) of the mapset.

A DFHMSD macro contains one or more map definition macros, each of which contains one or more field definition macros.

You should ensure that the names of maps, and names of fields within a mapset (or within multiple mapsets that are copied into one application program) are unique. Maps and mapsets must not have the same name.

Before CICS can load a physical map, you must define a physical map using an RDO transaction with the MAPSET attribute.

You assemble a BMS mapset definition to generate either a symbolic description map or a physical map. The *CICS/ESA System Definition Guide* tells you how to assemble and catalog the maps.

DFHMSD operands

BASE

specifies that the same storage base is used for the symbolic description maps from more than one mapset. The same name is specified for each mapset that is to share the same storage base. Because all mapsets with the same base describe the same storage, data related to a previously used mapset may be overwritten when a new mapset is used. Different maps within the same mapset also overlay one another.

This operand is not valid for assembler-language programs, and cannot be used when STORAGE=AUTO has been specified.

COLOR

indicates the individual color, or the default color for the mapset (where applicable). This is overridden by the COLOR operand of the DFHMDI macro, which is in turn overridden by the COLOR operand of the DFHMDF macro.

The valid colors are blue, red, pink, green, turquoise, yellow, and neutral.

The COLOR operand is ignored unless the terminal supports color, as indicated by FEATURE in the terminal control table TYPE=TERMINAL, or with the RDO TYPETERM definition COLOR.

CTRL

defines characteristics of IBM 3270 terminals. Use of **any** of the control options in the SEND MAP command overrides **all** control options in the DFHMDI macro, which in turn overrides **all** control options in the DFHMSD macro.

If CTRL is used with cumulative BMS paging (that is, the ACCUM option is used on the BMS SEND MAP commands), it must be specified on the last (or only) map of a page, unless it is overridden by the ALARM, FREEKB and so on, options on the SEND MAP or accumulated SEND CONTROL command.

PRINT

must be specified if the printer is to be started; if omitted, the data is sent to the printer buffer but is not printed. This operand is ignored if the mapset is used with 3270 displays without the Printer Adapter feature.

LENGTH

indicates the line length on the printer; length can be specified as L40, L64, L80, or HONEOM. L40, L64, and L80 force a new line after 40, 64, or 80 characters, respectively. HONEOM causes the default printer line length to be used. If this option is omitted, BMS sets the line length from the TCT page size.

DFHMSD

FREEKB causes the keyboard to be unlocked after the map is written. If FREEKB is not specified, the keyboard remains locked; data entry from the keyboard is inhibited until this status is changed.

ALARM activates the 3270 audible alarm. For non-3270 VTAM terminals, it sets the alarm flag in the FMH. (This feature is not supported by interactive and batch logical units.)

FRSET specifies that the modified data tags (MDTs) of all fields currently in the 3270 buffer are to be reset to a not-modified condition (that is, field reset) before map data is written to the buffer. This allows the DFHMDF macro with the ATTRB operand to control the final status of any fields written or rewritten in response to a BMS command.

CURSLOC

indicates that for all RECEIVE MAP operations using this map on 3270 terminals, BMS sets a flag in the application data structure element for the field where the cursor is located.

The flag may be tested by DFHBMCUR (see copybook DFHBMSCA in Appendix J, "BMS-related constants" on page 375).

To test the flag (COBOL example):

(DFHBMSCA)

```
...
02 DFHBMEOF PIC X VALUE X'80'.
02 DFHBMCUR PIC X VALUE X'02'.
02 DFHBMEC PIC X VALUE X'82'.
02 DFHBMFLG PIC X.
   88 DFHERASE VALUES ARE X'80', X'82'.
   88 DFHCURSR VALUES ARE X'02', X'82'.
MOVE FLD1F TO DFHBMFLG.
IF DFHERASE THEN ...
   ELSE ...
IF DFHCURSR THEN ...
   ELSE ...
```

Notes:

1. If CURSLOC=YES is specified for the MAP definitions, and there is no data for any field of the application data structure, but the cursor lies within a field known to the application data structure, BMS sets the cursor flag for the appropriate field, but the data for all fields in the application data structure is null, and the MAPFAIL condition does not occur. The unmapped data stream is not available to the application program unless it is a RECEIVE DATA FROM request.
2. A valid CURSLOC definition in DFHMDF overrides the definition in DFHMSD.

DATA

specifies the format of the data.

FIELD specifies that the data is passed as contiguous fields, each field having the format:

LL	A	data field
----	---	------------

"LL" is two bytes specifying the length of the data as input from the terminal (these two bytes are ignored in output processing). "A" is a byte into which the programmer can place an attribute to override that specified in the map used to process this data (see copybook DFHBMSCA in Appendix J, "BMS-related constants" on page 375).

BLOCK specifies that the data is passed as a continuous stream in the following format:

A	data field	space
---	------------	-------

This stream is processed as line segments of the length specified in the map used to process the data. The data is in the form that it appears on the terminal; that is, it contains data fields and interspersed blanks corresponding to any spaces that are to appear between the fields on output. You cannot use DSATTS=YES, if you specify DATA=BLOCK.

Block data is further discussed in the *CICS/ESA Application Programming Guide*.

DSATTS

specifies the attribute types to be included in the symbolic description map. These types can be one or more of the following: COLOR, HILIGHT, OUTLINE, PS, SOSI, TRANSP, and VALIDN. Any type included in DSATTS should also be included in MAPATTS.

EXTATT

this operand is supported for compatibility with previous releases. Each of the extended attributes can be defined individually. For new maps, the operands DSATTS and MAPATTS should be used instead.

NO is equivalent to specifying neither the DSATTS operand nor the MAPATTS operand.

YES is equivalent to:
MAPATTS=(COLOR,HILIGHT,PS,VALIDN)
DSATTS=(COLOR,HILIGHT,PS,VALIDN)

MAPONLY is equivalent to:
MAPATTS=(COLOR,HILIGHT,PS,VALIDN)

FLDSEP

specifies the field separator sequence (1–4 characters) for input from non-3270 devices. Input from non-3270 devices can be entered as a single string of data with the field separator sequence delimiting fields. The data

between the field separators is moved to the input fields in the map in order.

FOLD

specifies whether to generate lowercase or uppercase characters in C language programs.

FOLD is only available for programs written in C.

HIGHLIGHT

specifies the default highlighting attribute for all fields in all maps in a mapset. This is overridden by the HIGHLIGHT operand of the DFHMDI, which is in turn overridden by the HIGHLIGHT operand of the DFHMDF.

OFF is the default and indicates that no highlighting is used.

BLINK specifies that the field must blink.

REVERSE specifies that the character or field is displayed in reverse video, for example, on a 3278, black characters on a green background.

UNDERLINE

specifies that a field is underlined.

The HIGHLIGHT operand is ignored unless the terminal supports highlighting, as indicated by HIGHLIGHT(YES) on the RDO TYPETERM definition or by FEATURE in the terminal control table TYPE=TERMINAL.

HTAB

specifies one or more tab positions for use with interactive and batch logical units and SCS printers with horizontal forms control.

LANG

specifies the source language of the application programs into which the symbolic description maps in the mapset are copied. This option need only be coded for DFHMSD TYPE=DSECT. If a mapset is to be used by more than one program, and the programs are not all written in the same source language, a separate version of the mapset must be defined for each programming language.

LDC

specifies the code to be used by CICS to determine the logical device mnemonic to be used for a BMS output operation. If no LDC operand has been specified on any previous BMS output in the logical message, this LDC will be transmitted in the function management header to the logical unit. This operand is used only for TCAM and VTAM-supported 3600 terminals, and batch logical units.

MAPATTS

specifies the attribute types to be included in the physical map. These types can be one or more of the following: COLOR, HIGHLIGHT, OUTLINE, PS, SOSI, TRANSP, and VALIDN. This list must include all the attribute types to be specified for individual fields in the map (DFHMDF macro).

Where possible these values are deduced from operands already specified in the DFHMDI and DFHMSD macros. For example, if COLOR=BLUE has been specified, MAPATTS=COLOR is assumed.

MODE

specifies whether the mapset is to be used for input, output, or both.

OBFMT

specifies whether outboard formatting is to be used. This operand is available only for 3650 logical units, or for an 8100 series processor running DPS Release 2 and defined to CICS as an LUTYPE2 logical unit. For more information, see the *CICS/ESA Application Programming Guide*.

The OBFMT operand on DFHMSD is overridden by the OBFMT operand on DFHMDI.

YES specifies that all maps within this mapset can be used in outboard formatting, except those for which OBFMT=NO is specified in the DFHMDI macro.

NO specifies that no maps within this mapset can be used in outboard formatting, except those for which OBFMT=YES is specified in DFHMDI.

OUTLINE

allows lines to be included above, below, to the left, or to the right of a field. You can use these lines in any combination to construct boxes around fields or groups of fields.

PARTN

specifies the default partition to be associated with maps in this mapset. If the ACTIVATE option is specified, the specified partition is also activated when maps in this mapset are output to a terminal that supports partitions. This option is overridden by the PARTN operand of the DFHMDI macro, which is in turn overridden by any OUTPARTN or ACTPARTN option on the SEND MAP command, or the INPARTN option on a RECEIVE MAP command.

The PARTN operand is ignored if the target terminal does not support partitions, or if there is no partition set associated with the transaction.

PS specifies that programmed symbols are to be used. This is overridden by the PS operand of the DFHMDI macro, which is in turn overridden by the PS operand of the DFHMDF macro.

BASE specifies that the base symbol set is to be used.

psid specifies a single EBCDIC character, or a hexadecimal code of the form X'nn', that identifies the set of programmed symbols to be used.

DFHMSD

The PS operand is ignored unless the terminal supports programmed symbols, as indicated by PROGSYMBOLS(YES) on the RDO TYPETERM definition, or by FEATURE on the terminal control table TYPE=TERMINAL.

SOSI

indicates that the field may contain a mixture of EBCDIC and DBCS data. The DBCS subfields within an EBCDIC field are delimited by SO (shift out) and SI (shift in) characters. SO and SI both occupy a single screen position (normally displayed as a blank). They can be included in any non-DBCS field on output provided they are correctly paired. The terminal user can transmit them inbound if they are already present in the field, but can add them to an EBCDIC field only if the field has the SOSI attribute.

STORAGE

The meaning of this operand depends upon the language in which application programs are written, as follows:

For a **COBOL** program, STORAGE=AUTO specifies that the symbolic description maps in the mapset are to occupy separate (that is, not redefined) areas of storage. This operand is used when the symbolic description maps are copied into the working-storage section and the storage for the separate maps in the mapset is to be used concurrently.

For a **C** program, STORAGE=AUTO specifies that the symbolic description maps are to be defined as having the automatic storage class. If STORAGE=AUTO is not specified, they are declared as pointers. You cannot specify both BASE=name and STORAGE=AUTO for the same mapset. If STORAGE=AUTO is specified and TIOAPFX is not, TIOAPFX=YES is assumed.

For a **PL/I** program, STORAGE=AUTO specifies that the symbolic description maps are to be declared as having the AUTOMATIC storage class. If STORAGE=AUTO is not specified, they are declared as BASED. You cannot specify both BASE=name and STORAGE=AUTO for the same mapset. If STORAGE=AUTO is specified and TIOAPFX is not, TIOAPFX=YES is assumed.

For an **assembler-language** program, STORAGE=AUTO specifies that individual maps within a mapset are to occupy separate areas of storage instead of overlaying one another.

SUFFIX

specifies a 1-character, user-defined, device-dependent suffix for this mapset, as an alternative to a suffix generated by the TERM operand. The suffix specified by this operand should match the value of a transaction defined with ALTSUFFIX, or ALTSFX in the terminal control table TYPE=TERMINAL. Use a numeric value to avoid conflict with suffixes generated by the TERM operand.

TERM

specifies the type of terminal or logical unit (LU) associated with the mapset. If no terminal type or LU is specified, 3270 is assumed. The terminal types and LUs you can specify, together with their generated suffixes, are shown in Table 27.

Table 27. BMS terminal types

TYPE	Suffix
CRLP ¹	A
TAPE	B
DISK	C
TWX	D
1050	E
2740	F
2741	G
2770	I
2780	J
3780	K
3270-1 (40-column)	L
3270-2 (80-column)	M
INTLU/3767/3770I/SCS ²	P
2980	Q
2980-4	R
3270 ³	blank
3601	U
3653 ⁴	V
3650UP ⁵	W
3650/3270 ⁶	X
BCHLU/3770B ⁷	Y
ALL (all the above)	blank

Notes:

1. Card-reader-in/line-printer-out.
2. All interactive LUs including 3790 full-function LU and SCS printer LUs (3270 and 3790).
3. Default if TERM omitted.
Same as ALL; used when no need to distinguish between models.
4. Plus host-conv (3653) LU.
5. Plus interpreter LU.
6. Plus host-conv (3270) LU.
7. Plus all batch and BDI LUs.

In addition, you should note the following:

For TCAM-connected terminals (other than 3270 or SNA devices), use either CRLP or ALL; for TCAM-connected 3270s or SNA devices, select the appropriate parameter in the normal way.

If ALL is specified, ensure that device dependent characters are not included in the mapset and that format characteristics such as page size are suitable for all input/output operations (and all terminals) in which the mapset is applied. For example, some terminals are limited to 480 bytes, others to 1920 bytes; the 3604 is limited to six lines of 40 characters each. Within these guidelines, use of ALL can offer important advantages. Because an assembly run is required for each map generation, the use of ALL, indicating that one map is to be used for more than one terminal, can result in significant time and storage savings.

However, better run-time performance for maps used by single terminal types is achieved if the terminal type (rather than ALL) is specified. Alternatively, BMS support for device-dependent mapsets can be bypassed by specifying NODDS in the BMS operand of the system initialization parameters. For more information, see the *CICS/ESA Resource Definition Guide*.

TIOAPFX

specifies whether BMS should include a filler in the symbolic description maps to allow for the unused TIOA prefix.

YES specifies that the filler should be included in the symbolic description maps. If TIOAPFX=YES is specified, all maps within the mapset have the filler, except when TIOAPFX=NO is specified on the DFHMDI macro. TIOAPFX=YES should **always** be used for command-level application programs.

NO is the default and specifies that the filler is not to be included. The filler may still be included for a map if TIOAPFX=YES is specified on DFHMDI.

TRANSP

determines whether the background of an alphanumeric field is transparent or opaque, that is, whether an underlying (graphic) presentation space is visible between the characters.

TRIGRAPH

specifies trigraph sequences to be used in C/370 language symbolic description maps.

When TRIGRAPH=YES, trigraph sequences are produced:

```
{ prints as ??<
} prints as ??>
[ prints as ??(
] prints as ??)
```

This option is only available for programs written in C.

TYPE

specifies the type of map to be generated using the definition. Both types of map must be generated before the mapset can be used by an application program. If aligned symbolic description maps are required, you should ensure that you specify SYSPARM=ADSECT and SYSPARM=AMAP when you assemble the symbolic and physical maps respectively.

DSECT specifies that a symbolic description map is to be generated. Symbolic description maps must be copied into the source program before it is translated and compiled.

MAP specifies that a physical map is to be generated. Physical maps must be assembled or compiled, link-edited, and cataloged in the CICS program library

before an application program can use them.

If both map and DSECT are to be generated in the same job, the SYSPARM option can be used in the assembler job execution step, as described in the *CICS/ESA Installation Guide*.

VALIDN

specifies:

- # • the validation that is to be used on an 8775 terminal
- # • that this field is to be processed by the BMS global user exits

This is overridden by the VALIDN operand of the DFHMDI macro, which is in turn overridden by the VALIDN operand of the DFHMDF macro.

MUSTFILL specifies that the field must be filled completely with data. An attempt to move the cursor from the field before it has been filled, or to transmit data from an incomplete field, raises the INHIBIT INPUT condition.

MUSTENTER

specifies that data must be entered into the field, though need not fill it. An attempt to move the cursor from an empty field raises the INHIBIT INPUT condition.

TRIGGER

specifies that this field is a trigger field. Trigger fields are discussed in the *CICS/ESA Application Programming Guide*.

Apar PQ12071

Documentation for Apar PQ12071
added 12/03/98

USEREXIT

specifies that this field is to be processed by the BMS global user exits, XBMIN and XBMOU, if this field is received or transmitted in a 3270 datastream when the respective exit is enabled. For further information on the use of the BMS global user exits, refer to the *CICS/ESA Customization Guide*.

The MUSTFILL, MUSTENTER and TRIGGER specifications are valid only for terminals that support the field validation extended attribute, otherwise they are ignored. The USEREXIT specification applies to all 3270 devices.

Note: The USEREXIT specification is totally unconnected with the field validation extended attribute as defined in the 3270 datastream architecture.

