

CICS Transaction Server for z/OS



CICS Application Programming Reference

Version 3 Release 1

CICS Transaction Server for z/OS



CICS Application Programming Reference

Version 3 Release 1

Contents

Preface	xi
What this book is about	xi
Who should read this book	xi
What you need to know to understand this book	xi
How to use this book	xi
What this book does not cover	xi
Notes on terminology	xi
Summary of changes	xiii
Changes for CICS Transaction Server for z/OS, Version 3 Release 1	xiii
Changes for CICS Transaction Server for z/OS, Version 2 Release 3	xv
Changes for CICS Transaction Server for z/OS, Version 2 Release 2 and CICS Transaction Server for z/OS, Version 2 Release 1	xv
Changes for the CICS Transaction Server for OS/390 Version 1 Release 3 edition	xvi
Figures	xvii
Tables	xix
CICS API commands	1
About the CICS API commands	1
CICS API command format	1
CICS command syntax notation	2
CICS command argument values	3
CICS command restrictions	8
LENGTH options in CICS commands	8
NOHANDLE option	9
RESP and RESP2 options	9
Translated code for CICS commands	10
CICS-value data areas (cvdas)	16
CICS threadsafe commands in the API	16
Threadsafe command list	17
CICS command summary	19
ABEND	27
ACQUIRE	29
ADD SUBEVENT	32
ADDRESS	34
ADDRESS SET	36
ALLOCATE (APPC)	37
ALLOCATE (LUTYPE6.1)	41
ALLOCATE (MRO)	43
ASKTIME	45
ASSIGN	46
BIF DEEDIT	60
BUILD ATTACH (LUTYPE6.1)	62
BUILD ATTACH (MRO)	65
CANCEL	68
CANCEL (BTS)	70
CHANGE PASSWORD	73
CHANGE TASK	75
CHECK ACQPROCESS	76
CHECK ACTIVITY	78

CHECK TIMER	81
CONNECT PROCESS	83
CONVERSE (VTAM default)	86
CONVERSE (APPC)	86
CONVERSE (LUTYPE2/LUTYPE3)	87
CONVERSE (LUTYPE4)	87
CONVERSE (LUTYPE6.1)	88
CONVERSE (SCS)	88
CONVERSE (3270 logical)	89
CONVERSE (3600-3601)	90
CONVERSE (3600-3614)	91
CONVERSE (3650 interpreter)	91
CONVERSE (3650-3270)	92
CONVERSE (3650-3653)	92
CONVERSE (3650-3680)	93
CONVERSE (3767)	93
CONVERSE (3770)	94
CONVERSE (3790 full-function or inquiry)	94
CONVERSE (3790 3270-display)	95
CONVERSE: VTAM options	95
CONVERSE (non-VTAM default)	100
CONVERSE (MRO)	101
CONVERSE (2260)	101
CONVERSE (3270 display)	102
CONVERSE: non-VTAM options	102
CONVERTTIME	107
DEFINE ACTIVITY	109
DEFINE COMPOSITE EVENT	112
DEFINE COUNTER and DEFINE DCOUNTER	114
DEFINE INPUT EVENT	118
DEFINE PROCESS	119
DEFINE TIMER	122
DELAY	125
DELETE	128
DELETE ACTIVITY	135
DELETE CONTAINER (BTS)	137
DELETE CONTAINER (CHANNEL)	139
DELETE COUNTER and DELETE DCOUNTER	140
DELETE EVENT	142
DELETE TIMER	144
DELETEQ TD	145
DELETEQ TS	147
DEQ	149
DOCUMENT CREATE	151
DOCUMENT INSERT	155
DOCUMENT RETRIEVE	158
DOCUMENT SET	160
DUMP TRANSACTION	163
ENDBR	168
ENDBROWSE ACTIVITY	170
ENDBROWSE CONTAINER	171
ENDBROWSE EVENT	172
ENDBROWSE PROCESS	173
ENQ	174
ENTER TRACENUM	178
EXTRACT ATTACH (LUTYPE6.1)	180

EXTRACT ATTACH (MRO)	184
EXTRACT ATTRIBUTES (APPC)	188
EXTRACT ATTRIBUTES (MRO)	190
EXTRACT CERTIFICATE	192
EXTRACT LOGONMSG	195
EXTRACT PROCESS	197
EXTRACT TCPIP	199
EXTRACT TCT	202
FORCE TIMER	203
FORMATTIME	205
FREE	210
FREE (APPC)	211
FREE (LUTYPE6.1)	213
FREE (MRO)	214
FREEMAIN	216
GDS ALLOCATE	219
GDS ASSIGN	222
GDS CONNECT PROCESS	223
GDS EXTRACT ATTRIBUTES	226
GDS EXTRACT PROCESS	228
GDS FREE	230
GDS ISSUE ABEND	232
GDS ISSUE CONFIRMATION	234
GDS ISSUE ERROR	236
GDS ISSUE PREPARE	238
GDS ISSUE SIGNAL	240
GDS RECEIVE	242
GDS SEND	245
GDS WAIT	248
GET CONTAINER (BTS)	250
GET CONTAINER (CHANNEL)	253
GET COUNTER and GET DCOUNTER	256
GETMAIN	261
GETNEXT ACTIVITY	265
GETNEXT CONTAINER	267
GETNEXT EVENT	269
GETNEXT PROCESS	271
HANDLE ABEND	273
HANDLE AID	275
HANDLE CONDITION	277
IGNORE CONDITION	279
INQUIRE ACTIVITYID	280
INQUIRE CONTAINER	283
INQUIRE EVENT	285
INQUIRE PROCESS	287
INQUIRE TIMER	288
INVOKE WEBSERVICE	290
ISSUE ABEND	293
ISSUE ABORT	295
ISSUE ADD	297
ISSUE CONFIRMATION	299
ISSUE COPY (3270 logical)	301
ISSUE DISCONNECT (default)	303
ISSUE DISCONNECT (LUTYPE6.1)	305
ISSUE END	306
ISSUE ENDFILE	308

ISSUE ENDOUTPUT	309
ISSUE EODS	310
ISSUE ERASE	311
ISSUE ERASEAUP	313
ISSUE ERROR	315
ISSUE LOAD	317
ISSUE NOTE	318
ISSUE PASS	320
ISSUE PREPARE	322
ISSUE PRINT	324
ISSUE QUERY	325
ISSUE RECEIVE	327
ISSUE REPLACE	329
ISSUE RESET	332
ISSUE SEND	333
ISSUE SIGNAL (APPC)	336
ISSUE SIGNAL (LUTYPE6.1)	338
ISSUE WAIT	339
JOURNAL	341
LINK	342
LINK ACQPROCESS	350
LINK ACTIVITY	353
LOAD	357
MONITOR	360
MOVE CONTAINER (BTS)	363
MOVE CONTAINER (CHANNEL)	366
POINT	369
POP HANDLE	370
POST	371
PURGE MESSAGE	375
PUSH HANDLE	376
PUT CONTAINER (BTS)	377
PUT CONTAINER (CHANNEL)	380
QUERY COUNTER and QUERY DCOUNTER	384
QUERY SECURITY	387
READ	391
READNEXT	403
READPREV	414
READQ TD	424
READQ TS	428
RECEIVE (VTAM default)	432
RECEIVE (APPC)	432
RECEIVE (LUTYPE2/LUTYPE3)	433
RECEIVE (LUTYPE4)	433
RECEIVE (LUTYPE6.1)	434
RECEIVE (3270 logical)	434
RECEIVE (3600 pipeline)	435
RECEIVE (3600-3601)	435
RECEIVE (3600-3614)	436
RECEIVE (3650)	436
RECEIVE (3767)	437
RECEIVE (3770)	438
RECEIVE (3790 full-function or inquiry)	438
RECEIVE: VTAM options	439
RECEIVE (non-VTAM default)	442
RECEIVE (MRO)	443

RECEIVE (2260)	443
RECEIVE (2980)	444
RECEIVE (3270 display)	446
RECEIVE (3790 3270-display)	447
RECEIVE: non-VTAM options	447
RECEIVE MAP	451
RECEIVE MAP MAPPINGDEV	455
RECEIVE PARTN	458
RELEASE	461
REMOVE SUBEVENT	463
RESET ACQPROCESS	464
RESET ACTIVITY	466
RESETBR	468
RESUME	473
RETRIEVE	475
RETRIEVE REATTACH EVENT	479
RETRIEVE SUBEVENT	481
RETURN	483
REWIND COUNTER and REWIND DCOUNTER	487
REWRITE	490
ROUTE	495
RUN	500
SEND (VTAM default)	505
SEND (APPC)	505
SEND (LUTYPE2/LUTYPE3)	506
SEND (LUTYPE4)	506
SEND (LUTYPE6.1)	507
SEND (SCS)	507
SEND (3270 logical)	508
SEND (3600 pipeline)	508
SEND (3600-3601)	509
SEND (3600-3614)	510
SEND (3650 interpreter)	510
SEND (3650-3270)	511
SEND (3650-3653)	511
SEND (3650-3680)	512
SEND (3767)	512
SEND (3770)	513
SEND (3790 full-function or inquiry)	513
SEND (3790 SCS)	514
SEND (3790 3270-display)	514
SEND (3790 3270-printer)	515
SEND: VTAM options	515
SEND (non-VTAM default)	519
SEND (MRO)	519
SEND (2260)	520
SEND (2980)	520
SEND (3270 display)	521
SEND: non-VTAM options	521
SEND CONTROL	525
SEND MAP	530
SEND MAP MAPPINGDEV	538
SEND PAGE	541
SEND PARTNSET	545
SEND TEXT	546
SEND TEXT MAPPED	553