The VALIDN operand is ignored unless the terminal supports validation, as indicated by VALIDATION(YES)

DFHMSD

in the RDO TYPETERM definition or by FEATURE in the terminal control table TYPE=TERMINAL.

VTAB

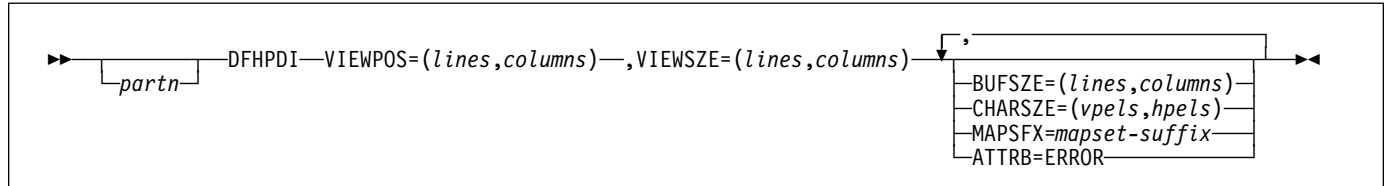
specifies one or more tab positions for use with interactive and batch logical units and SCS printers having vertical forms control.

DFHPDI

Function

Partition definition.

Command syntax



A partition set contains one or more partitions. Each partition is defined by coding a partition definition macro.

“partn” is a partition name (1–2 characters). It allows you to refer to the partition in your application programs.

Every partition in a partition set must have a different name. Only the error partition can be unnamed (see ATTRB=ERROR operand).

Partitions are defined by coding the macros DFHPSD (partition set definition) and DFHPDI (partition definition). Each partition definition must be part of a partition set definition.

DFHPDI operands

ATTRB

specifies that error messages are to be directed to this partition whenever possible. The partition is cleared before an error message is displayed. The RDO option ERRHIGHLIGHT, or ERRATT in the terminal control table TYPE=TERMINAL is honored, but the LASTLINE option is ignored.

BUFSIZE=(lines,columns)

specifies the size of the presentation space for the partition. Device limitations mean that the “columns” value must be equal to the “columns” value specified by the VIEWSIZE operand. The “lines” value can be greater than or, by default, equal to the value specified by the VIEWSIZE operand. A greater lines value implies that the target terminal supports vertical scrolling.

CHARSIZE(vpels, hpels)

specifies the size of the character cell to be reserved for each character displayed in a partition. You specify the size as numbers of vertical picture elements (vpels) and numbers of horizontal picture elements (hpels). You can specify this operand on either the DFHPSD macro only, or on both the DFHPSD and DFHPDI macros. The values specified in the DFHPSD become the defaults for all partitions in the partition set. You can override these

defaults for individual partitions by coding CHARSIZE in the DFHPDI macro.

MAPSFX=mapset-suffix

specifies the partition’s 1-character mapset suffix. BMS uses the suffix to select mapset versions in the same way as for the RDO option ALTSUFFIX, or ALTSFX in the terminal control terminal TYPE=TERMINAL macro. If this operand is omitted, a suffix L is assumed if the “columns” value of the BUFSIZE operand is less than or equal to 40; otherwise M is assumed.

VIEWPOS=(lines,columns)

specifies the position of the top left-hand corner of this partition’s viewport. You specify the position in numbers of lines and numbers of columns.

The DFHPDI macro checks that viewports do not overlap. If you have coded the RDO ALTSCREEN option, or the ALTSCRN operand of the DFHPSD macro, DFHPDI also checks that all viewports fit within the usable area of the terminal screen.

Note: The information given here on positioning viewports is necessarily brief. For more information you should consult the component description for the device you are using.

VIEWSIZE=(lines,columns)

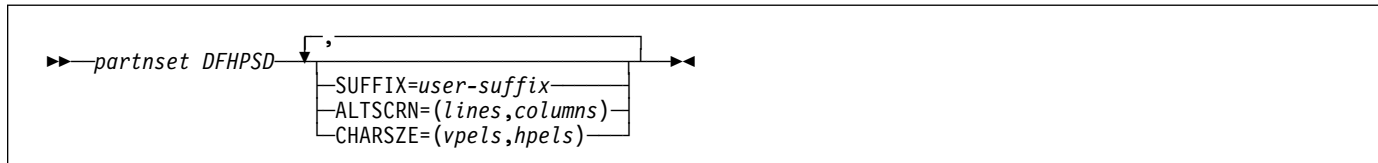
specifies the size, in lines and columns, of the partition’s viewport. The DFHPDI macro checks that viewports do not overlap. If you code the RDO ALTSCREEN option, or the ALTSCRN operand of the DFHPSD macro partition set definition macro, DFHPDI checks that the partitions all fit within the usable area of the display screen.

DFHPSD

Function

Partition set definition.

Command syntax



Each partition set definition contains a single DFHPSD macro followed by one or more DFHPDI macros, and ending with a DFHPSD TYPE=FINAL partition set definition macro.

“partnset” is a partition set name (1–6 characters).

Partitions are defined by coding the macros DFHPSD (partition set definition) and DFHPDI (partition definition). Each partition definition must be part of a partition set definition.

Ending DFHPSD

```
[partnset] DFHPSD TYPE=FINAL
```

The PARTNSET name (if specified) must match that specified on the DFHPSD macro that started the partition set definition.

DFHPSD operands

ALTSCRN(lines,columns)

specifies the size, in characters, of the usable area of the target terminal. This is normally the same as the RDO ALTSCREEN option, or the ALTSCRN operand of the terminal control table TYPE=TERMINAL entry for the terminal. You use ALTSCRN to ensure that the viewports of partitions within a partition set fit into the usable area of the screen.

CHARSIZE(vpels,hpels)

specifies the size of the character cell to be reserved for each character displayed in a partition. You specify the size as numbers of vertical picture elements (vpels) and numbers of horizontal picture elements (hpels). You can specify this operand on either the DFHPSD macro only, or on both the DFHPSD and DFHPDI macros. The values specified in this operand become the defaults for all partitions in the partition set. You can override this default for individual partitions by coding CHARIZE in the DFHPDI macro.

SUFFIX=user-suffix

specifies a 1-character user suffix for this version of the partition set. It allows different versions of a partition set to be associated with different terminals. When the partition set is to be loaded, CICS looks for a version whose suffix matches the RDO option ALTSUFFIX or the ALTSFX in the terminal control table TYPE=TERMINAL. If it cannot find the correct partition set version, it loads a version with a default suffix (M or L). If it cannot find a suffixed version either, it loads an unsuffixed one. If it cannot find this, it abends with APCT.

Index

Numerics

- 16MB line 121
- 2260 Display Station
 - CONVERSE 55
 - ISSUE RESET 161
 - RECEIVE 212
 - SEND 257
- 2265 Display Station
 - CONVERSE 55
 - ISSUE RESET 161
 - RECEIVE 212
 - SEND 257
- 2741 Communication Terminal
 - CONVERSE 56
 - ISSUE RESET 161
 - RECEIVE 212
 - SEND 257
- 2770 Data Communication System
 - CONVERSE 54, 57
 - RECEIVE 211
 - SEND 256
- 2780 Data Transmission Terminal
 - CONVERSE 54, 57
 - RECEIVE 211
 - SEND 256
- 2980 General Banking Terminal System
 - DFH2980 structure 214
 - output control 213
 - output to common buffer 214
 - passbook control 213
 - RECEIVE/SEND commands 213, 258
- 3270 Information Display System
 - (BTAM or TCAM supported) 58, 215, 258
 - (BTAM supported) 137, 161
 - logical unit 44, 138, 203, 245
- 3600 Finance Communication System
 - 3601 logical unit 44, 204, 246
 - 3614 logical unit 45, 205, 246
 - CONVERSE command 58
 - pipeline logical units 245
 - RECEIVE command 215
 - SEND command 259
- 3630 Plant Communication System
 - RECEIVE 204
 - SEND 246
- 3650 Host Command Processor
 - CONVERSE 47
- 3650 Logical Units
 - RECEIVE 205
- 3650 Store System
 - host conversational
 - LU 3270 247
 - LU 3653 248
 - 3650 Store System (*continued*)
 - interpreter logical unit 45, 145, 150, 205, 247
- 3650/3680 Full-function Logical Unit
 - RECEIVE 207
 - SEND 250
- 3650/3680 Store System
 - host command processor LU 248
- 3660 Supermarket Scanning System
 - CONVERSE 54
 - RECEIVE 211
 - SEND 256
- 3680 Host Command Processor
 - CONVERSE 47
- 3680 Programmable Store System
 - host command processor LU 248
- 3735 Programmable Buffered Terminal 60, 216, 259
- 3740 Data Entry System 60, 143, 144, 216
- 3767 Communication Terminal
 - interactive logical unit 47, 206, 249
- 3770 Data Communication System
 - batch logical unit 48, 206, 249
- 3770 Full Function Logical Unit
 - RECEIVE 207
 - SEND 250
- 3770 interactive logical unit
 - RECEIVE 206
 - SEND 249
- 3780 Data Communication Terminal
 - CONVERSE 54
 - RECEIVE 211
 - SEND 256
- 3790 Communication System
 - 3270-display logical unit 49, 217, 251
 - Full Function Logical Unit 48
 - full-function logical unit 207, 250
 - SCS printer logical unit 250
- 4700 Finance Communication System
 - CONVERSE command 58
 - RECEIVE command 215
 - SEND command 259

A

- ABCODE option
 - ABEND 12
 - ASSIGN 23
- ABDUMP option
 - ASSIGN 23
- ABEND AICA timeout 2
- ABEND command 12
- ABEND exit, reactivating 123
- abend support commands 9

- abnormal termination, task 123
- ABPROGRAM option
 - ASSIGN 23
- absolute expressions 4
- ABSTIME option
 - ASKTIME 21
 - FORMATIME 92
- access to system information
 - ADDRESS command 13
 - ADDRESS SET command 15
 - ASSIGN command 22
 - CICS storage areas 13, 15
- ACCUM option
 - SEND CONTROL 264
 - SEND MAP 268
 - SEND TEXT 280
- ACEE option
 - ADDRESS 13
- ACTION option
 - WRITE OPERATOR 334
- ACTPARTN option
 - SEND CONTROL 264
 - SEND MAP 269
 - SEND TEXT 280
- ADDRESS command 13
- ADDRESS SET command 15
- address, cursor 361
- AFTER option
 - POST 179
 - ROUTE 239
 - START 306
- AID option
 - RECEIVE MAP MAPPINGDEV command 223
- ALARM option
 - SEND CONTROL 264
 - SEND MAP 269
 - SEND MAP MAPPINGDEV command 274
 - SEND TEXT 281
 - SEND TEXT NOEDIT 286
- ALARM value
 - DFHMDI 392
 - DFHMSD 398
- ALIGNED attribute
 - PL/I 4
- ALL option
 - SEND PAGE 276
- ALLOCATE (APPC) command 16
- ALLOCATE (LUTYPE6.1) command 18
- ALLOCATE (MRO) command 20
- ALTER option
 - QUERY SECURITY 182
- ALTERNATE option
 - CONVERSE (non-VTAM) 61
 - CONVERSE (VTAM) 50
 - SEND (non-VTAM) 261
 - SEND (VTAM) 252
- ALTERNATE option (*continued*)
 - SEND CONTROL 265
 - SEND MAP 269
 - SEND TEXT 281
 - SEND TEXT NOEDIT 286
- ALTSCRN operand
 - DFHPSD 404
- ALTSCRNH option
 - ASSIGN 23
- ALTSCRNWD option
 - ASSIGN 23
- ANYKEY option
 - HANDLE AID 125
- APLKYBD option
 - ASSIGN 23
- APLTEXT option
 - ASSIGN 23
- APPC basic conversations
 - commands 9
- APPC logical unit
 - acquiring session to 16
 - initiating conversation with 39
 - returning mapped sessions to CICS 96
 - sending and receiving 41
- APPC mapped conversations
 - abending 130
 - commands 9
 - ensuring transmission of accumulated data 320
 - extracting attributes of 86
 - informing partner of error 149
 - issuing a positive response 135
 - receiving data 201
 - requesting change of direction 164
 - retrieving values from attach header 89
 - returning sessions to CICS 96
 - sending data 242
- application performance, monitoring 174
- APPLID option
 - ASSIGN 23
- argument values
 - assembler language 4
 - C 2
 - COBOL 2
 - PL/I 3
- ASA option
 - SPOOLOPEN 296
- ASIS option
 - CONVERSE (non-VTAM) 61
 - CONVERSE (VTAM) 50
 - RECEIVE (non-VTAM) 218
 - RECEIVE (VTAM) 208
 - RECEIVE MAP 220
 - RECEIVE PARTN 225
 - SEND (non-VTAM) 261
- ASKIP value
 - DFHMDF 384

ASKTIME command 21
 ASRAINTRPT option
 ASSIGN 23
 ASRAKEY option
 ASSIGN 23
 ASRAPSW option
 ASSIGN 23
 ASRAREGS option
 ASSIGN 23
 ASRASPC option
 ASSIGN 24
 ASRASTG option
 ASSIGN 24
 assembler language
 argument values 4
 LENGTH option default 4
 program exit 354
 register contents 353
 translated code 353
 ASSIGN command 22
 asynchronous interrupt 358
 AT option
 POST 179
 ROUTE 239
 START 306
 ATTACH option
 START 306
 ATTACHID option
 BUILD ATTACH (LUTYPE6.1) 31
 BUILD ATTACH (MRO) 33
 CONVERSE (non-VTAM) 61
 CONVERSE (VTAM) 50
 EXTRACT ATTACH (LUTYPE6.1) 82
 EXTRACT ATTACH (MRO) 84
 SEND (non-VTAM) 261
 SEND (VTAM) 252
 attention identifier (AID) 125
 ATTRB operand
 DFHMDF 384
 DFHPDI 403
 attributes
 control character list, DFHBMSCA 375
 authentication commands 9
 AUTOPAGE option
 SEND PAGE 276
 AUXILIARY option
 WRITEQ TS 338

B

back out to a syncpoint 315
 BASE operand
 DFHMDS 397
 BASE value
 DFHMDF 389
 DFHMDI 394
 BASE value (*continued*)
 DFHMDS 399
 basic mapping support (BMS)
 commands 9
 completing a logical message 276
 deleting a logical message 180
 determining input partition 225
 field definition macro 381, 383
 full BMS
 RECEIVE MAP 220
 RECEIVE PARTN 225
 SEND CONTROL 264
 SEND MAP 268
 SEND PAGE 276
 SEND PARTNSET 279
 SEND TEXT 280
 SEND TEXT MAPPED 284
 SEND TEXT NOEDIT 286
 full-function BMS
 PURGE MESSAGE 180
 map definition macro 381, 391
 mapping input data 220
 mapping input data with MAPPINGDEV 223
 mapset definition macro 381, 397
 minimum BMS
 RECEIVE MAP 220
 RECEIVE MAP MAPPINGDEV command 223
 SEND CONTROL 264
 SEND MAP 268
 SEND MAP MAPPINGDEV command 274
 partition definition macro 381, 403
 partition set definition macro 381, 404
 related constants 375
 routing a logical message 239
 sending previously mapped data 284
 sending user-defined data stream 286
 standard BMS
 RECEIVE MAP 220
 RECEIVE PARTN 225
 SEND CONTROL 264
 SEND MAP 268
 SEND PARTNSET 279
 SEND TEXT 280
 batch data interchange (BDI)
 add record to data set 133
 commands 9
 delete a record from data set 146
 exceptional conditions 151
 read record from data set 157
 request next record number 151
 send data to output device 162
 terminate data set 131, 141
 update a record in data set 159
 wait for function completion 166
 batch logical unit, 3770 48, 206, 249

- batch mode application, 3740 60, 216, 260
- BELOW option
 - GETMAIN 121
- BIF DEEDIT command 30
- BLANK value
 - DFHMDF 386
- BLINK value
 - DFHMDF 386
 - DFHMDI 393
 - DFHMSD 399
- BLOCK value
 - DFHMDI 392
 - DFHMSD 398
- BMS
 - See basic mapping support
- BOTTOM value
 - DFHMDI 393
- browse operation
 - ending 77
 - read next record 189
 - read previous record 193
 - reset starting point 229
 - starting 309
- BRT value
 - DFHMDF 384
- BTAM programmable terminals 358
- BTRANS option
 - ASSIGN 24
- BUFFER option
 - GDS RECEIVE 115
 - RECEIVE (non-VTAM) 218
 - RECEIVE (VTAM) 208
- BUFSZE operand
 - DFHPDI macro 403
- BUILD ATTACH (LUTYPE6.1) command 31
- BUILD ATTACH (MRO) command 33
- built-in functions
 - commands 9