SEND TEXT NOEDIT	556
SIGNOFF	560
SIGNON	561
SOAPFAULT ADD	565
SOAPFAULT CREATE	568
SOAPFAULT DELETE.	571
SPOOLCLOSE	572
SPOOLOPEN INPUT	574
SPOOLOPEN OUTPUT	577
SPOOLREAD	582
SPOOLWRITE	585
START	588
START ATTACH	598
START BREXIT	600
START CHANNEL	603
STARTBR	608
STARTBROWSE ACTIVITY.	614
STARTBROWSE CONTAINER	616
STARTBROWSE EVENT.	618
STARTBROWSE PROCESS	620
SUSPEND	622
SUSPEND (BTS)	623
SYNCPOINT	625
SYNCPOINT ROLLBACK	626
TEST EVENT	627
UNLOCK	628
UPDATE COUNTER and UPDATE DCOUNTER	632
VERIFY PASSWORD	636
WAIT CONVID (APPC)	639
WAIT EVENT	641
WAIT EXTERNAL	643
WAIT JOURNALNAME	646
WAIT JOURNALNUM	648
WAIT SIGNAL.	649
WAIT TERMINAL	650
WAITCICS	652
WEB CLOSE	654
WEB CONVERSE	657
WEB ENDBROWSE FORMFIELD	666
WEB ENDBROWSE HTTPHEADER	667
WEB EXTRACT	668
WEB OPEN	673
WEB PARSE URL	677
WEB READ FORMFIELD	680
WEB READ HTTPHEADER.	682
WEB READNEXT FORMFIELD	684
WEB READNEXT HTTPHEADER	686
WEB RECEIVE (Server)	688
WEB RECEIVE (Client)	693
WEB RETRIEVE.	697
WEB SEND (Server)	698
WEB SEND (Client)	706
WEB STARTBROWSE FORMFIELD	713
WEB STARTBROWSE HTTPHEADER	715
WEB WRITE HTTPHEADER	716
WRITE	719

WRITE JOURNALNAME	726
WRITE JOURNALNUM	730
WRITE OPERATOR	731
WRITEQ TD	734
WRITEQ TS	737
XCTL	742
Appendix. Detailed reference information for the CICS API commands	745
EXEC interface block	745
EIB fields	745
Codes returned by ASSIGN	762
ASSIGN TERMCODE	763
ASSIGN FCI	765
National language codes	766
Terminal control	767
Commands and options for terminals and logical units	767
TCAM-supported terminals and logical units	769
Teletypewriter programming	770
Display device operations	771
SAA Resource Recovery	773
SRRCMT	773
SRRBACK	773
Common Programming Interface Communications (CPI Communications)	774
CPI Communications language interfaces	774
API restrictions for distributed program link	774
Summary of the restricted API commands	774
API commands and distributed program link.	775
BMS-related constants	779
Magnetic slot reader (MSR) control value constants, DFHMSRCA	782
MSR control byte values	782
Attention identifier constants, DFHAID	784
BMS macros	784
Mapset, map, and field definition	784
Partition set definition	786
DFHMDF	788
DFHMDI	798
DFHMSD	807
DFHPDI	818
DFHPSD	819
Bibliography	821
The CICS Transaction Server for z/OS library	821
The entitlement set	821
PDF-only books	821
Other CICS books	823
Books from related libraries	823
MVS	823
Systems Network Architecture	823
SQL	823
Other related books.	824
Accessibility	825
Index	827
Notices	873

Programming interface information	874
Trademarks.	874

Preface

What this book is about

This book describes the CICS® Transaction Server for z/OS®, Version 3 Release 1 EXEC application programming interface. It contains **reference** information needed to prepare COBOL, C, PL/I, and assembler-language application programs, using EXEC CICS commands, to be executed under CICS. Guidance information is in the *CICS Application Programming Guide*. For information about debugging CICS applications, see the *CICS Problem Determination Guide*.

Who should read this book

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 S370 assembler language. The *CICS Application Programming Primer* and the *CICS Application Programming Guide* will help you to design and write CICS applications using the commands described in this book.

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.

What this book does not cover

The EXEC CICS commands for system programming; that is COLLECT, CREATE, DISABLE, ENABLE, INQUIRE, PERFORM, RESYNC, and SET are not covered in this book. You will find them in the *CICS System Programming Reference*.

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

The CICS C++ OO programming interface is not described in this book. It is defined in the *CICS C++ OO Class Libraries* manual.

The CICS Java™ programming interface is not described here, it is defined in Javadoc HTML provided in the CICS Information Center.

Notes on terminology

- **CICS** refers to IBM® CICS Transaction Server for z/OS, Version 3 Release 1
- **VTAM**® refers to IBM ACF/VTAM
- **IMS**™ refers to IBM IMS/ESA®
- **TCAM** refers to the DCB interface of ACF/TCAM.

Summary of changes

This book is based on the CICS Application Programming Reference for CICS Transaction Server for z/OS, Version 2 Release 3.

Changes for CICS Transaction Server for z/OS, Version 3 Release 1

The more significant changes for this edition are:

- New commands:
 - INVOKE WEBSERVICE
 - DELETE CONTAINER (CHANNEL)
 - GET CONTAINER (CHANNEL)
 - MOVE CONTAINER (CHANNEL)
 - PUT CONTAINER (CHANNEL)
 - SOAPFAULT ADD
 - SOAPFAULT CREATE
 - SOAPFAULT DELETE
 - START TRANSID CHANNEL
 - WEB CLOSE
 - WEB CONVERSE
 - WEB OPEN
 - WEB PARSE URL
 - WEB SEND (Client)
 - WEB RECEIVE (Client)
- New options:
 - CHANNEL added to:
 - ASSIGN
 - LINK
 - RETURN
 - STARTBROWSE CONTAINER
 - XCTL
 - SESSTOKEN (marker for CICS as an HTTP client) added to:
 - WEB EXTRACT
 - WEB SEND
 - WEB RECEIVE
 - WEB READ HTTPHEADER
 - WEB WRITE HTTPHEADER
 - WEB STARTBROWSE HTTPHEADER
 - WEB READNEXT HTTPHEADER
 - WEB ENDBROWSE HTTPHEADER
 - HOST, HOSTLENGTH, PORTNUMBER, SCHEME and URIMAP added to WEB EXTRACT
 - ACTION, CHARACTERSET, CLOSESTATUS, CHUNKLENGTH, FROM, FROMLENGTH, HOSTCODEPAGE, MEDIATYPE, SERVERCONV, STATUSLEN added to WEB SEND for CICS as an HTTP server
 - CHARACTERSET and SERVERCONV added to WEB RECEIVE for CICS as an HTTP server
- Existing commands that have been made threadsafe:
 - WEB ENDBROWSE FORMFIELD
 - WEB ENDBROWSE HTTPHEADER
 - WEB EXTRACT
 - WEB READ FORMFIELD
 - WEB READ HTTPHEADER
 - WEB READNEXT FORMFIELD