C

- C language
 - ADDRESS COMMAREA 13
 - ADDRESS EIB 13
 - argument values 2
 - LENGTH option default 3
 - translated code 353
- CANCEL command 35
- CANCEL option
 - ABEND 12
 - HANDLE ABEND 123
- CARD option
 - ISSUE ABORT 131
 - ISSUE END 141
 - ISSUE SEND 162
 - ISSUE WAIT 166

- CASE operand
 - DFHMDF 385
- CBIDERR condition
 - ALLOCATE (APPC) 16
 - ALLOCATE (LUTYPE6.1) 18
 - CONVERSE (non-VTAM) 63
 - CONVERSE (VTAM) 51
 - EXTRACT ATTACH (LUTYPE6.1) 83
 - EXTRACT ATTACH (MRO) 85
 - SEND (non-VTAM) 262
 - SEND (VTAM) 253
- CBUFF option
 - SEND (non-VTAM) 261
- CHANGE PASSWORD command 37
- CHANGE TASK command 38
- CHANGETIME option
 - VERIFY PASSWORD 318
- CHARSZ operand
 - DFHPDI 403
 - DFHPSD 404
- CICSDATAKEY option
 - GETMAIN 121
- CLASS option
 - SPOOL OPEN 296
 - SPOOL OPEN INPUT 294
- CLEAR option
 - HANDLE AID 125
- CLRPARTN option
 - HANDLE AID 125
- CMDSEC option
 - ASSIGN 24
- CNOTCOMPL option
 - SEND (non-VTAM) 261
 - SEND (VTAM) 252
- COBOL
 - argument values 2
 - translated code 353
- CODEREG operand 355
- COLOR operand
 - DFHMDF 385
 - DFHMDI 391
 - DFHMSD 397
- COLOR option
 - ASSIGN 24
- COLUMN operand
 - DFHMDI 391
- command language translator
 - translated code 353
- commands
 - format, arguments 1
 - security 10
 - spool 10
 - temporary storage control 10
- COMMAREA option
 - ADDRESS 13
 - LINK 169

COMMAREA option (*continued*)
 RETURN 235
 XCTL 341
 common buffer, output to, 2980 214
 common programming interface communications (CPI Communications) 365
 COMPLETE option
 DUMP TRANSACTION 74
 CONFIRM option
 GDS SEND 117
 SEND (VTAM) 252
 CONNECT PROCESS command 39
 CONSOLE option
 ISSUE ABORT 131
 ISSUE END 141
 ISSUE SEND 162
 ISSUE WAIT 166
 console support commands 9
 constants
 AID values, DFHAID 379
 attribute values, DFHBMSCA 375
 for 3270 attributes 375
 for examining EIBAID field 379
 for MSR control values 377
 for printer format controls 375
 MSR control, DFHtex read 377
 printer control values, DFHBMSCA 375
 CONTROL option
 QUERY SECURITY 182
 CONVDATA option
 GDS CONNECT PROCESS 105
 GDS EXTRACT ATTRIBUTES 107
 GDS FREE 109
 GDS ISSUE ABEND 110
 GDS ISSUE CONFIRMATION 111
 GDS ISSUE ERROR 112
 GDS ISSUE PREPARE 113
 GDS ISSUE SIGNAL 114
 GDS RECEIVE 115
 GDS SEND 117
 GDS WAIT 119
 CONVERSE (2260) command 55
 CONVERSE (2741) command 56
 CONVERSE (2770) command 57
 CONVERSE (2780) command 57
 CONVERSE (3270 display) command 58
 CONVERSE (3270 logical) command 44
 CONVERSE (3600 BTAM) command 58
 CONVERSE (3600-3601) command 44
 CONVERSE (3600-3614) command 45
 CONVERSE (3650 interpreter) command 45
 CONVERSE (3650-3270) command 46
 CONVERSE (3650-3653) command 46
 CONVERSE (3650-3680) command 47
 CONVERSE (3735) command 59
 CONVERSE (3740) command 60
 CONVERSE (3767) command 47
 CONVERSE (3770) command 48
 CONVERSE (3790 3270-display) command 49
 CONVERSE (3790 full-function or inquiry) command 48
 CONVERSE (APPC) command 41
 CONVERSE (LUTYPE2/LUTYPE3) command 42
 CONVERSE (LUTYPE4) command 42
 CONVERSE (LUTYPE6.1) command 43
 CONVERSE (MRO) command 53
 CONVERSE (non-VTAM default) command 53
 CONVERSE (SCS) command 43
 CONVERSE (System/3) command 54
 CONVERSE (System/7) command 54
 CONVERSE (VTAM default) command 41
 CONVERSE option
 ISSUE LOAD 150
 converse with terminal or LU 358
 CONVID option
 CONNECT PROCESS 39
 CONVERSE (non-VTAM) 61
 CONVERSE (VTAM) 50
 EXTRACT ATTACH (LUTYPE6.1) 82
 EXTRACT ATTACH (MRO) 84
 EXTRACT ATTRIBUTES (APPC) 86
 EXTRACT ATTRIBUTES (MRO) 87
 EXTRACT PROCESS 89
 FREE (APPC) 96
 FREE (LUTYPE6.1) 97
 FREE (MRO) 98
 GDS ALLOCATE 102
 GDS CONNECT PROCESS 105
 GDS EXTRACT ATTRIBUTES 107
 GDS EXTRACT PROCESS 108
 GDS FREE 109
 GDS ISSUE ABEND 110
 GDS ISSUE CONFIRMATION 111
 GDS ISSUE ERROR 112
 GDS ISSUE PREPARE 113
 GDS ISSUE SIGNAL 114
 GDS RECEIVE 115
 GDS SEND 117
 GDS WAIT 119
 ISSUE ABEND 130
 ISSUE CONFIRMATION 135
 ISSUE ERROR 149
 ISSUE PREPARE 154
 ISSUE SIGNAL (APPC) 164
 ISSUE SIGNAL (LUTYPE6.1) 165
 POINT 176
 RECEIVE (VTAM) 208
 SEND (VTAM) 252
 WAIT CONVID 320
 WAIT TERMINAL 326
 copy displayed information 361

- copybooks
 - DFHAID 379
 - DFHBMSCA 375
 - DFHEIBLK 355
 - DFHMSRCA 377
- CPI Communications (SAA) 365
- create a journal record 332
- CSMT transaction
 - resend message destination 59
- CSTL transaction
 - resend message destination 59
- CTLCHAR option
 - CONVERSE (non-VTAM) 61
 - CONVERSE (VTAM) 50
 - ISSUE COPY (3270 display) 137
 - ISSUE COPY (3270 logical) 138
 - SEND (non-VTAM) 261
 - SEND (VTAM) 252
- CTRL operand
 - DFHMDI 391
 - DFHMSD 397
- CURRENT option
 - SEND PAGE 276
- CURSLOC operand
 - DFHMDI 392
 - DFHMSD 398
- cursor address 361
- CURSLOC option
 - RECEIVE MAP MAPPINGDEV command 223
 - SEND CONTROL 265
 - SEND MAP 269
 - SEND MAP MAPPINGDEV command 274
 - SEND TEXT 281
- cursor position
 - terminal control 361
- CVDA (CICS-value data area)
 - argument values 2
 - command format 2
 - passing and receiving 5
 - values in alphabetic order 371
 - values in numeric order 371
- CVDA options
 - ACTION
 - WRITE OPERATOR 334
 - ALTER
 - QUERY SECURITY 182
 - ASRAKEY
 - ASSIGN 23
 - ASRASPC
 - ASSIGN 24
 - CONTROL
 - QUERY SECURITY 182
 - LOGMESSAGE
 - QUERY SECURITY 182
 - MAXLIFETIME
 - DEQ 72
 - ENQ 79

- CVDA options (*continued*)
 - PURGEABILITY
 - WAIT EXTERNAL 323
 - WAITCICS 328
 - READ
 - QUERY SECURITY 182
 - STATE 51, 63, 209, 219, 253, 262
 - ALLOCATE (APPC) 16
 - ALLOCATE (MRO) 20
 - CONNECT PROCESS 39
 - EXTRACT ATTRIBUTES (APPC) 86
 - EXTRACT ATTRIBUTES (MRO) 87
 - FREE (APPC) 96
 - FREE (MRO) 98
 - GDS ALLOCATE 102
 - GDS CONNECT PROCESS 105
 - GDS EXTRACT ATTRIBUTES 107
 - GDS FREE 109
 - GDS ISSUE ABEND 110
 - GDS ISSUE CONFIRMATION 111
 - GDS ISSUE ERROR 112
 - GDS ISSUE PREPARE 113
 - GDS ISSUE SIGNAL 114
 - GDS RECEIVE 116
 - GDS SEND 118
 - GDS WAIT 119
 - ISSUE ABEND 130
 - ISSUE CONFIRMATION 135
 - ISSUE ERROR 149
 - ISSUE PREPARE 154
 - ISSUE SIGNAL (APPC) 164
 - WAIT CONVID 320
 - UPDATE
 - QUERY SECURITY 183
- CVDA values
 - ALLOCATED
 - ALLOCATE (APPC) 16
 - ALLOCATE (MRO) 20
 - CONNECT PROCESS 39
 - EXTRACT ATTRIBUTES (APPC) 86
 - EXTRACT ATTRIBUTES (MRO) 87
 - FREE (APPC) 96
 - FREE (MRO) 98
 - GDS ALLOCATE 102
 - GDS CONNECT PROCESS 105
 - GDS EXTRACT ATTRIBUTES 107
 - GDS FREE 109
 - GDS ISSUE ABEND 110
 - GDS ISSUE CONFIRMATION 111
 - GDS ISSUE ERROR 112
 - GDS ISSUE PREPARE 113
 - GDS ISSUE SIGNAL 114
 - GDS RECEIVE 116
 - GDS SEND 118
 - GDS WAIT 119
 - ISSUE ABEND 130
 - ISSUE CONFIRMATION 135

CVDA values (continued)

ALLOCATED (continued)

ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
RECEIVE (MRO) 219
SEND (APPC) 253
SEND (MRO) 262
WAIT CONVID 320

ALTERABLE

QUERY SECURITY 182

BASESPACE

ASSIGN 24

CICSEXECKEY

ASSIGN 23

CONFFREE

CONNECT PROCESS 39
EXTRACT ATTRIBUTES (APPC) 86
FREE (APPC) 96
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
SEND (APPC) 253
WAIT CONVID 320

CONFRECEIVE

CONNECT PROCESS 39
EXTRACT ATTRIBUTES (APPC) 86
FREE (APPC) 96
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130

CVDA values (continued)

CONFRECEIVE (continued)

ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
SEND (APPC) 253
WAIT CONVID 320

CONFSEND

CONNECT PROCESS 39
EXTRACT ATTRIBUTES (APPC) 86
FREE (APPC) 96
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
SEND (APPC) 253
WAIT CONVID 320

CRITICAL

WRITE OPERATOR 334

CTRLABLE

QUERY SECURITY 182

EVENTUAL

WRITE OPERATOR 334

FREE

CONNECT PROCESS 39
EXTRACT ATTRIBUTES (APPC) 86
EXTRACT ATTRIBUTES (MRO) 87
FREE (APPC) 96
FREE (MRO) 98
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119

CVDA values (continued)

FREE (continued)

ISSUE ABEND 130
 ISSUE CONFIRMATION 135
 ISSUE ERROR 149
 ISSUE PREPARE 154
 ISSUE SIGNAL (APPC) 164
 RECEIVE (APPC) 209
 RECEIVE (MRO) 219
 SEND (APPC) 253
 SEND (MRO) 262
 WAIT CONVID 320

IMMEDIATE

WRITE OPERATOR 334

LOG

QUERY SECURITY 182

LUW

DEQ 72

ENQ 79

NOLOG

QUERY SECURITY 182

NONCICS

ASSIGN 23

NOTALTERABLE

QUERY SECURITY 182

NOTAPPLIC

ASSIGN 23, 24

NOTCTRLABLE

QUERY SECURITY 182

NOTPURGEABLE

WAIT EXTERNAL 323

WAITCICS 328

NOTREADABLE

QUERY SECURITY 182

NOTUPDATABLE

QUERY SECURITY 183

PENDFREE

CONNECT PROCESS 39
 EXTRACT ATTRIBUTES (APPC) 86
 EXTRACT ATTRIBUTES (MRO) 87
 FREE (APPC) 96
 FREE (MRO) 98
 GDS ALLOCATE 102
 GDS CONNECT PROCESS 105
 GDS EXTRACT ATTRIBUTES 107
 GDS FREE 109
 GDS ISSUE ABEND 110
 GDS ISSUE CONFIRMATION 111
 GDS ISSUE ERROR 112
 GDS ISSUE PREPARE 113
 GDS ISSUE SIGNAL 114
 GDS RECEIVE 116
 GDS SEND 118
 GDS WAIT 119
 ISSUE ABEND 130
 ISSUE CONFIRMATION 135
 ISSUE ERROR 149

CVDA values (continued)

PENDFREE (continued)

ISSUE PREPARE 154
 ISSUE SIGNAL (APPC) 164
 RECEIVE (APPC) 209
 RECEIVE (MRO) 219
 SEND (APPC) 253
 SEND (MRO) 262
 WAIT CONVID 320

PENDRECEIVE

CONNECT PROCESS 39
 EXTRACT ATTRIBUTES (APPC) 86
 FREE (APPC) 96
 GDS ALLOCATE 102
 GDS CONNECT PROCESS 105
 GDS EXTRACT ATTRIBUTES 107
 GDS FREE 109
 GDS ISSUE ABEND 110
 GDS ISSUE CONFIRMATION 111
 GDS ISSUE ERROR 112
 GDS ISSUE PREPARE 113
 GDS ISSUE SIGNAL 114
 GDS RECEIVE 116
 GDS SEND 118
 GDS WAIT 119
 ISSUE ABEND 130
 ISSUE CONFIRMATION 135
 ISSUE ERROR 149
 ISSUE PREPARE 154
 ISSUE SIGNAL (APPC) 164
 RECEIVE (APPC) 209
 SEND (APPC) 253
 WAIT CONVID 320

PURGEABLE

WAIT EXTERNAL 323
 WAITCICS 328

READABLE

QUERY SECURITY 182

RECEIVE

CONNECT PROCESS 39
 CONVERSE (MRO) 63
 EXTRACT ATTRIBUTES (APPC) 86
 EXTRACT ATTRIBUTES (MRO) 87
 FREE (APPC) 96
 FREE (MRO) 98
 GDS ALLOCATE 102
 GDS CONNECT PROCESS 105
 GDS EXTRACT ATTRIBUTES 107
 GDS FREE 109
 GDS ISSUE ABEND 110
 GDS ISSUE CONFIRMATION 111
 GDS ISSUE ERROR 112
 GDS ISSUE PREPARE 113
 GDS ISSUE SIGNAL 114
 GDS RECEIVE 116
 GDS SEND 118
 GDS WAIT 119

CVDA values (continued)

RECEIVE (continued)

ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
RECEIVE (MRO) 219
SEND (APPC) 253
SEND (MRO) 262
WAIT CONVID 320

ROLLBACK

CONNECT PROCESS 39
CONVERSE (MRO) 63
EXTRACT ATTRIBUTES (APPC) 86
EXTRACT ATTRIBUTES (MRO) 87
FREE (APPC) 96
FREE (MRO) 98
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
RECEIVE (MRO) 219
SEND (APPC) 253
SEND (MRO) 262
WAIT CONVID 320

SEND

CONNECT PROCESS 39
CONVERSE (MRO) 63
EXTRACT ATTRIBUTES (APPC) 86
EXTRACT ATTRIBUTES (MRO) 87
FREE (APPC) 96
FREE (MRO) 98
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114

CVDA values (continued)

SEND (continued)

GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
RECEIVE (MRO) 219
SEND (APPC) 253
SEND (MRO) 262
WAIT CONVID 320

SUBSPACE

ASSIGN 24

SYNCFREE

CONNECT PROCESS 39
CONVERSE (MRO) 63
EXTRACT ATTRIBUTES (APPC) 86
EXTRACT ATTRIBUTES (MRO) 87
FREE (APPC) 96
FREE (MRO) 98
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
RECEIVE (MRO) 219
SEND (APPC) 253
SEND (MRO) 262
WAIT CONVID 320

SYNCRECEIVE

CONNECT PROCESS 39
CONVERSE (MRO) 63
EXTRACT ATTRIBUTES (APPC) 86
EXTRACT ATTRIBUTES (MRO) 87
FREE (APPC) 96
FREE (MRO) 98
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109