- WEB READNEXT HTTPHEADER
- WEB RECEIVE
- WEB RETRIEVE
- WEB SEND
- WEB STARTBROWSE FORMFIELD
- WEB STARTBROWSE HTTPHEADER
- WEB WRITE HTTPHEADER
- For several CICS releases, BTAM terminals have been supported only indirectly; that is, by transaction routing from a back-level terminal-owning region to which the terminals were attached. In CICS Transaction Server for z/OS, Version 3 Release 1, this indirect support is removed. BTAM is no longer supported and the following BTAM-specific commands have been removed:
 - CONVERSE (SYSTEM/3)
 - CONVERSE (SYSTEM/7)
 - CONVERSE (2741)
 - CONVERSE (2770)
 - CONVERSE (2780)
 - CONVERSE (3600 BTAM)
 - CONVERSE (3735)
 - CONVERSE (3740)
 - ISSUE COPY (3270 display)
 - RECEIVE (SYSTEM/3)
 - RECEIVE (SYSTEM/7)
 - RECEIVE (2741)
 - RECEIVE (3600 BTAM)
 - RECEIVE (3735)
 - RECEIVE (3740)
 - SEND (SYSTEM/3)
 - SEND (SYSTEM/7)
 - SEND (2741)
 - SEND (3600 BTAM)
 - SEND (3735)
 - SEND (3740)
- The descriptions of the following CICS business transaction services (BTS) application programming commands have been moved to this manual from the *CICS Business Transaction Services* manual:
 - ACQUIRE
 - ADD SUBEVENT
 - CANCEL (BTS)
 - CHECK ACQPROCESS
 - CHECK ACTIVITY
 - CHECK TIMER
 - DEFINE ACTIVITY
 - DEFINE COMPOSITE EVENT
 - DEFINE INPUT EVENT
 - DEFINE PROCESS
 - DEFINE TIMER
 - DELETE ACTIVITY
 - DELETE CONTAINER (BTS)
 - DELETE EVENT
 - DELETE TIMER
 - ENDBROWSE ACTIVITY
 - ENDBROWSE CONTAINER
 - ENDBROWSE EVENT
 - ENDBROWSE PROCESS

- FORCE TIMER
- GET CONTAINER (BTS)
- GETNEXT ACTIVITY
- GETNEXT CONTAINER
- GETNEXT EVENT
- GETNEXT PROCESS
- INQUIRE ACTIVITYID
- INQUIRE CONTAINER
- INQUIRE EVENT
- INQUIRE PROCESS
- INQUIRE TIMER
- LINK ACQPROCESS
- LINK ACTIVITY
- MOVE CONTAINER (BTS)
- PUT CONTAINER (BTS)
- REMOVE SUBEVENT
- RESET ACQPROCESS
- RESET ACTIVITY
- RESUME
- RETRIEVE REATTACH EVENT
- RETRIEVE SUBEVENT
- RUN
- STARTBROWSE ACTIVITY
- STARTBROWSE CONTAINER
- STARTBROWSE EVENT
- STARTBROWSE PROCESS
- SUSPEND (BTS)
- TEST EVENT

Changes for CICS Transaction Server for z/OS, Version 2 Release 3

The more significant changes for this edition are:

- New options:
 - PRIVACY added to EXTRACT TCPIP
- Commands that are now threadsafe:
 - ASKTIME
 - CHANGE TASK
 - DOCUMENT CREATE
 - DOCUMENT INSERT
 - DOCUMENT RETRIEVE
 - DOCUMENT SET
 - FORMATTIME

Changes for CICS Transaction Server for z/OS, Version 2 Release 2 and CICS Transaction Server for z/OS, Version 2 Release 1

The more significant changes for these editions were:

- New commands:
 - WEB ENDBROWSE FORMFIELD
 - WEB READ FORMFIELD
 - WEB READNEXT FORMFIELD
 - WEB STARTBROWSE FORMFIELD
- New options
 - DELIMITER and UNESCAPED added to DOCUMENT CREATE

- DELIMITER and UNESCAPED added to DOCUMENT SET
- AUTHENTICATE added to EXTRACT TCPIP
- QUERYSTRING and QUERYSTRLEN added to WEB EXTRACT

Changes for the CICS Transaction Server for OS/390 Version 1 Release 3 edition

A new option, BREXIT, has been added to the START command to define a 3270 Bridge exit. To simplify use of the START command, it is now documented as 3 variations: START ATTACH, START BREXIT and the original Interval Control START. This change shows the limited number of options that you can use with BREXIT and ATTACH.

New DOCUMENT and WEB commands have been added for use with CICS Web support.

EXTRACT TCPIP has been added to support the CICS TCP/IP listener.

EXTRACT CERTIFICATE has been added to support the Secure Sockets Layer (SSL).

If the name specified in the RESOURCE option of an ENQ or DEQ command matches the name of an installed ENQMODEL resource, the ENQSCOPE attribute of the resource definition controls the scope of the ENQ or DEQ command, that is, whether it has local or sysplex scope. The syntax of the ENQ or DEQ command is not changed.

Changes have been made to the PICIN and PICOUT options of the BMS macro DFHMDF to support multi-character currency symbols, such as EUR.

Figures

1. Translated code for a CICS command	13
2. Translated code for user variables	14
3. Examples of coding user EMPs	362
4. ECBLIST option, EXEC CICS WAIT EXTERNAL.	645
5. ECBLIST option, EXEC CICS WAITCICS	654
6. Bytes 1 and 2 of EIBRCODE for SYSIDERR	756
7. Bytes 1 and 2 of EIBRCODE for SESSIONERR	756
8. Byte 3 of EIBRCODE for SYSBUSY	757
9. Bytes 1 or 3 of EIBRCODE for INVREQ.	757
10. Byte 1 of EIBRCODE for LENGERR	758
11. EIBRCODE for ILLOGIC or IOERR	758
12. Byte 3 of EIBRCODE for INVMPSZ	758

Tables

1. GDS ALLOCATE return codes	220
2. GDS ASSIGN return codes	222
3. GDS CONNECT PROCESS return codes	224
4. GDS EXTRACT ATTRIBUTES return codes	227
5. GDS EXTRACT PROCESS return codes	229
6. GDS FREE return codes	231
7. GDS ISSUE ABEND return codes	233
8. GDS ISSUE CONFIRMATION return codes	235
9. GDS ISSUE ERROR return codes	237
10. GDS ISSUE PREPARE return codes	239
11. GDS ISSUE SIGNAL return codes	241
12. GDS RECEIVE return codes	244
13. GDS SEND return codes	247
14. GDS WAIT return codes.	249
15. Data-key options on GETMAIN command	262
16. Symbolic identifier of resource being accessed	761
17. CICS language suffixes	766
18. Other IBM language codes.	767
19. Restricted API commands	774
20. Summary of the CICS API by functional area	775
21. Standard attribute and printer control character list, DFHBMSCA.	780
22. Bit map for attributes	781
23. Key to attributes and settings in Bit Map.	782
24. Standard list DFHMSRCA	783
25. Standard list DFHAID.	784
26. BMS terminal types	815

CICS API commands

This section 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 system programming interface (SPI) are primarily for the use of the system programmer; they are not described here. For details of these commands, refer to the *CICS System Programming Reference*.

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

About the CICS API commands

This section contains general information which applies to all the CICS API commands.

CICS API 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. 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 command syntax notation.”

Note: If you want to add comments against CICS commands, you can do this, in assembler only, by using a period or a comma as a delimiter after the last argument. For example:

```
EXEC CICS ADDRESS EIB(MYUEIB), @F1A
```

CICS command syntax notation

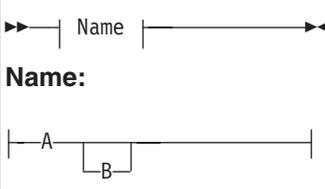
In CICS documentation, CICS commands are presented in a standard way.

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 must code. You may code more than one of them, in any sequence.
	A set of alternatives—one of which you may code.
	A set of alternatives — any number (including none) of which you may code once, in any sequence.
	Alternatives where A is the default.

Symbol	Action
	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).

CICS command 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
- hmmmss
- filename
- systemname

Data-areas and Data-values

Data-areas and data-values are the basic argument types. The difference between them is the direction in which information flows when a task executes a command. A **data-value** is always, and *exclusively* a sender; it conveys data to CICS that CICS uses to process the command. A **data-area** is a receiver; CICS uses it to return information to the caller. Note that a data-area can also be a sender, for example when the data to be conveyed to CICS is variable length (as in FROM), or where a field is used both for input and output.

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
 - Doubleword *unsigned* binary — PIC 9(18) 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
- Doubleword *unsigned* binary — PIC 9(18) 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 16.
- “ptr-value” can be replaced by a pointer variable or ADDRESS special register.
- “ptr-ref” can be replaced by a pointer variable or ADDRESS special register.
- “name” can be replaced by either of the following:
 - A character string specified as an alphanumeric literal. If this is shorter than the required length, it is padded with blanks.
 - A COBOL data area with a length equal to the required length 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, which might lead to unpredictable results.

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

“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 #.

- “label” can be replaced by any COBOL paragraph name or a section name.
- “hhmmss” 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 COBOL, 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
 - Doubleword binary — char[8]
 - 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
 - Doubleword binary — char[8]
 - 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 16.
 - “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 #.
- “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 #.
- “label” is not supported in the C language.
 - “hhmmss” 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)
 - Doubleword binary — CHAR (8)
 - 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)
 - Doubleword binary — CHAR (8)
 - 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 16.
- “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 #.

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

- “label” can be replaced by any PL/I expression whose value is a label.
- “hhmmss” 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 16.
- “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 #.
“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 #.
- “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)
```

- “hhmss” 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 0HHMSS+ 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)
```

CICS command restrictions

The following general restrictions apply to all CICS commands:

The restrictions that apply to CICS commands that access user data:

- 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.