CVDA values (continued)

SYNCRECEIVE (continued)

GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
RECEIVE (MRO) 219
SEND (APPC) 253
SEND (MRO) 262
WAIT CONVID 320

SYNCSEND

CONNECT PROCESS 39
CONVERSE (MRO) 63
EXTRACT ATTRIBUTES (APPC) 86
EXTRACT ATTRIBUTES (MRO) 87
FREE (APPC) 96
FREE (MRO) 98
GDS ALLOCATE 102
GDS CONNECT PROCESS 105
GDS EXTRACT ATTRIBUTES 107
GDS FREE 109
GDS ISSUE ABEND 110
GDS ISSUE CONFIRMATION 111
GDS ISSUE ERROR 112
GDS ISSUE PREPARE 113
GDS ISSUE SIGNAL 114
GDS RECEIVE 116
GDS SEND 118
GDS WAIT 119
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE ERROR 149
ISSUE PREPARE 154
ISSUE SIGNAL (APPC) 164
RECEIVE (APPC) 209
RECEIVE (MRO) 219
SEND (APPC) 253
SEND (MRO) 262
WAIT CONVID 320

TASK

DEQ 72
ENQ 79

UPDATABLE

QUERY SECURITY 183

USEREXECKEY

ASSIGN 23

CWA option

ADDRESS 13

CWALENG option

ASSIGN 24

D

Data Facility Product (DFP)

duct xi

DATA operand

DFHMDI 392

DFHMSD 398

DATA option

FREEMAIN 101

data sets

add records to 133

interrogating 156

processing termination 141

read records from 157

update records 159

data tables

CICS/user-maintained

DELETE 67

ENDBR 77

READ 185

READNEXT 189

READPREV 193

RESETBR 229

REWRITE 237

STARTBR 309

UNLOCK 316

WRITE 329

data to output device, sending 162

data-area argument

CICS command format 2

definition 1

data-value argument

CICS command format 2

definition 1

data, deleting

file control records 67

temporary storage queues 71

transient data queues 70

data, passing to new tasks 305

DATA1 option

MONITOR 175

DATA2 option

MONITOR 175

DATALength option

LINK 169

DATAONLY option

SEND MAP 269

SEND MAP MAPPINGDEV command 274

DATAPointer option

FREEMAIN 101

DATAREG operand 355
 DATASTR option
 BUILD ATTACH (LUTYPE6.1) 31
 BUILD ATTACH (MRO) 33
 EXTRACT ATTACH (LUTYPE6.1) 82
 EXTRACT ATTACH (MRO) 84
 DATE option
 FORMATTIME 92
 DATEFORM option
 FORMATTIME 92
 DATESEP option
 FORMATTIME 93
 DAYCOUNT option
 FORMATTIME 93
 DAYOFMONTH option
 FORMATTIME 93
 DAYOFWEEK option
 FORMATTIME 93
 DAYSLEFT option
 VERIFY PASSWORD 318
 DCT option
 DUMP TRANSACTION 74
 DDMMYY option
 FORMATTIME 93
 DDMMYYYY option
 FORMATTIME 93
 DEBKEY option
 READ 185
 STARTBR 309
 DEBREC option
 READ 185
 STARTBR 309
 DEFAULT option
 CONVERSE (non-VTAM) 61
 CONVERSE (VTAM) 50
 SEND (non-VTAM) 261
 SEND (VTAM) 252
 SEND CONTROL 265
 SEND MAP 269
 SEND TEXT 281
 SEND TEXT NOEDIT 286
 DEFRESP option
 CONVERSE (non-VTAM) 61
 CONVERSE (VTAM) 50
 ISSUE ADD 133
 ISSUE ERASE 146
 ISSUE REPLACE 159
 ISSUE SEND 162
 SEND (non-VTAM) 261
 SEND (VTAM) 252
 DEFSCRNHT option
 ASSIGN 24
 DEFSCRNWD option
 ASSIGN 24
 DELAY command 65
 delay processing, task 65
 DELETE command 67
 delete loaded program 227
 DELETE option
 SPOOLCLOSE 293
 delete records
 batch data interchange records 146
 DELETEQ TD command 70
 DELETEQ TS command 71
 deleting data
 temporary storage queues 71
 transient data queues 70
 DELIMITER option
 ASSIGN 24
 DEQ command 72
 dequeue from resource 72
 DEST option
 CONVERSE (non-VTAM) 61
 CONVERSE (VTAM) 50
 SEND (non-VTAM) 261
 SEND (VTAM) 252
 DESTCOUNT option
 ASSIGN 24
 DESTID option
 ASSIGN 24
 ISSUE ABORT 131
 ISSUE ADD 133
 ISSUE END 141
 ISSUE ERASE 146
 ISSUE NOTE 151
 ISSUE QUERY 156
 ISSUE REPLACE 159
 ISSUE SEND 162
 ISSUE WAIT 166
 DESTIDLENG option
 ASSIGN 25
 ISSUE ABORT 131
 ISSUE ADD 133
 ISSUE END 141
 ISSUE ERASE 146
 ISSUE NOTE 151
 ISSUE QUERY 156
 ISSUE REPLACE 159
 ISSUE SEND 162
 ISSUE WAIT 166
 DET value
 DFHMDF 384
 DFH2980 structure 214
 DFHAID attention identifier list 379
 DFHBMSCA, standard attribute and printer control character
 list, BMS 375
 DFHEAI interface processor 354
 DFHECALL macro 353
 DFHEIBLK copybook 355
 DFHEICAL macro, use DFHECALL 353

- DFHEIEND macro 353
- DFHEIENT macro
 - CODEREG 355
 - DATAREG 355
 - defaults 355
 - description 353
 - EIBREG 355
- DFHEIGBL macro 353
- DFHEIPLR symbolic register 356
- DFHEIRET macro 354
- DFHEISTG macro 355
- DFHMDF macro 383
- DFHMDI macro 390
- DFHMIRS 169
- DFHMSD macro 396
- DFHMSRCA, MSR control value constants 377
- DFHPDI macro 403
- DFHPSD macro 404
- DFHRESP, built-in function 5
- DFHVALUE, translator routine 5
- DFP, Data Facility Product xi
- diagnostic services commands 9
- DISABLED condition
 - DELETE 68
 - DELETEQ TD 70
 - READ 187
 - READQ TD 197
 - STARTBR 310
 - UNLOCK 316
 - WRITE 330
 - WRITEQ TD 336
- disconnect a switched line 358
- display-device operations
 - attention identifier (AID) 361
 - attention identifier list, DFHAID 379
 - copy displayed information 361
 - cursor address 361
 - erase all unprotected fields 361
 - input operation without data 361
 - pass control on receipt of an AID 125, 129
 - print displayed information 360
 - standard attribute and printer control character list, DFHBMSCA 375
 - terminal 360
- distributed program link (DPL)
 - ogram link 367
- DPL, distributed program link 367
- DRK value
 - DFHMDF 384
- DS3270 option
 - ASSIGN 25
- DSATTS operand
 - DFHMDI 393
 - DFHMSD 398
- DSECT value
 - DFHMSD 401

- DSSCS option
 - ASSIGN 25
- DSSTAT condition
 - ISSUE RECEIVE 157
- DUMP TRANSACTION command 74
- DUMPCODE option
 - DUMP TRANSACTION 74
- DUMPID option
 - DUMP TRANSACTION 74
- DUPKEY condition
 - DELETE 68
 - READ 187
 - READNEXT 191
 - READPREV 194
- DUPREC condition
 - REWRITE 237
 - WRITE 330
- dynamic allocation 296
- dynamic storage, extensions 355

E

- ECADDR option
 - WAIT EVENT 321
- ECBLIST option
 - WAIT EXTERNAL 323
 - WAITCICS 328
- EDF, execution diagnostic facility 308
- EIB fields
 - EIBAID 343
 - EIBATT 343
 - EIBCALEN 343
 - EIBCOMPL 343
 - EIBCONF 343
 - EIBCPOSN 343
 - EIBDATE 343
 - EIBDS 343
 - EIBEOC 343
 - EIBERR 344
 - EIBERRCD 344
 - EIBFMH 344
 - EIBFN 344
 - EIBFREE 345
 - EIBNODAT 345
 - EIBRCODE 345
 - EIBRECV 348
 - EIBREQID 348
 - EIBRESP 349
 - EIBRESP2 349
 - EIBRLDBK 349
 - EIBRSRCE 349
 - EIBSIG 349
 - EIBSYNC 350
 - EIBSYNRB 350
 - EIBTASKN 350
 - EIBTIME 350

EIB fields (*continued*)
 EIBTRMID 350
 EIBTRNID 350
 EIB option
 ADDRESS 13
 EIBAID 361
 examining contents of field 379
 EIBREG operand 355
 ENDBR command 77
 ENDDATA condition
 RETRIEVE 234
 ENDFILE condition
 READNEXT 191
 READPREV 194
 ENDFILE option
 ISSUE ENDOUTPUT 144
 ENDINPT condition
 RECEIVE (non-VTAM) 219
 ENDOUTPUT option
 ISSUE ENDFILE 143
 English and katakana characters, mixed 61, 220, 225
 ENQ command 78
 ENQBUSY condition
 ENQ 79
 ENTER option
 HANDLE AID 125
 ENTER TRACEID
 monitoring aspects replaced by MONITOR 174
 tracing aspects replaced by ENTER TRACENUM 80
 ENTER TRACENUM command 80
 ENTRY option
 LOAD 172
 entry to assembler-language program 353
 ENTRYNAME option
 MONITOR 175
 ENVDEFERR condition
 RETRIEVE 234
 environment services
 commands 9
 EOC condition
 ALLOCATE (LUTYPE6.1) 18
 CONVERSE (non-VTAM) 64
 CONVERSE (VTAM) 52
 ISSUE RECEIVE 157
 RECEIVE (non-VTAM) 219
 RECEIVE (VTAM) 209
 RECEIVE MAP 221
 RECEIVE PARTN 226
 EODS condition
 CONVERSE (VTAM) 52
 ISSUE RECEIVE 158
 RECEIVE (VTAM) 209
 RECEIVE MAP 221
 RECEIVE PARTN 226
 EOF condition
 CONVERSE (non-VTAM) 64
 EOF condition (*continued*)
 RECEIVE (non-VTAM) 219
 EQUAL option
 READ 185
 RESETBR 229
 STARTBR 309
 equated symbols 4
 erase all unprotected fields 361
 ERASE option
 CONVERSE (non-VTAM) 61
 CONVERSE (VTAM) 50
 SEND (non-VTAM) 261
 SEND (VTAM) 252
 SEND CONTROL 265
 SEND MAP 269
 SEND MAP MAPPINGDEV command 274
 SEND TEXT 281
 SEND TEXT NOEDIT 286
 ERASEAUP option
 SEND CONTROL 265
 SEND MAP 269
 SEND MAP MAPPINGDEV command 274
 ERRTERM option
 ROUTE 239
 ESDS (entry-sequenced data set)
 DELETE 68
 READ 186
 STARTBR 310
 WRITE 329
 ESM
 ACEE pointer 13
 QUERY SECURITY, NOTFND 184
 QUERY SECURITY, RESCLASS 182
 USERNAME 29
 ESM, external security manager 308
 ESMREASON option
 CHANGE PASSWORD 37
 SIGNON 290
 VERIFY PASSWORD 318
 ESMRESP option
 CHANGE PASSWORD 37
 SIGNON 290
 VERIFY PASSWORD 318
 events, timer
 control area, timer 178
 monitoring point 174
 waiting for 321
 EWASUPP option
 ASSIGN 25
 EXCEPTION option
 ENTER TRACENUM 80
 exception support commands 9
 exclusive control release, UNLOCK command 316
 EXEC CICS command format 1
 EXEC CICS DLI interface
 No longer documented xi

- execution diagnostic facility (EDF) 308
- exit from ASM program 354
- exit, abnormal termination recovery 123
- expiration time, notification when reached 178
- EXPIRED condition
 - DELAY 66
 - POST 179
 - WRITE OPERATOR 335
- EXPIRYTIME option
 - VERIFY PASSWORD 318
- EXTATT operand
 - DFHMDI 393
 - DFHMSD 398
- EXTDS option
 - ASSIGN 25
- external security manager (ESM) 182, 308
- EXTRACT ATTACH (LUTYPE6.1) command 82
- EXTRACT ATTACH (MRO) command 84
- EXTRACT ATTRIBUTES (APPC) command 86
- EXTRACT ATTRIBUTES (MRO) command 87
- EXTRACT LOGONMSG command 88
- EXTRACT PROCESS command 89
- EXTRACT TCT command 91

F

- FACILITY option
 - ASSIGN 25
- FCI option
 - ASSIGN 25, 351
- FCT option
 - DUMP TRANSACTION 74
- field definition macro, BMS 381
- field edit built-in function
 - See BIF DEEDIT
- FIELD option
 - BIF DEEDIT 30
- field separator operand 393, 398
- FIELD value
 - DFHMDI 392
 - DFHMSD 398
- FIELDS operand
 - DFHMDI 393
- file control
 - commands 10
 - deleting VSAM records 67
 - end browse operation 77
 - read next record 189
 - read previous record 193
 - release exclusive control 316
 - specify start for browse 309
 - update a record 237
 - writing new record 329
- FILE option
 - DELETE 67
 - ENDBR 77

- FILE option (*continued*)
 - READ 185
 - READNEXT 189
 - READPREV 193
 - RESETBR 229
 - REWRITE 237
 - STARTBR 309
 - UNLOCK 316
 - WRITE 329
- filename
 - definition 2, 3, 4
- filename argument, CICS command format 2
- FILENOTFOUND condition
 - DELETE 68
 - ENDBR 77
 - READ 187
 - READNEXT 191
 - READPREV 194
 - RESETBR 230
 - REWRITE 237
 - STARTBR 310
 - UNLOCK 316
 - WRITE 330
- FIRST value
 - DFHMDI 393
- FLDSEP operand
 - DFHMDI 393
 - DFHMSD 398
- FLENGTH option
 - DUMP TRANSACTION 74
 - fullword alternative to LENGTH 357
 - GDS RECEIVE 115
 - GDS SEND 117
 - GETMAIN 121
 - LOAD 172
 - RECEIVE (non-VTAM) 218
 - RECEIVE (VTAM) 208
 - SEND (non-VTAM) 261
 - SEND (VTAM) 252
 - SPOOLWRITE 301
- FMH option
 - CONVERSE (non-VTAM) 62
 - CONVERSE (VTAM) 50
 - SEND (non-VTAM) 261
 - SEND (VTAM) 252
 - START 306
- FMHPARM option
 - SEND MAP 269
 - SEND PAGE 276
 - SEND TEXT 281
- FOLD operand
 - DFHMSD 399
- FOR option
 - DELAY 65
- FORMATTIME command 92

FORMFEED option
 SEND CONTROL 265
 SEND MAP 269
 SEND MAP MAPPINGDEV command 275
 SEND TEXT 281
 FREE (APPC) command 96
 FREE (LUTYPE6.1) command 97
 FREE (MRO) command 98
 FREE command 95
 free main storage 99
 FREEKB option
 SEND CONTROL 265
 SEND MAP 269
 SEND MAP MAPPINGDEV command 275
 SEND TEXT 281
 SEND TEXT NOEDIT 286
 FREEKB value
 DFHMDI 392
 DFHMSD 398
 FREEMAIN command 99
 FROM option
 CONVERSE (non-VTAM) 62
 CONVERSE (VTAM) 50
 DUMP TRANSACTION 74
 ENTER TRACENUM 80
 GDS SEND 117
 ISSUE ADD 133
 ISSUE PASS 152
 ISSUE REPLACE 159
 ISSUE SEND 162
 RECEIVE MAP 220
 RECEIVE MAP MAPPINGDEV command 223
 REWRITE 237
 SEND (non-VTAM) 261
 SEND (VTAM) 252
 SEND MAP 269
 SEND MAP MAPPINGDEV command 275
 SEND TEXT 281
 SEND TEXT MAPPED 284
 SEND TEXT NOEDIT 286
 SPOOLWRITE 301
 START 306
 WRITE 329
 WRITE JOURNALNUM 332
 WRITEQ TD 336
 WRITEQ TS 338
 FROMLENGTH option
 CONVERSE (non-VTAM) 62
 CONVERSE (VTAM) 50
 fullword alternative to FROMLENGTH 357
 FROMLENGTH option
 CONVERSE (non-VTAM) 62
 CONVERSE (VTAM) 50
 ENTER TRACENUM 80
 fullword length alternative (FROMLENGTH) 357

FRSET option
 SEND CONTROL 265
 SEND MAP 269
 SEND MAP MAPPINGDEV command 275
 FRSET value
 DFHMDI 392
 DFHMSD 398
 FSET value
 DFHMDF 384
 Full Function Logical Unit, 3790 48, 207, 250
 FULLDATE option
 FORMATIME 93
 fullword length option 357
 FUNCERR condition
 ISSUE ABORT 131
 ISSUE ADD 133
 ISSUE END 141
 ISSUE ERASE 147
 ISSUE NOTE 151
 ISSUE QUERY 156
 ISSUE REPLACE 160
 ISSUE SEND 163
 ISSUE WAIT 166
 function management header
 See FMH

G

GCHARS option
 ASSIGN 25
 GCODES option
 ASSIGN 25
 GDS (generalized data stream) 9
 GDS ALLOCATE command 102
 GDS ASSIGN command 104
 GDS CONNECT PROCESS command 105
 GDS EXTRACT ATTRIBUTES command 107
 GDS EXTRACT PROCESS command 108
 GDS FREE command 109
 GDS ISSUE ABEND command 110
 GDS ISSUE CONFIRMATION command 111
 GDS ISSUE ERROR command 112
 GDS ISSUE PREPARE command 113
 GDS ISSUE SIGNAL command 114
 GDS RECEIVE command 115
 GDS SEND command 117
 GDS WAIT command 119
 General Banking Terminal System (2980 General Banking Terminal System) 213
 generalized data stream (GDS) 9
 generic applid, XRF 23
 GENERIC option
 DELETE 67
 READ 186
 RESETBR 229
 STARTBR 309

get main storage 120
GETMAIN command 120
GINIT operand
 DFHMDF 385
GMMI option
 ASSIGN 25
GROUPID option
 SIGNON 290
GRPNAME operand
 DFHMDF 385
GTEQ option
 READ 186
 RESETBR 229
 STARTBR 309

H

HANDLE ABEND command 123
HANDLE AID command 125
HANDLE CONDITION command 127
HEADER operand
 DFHMDI 393
HEADER option
 SEND TEXT 281
hmmss argument, CICS command format 2
HIGHLIGHT operand
 DFHMDF 386
 DFHMDI 393
 DFHMSD 399
HIGHLIGHT option
 ASSIGN 25
HOLD option
 LOAD 172
HONEYMOM option
 SEND CONTROL 265
 SEND MAP 269
 SEND TEXT 281
 SEND TEXT NOEDIT 286
host command processor LU, 3650/3680 248
host conversational LU 3650
 (3270) 46, 247
 (3653) 46, 248
HOURS option
 DELAY 65
 POST 179
 ROUTE 239
 START 306
HTAB operand
 DFHMSD 399

I

IC value
 DFHMDF 385
IGNORE CONDITION command 129

IGREQCD condition
 CONVERSE (VTAM) 52
 ISSUE SEND 163
 SEND (VTAM) 253
 SEND CONTROL 266
 SEND MAP 271
 SEND PAGE 277
 SEND TEXT 283
 SEND TEXT MAPPED 284
 SEND TEXT NOEDIT 287
IGREQID condition
 SEND CONTROL 266
 SEND MAP 271
 SEND TEXT 283
 SEND TEXT MAPPED 285
 SEND TEXT NOEDIT 287
ILLOGIC condition
 DELETE 68
 EIBRCODE 348
 ENDBR 77
 READ 187
 READNEXT 191
 READPREV 194
 RESETBR 230
 REWRITE 237
 STARTBR 310
 UNLOCK 316
 WRITE 330
IMMEDIATE option
 RETURN 235
implicit SPOOLCLOSE 294
INBFMH condition
 CONVERSE (non-VTAM) 64
 CONVERSE (VTAM) 52
 RECEIVE (non-VTAM) 219
 RECEIVE (VTAM) 209
INITIAL operand
 DFHMDF 386
initialize main storage 120
initiate a task 305
INITIMG option
 GETMAIN 121
INITPARM option
 ASSIGN 25
INITPARMLEN option
 ASSIGN 25
INPARTN option
 ASSIGN 25
 RECEIVE MAP 220
input operation without data 361
INPUTMSG option
 LINK 169
 RETURN 235
 XCTL 341
INPUTMSGLEN option
 LINK 169

INPUTMSGLEN option (*continued*)
 RETURN 235
 XCTL 341
 interactive logical units 47, 206, 249
 interface processor DFHEAI 354
 interface to JES 6
 interpreter logical unit, 3650
 CONVERSE 45
 ISSUE EODS 145
 ISSUE LOAD 150
 RECEIVE 205
 SEND (VTAM) 247
 interrogate a data set 156
 interval control
 ASKTIME options 21
 cancel interval control command 35
 CANCEL options 35
 commands 10
 DELAY options 65
 delay processing of task 65
 FORMATTIME options 92
 notification when specified time expires 178
 request current time of day 21
 retrieve data stored for task 232
 start a task 303
 wait for event to occur 321
 INTERVAL option
 DELAY 65
 POST 179
 ROUTE 239
 START 306
 INTO option
 CONVERSE (non-VTAM) 62
 CONVERSE (VTAM) 50
 EXTRACT LOGONMSG 88
 GDS RECEIVE 115
 ISSUE RECEIVE 157
 READ 186
 READNEXT 189
 READPREV 193
 READQ TD 196
 READQ TS 198
 RECEIVE (non-VTAM) 218
 RECEIVE (VTAM) 208
 RECEIVE MAP 221
 RECEIVE MAP MAPPINGDEV command 223
 RECEIVE PARTN 225
 RETRIEVE 233
 SPOOLREAD 299
 INVALIDCOUNT option
 VERIFY PASSWORD 318
 INVERRTERM condition
 ROUTE 241
 INVITE option
 GDS SEND 117
 SEND (non-VTAM) 261
 INVITE option (*continued*)
 SEND (VTAM) 252
 INVLDC condition
 ROUTE 241
 SEND CONTROL 266
 SEND MAP 271
 SEND TEXT 283
 INVMPSTZ condition
 EIBRCODE byte 3 348
 RECEIVE MAP 221
 RECEIVE MAP MAPPINGDEV command 224
 SEND MAP 272
 SEND MAP MAPPINGDEV command 275
 INVOKINGPROG option
 ASSIGN 25
 INVPARTN condition
 RECEIVE MAP 221
 RECEIVE PARTN 226
 SEND CONTROL 266
 SEND MAP 272
 SEND TEXT 283
 SEND TEXT NOEDIT 287
 INVPARTNSET condition
 SEND PARTNSET 279
 INVREQ condition
 ADDRESS 14
 ALLOCATE (APPC) 16
 ALLOCATE (LUTYPE6.1) 18
 ALLOCATE (MRO) 20
 ASSIGN 29
 CHANGE PASSWORD 37
 CHANGE TASK 38
 CONNECT PROCESS 40
 CONVERSE (VTAM) 52
 DELAY 66
 DELETE 68
 DELETEQ TD 70
 DELETEQ TS 71
 DEQ 72
 DUMP TRANSACTION 76
 EIBRCODE bytes 1-3 348
 ENDBR 77
 ENQ 79
 ENTER TRACENUM 80
 EXTRACT ATTACH (LUTYPE6.1) 83
 EXTRACT ATTACH (MRO) 85
 EXTRACT ATTRIBUTES (APPC) 86
 EXTRACT ATTRIBUTES (MRO) 87
 EXTRACT PROCESS 89
 EXTRACT TCT 91
 FORMATTIME 94
 FREE (APPC) 96
 FREE (LUTYPE6.1) 97
 FREE (MRO) 98
 FREEMAIN 101
 HANDLE AID 126

INVREQ condition (*continued*)

ISSUE ABEND 130
 ISSUE ABORT 132
 ISSUE ADD 134
 ISSUE CONFIRMATION 135
 ISSUE END 142
 ISSUE ENDFILE 143
 ISSUE ENDOUTPUT 144
 ISSUE EODS 145
 ISSUE ERASE 147
 ISSUE ERASEAUP 148
 ISSUE ERROR 149
 ISSUE NOTE 151
 ISSUE PASS 153
 ISSUE PREPARE 154
 ISSUE PRINT 155
 ISSUE QUERY 156
 ISSUE RECEIVE 158
 ISSUE REPLACE 160
 ISSUE RESET 161
 ISSUE SEND 163
 ISSUE SIGNAL (APPC) 164
 ISSUE WAIT 167
 LINK 170
 LOAD 173
 MONITOR 175
 POP HANDLE 177
 POST 179
 PURGE MESSAGE 180
 QUERY SECURITY 183
 READ 187
 READNEXT 191
 READPREV 195
 READQ TD 197
 READQ TS 199
 RECEIVE (non-VTAM) 219
 RECEIVE (VTAM) 209
 RECEIVE MAP 221
 RECEIVE MAP MAPPINGDEV command 224
 RECEIVE PARTN 226
 RELEASE 227
 RESETBR 230
 RETRIEVE 234
 RETURN 236
 REWRITE 238
 ROUTE 241
 SEND (non-VTAM) 262
 SEND (VTAM) 253
 SEND CONTROL 266
 SEND MAP 272
 SEND MAP MAPPINGDEV command 275
 SEND PAGE 277
 SEND PARTNSET 279
 SEND TEXT 283
 SEND TEXT MAPPED 285
 SEND TEXT NOEDIT 287

INVREQ condition (*continued*)

SIGNOFF 289
 SIGNON 291
 START 307
 STARTBR 310
 SYNCPOINT 314
 SYNCPOINT ROLLBACK 315
 UNLOCK 316
 VERIFY PASSWORD 319
 WAIT CONVID 320
 WAIT EVENT 321
 WAIT EXTERNAL 323
 WAIT JOURNALNUM 324
 WAIT TERMINAL 326
 WAITCICS 328
 WRITE 330
 WRITE OPERATOR 335
 WRITEQ TD 336
 WRITEQ TS 339
 XCTL 341

IOERR condition

DELETE 69
 DUMP TRANSACTION 76
 EIBRCODE 348
 READ 188
 READNEXT 191
 READPREV 195
 READQ TD 197
 READQ TS 199
 RESETBR 230
 RETRIEVE 234
 REWRITE 238
 START 308
 STARTBR 311
 UNLOCK 316
 WAIT JOURNALNUM 324
 WRITE 330
 WRITE JOURNALNUM 333
 WRITEQ TD 336
 WRITEQ TS 339

ISCINVREQ condition

CANCEL 36
 DELETE 69
 DELETEQ TD 70
 DELETEQ TS 71
 ENDBR 77
 READ 188
 READNEXT 191
 READPREV 195
 READQ TD 197
 READQ TS 199
 RESETBR 230
 REWRITE 238
 START 308
 STARTBR 311
 UNLOCK 317

ISCINVREQ condition (*continued*)
 WRITE 331
 WRITEQ TD 336
 WRITEQ TS 339
 ISSUE ABEND command 130
 ISSUE ABORT command 131
 ISSUE ADD command 133
 ISSUE CONFIRMATION command 135
 ISSUE COPY (3270 display) command 137
 ISSUE COPY (3270 logical) command 138
 ISSUE COPY command
 general information 361
 ISSUE DISCONNECT (default) command 139
 ISSUE DISCONNECT (LUTYPE6.1) command 140
 ISSUE DISCONNECT command
 general information 358
 ISSUE END command 141
 ISSUE ENDFILE command 143
 ISSUE ENDOUTPUT command 144
 ISSUE EODS command 145
 ISSUE ERASE command 146
 ISSUE ERASEAUP command 148
 general information 361
 ISSUE ERROR command 149
 ISSUE LOAD command 150
 ISSUE NOTE command 151
 ISSUE PASS command 152
 ISSUE PREPARE command 154
 ISSUE PRINT command 155
 general information 360
 ISSUE QUERY command 156
 ISSUE RECEIVE command 157
 ISSUE REPLACE command 159
 ISSUE RESET command 161
 general information 358
 ISSUE SEND command 162
 ISSUE SIGNAL (APPC) command 164
 ISSUE SIGNAL (LUTYPE6.1) command 165
 ISSUE SIGNAL command
 general information 358
 ISSUE WAIT command 166
 ITEM option
 READQ TS 198
 WRITEQ TS 338
 ITEMERR condition
 READQ TS 199
 WRITEQ TS 340
 IUTYPE option
 BUILD ATTACH (LUTYPE6.1) 31
 BUILD ATTACH (MRO) 33
 EXTRACT ATTACH (LUTYPE6.1) 82
 EXTRACT ATTACH (MRO) 84

J

JES (job entry subsystem)
 CICS interface to 6
 exits 8
 input 8
 internal limits 6
 output 8
 RESP and RESP2 options 8
 retrieve data from JES spool 7
 send file to destination 7
 spooler commands 8
 write directly to JES spool 7
 JIDERR condition
 WAIT JOURNALNUM 324
 WRITE JOURNALNUM 333
 journal record, creating 332
 journaling commands 10
 JOURNALNUM option
 WAIT JOURNALNUM 324
 WRITE JOURNALNUM 332
 JTYPEID option
 WRITE JOURNALNUM 332
 JUSFIRST option
 SEND TEXT 281
 JUSLAST option
 SEND TEXT 281
 JUSTIFY operand
 DFHMDF 386
 DFHMDFI 393
 JUSTIFY option
 SEND TEXT 281

K

katakana and English characters, mixed 61, 225
 KATAKANA option
 ASSIGN 26
 katakana terminals
 CONVERSE (3270 display) 61
 CONVERSE (3270 logical) 50
 CONVERSE (3600 BTAM) 61
 CONVERSE (3735) 61
 CONVERSE (3740) 61
 CONVERSE (LUTYPE2/LUTYPE3) 50
 CONVERSE (System/3) 61
 CONVERSE (System/7) 61
 RECEIVE (3270 display) 218
 RECEIVE (3270 logical) 208
 RECEIVE (3790 3270-display) 218
 RECEIVE (LUTYPE2/LUTYPE3) 208
 RECEIVE (System/3) 218
 RECEIVE (System/7) 218
 RECEIVE MAP 220
 RECEIVE PARTN 225
 SEND (3600 BTAM) 261

katakana terminals (*continued*)

SEND (3735) 261
SEND (3740) 261
SEND (System/3) 261
SEND (System/7) 261

KEEP option

SPOOLCLOSE 293

KEYLENGTH option

DELETE 67
ISSUE ERASE 146
ISSUE REPLACE 159
READ 186
READNEXT 189
READPREV 193
RESETBR 229
STARTBR 310
WRITE 329

KEYNUMBER option

ISSUE ERASE 146
ISSUE REPLACE 159

keyword length 357

L

L40, L64, or L80 options

SEND CONTROL 265
SEND MAP 270
SEND TEXT 282
SEND TEXT NOEDIT 287

label argument, CICS command format 2

LABEL option

HANDLE ABEND 123

LANG operand

DFHMSD 399

LANGINUSE option

ASSIGN 26
SIGNON 291

language codes 373

LANGUAGECODE option

SIGNON 290

LAST option

GDS SEND 117
SEND (non-VTAM) 261
SEND (VTAM) 252
SEND CONTROL 265
SEND MAP 270
SEND PAGE 276
SEND TEXT 282
SEND TEXT MAPPED 284
SEND TEXT NOEDIT 287

LAST value

DFHMDI 393

LASTUSETIME option

VERIFY PASSWORD 318

LDC operand

DFHMSD 399

LDC option

CONVERSE (VTAM) 50
ROUTE 239
SEND (VTAM) 253
SEND CONTROL 265
SEND MAP 270
SEND TEXT 282

LDCMNEM option

ASSIGN 26

LDCNUM option

ASSIGN 26

LEAVEKB option

CONVERSE (non-VTAM) 62
RECEIVE (non-VTAM) 218
SEND (non-VTAM) 261

LEFT value

DFHMDF 386
DFHMDI 393

LENGERR condition

BIF DEEDIT 30
CONNECT PROCESS 40
CONVERSE (non-VTAM) 64
CONVERSE (VTAM) 52
DEQ 72
EIBRCODE byte 1 348
ENQ 79
ENTER TRACENUM 81
EXTRACT PROCESS 89
GETMAIN 122
ISSUE COPY (3270 logical) 138
ISSUE PASS 153
ISSUE RECEIVE 158
LINK 170
LOAD 173
QUERY SECURITY 183
READ 188
READNEXT 192
READPREV 195
READQ TD 197
READQ TS 199
RECEIVE (non-VTAM) 219
RECEIVE (VTAM) 209
RECEIVE PARTN 226
RETRIEVE 234
RETURN 236
REWRITE 238
SEND (non-VTAM) 262
SEND (VTAM) 253
SEND TEXT 283
START 308
WRITE 331
WRITE JOURNALNUM 333
WRITE OPERATOR 335
WRITEQ TD 336
WRITEQ TS 340
XCTL 341