LENGTH options in CICS commands

In COBOL, PL/I, and assembler language, the translator defaults certain lengths, if the NOLENGTH translator option is not specified. This means they are optional in programs that specify data areas.

In 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.

This advisory 24KB limit does not apply to the FLENGTH option on CICS
commands (except for terminal-related SEND and RECEIVE commands, due to architectural limitations). The FLENGTH option is used on commands relating to containers and journals, among others.

For **temporary storage**, **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 Application Programming Guide*.

Note that using the C or 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.

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 translation output

EXEC CICS commands are converted to calls to the CICS interface DFHEI1.

The following example shows how the EXEC statement:
is translated to:

```
EXEC CICS RETURN TRANSID('fred')
      COMMAREA(mycommarea) END-EXEC.
```

```
Move length of mycommarea to dfhb0020
Call 'DFHEI1' using by content
      x'0e08e0000700001000f0f0f0f2f7404040'
      by content 'fred' by reference mycommarea
      by reference dfhb0020 end-call.
```

Copybook DFHEIBLC: This new copybook is a lower case version of the existing DFHEIBLK copybook.

A difference is that in DFHEIBLK the top level name is

```
01 EIBLK.
```

whereas in DFHEIBLC the top level name is

```
01 dfheiblk.
```

This is consistent with the name generated by the translator today, and also conforms to the rule that CICS reserved words should start with DFH.

C translation output

For a C application program, each command is replaced by reassignment statements followed by a dfhexec statement that passes the parameters.

PL/I translation output

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 translation output

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.

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 an EXEC CICS 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 1 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$AGA') MAPONLY ERASE
        DFHECALL =X'1804C00008000000000046204000020',
              (CHA7,=CL7'DFH$AGA*'),(____RF,DFHEIV00)
        DFHEIRET           INSERTED BY TRANSLATOR
        DFHEISTG           INSERTED BY TRANSLATOR
        DFHEIEND           INSERTED BY TRANSLATOR
        END
```

Figure 1. 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 65 264 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. At link-edit time, use the STORAGE option of the CEEXOPT macro to ensure that the DFHEISTG storage is initialized to x'00', for example CEEXOPT STORAGE=(, ,00). Make sure that your application propagates or initializes any constants that are defined in the user DFHEISTG area.

#

The example in Figure 2 on page 14 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 2. 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 in place of the first CSECT or START instruction in your source program. The operands you can use to specify base registers are as follows:

- CODEREG - base registers.
- DATAREG - dynamic-storage registers.
- EIBREG - register to address the EIB.

For example, the source code shown in Figure 1 on page 13 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.

You can also use the DFHEIENT macro to specify that you want to use relative addressing instructions in your program. When you use relative addressing, you do not need to use *any* base registers to address your program instructions, but you do need to use at least one base register to address static data within your program. You should specify the following operands on the DFHEIENT macro:

- CODEREG=0 to specify that no registers are to be used to address program instructions.
- STATREG to specify one or more registers to address the static data area in your program.
- STATIC to specify the address of the start of the static data within your program.

If you use relative addressing you should also include a COPY statement for one of the copybooks DFHKEBRC (provided by CICS) or IEABRC (provided by z/OS) to redefine the assembler mnemonics for branch instructions to use relative branch instructions. You should also ensure that any LTORG statements, and instructions that are the target of EXECUTE instructions, appear after the label specified in the STATIC operand. For example:

```

COPY DFHKEBRC          Define relative branch mnemonics
RELATIVE DFHEIENT CODEREG=0,STATREG=(8,9),STATIC=MYSTATIC
....
EX      R2,VARMOVE      Execute instruction in static area
....

MYSTATIC DS    0D          Static data area
MYCONST DC    C'constant'  Static data value
VARMOVE MVC   WORKA(0),WORKB Executed instruction
          LTORG ,          Literal pool

```

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.

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) ....
```

CICS System Programming Reference lists the CVDA values and their numeric equivalents.

CICS threadsafe commands in the API

If your application program is defined as threadsafe, it can receive control on an open transaction environment (OTE) TCB. This happens if a program in the task issues a DB2® SQL request that causes CICS to pass control to the CICS DB2 adaptor on an L8 open TCB. Although the task is attached and runs initially on the CICS QR TCB, CICS switches it to an L8 TCB for the execution of the DB2 request. If you define the application program issuing the SQL request as threadsafe, CICS leaves the task running on the L8 open TCB on return from DB2, to avoid a costly TCB switch. For more information, see the *CICS DB2 Guide*.

To obtain the maximum performance benefit from OTE, write your CICS DB2 application programs in a threadsafe manner to avoid CICS having to switch TCBs. However, be aware that not all EXEC CICS commands are threadsafe, and issuing any of the non-threadsafe commands causes CICS to switch your task back to the QR TCB to ensure serialization. The commands that are threadsafe are indicated in the command syntax diagrams in this programming reference with the statement: “This command is threadsafe”, and are listed in “Threadsafe command list” on page 17.

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

Threadsafe command list

Not all EXEC CICS commands are threadsafe, and issuing any of the non-threadsafe commands causes CICS to use the QR TCB to ensure serialization. The commands that are threadsafe are indicated in the command syntax diagrams in this programming reference with the statement: “This command is threadsafe”, and are listed here. See “CICS threadsafe commands in the API” on page 16 for more guidance on the use of threadsafe commands and see the *CICS Application Programming Guide* for information about writing threadsafe application programs.

Threadsafe commands:

- ABEND
- ADDRESS
- ASKTIME
- ASSIGN
- CHANGE TASK
- CONVERTTIME
- DELETEQ TS
- DEQ (This command is threadsafe if it is defined as LOCAL. It is non-threadsafe if it is defined as GLOBAL.)
- DOCUMENT CREATE
- DOCUMENT INSERT
- DOCUMENT RETRIEVE
- DOCUMENT SET
- ENQ (This command is threadsafe if it is defined as LOCAL. It is non-threadsafe if it is defined as GLOBAL.)
- ENTER TRACENUM
- FORMATTIME
- FREEMAIN
- GETMAIN
- HANDLE ABEND
- HANDLE AID
- HANDLE CONDITION
- IGNORE CONDITION
- INVOKE WEBSERVICE
- LINK (This command is threadsafe when it is used to link to a program in a local CICS region. It is non-threadsafe when it is used to link to a program in a remote CICS region.)
- LOAD
- MONITOR
- POP HANDLE
- PUSH HANDLE
- READQ TS
- RELEASE
- RETURN
- SOAPFAULT ADD

#

#

- #
- #
- SOAPFAULT CREATE
- SOAPFAULT DELETE
- SUSPEND
- WAIT EXTERNAL
- WEB CLOSE
- WEB CONVERSE
- WEB ENDBROWSE FORMFIELD
- WEB ENDBROWSE HTTPHEADER
- WEB EXTRACT
- WEB OPEN
- WEB PARSE URL
- WEB READ FORMFIELD
- WEB READ HTTPHEADER
- WEB READNEXT FORMFIELD
- WEB READNEXT HTTPHEADER
- WEB RECEIVE
- WEB RETRIEVE
- WEB SEND
- WEB STARTBROWSE FORMFIELD
- WEB STARTBROWSE HTTPHEADER
- WEB WRITE HTTPHEADER
- WRITEQ TS
- XCTL

CICS command summary

This list shows the 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 MAP MAPPINGDEV
- RECEIVE PARTN
- ROUTE
- SEND CONTROL
- SEND MAP
- SEND MAP MAPPINGDEV
- SEND PAGE
- SEND PARTNSET
- SEND TEXT
- SEND TEXT MAPPED
- SEND TEXT NOEDIT

Built-in functions

- BIF DEEDIT

CICS business transaction services (BTS)

ACQUIRE
ADD SUBEVENT
CANCEL
CHECK ACQPROCESS
CHECK ACTIVITY
CHECK TIMER
DEFINE ACTIVITY
DEFINE COMPOSITE EVENT
DEFINE INPUT EVENT
DEFINE PROCESS
DEFINE TIMER
DELETE ACTIVITY
DELETE CONTAINER (BTS)
DELETE EVENT
DELETE TIMER
ENDBROWSE ACTIVITY
ENDBROWSE CONTAINER
ENDBROWSE EVENT
ENDBROWSE PROCESS

FORCE TIMER
GET CONTAINER (BTS)
GETNEXT ACTIVITY
GETNEXT CONTAINER
GETNEXT EVENT
GETNEXT PROCESS
INQUIRE ACTIVITYID
INQUIRE CONTAINER
INQUIRE EVENT
INQUIRE PROCESS
INQUIRE TIMER
LINK ACQPROCESS
LINK ACTIVITY
MOVE CONTAINER (BTS)
PUT CONTAINER (BTS)
REMOVE SUBEVENT
RESET ACQPROCESS
RESET ACTIVITY
RESUME
RETRIEVE REATTACH EVENT
RETRIEVE SUBEVENT
RUN
STARTBROWSE ACTIVITY
STARTBROWSE CONTAINER
STARTBROWSE EVENT
STARTBROWSE PROCESS
SUSPEND (BTS)
TEST EVENT