LENGTH operand
 DFHMDF 386
 LENGTH option
 BIF DEEDIT 30
 built-in function 30
 default (assembler language) 4
 default (C) 3
 default (PL/I) 4
 DEQ 72
 DUMP TRANSACTION 75
 ENQ 78
 EXTRACT LOGONMSG 88
 fullword length alternative (FLENGTH) 357
 GETMAIN 121
 ISSUE ADD 133
 ISSUE PASS 152
 ISSUE RECEIVE 157
 ISSUE REPLACE 159
 ISSUE SEND 162
 LINK 169
 LOAD 172
 READ 186
 READNEXT 190
 READPREV 193
 READQ TD 196
 READQ TS 198
 RECEIVE (non-VTAM) 218
 RECEIVE (VTAM) 208
 RECEIVE MAP 221
 RECEIVE MAP MAPPINGDEV command 223
 RECEIVE PARTN 225
 RETRIEVE 233
 RETURN 235
 REWRITE 237
 SEND (non-VTAM) 261
 SEND (VTAM) 253
 SEND MAP 270
 SEND MAP MAPPINGDEV command 275
 SEND TEXT 282
 SEND TEXT MAPPED 284
 SEND TEXT NOEDIT 287
 START 306
 WRITE 329
 WRITE JOURNALNUM 332
 WRITEQ TD 336
 WRITEQ TS 338
 XCTL 341
 LENGTH value
 DFHMDI 391
 DFHMSD 397
 LENGTHLIST option
 DUMP TRANSACTION 75
 LIGHTPEN option
 HANDLE AID 125
 LINE option
 SPOOLWRITE 301
 LINE value
 DFHMDI 394
 line, telecommunication 358
 LINEADDR option
 CONVERSE (non-VTAM) 62
 SEND (non-VTAM) 261
 LINK command 168
 link to program expecting return 168
 LIST option
 ROUTE 240
 literal constants 4
 LLID option
 GDS RECEIVE 115
 LOAD command 172
 load programs, tables, or maps 172
 LOADING condition
 DELETE 69
 READ 188
 STARTBR 311
 WRITE 331
 logical device code (LDC) 44, 246
 logical messages, BMS
 completing a logical message 276
 full BMS
 ROUTE 239
 purging a logical message 180
 routing a logical message 239
 LOGMESSAGE option
 QUERY SECURITY 182
 LOGMODE option
 ISSUE PASS 152
 LOGONLOGMODE option
 ISSUE PASS 152
 LU (logical unit)
 3270 Information Display System 44, 138, 203, 245
 3270 SCS Printer 43, 244
 3270-Display, LUTYPE2 42, 202, 243
 3270-Display, LUTYPE3 202, 243
 3600 (3601) 44, 204, 246
 3600 (3614) 45, 205, 246
 3600 pipeline 204, 245
 3650 host conversational (3270) 46, 247
 3650 host conversational (3653) 46, 248
 3650 interpreter 45, 145, 150, 205, 247
 3650/3680 host command processor 248
 3770 batch 48, 206, 249
 3790 (3270-display) 49, 217, 251
 3790 (3270-printer) 251
 3790 full-function 207, 250
 3790 full-function or inquiry 48
 3790 SCS printer 250
 batch 48, 206, 249
 conversing with (CONVERSE) 358
 interactive 47
 reading data from 157, 357
 writing data to 133, 357

- LUNAME option
 - ISSUE PASS 152
- LUTYPE2, 3270-Display LU 42, 202, 243
- LUTYPE3, 3270-Display LU 202, 243
- LUTYPE4
 - logical unit 42, 202, 243
- LUTYPE6.1 logical unit
 - acquiring a session 18
 - communicating on LUTYPE6.1 session 43
 - converting 8-character names to 4 characters 91
 - disconnecting 140
 - ensuring terminal operation has completed 326
 - getting information about 176
 - receiving data 203
 - requesting change of direction 165
 - retrieving values from an LUTYPE6.1 header 82
 - sending data 244
 - specifying values for an MRO attach header 33
 - specifying values for LUTYPE6.1 attach header 31

M

- macros, BMS, summary 381
- magnetic slot reader (MSR) 377
- MAIN option
 - WRITEQ TS 338
- main storage 120
- map definition macro, BMS 381, 391
- MAP option
 - RECEIVE MAP 221
 - RECEIVE MAP MAPPINGDEV command 224
 - SEND MAP 270
 - SEND MAP MAPPINGDEV command 275
- MAP value
 - DFHMSD 401
- MAPATTS operand
 - DFHMDI 394
 - DFHMSD 399
- MAPCOLUMN option
 - ASSIGN 26
- MAPFAIL condition
 - RECEIVE MAP 221
 - RECEIVE MAP MAPPINGDEV command 224
- MAPHEIGHT option
 - ASSIGN 26
- MAPLINE option
 - ASSIGN 26
- MAPONLY option
 - SEND MAP 270
 - SEND MAP MAPPINGDEV command 275
- MAPONLY value
 - DFHMDI 393
 - DFHMSD 398
- MAPPINGDEV option
 - RECEIVE MAP MAPPINGDEV command 224
 - SEND MAP MAPPINGDEV command 275

- maps, loading 172
- mapset definition macro (DFHMSD) 381, 397
- MAPSET option
 - RECEIVE MAP 221
 - RECEIVE MAP MAPPINGDEV command 224
 - SEND MAP 270
 - SEND MAP MAPPINGDEV command 275
- MAPSFX operand
 - DFHPDI 403
- MAPWIDTH option
 - ASSIGN 26
- MASSINSERT option
 - WRITE 330
- MAXLENGTH option
 - CONVERSE (non-VTAM) 62
 - CONVERSE (VTAM) 51
 - fullword alternative to MAXLENGTH 357
 - GDS RECEIVE 115
 - RECEIVE (non-VTAM) 218
 - RECEIVE (VTAM) 208
 - SPOOLREAD 299
- MAXLENGTH option
 - CONVERSE (non-VTAM) 62
 - CONVERSE (VTAM) 51
 - fullword length alternative (MAXLENGTH) 357
 - RECEIVE (non-VTAM) 218
 - RECEIVE (VTAM) 208
 - WRITE OPERATOR 335
- MAXLIFETIME option
 - DEQ 72
 - ENQ 79
- MAXPROCLN option
 - EXTRACT PROCESS 89
 - GDS EXTRACT PROCESS 108
- MCC option
 - SPOOLOPEN 296
- MINUTES option
 - DELAY 65
 - POST 179
 - ROUTE 240
 - START 306
- MMDDYY option
 - FORMATTIME 93
- MMDDYYYY option
 - FORMATTIME 93
- MODE operand
 - DFHMSD 399
- model codes (terminal) 351
- MODENAME option
 - GDS ALLOCATE 102
- MONITOR command 174
- monitoring application performance 174
- monitoring commands 10
- MONTHOFYEAR option
 - FORMATTIME 93

MSR (magnetic slot reader)
 control byte values and constants 377
 DFHMSRCA, 377
 MSR option
 SEND CONTROL 265
 SEND MAP 270
 SEND TEXT 282
 MSRCONTROL option
 ASSIGN 26
 multi region operation (MRO) commands
 ALLOCATE 20
 BUILD ATTACH 33
 CONVERSE 53
 EXTRACT ATTACH 84
 EXTRACT ATTRIBUTES 87
 FREE 98
 RECEIVE 210
 SEND 255
 multiple base registers 355
 MUSTENTER
 DFHMDF 389
 DFHMDI 395
 DFHMDS 401
 MUSTFILL
 DFHMDF 389
 DFHMDI 395
 DFHMDS 401

N

name argument, CICS command format 2
 NAME option
 WAIT EVENT 321
 WAIT EXTERNAL 323
 WAITCICS 328
 naming restriction 353
 National language codes 373
 NATLANG option
 SIGNON 291
 NATLANGINUSE option
 ASSIGN 26
 SIGNON 291
 NETNAME option
 ASSIGN 26
 EXTRACT TCT 91
 NETNAMEIDERR condition
 ALLOCATE (APPC) 17
 new tasks, passing data to 305
 NEWPASSWORD option
 CHANGE PASSWORD 37
 SIGNON 291
 NEXT option
 READQ TS 198
 NEXT value
 DFHMDI 391, 394

NEXTTRANSID option
 ASSIGN 26
 NLEOM option
 ROUTE 240
 SEND MAP 270
 SEND TEXT 282
 NO value
 DFHMDI 393, 394, 395
 DFHMDS 398, 401
 NOAUTOPAGE option
 SEND PAGE 276
 NOCC option
 SPOOLOPEN 296
 NOCHECK option
 START 306
 NODE option
 SPOOLOPEN 296
 NODUMP option
 ABEND 12
 NOFLUSH option
 SEND MAP 271
 NOHANDLE option
 deactivating HANDLE CONDITION 127
 option 5
 overriding HANDLE AID 6
 NOJBUFSP condition
 WRITE JOURNALNUM 333
 NONVAL condition
 ISSUE LOAD 150
 NOPASSBKRD condition
 RECEIVE (non-VTAM) 219
 NOPASSBKWR condition
 SEND (non-VTAM) 262
 NOQUEUE option
 ALLOCATE (APPC) 16
 ALLOCATE (LUTYPE6.1) 18
 ALLOCATE (MRO) 20
 GDS ALLOCATE 102
 NOQUIESCE
 ISSUE PASS 152
 NORM value
 DFHMDF 385
 NOSPACE condition
 DUMP TRANSACTION 76
 REWRITE 238
 WRITE 331
 WRITEQ TD 337
 WRITEQ TS 340
 NOSTART condition
 ISSUE LOAD 150
 NOSTG condition
 DUMP TRANSACTION 76
 GETMAIN 122
 NOSUSPEND option
 ALLOCATE (APPC) 16
 ALLOCATE (LUTYPE6.1) 18

NOSUSPEND option (*continued*)

ENQ 79
GETMAIN 121
READQ TD 196
WRITE JOURNALNUM 332
WRITEQ TS 339

NOTALLOC condition

CONNECT PROCESS 40
CONVERSE (non-VTAM) 64
CONVERSE (VTAM) 52
EXTRACT ATTACH (LUTYPE6.1) 83
EXTRACT ATTACH (MRO) 85
EXTRACT ATTRIBUTES (APPC) 86
EXTRACT ATTRIBUTES (MRO) 87
EXTRACT LOGONMSG 88
EXTRACT PROCESS 90
EXTRACT TCT 91
FREE 95
FREE (APPC) 96
FREE (LUTYPE6.1) 97
FREE (MRO) 98
ISSUE ABEND 130
ISSUE CONFIRMATION 135
ISSUE COPY (3270 display) 137
ISSUE COPY (3270 logical) 138
ISSUE DISCONNECT (LUTYPE6.1) 140
ISSUE ENDFILE 143
ISSUE ENDOUTPUT 144
ISSUE EODS 145
ISSUE ERASEAUP 148
ISSUE ERROR 149
ISSUE LOAD 150
ISSUE PASS 153
ISSUE PREPARE 154
ISSUE PRINT 155
ISSUE RESET 161
ISSUE SIGNAL (APPC) 164
ISSUE SIGNAL (LUTYPE6.1) 165
POINT 176
RECEIVE (non-VTAM) 219
RECEIVE (VTAM) 209
SEND (non-VTAM) 262
SEND (VTAM) 253
WAIT CONVID 320
WAIT SIGNAL 325
WAIT TERMINAL 326

NOTAUTH condition

CANCEL 36
CHANGE PASSWORD 37
DELETE 69
DELETEQ TD 70
DELETEQ TS 71
ENDBR 77
HANDLE ABEND 123
LINK 170
LOAD 173

NOTAUTH condition (*continued*)

READ 188
READNEXT 192
READPREV 195
READQ TD 197
READQ TS 199
RELEASE 227
RESETBR 230
REWRITE 238
SIGNON 291
START 308
STARTBR 311
UNLOCK 317
VERIFY PASSWORD 319
WRITE 331
WRITE JOURNALNUM 333
WRITEQ TD 337
WRITEQ TS 340
XCTL 342

NOTFND condition

CANCEL 36
DELETE 69
QUERY SECURITY 183
READ 188
READNEXT 192
READPREV 195
RESETBR 230
RETRIEVE 234
REWRITE 238
STARTBR 311

NOTOPEN condition

DELETE 69
DUMP TRANSACTION 76
READ 188
READQ TD 197
STARTBR 311
UNLOCK 317
WAIT JOURNALNUM 324
WRITE 331
WRITE JOURNALNUM 333
WRITEQ TD 337

NOTTRUNCATE option

CONVERSE (non-VTAM) 62
CONVERSE (VTAM) 51
RECEIVE (non-VTAM) 218
RECEIVE (VTAM) 208

NOWAIT option

ISSUE ADD 133
ISSUE ERASE 146
ISSUE REPLACE 159
ISSUE SEND 162

NUM value

DFHMDF 385

NUMBER value

DFHMDF 391, 394

NUMEVENTS option
 WAIT EXTERNAL 323
 WAITCICS 328
 NUMITEMS option
 READQ TS 199
 WRITEQ TS 339
 NUMREC option
 DELETE 68
 ISSUE ADD 133
 ISSUE ERASE 146
 ISSUE REPLACE 159
 NUMROUTES option
 WRITE OPERATOR 335
 NUMSEGMENTS option
 DUMP TRANSACTION 75
 NUMTAB option
 ASSIGN 26

O

OBFMT operand
 DFHMDI 394
 DFHMSD 399
 OCCURS operand
 DFHMDF 386
 OFF value
 DFHMDF 386
 DFHMDI 393
 DFHMSD 399
 OIDCARD option
 SIGNON 291
 OPCLASS option
 ASSIGN 26
 ROUTE 240
 OPENERR condition
 DUMP TRANSACTION 76
 OPERID option
 HANDLE AID 125
 OPERKEYS option
 ASSIGN 26
 OPERPURGE option
 SEND PAGE 277
 OPID option
 ASSIGN 26
 OPSECURITY option
 ASSIGN 27
 options
 BMS 220, 223, 280
 length 357
 OPTIONS(MAIN)
 in PL/I 353
 ORGABCODE option
 ASSIGN 27
 OUTDESCR option
 SPOOLOPEN 296

OUTLINE operand
 DFHMDF 387
 DFHMDI 394
 DFHMSD 399
 OUTLINE option
 ASSIGN 27
 OUTPARTN option
 SEND CONTROL 266
 SEND MAP 271
 SEND TEXT 282
 SEND TEXT NOEDIT 287
 output control, 2980 General Banking Terminal System 213
 output to common buffer, 2980 214
 OVERFLOW condition
 SEND MAP 272