Channel commands

DELETE CONTAINER (CHANNEL)
GET CONTAINER (CHANNEL)
MOVE CONTAINER (CHANNEL)
PUT CONTAINER (CHANNEL)
START TRANSID (CHANNEL)

Console support

- WRITE OPERATOR

Diagnostic services

- DUMP TRANSACTION
- ENTER TRACENUM

Document services

- DOCUMENT CREATE
- DOCUMENT INSERT
- DOCUMENT RETRIEVE
- DOCUMENT SET

Environment services

- ADDRESS
- ADDRESS SET
- ASSIGN

Exception support

- HANDLE CONDITION
- IGNORE CONDITION
- POP HANDLE
- PUSH HANDLE

File control services

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

Interval control services

- ASKTIME
- CANCEL
- DELAY
- FORMATTIME
- POST
- RETRIEVE
- START
- WAIT EVENT

Journaling

- WAIT JOURNALNAME
- WAIT JOURNALNUM
- WRITE JOURNALNAME
- WRITE JOURNALNUM

Monitoring

- MONITOR

Named counter server

- DEFINE COUNTER
- DEFINE DCOUNTER
- DELETE COUNTER
- DELETE DCOUNTER
- GET COUNTER
- GET DCOUNTER
- QUERY COUNTER
- QUERY DCOUNTER
- REWIND COUNTER

- REWIND DCOUNTER
- UPDATE COUNTER
- UPDATE DCOUNTER
-

Program control

- LINK
- LOAD
- RELEASE
- RETURN
- XCTL

Scheduling services

- START ATTACH
- START BREXIT

Security services

- 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

TCP/IP services

- EXTRACT CERTIFICATE
- EXTRACT TCPIP

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 (2260)
- CONVERSE (3270 display)
- CONVERSE (3270 logical)
- CONVERSE (3600-3601)
- CONVERSE (3600-3614)
- CONVERSE (3650 interpreter)
- CONVERSE (3650-3270)
- CONVERSE (3650-3653)
- CONVERSE (3650-3680)
- 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 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 (2260)
- RECEIVE (2770)
- RECEIVE (2780)
- RECEIVE (2980)
- RECEIVE (3270 display)
- RECEIVE (3270 logical)
- RECEIVE (3600-3601)
- RECEIVE (3600-3614)
- RECEIVE (3650)
- 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 (2260)
- SEND (2770)
- SEND (2880)
- SEND (2980)
- SEND (3270 display)
- SEND (3270 logical)
- SEND (3600 pipeline)
- SEND (3600-3601)
- SEND (3600-3614)
- SEND (3650 interpreter)
- SEND (3650-3270)
- SEND (3650-3653)
- SEND (3650-3680)
- 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

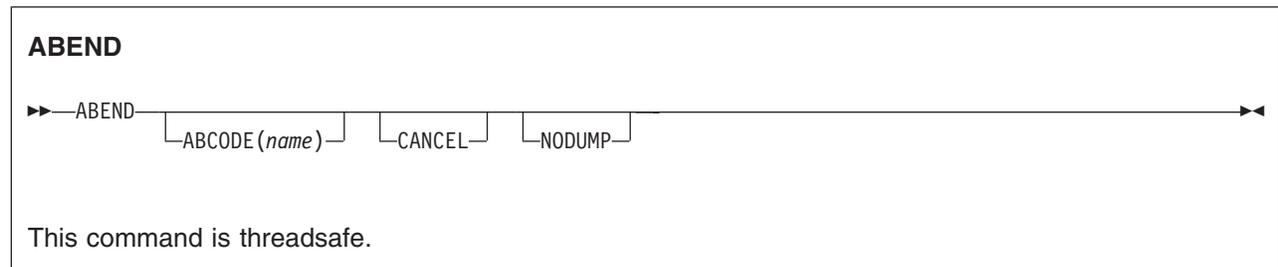
- DELETEDQ TD
- READQ TD
- WRITEQ TD

Web support

- CONVERTTIME
- WEB CLOSE
- WEB CONVERSE
- WEB ENDBROWSE FORMFIELD
- WEB ENDBROWSE HTTPHEADER
- WEB EXTRACT
- WEB OPEN
- WEB PARSE URL
- WEB READ FORMFIELD
- WEB READ HTTPHEADER
- WEB READNEXT FORMFIELD
- WEB READNEXT HTTPHEADER
- WEB RECEIVE (Server and Client versions)
- WEB RETRIEVE
- WEB SEND (Server and Client versions)
- WEB STARTBROWSE FORMFIELD
- WEB STARTBROWSE HTTPHEADER
- WEB WRITE HTTPHEADER

ABEND

Terminate a task abnormally.



Description

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.

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. ABCODE follows the format rules for DUMPCODE. The EXEC CICS DUMP TRANSACTION command gives the format rules that apply to DUMPCODE, if these rules are not followed, ABEND does not produce a dump.

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

Note: If ABCODE is not used, the effect is the same as NODUMP.

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

#

allows you to request an abend without causing a dump to be taken. For programs link-edited using the Language Environment® SCEELKED library, when NODUMP is specified, a dump is never taken, regardless of any setting in the transaction dump table. For programs not link-edited with Language Environment, if the transaction dump table already has an entry for the abend code, or if the abend is in Language Environment run-unit initialization or termination, the NODUMP option is ignored.

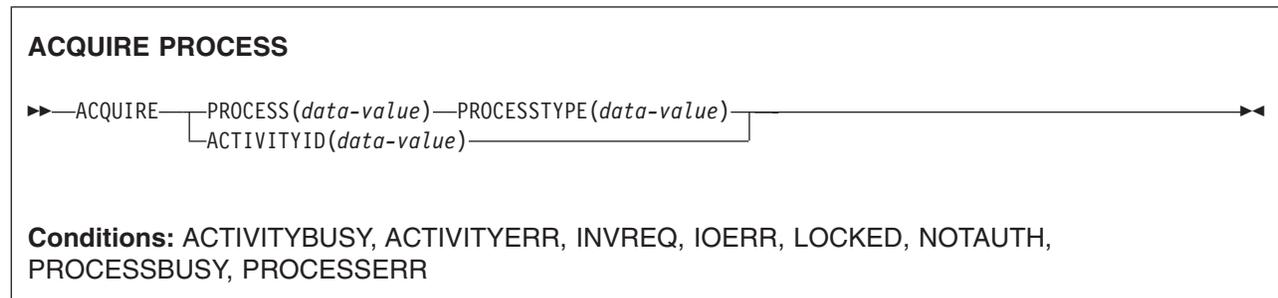
Examples

The following example shows how to terminate a task abnormally:

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

ACQUIRE

Acquire access to a BTS activity from outside the process that contains it.



Description

ACQUIRE enables a program that is executing outside a particular BTS process to access an activity within the process. It allows the program to:

- Read and write to the activity's data-containers
- Issue various commands, such as RUN and LINK, against the activity.¹

An activity that a program gains access to by means of an ACQUIRE command is known as an **acquired activity**. A program can acquire only one activity per unit of work. The activity remains acquired until the next syncpoint.

ACQUIRE ACTIVITYID acquires the specified descendant (non-root) activity.

ACQUIRE PROCESS acquires the root activity of the specified process.

Note: When a program defines a process, it is automatically given access to the process's root activity. (This enables the defining program to access the process containers and root activity containers before running the process.) When a program gains access to a root activity by means of *either* a DEFINE PROCESS or an ACQUIRE PROCESS command, the process is known as the **acquired process**.

Rules

1. A program can acquire only one activity within the same unit of work. The activity remains acquired until the next syncpoint. This means, for example, that a program:
 - Cannot issue both a DEFINE PROCESS and an ACQUIRE PROCESS command within the same unit of work.
 - Cannot issue both an ACQUIRE PROCESS and an ACQUIRE ACTIVITYID command within the same unit of work. That is, it can acquire *either* a descendant activity or a root activity, not one of each.
2. If a program is executing as an activation of an activity, it cannot:
 - Acquire an activity in the same process as itself. It cannot, for example, issue ACQUIRE PROCESS for the current process.
 - Use a LINK command to activate the activity that it has acquired.

1. If the acquired activity is a root activity, against the process.

3. An acquired activity's process is accessible in the same way as the activity itself can access it. Thus, if the acquired activity is a descendant activity:
 - Its process's containers may be read but not updated.
 - The process may not be the subject of any command—such as RUN, LINK, SUSPEND, RESUME, or RESET—that directly manipulates the process or its root activity.

Conversely, if the acquired activity is a root activity:

- Its process's containers may be both read and updated.
- The process may be the subject of commands such as RUN, LINK, SUSPEND, RESUME, or RESET. The ACQPROCESS keyword on the command identifies the subject process as the one the program that issues the command has acquired in the current unit of work.

Options

ACTIVITYID(data-value)

specifies the identifier (1–52 characters) of the descendant activity to be acquired.

PROCESS(data-value)

specifies the name (1–36 characters) of the process whose root activity is to be acquired.

PROCESSTYPE(data-value)

specifies the process-type (1–8 characters) of the process whose root activity is to be acquired.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19 The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8 The activity referred to by the ACTIVITYID option could not be found.

INVREQ

RESP2 values:

- 22 The unit of work that issued the ACQUIRE command has already acquired an activity; a unit of work can acquire only one activity.

IOERR

RESP2 values:

- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

NOTAUTH

RESP2 values:

- 101** The user associated with the issuing task is not authorized to access the file associated with the BTS repository data set on which details of the process are stored.

PROCESSBUSY

RESP2 values:

- 13** The request timed out. It may be that another task using this process-record has been prevented from ending.

PROCESSERR

RESP2 values:

- 5** The process named in the PROCESS option could not be found.
- 9** The process-type named in the PROCESSTYPE option could not be found.

Usage examples

ACQUIRE ACTIVITYID can be used to implement user-related activities. For example, on its first activation an activity might:

1. Define an input event to represent a particular user-interaction
2. Issue an ASSIGN command to obtain the identifier of its own activity-instance
3. Save the input event and activity identifier on a data base
4. Return without completing.

Later, when a user is ready to process the work represented by the activity, he or she starts a transaction. This transaction, which executes outside the BTS process:

1. Retrieves the input event and activity identifier from the data base
2. Uses the ACQUIRE ACTIVITYID command to acquire access to the activity
3. Places the information required to complete the activity in an input data-container, and runs the activity. The INPUTEVENT option of the RUN command tells the activity why it is being activated.

For an example of the use of ACQUIRE ACTIVITYID, see the *CICS Business Transaction Services* manual.

ACQUIRE PROCESS can be used to implement client/server processing. For example, a client program might use the DEFINE PROCESS and RUN commands to create and run a server process, which carries out some work, defines one or more input events, and returns without completing. The client issues a syncpoint or returns. To run the same server process again, the client uses the ACQUIRE PROCESS and RUN commands.

For an example of the use of ACQUIRE PROCESS, see the *CICS Business Transaction Services* manual.

ADD SUBEVENT

Add a sub-event to a BTS composite event.

ADD SUBEVENT

▶—ADD—SUBEVENT(*data-value*)—EVENT(*data-value*)—▶

Conditions: EVENTERR, INVREQ

Description

ADD SUBEVENT adds a sub-event to a BTS composite event. The sub-event:

- Must be an atomic (not a composite) event
- Cannot be a system event
- Must not currently be part of a composite event
- Cannot, if the predicate of the composite event uses the AND Boolean operator, be an input event.

Adding a sub-event causes the composite's predicate to be re-evaluated.

Options

EVENT(*data-value*)

specifies the name (1–16 characters) of the composite event. This must previously have been defined to the current activity, using the DEFINE COMPOSITE EVENT command.

SUBEVENT(*data-value*)

specifies the name (1–16 characters) of the atomic event to be added to the composite event as a sub-event. The sub-event must previously have been defined to the current activity, using one of the following commands:

- DEFINE ACTIVITY
- DEFINE INPUT EVENT
- DEFINE TIMER

It:

- Must not currently be part of a composite event
- Cannot, if the predicate of the composite event uses the AND Boolean operator, be an input event.

Conditions

EVENTERR

RESP2 values:

- 4** The event specified on the EVENT option is not recognized by BTS.
- 5** The sub-event specified on the SUBEVENT option is not recognized by . BTS.

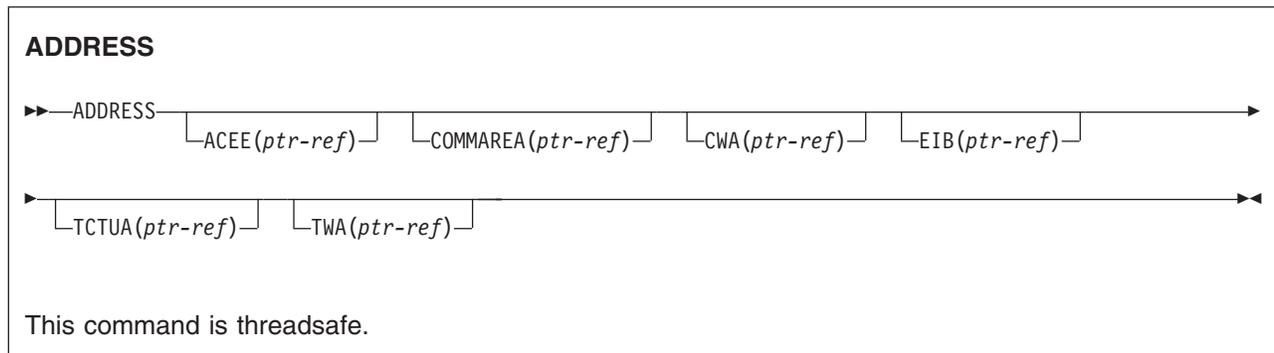
INVREQ

RESP2 values:

- 1 The command was issued outside the scope of an activity.
- 2 The event specified on the EVENT option is invalid—it is not a composite event.
- 3 The sub-event specified on the SUBEVENT option is invalid. Specifying any of the following as a sub-event produces this error:
 - A composite event
 - A system event
 - A sub-event of another composite event
 - A sub-event of this composite event—that is, an atomic event that has already been added to this composite event
 - An input event, if the composite uses the AND Boolean operator.

ADDRESS

Obtain access to CICS storage areas.



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 Application Programming Guide* for more information about transaction affinities.

Description

ADDRESS accesses the following areas:

- The access control environment element (ACEE)
- The communication area available to the invoked program (COMMAREA)
- 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.

Options

ACEE(*ptr-ref*)

returns 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'.

For information on how to map the ACEE data area, see the mapping macro IHAACEE supplied in SYS1.MACLIB.

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*)

returns a pointer reference, set to the address of the communication area (COMMAREA) available to the currently executing program. COMMAREA is

used to pass information between application programs. 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*)

returns a pointer reference, set to the address of the common work area (CWA). This area makes information available to applications running in a single CICS system. If a CWA does not exist, CICS sets the pointer reference to the null value, X'FF000000'.

EIB (*ptr-ref*)

returns a pointer reference set to the address of the EXEC interface block (EIB). You must use this option to get addressability to the EIB 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 are using a command that includes the RESP or RESP2 option.

TCTUA (*ptr-ref*)

returns a pointer reference, set to the address of the terminal control table user area (TCTUA) for the principal facility, not that for any alternate facility that may have been allocated. This area is used for passing information between application programs, but only if the same terminal is associated with the application programs involved. If a TCTUA does not exist, the pointer reference is set to the null value, X'FF000000'.

TWA (*ptr-ref*)

returns a pointer reference, set to the address of the transaction work area (TWA). This area is used for passing information between application programs, but only if they are in the same task. 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 SET

Set the address of a structure or pointer.

ADDRESS SET



Description

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

Options

SET (*data-area/ptr-ref*)
sets a pointer reference.

USING (*ptr-ref/data-area*)
supplies a pointer value.

COBOL example of ADDRESS SET

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

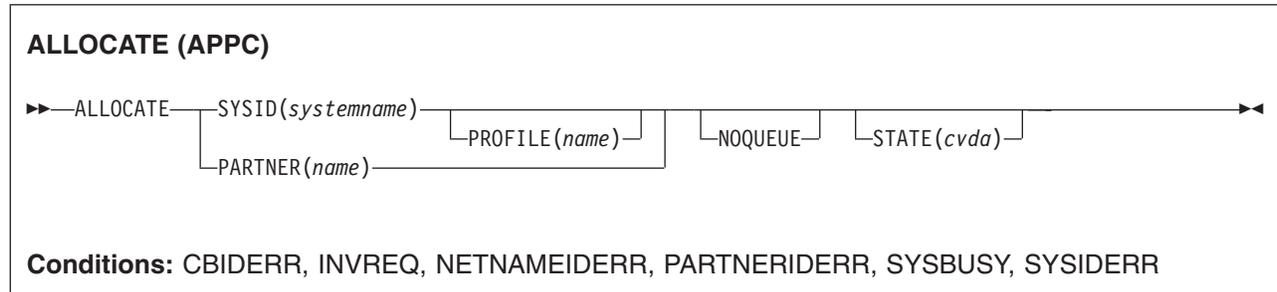
```
EXEC CICS ADDRESS SET(address of struc)
        USING(ptr)
```

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

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

ALLOCATE (APPC)

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



Description

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

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

If a session to the requested APPC LU is not available, the application is suspended until a 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 immediately available for allocation only if it is all of the following:

- A contention winner
- Already bound
- Not already allocated

CICS attempts to satisfy a request for a session by choosing among sessions in the following order of preference:

- # 1. Contention winner that is bound and not already allocated (CICS allocates it).
This is a session that is immediately available.
- # 2. Contention winner that is unbound (CICS binds it and allocates it).
- # 3. Contention loser that is bound and not already allocated (CICS bids for it).
- # 4. Contention loser that is unbound (CICS binds it and bids for it).