P

PA1-PA3 option
 HANDLE AID 125
 PAGE option
 SPOOLWRITE 301
 PAGENUM option
 ASSIGN 27
 PAGING option
 SEND CONTROL 266
 SEND MAP 271
 SEND TEXT 282
 SEND TEXT MAPPED 284
 SEND TEXT NOEDIT 287
 partition definition macro (DFHPDI) 381, 403
 partition set definition macro (DFHPSD) 381, 404
 PARTN operand
 DFHMDI 394
 DFHMSD 399
 PARTN option
 RECEIVE PARTN 225
 PARTNER option
 ALLOCATE(APPC) 16
 CONNECT PROCESS 39
 GDS ALLOCATE 102
 GDS CONNECT PROCESS 105
 PARTNERIDERR condition
 ALLOCATE (APPC) 17
 CONNECT PROCESS 40
 PARTNFAIL condition
 RECEIVE MAP 221
 PARTNPAGE option
 ASSIGN 27
 PARTNS option
 ASSIGN 27
 PARTNSET option
 ASSIGN 27
 PASSBK option
 RECEIVE (non-VTAM) 218
 SEND (non-VTAM) 262

passbook control, 2980 213
 passing a session 152
 passing control
 expecting return (LINK) 168
 on receipt of an AID (HANDLE AID) 125
 on receipt of an AID (IGNORE AID) 129
 without return (XCTL) 341
 passing data to new tasks 305
 PASSWORD option
 CHANGE PASSWORD 37
 SIGNON 291
 VERIFY PASSWORD 318
 PCT option
 DUMP TRANSACTION 75
 performance, application, monitoring 174
 PF1–24 option
 HANDLE AID 125
 PFXLENG option
 WRITE JOURNALNUM 332
 PGMIDERR condition
 HANDLE ABEND 124
 LINK 170
 LOAD 173
 RELEASE 228
 XCTL 342
 PICIN operand
 DFHMDF 387
 PICOUT operand
 DFHMDF 388
 pipeline logical units 204, 245
 PIPELENGTH option
 CONNECT PROCESS 39
 EXTRACT PROCESS 89
 GDS CONNECT PROCESS 105
 GDS EXTRACT PROCESS 108
 PIPLIST option
 CONNECT PROCESS 39
 EXTRACT PROCESS 89
 GDS CONNECT PROCESS 105
 GDS EXTRACT PROCESS 108
 PL/I language
 argument values 3
 LENGTH option default 4
 PROCEDURE statement 353
 STAE option 12
 translated code 353
 POINT command 176
 POINT option
 MONITOR 175
 pointer-ref argument, CICS command format 2
 pointer-value argument, CICS command format 2
 POP HANDLE command 177
 POS operand 382
 DFHMDF 388
 POST command 178
 posting timer-event control area 178
 PPT option
 DUMP TRANSACTION 75
 PREFIX option
 WRITE JOURNALNUM 332
 PRINCONVID option
 GDS ASSIGN 104
 PRINSYSID option
 ASSIGN 27
 GDS ASSIGN 104
 print displayed information 360
 PRINT option
 ISSUE ABORT 131
 ISSUE END 141
 ISSUE SEND 162
 ISSUE WAIT 166
 SEND CONTROL 266
 SEND MAP 271
 SEND MAP MAPPINGDEV command 275
 SEND TEXT 282
 SEND TEXT NOEDIT 287
 SPOOLOPEN 297
 PRINT value
 DFHMDF 391
 DFHMDF 397
 printer control character list, DFHBMSCA 375
 priority of task, changing 38
 PRIORITY option
 CHANGE TASK 38
 PROCESS option
 BUILD ATTACH (LUTYPE6.1) 32
 BUILD ATTACH (MRO) 33
 EXTRACT ATTACH (LUTYPE6.1) 82
 EXTRACT ATTACH (MRO) 84
 processing task, control delay of 65
 PROCLENGTH option
 CONNECT PROCESS 39
 EXTRACT PROCESS 89
 GDS CONNECT PROCESS 105
 GDS EXTRACT PROCESS 108
 PROCNAME option
 CONNECT PROCESS 39
 EXTRACT PROCESS 89
 GDS CONNECT PROCESS 105
 GDS EXTRACT PROCESS 108
 PROFILE option
 ALLOCATE (APPC) 16
 ALLOCATE (LUTYPE6.1) 18
 program control
 commands 10
 deleting loaded program 227
 LINK options 169
 linking to another program 168
 load a program, table, or map 172
 returning program control 235
 transfer program control 341

PROGRAM option
 ASSIGN 27
 DUMP TRANSACTION 75
 HANDLE ABEND 123
 ISSUE LOAD 150
 LINK 169
 LOAD 172
 RELEASE 227
 XCTL 341

PROT value
 DFHMDF 385

PROTECT option
 START 306

PS operand
 DFHMDF 389
 DFHMDI 394
 DFHMSD 399

PS option
 ASSIGN 27

PSEUDOBIN option
 CONVERSE (non-VTAM) 62
 RECEIVE (non-VTAM) 219
 SEND (non-VTAM) 262

PUNCH option
 SPOOLOPEN 297

PURGE MESSAGE command 180

PURGEABILITY option
 WAIT EXTERNAL 323
 WAITCICS 328

PUSH HANDLE command 181

Q

QBUSY condition
 READQ TD 197

QIDERR condition
 DELETEQ TD 70
 DELETEQ TS 71
 QUERY SECURITY 184
 READQ TD 197
 READQ TS 199
 WRITEQ TD 337
 WRITEQ TS 340

QNAME option
 ASSIGN 27

QUERY SECURITY command 182

QUEUE option
 BUILD ATTACH (LUTYPE6.1) 32
 BUILD ATTACH (MRO) 34
 DELETEQ TD 70
 DELETEQ TS 71
 EXTRACT ATTACH (LUTYPE6.1) 83
 EXTRACT ATTACH (MRO) 85
 READQ TD 196
 READQ TS 199
 RETRIEVE 233

QUEUE option (*continued*)
 START 306
 WRITEQ TD 336
 WRITEQ TS 339

QZERO condition
 READQ TD 197

R

RBA option
 DELETE 68
 READ 186
 READNEXT 190
 READPREV 194
 RESETBR 229
 STARTBR 310
 WRITE 330

RDATT condition
 CONVERSE (non-VTAM) 64
 RECEIVE (non-VTAM) 219
 RECEIVE MAP 221

reactivate an ABEND exit 123

read attention 56

READ command 185

READ option
 QUERY SECURITY 182

reading records
 batch data interchange 157
 browsing, next 189
 browsing, previous (VSAM) 193
 file control 185
 from temporary storage queue 198
 from terminal or LU 357
 from transient data queue 196

READNEXT command 189

READPREV command 193

READQ TD command 196

READQ TS command 198

RECEIVE (2260) command 212

RECEIVE (2741) command 212

RECEIVE (2980) command 213

RECEIVE (3270 display) command 215

RECEIVE (3270 logical) command 203

RECEIVE (3600 BTAM) command 215

RECEIVE (3600 pipeline) command 204

RECEIVE (3600-3601) command 204

RECEIVE (3600-3614) command 205

RECEIVE (3650) command 205

RECEIVE (3735) command 216

RECEIVE (3740) command 216

RECEIVE (3767) command 206

RECEIVE (3770) command 206

RECEIVE (3790 3270-display) command 217

RECEIVE (3790 full-function or inquiry) command 207

RECEIVE (APPC) command 201

RECEIVE (LUTYPE2/LUTYPE3) command 202
 RECEIVE (LUTYPE4) command 202
 RECEIVE (LUTYPE6.1) command 203
 RECEIVE (MRO) command 210
 RECEIVE (non-VTAM) command 210
 RECEIVE (System/3) command 211
 RECEIVE (System/7) command 211
 RECEIVE (VTAM default) command 201
 RECEIVE command
 input operation without data 361
 read from terminal or logical unit 357
 RECEIVE MAP command 220
 RECEIVE MAP MAPPINGDEV command 223
 RECEIVE PARTN command 225
 RECFM option
 BUILD ATTACH (LUTYPE6.1) 32
 BUILD ATTACH (MRO) 34
 EXTRACT ATTACH (LUTYPE6.1) 83
 EXTRACT ATTACH (MRO) 85
 RECORDLENGTH option
 SPOOLOPEN 297
 records
 deleting VSAM 67
 reading 157, 185
 release exclusive control 316
 requesting next number 151
 updating 159, 237
 writing new 329
 writing new (adding) 133
 register contents in assembler language 353
 relative byte address (RBA) 68
 RELEASE command 227
 RELEASE option
 SEND PAGE 277
 relocatable expression 4
 REPLY option
 WRITE OPERATOR 335
 REPLYLENGTH option
 WRITE OPERATOR 335
 REQID option
 CANCEL 35
 DELAY 66
 ENDBR 77
 POST 179
 READNEXT 190
 READPREV 194
 RESETBR 230
 ROUTE 240
 SEND CONTROL 266
 SEND MAP 271
 SEND TEXT 282
 SEND TEXT MAPPED 284
 SEND TEXT NOEDIT 287
 START 306
 STARTBR 310
 WAIT JOURNALNUM 324
 REQID option (*continued*)
 WRITE JOURNALNUM 332
 RESCLASS option
 QUERY SECURITY 182
 RESET option
 HANDLE ABEND 123
 reset start for browse 229
 RESETBR command 229
 RESID option
 QUERY SECURITY 182
 RESIDLENGTH option
 QUERY SECURITY 182
 RESOURCE option
 BUILD ATTACH (LUTYPE6.1) 32
 BUILD ATTACH (MRO) 34
 DEQ 72
 ENQ 79
 ENTER TRACENUM 80
 EXTRACT ATTACH (LUTYPE6.1) 83
 EXTRACT ATTACH (MRO) 85
 resource scheduling 72
 RESP
 deactivating NOHANDLE 127
 option 5
 values in EIBRESP 349
 RESP and RESP2 options
 for interface to JES 8
 RESP2
 EXPIRED in messages to console operators 335
 INVREQ in messages to console operators 335
 INVREQ in SIGNOFF (Security control) 289
 INVREQ in SIGNON (Security control) 291
 INVREQ in WAIT EXTERNAL 323
 INVREQ on WAITCICS 328
 LENGERR in messages to console operators 335
 NOTAUTH in SIGNON (Security control) 291
 option 5
 USERIDERR in SIGNON (Security control) 292
 values in EIBRESP2 349
 RESSEC option
 ASSIGN 27
 RESTART option
 ASSIGN 27
 RESTYPE option
 QUERY SECURITY 182
 RETAIN option
 SEND PAGE 277
 RETCODE option
 GDS ALLOCATE 102
 GDS ASSIGN 104
 GDS CONNECT PROCESS 105
 GDS EXTRACT ATTRIBUTES 107
 GDS EXTRACT PROCESS 108
 GDS FREE 109
 GDS ISSUE ABEND 110
 GDS ISSUE CONFIRMATION 111

- RETCODE option (*continued*)
 - GDS ISSUE ERROR 112
 - GDS ISSUE PREPARE 113
 - GDS ISSUE SIGNAL 114
 - GDS RECEIVE 115
 - GDS SEND 117
 - GDS WAIT 119
- RETPAGE condition
 - SEND CONTROL 267
 - SEND MAP 272
 - SEND PAGE 277
 - SEND TEXT 283
- RETRIEVE command 232
- retrieve data stored for task 232
- RETURN command 235
- return program control 235
- RETURNPROG option
 - ASSIGN 27
- REVERSE value
 - DFHMDF 386
 - DFHMDI 393
 - DFHMSD 399
- REWRITE command 237
- REWRITE option
 - WRITEQ TS 339
- RIDFLD option
 - DELETE 68
 - ISSUE ADD 133
 - ISSUE ERASE 146
 - ISSUE NOTE 151
 - ISSUE REPLACE 159
 - READ 186
 - READNEXT 190
 - READPREV 194
 - RESETBR 230
 - STARTBR 310
 - WRITE 330
- RIGHT value
 - DFHMDF 386
 - DFHMDI 393
- ROLLBACK option
 - SYNCPOINT ROLLBACK 315
- ROLLEDBACK condition
 - LINK 170
 - SYNCPOINT 314
- ROUTE command 239
- ROUTECODES option
 - WRITE OPERATOR 335
- RPROCESS option
 - BUILD ATTACH (LUTYPE6.1) 32
 - BUILD ATTACH (MRO) 34
 - EXTRACT ATTACH (LUTYPE6.1) 83
 - EXTRACT ATTACH (MRO) 85
- RRESOURCE option
 - BUILD ATTACH (LUTYPE6.1) 32
 - BUILD ATTACH (MRO) 34

- RRESOURCE option (*continued*)
 - EXTRACT ATTACH (LUTYPE6.1) 83
 - EXTRACT ATTACH (MRO) 85
- RRN option
 - DELETE 68
 - ISSUE ADD 133
 - ISSUE ERASE 146
 - ISSUE NOTE 151
 - ISSUE REPLACE 159
 - READ 186
 - READNEXT 191
 - READPREV 194
 - RESETBR 230
 - STARTBR 310
 - WRITE 330
- RTEFAIL condition
 - ROUTE 241
- RTERMID option
 - RETRIEVE 233
 - START 307
- RTESOME condition
 - ROUTE 241
- RTRANSID option
 - RETRIEVE 233
 - START 307

S

- SAA (Systems Application Architecture)
 - Communications (CPI) 365
 - Resource Recovery 363
- SAME value
 - DFHMDI 391, 394
- schedule use of resource by task 72, 78
- SCRNHT option
 - ASSIGN 28
- SCRNWD option
 - ASSIGN 28
- SCS (SNA character string)
 - CONVERSE 43
 - SEND 244
 - SEND (VTAM) 250
- SCS printer logical unit, 3790 250
- SECONDS option
 - DELAY 66
 - POST 179
 - ROUTE 240
 - START 307
- security commands 10
- SEGMENTLIST option
 - DUMP TRANSACTION 75
- SELNERR condition
 - ISSUE ABORT 132
 - ISSUE ADD 134
 - ISSUE END 142
 - ISSUE ERASE 147

SELNERR condition (*continued*)

- ISSUE NOTE 151
- ISSUE QUERY 156
- ISSUE REPLACE 160
- ISSUE SEND 163
- ISSUE WAIT 167
- SEND (2260) command 257
- SEND (2741) command 257
- SEND (2980) command 258
- SEND (3270 display) command 258
- SEND (3270 logical) command 245
- SEND (3600 BTAM) command 259
- SEND (3600 pipeline) command 245
- SEND (3600-3601) command 246
- SEND (3600-3614) command 246
- SEND (3650 interpreter) command 247
- SEND (3650-3270) command 247
- SEND (3650-3653) command 248
- SEND (3650-3680) command 248
- SEND (3735) command 259
- SEND (3740) command 260
- SEND (3767) command 249
- SEND (3770) command 249
- SEND (3790 3270-display) command 251
- SEND (3790 3270-printer) command 251
- SEND (3790 full-function or inquiry) command 250
- SEND (3790 SCS) command 250
- SEND (APPC) command 242
- SEND (LUTYPE2/LUTYPE3) command 243
- SEND (LUTYPE4) command 243
- SEND (LUTYPE6.1) command 244
- SEND (MRO) command 255
- SEND (non-VTAM default) command 255
- SEND (SCS) command 244
- SEND (System/3) command 256
- SEND (System/7) command 256
- SEND (vtam default) command 242
- send asynchronous interrupt 358
- SEND command
 - write to terminal 357
- SEND CONTROL command 264
- SEND MAP command 268
- SEND MAP MAPPINGDEV command 274
- SEND PAGE command 276
- SEND PARTNSET command 279
- SEND TEXT command 280
- SEND TEXT MAPPED command 284
- SEND TEXT NOEDIT command 286
- sending data to output device 162
- sequential retrieval, browsing
 - reading records 185
- SESSBUSY condition
 - ALLOCATE (LUTYPE6.1) 18
- SESSION option
 - ALLOCATE (LUTYPE6.1) 18
 - CONNECT PROCESS 39

SESSION option (*continued*)

- CONVERSE (non-VTAM) 62
- CONVERSE (VTAM) 51
- EXTRACT ATTACH (LUTYPE6.1) 83
- EXTRACT ATTACH (MRO) 85
- EXTRACT ATTRIBUTES (MRO) 87
- FREE (LUTYPE6.1) 97
- FREE (MRO) 98
- ISSUE DISCONNECT (LUTYPE6.1) 140
- ISSUE SIGNAL (LUTYPE6.1) 165
- POINT 176
- RECEIVE (non-VTAM) 219
- RECEIVE (VTAM) 208
- SEND (non-VTAM) 262
- SEND (VTAM) 253
- WAIT TERMINAL 326
- session, passing 152
- SESSIONERR condition
 - ALLOCATE (LUTYPE6.1) 18
 - EIBRCODE bytes 1-2 347
- SET option
 - ADDRESS SET 15
 - CONVERSE (non-VTAM) 62
 - CONVERSE (VTAM) 51
 - EXTRACT LOGONMSG 88
 - GDS RECEIVE 115
 - GETMAIN 122
 - ISSUE RECEIVE 157
 - LOAD 172
 - POST 179
 - READ 186
 - READNEXT 191
 - READPREV 194
 - READQ TD 196
 - READQ TS 199
 - RECEIVE (non-VTAM) 219
 - RECEIVE (VTAM) 208
 - RECEIVE MAP 221
 - RECEIVE MAP MAPPINGDEV command 224
 - RECEIVE PARTN 225
 - RETRIEVE 233
 - SEND CONTROL 266
 - SEND MAP 271
 - SEND MAP MAPPINGDEV command 275
 - SEND PAGE 277
 - SEND TEXT 283
- SHARED option
 - GETMAIN 122
- SIGDATA option
 - ASSIGN 28
- SIGNAL condition
 - CONVERSE (VTAM) 52
 - ISSUE CONFIRMATION 136
 - ISSUE DISCONNECT (default) 139
 - ISSUE ERROR 149
 - RECEIVE (VTAM) 209