The action taken by CICS if no session is 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. In these situations, CICS does not bid for sessions or bind additional sessions. It looks for a session that is immediately available (that is, a contention

winner that is bound and not already allocated), and if one is not available, the SYSBUSY condition is returned. 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.

No HANDLE issued for SYSBUSY condition

If you have specified NOQUEUE (or NOSUSPEND), 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.

The HANDLE for SYSBUSY condition therefore has the same effect as the NOQUEUE option, except for where control is returned in the application. If the HANDLE command is used, control is returned to the label, and if it is not used, control is returned to the instruction following the ALLOCATE command.

If you have omitted the NOQUEUE option, and you have not issued a HANDLE command for the SYSBUSY option, then if no session is immediately available, CICS queues the request (and your application waits) until a session is available. The request is allocated a session either when a winner session has become available, or when CICS has successfully bid for a loser session. Omit the NOQUEUE option when you want all winner or loser sessions to be considered for allocation to the request. You can use the QUEUELIMIT and MAXQTIME attributes of the CONNECTION resource definitions to limit the length of the queue of requests, and the time that requests spend on the queue. *CICS Intercommunication Guide* has more information about allocate queues. The DTIMOUT value specified on the transaction definition can be used to limit the wait time for individual requests.

Options

NOQUEUE

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

Note, however, that if a HANDLE CONDITION for SYSBUSY is active when the command is executed, this also overrides the default action, and control is passed to the user label supplied in the HANDLE CONDITION. This takes precedence over the NOQUEUE option but is, of course, negated by either NOHANDLE or RESP.

If an APPC ALLOCATE request is issued against a single session connection from the contention loser end, the NOQUEUE option always causes SYSBUSY to be returned rather than allowing the request to bid for the session. If the NOQUEUE option is absent, the request is able to bid for the session.

If an APPC ALLOCATE request is issued against a parallel session connection, and the NOQUEUE option is specified, only sessions that are immediately available (that is, a contention winner that is bound and not already allocated) can be allocated to the request. If no such session is available, then SYSBUSY is returned. If the NOQUEUE option is absent, the request is able to bid for a loser session, or bind unbound winner sessions.

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.

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, or a HANDLE CONDITION for SYSBUSY is active.
- 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.
- All the sessions are busy and the (queued) allocates have been purged or rejected.

Default action: terminate the task abnormally.

ALLOCATE (LUTYPE6.1)

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

ALLOCATE (LUTYPE6.1)

▶—ALLOCATE—SESSION(*name*)—
 SYSID(*systemname*)—
 PROFILE(*name*)—
 NOQUEUE—▶

Conditions: CBIDERR, EOC, INVREQ, SESSBUSY, SESSIONERR, SYSBUSY, SYSIDERR

Description

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.

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.

Note, however, that if a HANDLE CONDITION for SESSBUSY or SYSBUSY is active when the command is executed, this also overrides the default action, and control is passed to the user label supplied in the HANDLE CONDITION. This takes precedence over the NOQUEUE option but is, of course, negated by either NOHANDLE or RESP.

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.

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 the NOQUEUE option is set, or a HANDLE CONDITION for SESSBUSY is active.

Default action: ignore the condition.

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 the NOQUEUE option is set, or a HANDLE CONDITION for SYSBUSY is active.
- 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.
- All the sessions are busy and the (queued) allocates have been purged or rejected.

Default action: terminate the task abnormally.

ALLOCATE (MRO)

Acquire an MRO session.

ALLOCATE (MRO)

```
▶▶—ALLOCATE—SYSID(systemname)—┐┐—┐┐—┐┐—▶▶  
                                └─PROFILE(name)─┘ └─NOQUEUE─┘ └─STATE(cvda)─┘
```

Conditions: INVREQ, SYSBUSY, SYSIDERR

Description

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 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 Intercommunication Guide*.

Options

NOQUEUE

overrides the default action when a SYSBUSY condition arises. This condition indicates 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.

Note, however, that if a HANDLE CONDITION for SYSBUSY is active when the command is executed, this also overrides the default action, and control is passed to the user label supplied in the HANDLE CONDITION. This takes precedence over the NOQUEUE option but is, of course, negated by either NOHANDLE or RESP.

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 option. If the PROFILE option is omitted, 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) of a system TCTSE. This option specifies that one of the sessions to the named system is to be allocated.

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 the NOQUEUE option is set, or a HANDLE CONDITION for SYSBUSY is active.

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.
- All the sessions are busy and the (queued) allocates have been purged or rejected.

Default action: terminate the task abnormally.

ASKTIME

Request current date and time of day.

ASKTIME



This command is threadsafe.

Description

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.

In response to an ASKTIME command, CICS issues an MVS™ STCK macro and modifies this by a local time difference. Thus, if your MVS TOD (hardware) clock is set to, say, GMT, and the local time is defined as British Summer Time (BST), it is BST that is stored in the EIBTIME field.

Refer to “EXEC interface block” on page 745 for details of the EIB.

Options

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.

Examples

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

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

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

Description

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 Intercommunication Guide*.

Options

ABCODE(*data-area*)

returns a 4-character current abend code. (Abend codes are documented in the *CICS 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 an EXEC CICS ABEND ABCODE command was issued without the NODUMP option 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.

When the latest abend is an APCT (resulting from an unsuccessful attempt to load a program, mapset or partitionset), the name is taken from the program, mapset or partitionset that was not loaded.

ACTIVITY(*data-area*)

returns, if this program is executing on behalf of a CICS business transaction services (BTS) activity, the 16-character name of the activity.

BTS is described in the *CICS Business Transaction Services* manual.

ACTIVITYID(*data-area*)

returns, if this program is executing on behalf of a BTS activity, the 52-character, CICS-assigned, identifier of the activity-instance.

If a program, executing outside the current process, wants to acquire control of this activity-instance, it must specify this identifier on an ACQUIRE ACTIVITYID command.

BTS is described in the *CICS Business Transaction Services* manual.

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 data-area containing the ILC (Instruction length code) and the PIC (Program interrupt code) 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. When valid, the contents of the eight bytes returned are: ILC (2 bytes binary), PIC (2 bytes binary), filler (4 bytes binary, always zero).

ASRAKEY(*cvda*)

returns the execution key at the time of the last ASRA, ASRB, AICA, or AEYD, abend, if any. CVDA values are:

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 data-area containing the 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.

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 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:

- 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 OC4 abend.
- An ASRB or AICA abend has occurred since the last ASRA or AEYD abend.

BRIDGE(*data-area*)

returns the 4-character TRANSID of the bridge monitor transaction that issued a START BREXIT TRANSID command to start the user transaction that issued this command. Blanks are returned if:

- The user transaction was not started by a bridge monitor transaction.
- This command was issued by a program started by a distributed program link (DPL) request.

Note: If the START BREXIT command was issued from a bridge exit, the TRANSID returned is the that of the bridge monitor that issued a START BREXIT naming the bridge exit.

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.

CHANNEL(*data-area*)

returns the 16-character name of the program's current channel, if one exists; otherwise blanks.

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. This 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”.)

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

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”.)

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

EWASUPP(*data-area*)

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

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

EXTDS(*data-area*)

returns a 1-byte indicator showing whether the terminal accepts the 3270 extended data stream, (X'FF') or not (X'00'). 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 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 principal facility that initiated the transaction issuing this command. If this option is specified, and there is no allocated facility, INVREQ occurs.

Note: You can use the QNAME option to get the name of the transient data intrapartition queue if the transaction was initiated by expiry of a transient data trigger level.

FCI(*data-area*)

returns a 1-byte facility control indicator, see “Codes returned by ASSIGN” on page 762

page 762. 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 65 534 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 65 534 representing the EBCDIC or ASCII 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 System Definition Guide* for further information about the INITPARM parameter.)

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

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.

LANGINUSE (*data-area*)

returns a 3-byte mnemonic code showing the language in use. The 3-byte mnemonic has a 1:1 correspondence with the 1-byte NATLANGINUSE option. See "National language codes" on page 766 for possible values of the code.

LDCMNE (*data-area*)

returns a 2-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. (NATLANGINUSE does not show the system default language as specified on the NATLANG system initialization parameter.)