SIGNAL condition (*continued*)
 SEND (VTAM) 253
 WAIT SIGNAL 325
 WAIT TERMINAL 326
 SIGNOFF command 289
 SIGNON command 290
 single thread used with JES 294
 SIT option
 DUMP TRANSACTION 75
 SIZE operand
 DFHMDI 394
 SNA character string
 See SCS
 SOSI operand
 DFHMDF 389
 DFHMDI 395
 DFHMSD 400
 SOSI option
 ASSIGN 28
 SPCOMMAND
 RESID value not valid 183
 RESTYPE values 183
 Spool commands 10
 SPOOLCLOSE command 293
 SPOOLCLOSE, implicit 294
 spooler, JES 6
 SPOOLOPEN INPUT command 294
 SPOOLOPEN OUTPUT 296
 SPOOLREAD command 299
 SPOOLWRITE command 301
 STAE option, PL/I 12
 standard attribute and printer control character list, BMS
 (DFHBMSCA) 375
 START command 303
 STARTBR command 309
 STARTCODE option
 ASSIGN 28
 STARTIO option
 WAIT JOURNALNUM 324
 WRITE JOURNALNUM 333
 STATE option
 ALLOCATE (APPC) 16
 ALLOCATE (MRO) 20
 CONNECT PROCESS 39
 CONVERSE (non-VTAM) 63
 CONVERSE (VTAM) 51
 EXTRACT ATTRIBUTES (APPC) 86
 EXTRACT ATTRIBUTES (MRO) 87
 FREE (APPC) 96
 FREE (MRO) 98
 GDS ALLOCATE 102
 GDS CONNECT PROCESS 105
 GDS EXTRACT ATTRIBUTES 107
 GDS FREE 109
 GDS ISSUE ABEND 110
 GDS ISSUE CONFIRMATION 111
 STATE option (*continued*)
 GDS ISSUE ERROR 112
 GDS ISSUE PREPARE 113
 GDS ISSUE SIGNAL 114
 GDS RECEIVE 116
 GDS SEND 118
 GDS WAIT 119
 ISSUE ABEND 130
 ISSUE CONFIRMATION 135
 ISSUE ERROR 149
 ISSUE PREPARE 154
 ISSUE SIGNAL (APPC) 164
 RECEIVE (non-VTAM) 219
 RECEIVE (VTAM) 209
 SEND (non-VTAM) 262
 SEND (VTAM) 253
 WAIT CONVID 320
 STATIONID option
 ASSIGN 28
 storage area length 22
 storage control commands 10
 STORAGE operand
 DFHMSD 400
 STORAGE option
 DUMP TRANSACTION 75
 storage, dynamic 355
 STRFIELD option
 CONVERSE (non-VTAM) 63
 CONVERSE (VTAM) 51
 SEND (non-VTAM) 262
 SEND (VTAM) 253
 stub, program 354
 SUBADDR option
 ISSUE ABORT 131
 ISSUE END 141
 ISSUE SEND 162
 ISSUE WAIT 166
 SUFFIX operand
 DFHMSD 400
 DFHPSD 404
 SUPPRESSED condition
 DUMP TRANSACTION 76
 WRITE 331
 SUSPEND command 313
 switched line disconnection 358
 symbolic register DFHEIPLR 356
 synchronization levels
 basic conversations 108
 synchronize, action
 journal output (WAIT JOURNALNUM) 324
 terminal input/output 357
 SYNCLEVEL option
 CONNECT PROCESS 40
 EXTRACT PROCESS 89
 GDS CONNECT PROCESS 106
 GDS EXTRACT PROCESS 108

SYNCONRETURN option
 LINK 169
 syncpoint
 backing out 315
 commands 10
 establishing 314
 SYNCPOINT command 314
 SYNCPOINT ROLLBACK command 315
 syntax notation 1
 SYSBUSY condition
 ALLOCATE (APPC) 17
 ALLOCATE (LUTYPE6.1) 18
 ALLOCATE (MRO) 20
 EIBRCODE byte 3 348
 SYSID option
 ALLOCATE (APPC) 16
 ALLOCATE (LUTYPE6.1) 18
 ALLOCATE (MRO) 20
 ASSIGN 28
 CANCEL 35
 DELETE 68
 DELETEQ TD 70
 DELETEQ TS 71
 ENDBR 77
 EXTRACT TCT 91
 GDS ALLOCATE 102
 LINK 169
 READ 187
 READNEXT 191
 READPREV 194
 READQ TD 197
 READQ TS 199
 RESETBR 230
 REWRITE 237
 START 307
 STARTBR 310
 UNLOCK 316
 WRITE 330
 WRITEQ TD 336
 WRITEQ TS 339
 SYSIDERR condition
 ALLOCATE (APPC) 17
 ALLOCATE (LUTYPE6.1) 19
 ALLOCATE (MRO) 20
 CANCEL 36
 DELETE 69
 DELETEQ TD 70
 DELETEQ TS 71
 EIBRCODE bytes 1-2 347
 ENDBR 77
 LINK 171
 READ 188
 READNEXT 192
 READPREV 195
 READQ TD 197
 READQ TS 200

SYSIDERR condition (*continued*)
 RESETBR 231
 REWRITE 238
 START 308
 STARTBR 312
 UNLOCK 317
 WRITE 331
 WRITEQ TD 337
 WRITEQ TS 340
 System/3
 CONVERSE 54
 RECEIVE 211
 SEND 256
 System/7
 CONVERSE 54
 RECEIVE 211
 SEND 256
 systemname
 definition 2, 3, 4
 systemname argument, CICS command format 2

T

TABLES option
 DUMP TRANSACTION 75
 tables, loading 172
 task control commands 10
 task initiation 305
 TASK option
 DUMP TRANSACTION 75
 task, abnormal termination 123
 task, delay processing of 65
 TASKDATALOC resource definition option 13
 TASKPRIORITY option
 ASSIGN 28
 TCAM-supported terminals and logical units 358
 TCT option
 DUMP TRANSACTION 76
 TCTUA option
 ADDRESS 14
 TCTUALENG option
 ASSIGN 28
 telecommunication line, relinquishing 358
 teletypewriter
 messages 359
 programming 359
 TELLERID option
 ASSIGN 28
 temporary storage control commands 10
 TERM operand
 DFHMDI 395
 DFHMSD 400
 TERMCODE option
 ASSIGN 28, 351
 TERMERR condition
 CONNECT PROCESS 40

TERMERR condition (*continued*)
 CONVERSE (non-VTAM) 64
 CONVERSE (VTAM) 52
 ISSUE ABEND 130
 ISSUE CONFIRMATION 136
 ISSUE COPY (3270 logical) 138
 ISSUE DISCONNECT (default) 139
 ISSUE DISCONNECT (LUTYPE6.1) 140
 ISSUE EODS 145
 ISSUE ERASEAUP 148
 ISSUE ERROR 149
 ISSUE LOAD 150
 ISSUE PREPARE 154
 ISSUE PRINT 155
 ISSUE SIGNAL (APPC) 164
 ISSUE SIGNAL (LUTYPE6.1) 165
 LINK 171
 RECEIVE (non-VTAM) 219
 RECEIVE (VTAM) 209
 SEND (non-VTAM) 263
 SEND (VTAM) 254
 WAIT SIGNAL 325
 TERMID option
 EXTRACT TCT 91
 ISSUE COPY (3270 display) 137
 ISSUE COPY (3270 logical) 138
 START 307
 TERMIDERR condition
 ISSUE COPY (3270 display) 137
 START 308
 terminal control 357
 commands 10
 terminal model codes 351
 terminal operator paging, initiate paging transaction 276
 TERMINAL option
 DUMP TRANSACTION 76
 RECEIVE MAP 221
 SEND CONTROL 266
 SEND MAP 271
 SEND TEXT 283
 SEND TEXT MAPPED 284
 SEND TEXT NOEDIT 287
 terminal type codes 351
 terminate data set processing
 abnormal 131
 normal 141
 terminology notes ix
 TERMPRIORITY option
 ASSIGN 28
 TEXT option
 WRITE OPERATOR 335
 TEXTKYBD option
 ASSIGN 28
 TEXTLENGTH option
 WRITE OPERATOR 335
 TEXTPRINT option
 ASSIGN 29
 time of day, requesting 21
 TIME option
 DELAY 66
 FORMATTIME 93
 POST 179
 ROUTE 240
 START 307
 TIMEOUT option
 WRITE OPERATOR 335
 timer-event control area 178
 TIMESEP option
 FORMATTIME 93
 TIOAPFX operand
 DFHMDI 395
 DFHMSD 401
 TITLE option
 ROUTE 241
 TOFLENGTH option
 CONVERSE (non-VTAM) 63
 CONVERSE (VTAM) 51
 fullword alternative to TOLENGTH 357
 SPOOLREAD 299
 TOKEN option
 DELETE 68
 READ 187
 REWRITE 237
 SPOOLCLOSE 293
 SPOOLOPEN 294, 297
 SPOOLREAD 299
 SPOOLWRITE 301
 UNLOCK 316
 TOLENGTH option
 CONVERSE (non-VTAM) 63
 CONVERSE (VTAM) 51
 fullword length alternative (TOFLENGTH) 357
 TRACENUM option
 ENTER TRACENUM 80
 TRAILER operand
 DFHMDI 395
 TRAILER option
 SEND PAGE 277
 SEND TEXT 283
 TRANPRIORITY option
 ASSIGN 29
 transfer program control 341
 TRANSID option
 CANCEL 35
 LINK 169
 RETURN 236
 SEND PAGE 277
 START 307
 TRANSIDERR condition
 START 308

- transient data commands 11
- transient data control
 - delete intrapartition queue 70
 - read data from TD queue 196
 - write data to TD queue 336
- translated code 353
- TRANSP operand
 - DFHMDF 389
 - DFHMDI 395
 - DFHMSD 401
- TRIGGER option
 - HANDLE AID 125
- TRIGGER value
 - DFHMDF 389
 - DFHMDI 395
 - DFHMSD 401
- TRIGRAPH operand
 - DFHMSD 401
- TRT option
 - DUMP TRANSACTION 76
- TSIOERR condition
 - PURGE MESSAGE 180
 - SEND CONTROL 267
 - SEND MAP 273
 - SEND PAGE 277
 - SEND TEXT 283
 - SEND TEXT MAPPED 285
 - SEND TEXT NOEDIT 288
- TWA option
 - ADDRESS 14
- TWALENG option
 - ASSIGN 29
- type codes (terminal) 351
- TYPE operand
 - DFHMSD 401

U

- UNATTEND option
 - ASSIGN 29
- UNDERLINE value
 - DFHMDF 386
 - DFHMDI 393
 - DFHMSD 399
- UNEXPIN condition
 - ISSUE ABORT 132
 - ISSUE ADD 134
 - ISSUE END 142
 - ISSUE ERASE 147
 - ISSUE NOTE 151
 - ISSUE QUERY 156
 - ISSUE RECEIVE 158
 - ISSUE REPLACE 160
 - ISSUE SEND 163
 - ISSUE WAIT 167
 - RECEIVE MAP 222

- UNLOCK command 316
- UNPROT value
 - DFHMDF 385
- UNTIL option
 - DELAY 66
- UPDATE option
 - QUERY SECURITY 183
 - READ 187
- updating records
 - batch data interchange 159
 - file control 237
- USERDATAKEY option
 - GETMAIN 122
- USEREXIT value
 - DFHMDF 389
 - DFHMDI 395
 - DFHMSD 401
- USERID option
 - ASSIGN 29
 - CHANGE PASSWORD 37
 - SIGNON 291
 - SPOOLCLOSE 294
 - SPOOLOPEN 297
 - START 307
 - VERIFY PASSWORD 318
- USERIDERR condition
 - CHANGE PASSWORD 37
 - SIGNON 292
 - START 308
 - VERIFY PASSWORD 319
- USERNAME option
 - ASSIGN 29
- USERPRIORITY option
 - ASSIGN 29
- USING option
 - ADDRESS SET 15

V

- VALIDATION option
 - ASSIGN 29
- VALIDN operand
 - DFHMDF 389
 - DFHMDI 395
 - DFHMSD 401
- VERIFY PASSWORD command 318
- VIEWPOS operand
 - DFHPDI 403
- VIEWSIZE operand
 - DFHPDI 403
- VOLUME option
 - ISSUE ABORT 131
 - ISSUE ADD 133
 - ISSUE END 141
 - ISSUE ERASE 146
 - ISSUE NOTE 151

VOLUME option (*continued*)

ISSUE QUERY 156
ISSUE REPLACE 159
ISSUE SEND 162
ISSUE WAIT 166

VOLUMELENG option

ISSUE ABORT 131
ISSUE ADD 133
ISSUE END 141
ISSUE ERASE 146
ISSUE NOTE 151
ISSUE QUERY 156
ISSUE REPLACE 160
ISSUE SEND 163
ISSUE WAIT 166

VSAM WRITE MASSINSERT

DISABLED cannot occur 316
NOTOPEN cannot occur 317
terminate operation 316

VTAB operand

DFHMSD 402

VTAM logon data, access to 88

W

WAIT CONVID (APPC) command 320

WAIT EVENT command 321

WAIT EXTERNAL command 322

WAIT JOURNALNUM command 324

WAIT option

GDS SEND 118
ISSUE COPY (3270 display) 137
ISSUE COPY (3270 logical) 138
ISSUE ERASEAUP 148
RETRIEVE 233
SEND (non-VTAM) 262
SEND (VTAM) 253
SEND command 357
SEND CONTROL 266
SEND MAP 271
SEND TEXT 283
SEND TEXT MAPPED 284
SEND TEXT NOEDIT 287
terminal control 357
WRITE JOURNALNUM 333

WAIT SIGNAL command 325

WAIT TERMINAL command 326

general information 357

WAITCICS command 327

waits

batch data interchange 166
for event to occur 321
terminal control operation 357

WPMEDIA option

ISSUE ABORT 131
ISSUE END 141

WPMEDIA option (*continued*)

ISSUE SEND 163

ISSUE WAIT 166

WRBRK condition

CONVERSE (non-VTAM) 64

SEND (non-VTAM) 263

SEND CONTROL 267

SEND MAP 273

SEND PAGE 277

SEND TEXT 283

SEND TEXT MAPPED 285

SEND TEXT NOEDIT 288

write break 56

WRITE command 329

WRITE JOURNALNUM command 332

WRITE OPERATOR command 334

critical action 334

eventual action 334

immediate action 334

WRITEQ TD command 336

WRITEQ TS command 338

writing data

to temporary storage queue 338

to terminal or logical unit 357

to transient data queue 336

writing records to data sets

batch data interchange 133

file control 329

X

XCTL command 341

XINIT operand

DFHMDF 389

XRF, generic applid 23

Y

YEAR option

FORMATTIME 93

YES value

DFHMDDI 393, 394, 395

DFHMSD 398, 401

YYDDD option

FORMATTIME 93

YYDDMM option

FORMATTIME 93

YYMMDD option

FORMATTIME 93

YYYYDDD option

FORMATTIME 93

YYYYDDMM option

FORMATTIME 94

YYYYMMDD option

FORMATTIME 94

Z

ZERO value

DFHMDF 386

Sending your comments to IBM

CICS for MVS/ESA.

Application Programming Reference

SC33-1170-03

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book. Please limit your comments to the information in this book and the way in which the information is presented.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

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

You can send your comments to IBM in any of the following ways:

- By mail, use the Readers' Comment Form
- By fax:
 - From outside the U.K., after your international access code use 44 1962 870229
 - From within the U.K., use 01962 870229
- Electronically, use the appropriate network ID:
 - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
 - IBMLink: WINVMJ(IDRCF)
 - Internet: idrcf@winvmj.vnet.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The page number or topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Readers' Comments

CICS for MVS/ESA.

Application Programming Reference

SC33-1170-03

Use this form to tell us what you think about this manual. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer. This form is provided for comments about the information in this manual and the way it is presented.

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

Be sure to print your name and address below if you would like a reply.

Name

Address

Company or Organization

Telephone

Email



You can send your comments POST FREE on this form from any one of these countries:

Australia	Finland	Iceland	Netherlands	Singapore	United States
Belgium	France	Israel	New Zealand	Spain	of America
Bermuda	Germany	Italy	Norway	Sweden	
Cyprus	Greece	Luxembourg	Portugal	Switzerland	
Denmark	Hong Kong	Monaco	Republic of Ireland	United Arab Emirates	

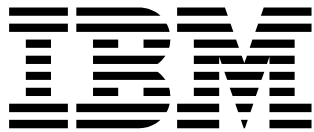
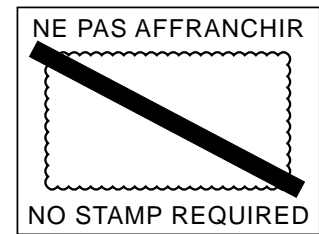
1 Cut along this line

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.).

2 Fold along this line

By air mail
Par avion

IBRS/CCRI NUMBER: PHQ - D/1348/SO



REPONSE PAYEE
GRANDE-BRETAGNE

IBM United Kingdom Laboratories Limited
Information Development Department (MP 095)
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.

SC33-1170-03



Spine information:



CICS for MVS/ESA.

Application Programming Ref

Version 4 Release 1