See “National language codes” on page 766 for possible values of the code.

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, CICS no longer returns a null string but the netname of the remote terminal.

If this command was issued by a user transaction which was started by a 3270 bridge transaction, the value returned is the termid of the bridge facility.

If the CICS region supports VTAM LU aliases, the NETNAME returned by CICS could be an LU alias, either dynamically allocated by VTAM or predefined on the LUALIAS parameter of a CDRSC definition.

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 of the External Security Manager.

OPERKEYS(*data-area*)

is accepted for compatibility with previous releases. If specified, a 64-bit null string is returned.

OPID(*data-area*)

returns the 3-character operator identification. This is used by BMS for routing terminal messages, as defined in the CICS segment of the External Security Manager.

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 the INQUIRE TERMINAL system programming command, which is described in the *CICS System Programming Reference* manual.

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. If specified, a 24-bit null string is returned.

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 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 Intercommunication Guide* for more information)

PROCESS (*data-area*)

returns, if this program is executing on behalf of a CICS business transaction services (BTS) activity, the 36-character name of the BTS process that contains the activity.

BTS is described in the *CICS Business Transaction Services* manual.

PROCESSTYPE (*data-area*)

returns, if this program is executing on behalf of a BTS activity, the 8-character process-type of the BTS process that contains the activity.

BTS is described in the *CICS Business Transaction Services* manual.

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.

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 user-replaceable module, or a program list table program, CICS returns eight blanks.
- 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-character value indicating how the transaction that issued the request was started. Possible values are:

Code Transaction started by

- | | |
|-----------|--|
| D | A distributed program link (DPL) request that did not specify the SYNCONRETURN option. The task 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, that did specify the SYNCONRETURN option. The task can issue syncpoint requests. |
| QD | Transient data trigger level. |

- S** START command that did not pass data in the FROM option. It may or may not have passed a channel.
- SD** START command that passed data in the FROM option.
- SZ** FEPI START command.
- TD** Terminal input or permanent transid.
- U** User-attached task.

Note for IIOp: When the IIOp request processor is run locally, the startcode for an ASSIGN command is U. When the IIOp request processor is run remotely, over an MRO link, the startcode for the command is TD. (If you attempt to run the IIOp request processor remotely over any other type of connection, the routing request is not accepted, so startcodes for these commands are not relevant in this situation).

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 field indicating the 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. 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 terminal resource definition. This is the DEVICE attribute (described in the *CICS 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 “Codes returned by ASSIGN” on page 762.

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 Intercommunication Guide* for more information about the ASSIGN command for LUTYPE6.1, APPC, and MRO.

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.

Conditions

INVREQ

RESP2 values:

- 1** The task does not have a signed-on user.
- 2** No BMS command has yet been issued, BMS routing is in effect, or no map has yet been positioned.
- 3** No batch data interchange (BDI) command has yet been issued.
- 4** The task is not initiated by automatic transaction initiation (ATI).
- 5** 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.
- 6** A CICS BTS request was issued from outside the CICS BTS environment. (Therefore, the transaction is not executing on behalf of a BTS activity.)
- 200** Command syntax options are not allowed in a server program invoked by a distributed program link.

Default action: terminate the task abnormally.

Examples

An example of RETURNPROG:

```
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.
```

BIF DEEDIT

Deediting (built-in function).

BIF DEEDIT

►—BIF DEEDIT—FIELD(*data-area*)—┐LENGTH(*data-value*)┘◄

Condition: LENGERR

Description

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.

Options

FIELD(*data-area*)
specifies the field to be edited.

LENGTH(*data-value*)
specifies the field length in bytes.

Conditions

LENGERR
occurs if the LENGTH value is less than 1.
Default action: terminate the task abnormally.

Examples

```
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.

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.)

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.

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.)

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*)

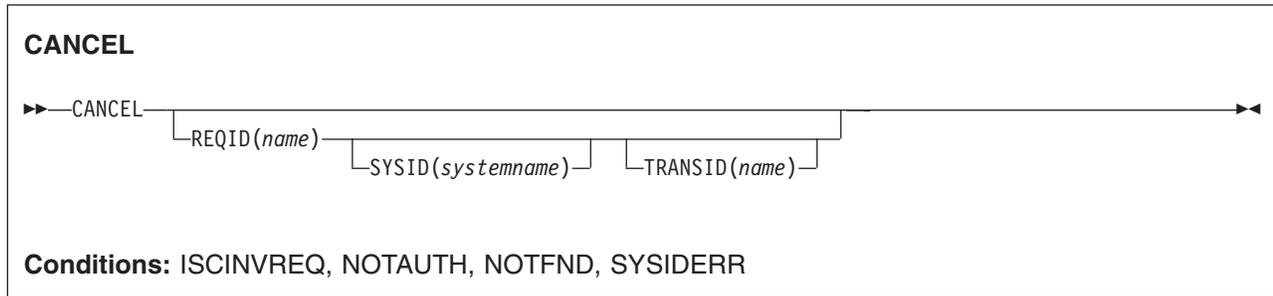
corresponds to the return-resource name, ATTRPRN, in an LUTYPE6.1 attach FMH.

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

Cancel interval control requests.



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 Application Programming Guide* for more information about transaction affinities.

Description

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 associated with the original command must be specified. The effect of the cancelation is as if the original command had never been issued. The cancelation is effective only before the original command has been honored.

Note: A NOTFND response to a CANCEL command of a START with REQID signifies that the start request is no longer outstanding. It does not imply that the task to be started has completed by this point in time; neither does it imply that the started task has issued a RETRIEVE command to read the FROM data from the REQID queue. A subsequent START command reusing the same REQID value may fail with an AEIQ abend (IOERR condition), if the REQID queue still exists at this time.

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.

Conditions

ISCVREQ

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.

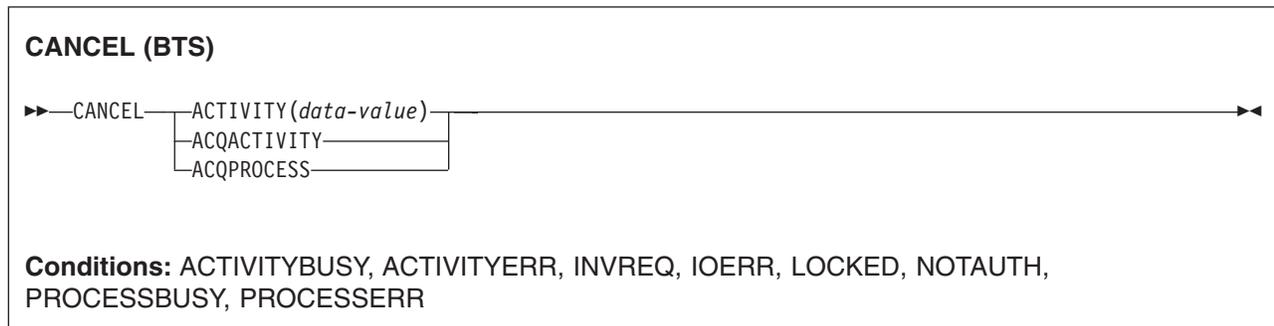
SYSIDERR

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.

CANCEL (BTS)

Cancel a BTS activity or process.



Description

CANCEL (BTS) forces a BTS activity or process, and all its descendant activities, into COMPLETE mode.

Options

ACQACTIVITY

specifies that the activity to be canceled is the one that the current unit of work has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the process that the current unit of work has acquired is to be canceled.

ACTIVITY (data-value)

specifies the name (1–16 characters) of the child activity to be canceled.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19** One or more of the descendant activities of the activity to be canceled are inaccessible or in CANCELLING mode.

ACTIVITYERR

RESP2 values:

- 8** The activity named on the ACTIVITY option could not be found.
- 14** The activity to be canceled is not in INITIAL or DORMANT mode.

INVREQ

RESP2 values:

- 4** The ACTIVITY option was used to name a child activity, but the command was issued outside the scope of a currently-active activity.
- 15** The ACQPROCESS option was used, but the issuing task has not acquired a process.
- 24** The ACQACTIVITY option was used, but the issuing task has not acquired an activity.

IOERR

RESP2 values:

- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

NOTAUTH

RESP2 values:

- 101** The user associated with the issuing task is not authorized to access the file associated with the BTS repository data set on which details of the process or activity are stored.

PROCESSBUSY

RESP2 values:

- 13** One or more of the activities that make up the process to be canceled are inaccessible or in CANCELLING mode.

PROCESSERR

RESP2 values:

- 9** The process—type could not be found.
- 14** The process to be canceled is not in INITIAL, DORMANT, or COMPLETE mode.

Activities

The only activities a program can cancel are as follows:

- If it is running as the activation of an activity, its own child activities. It can cancel several of its child activities within the same unit of work.
- The activity it has acquired, by means of an ACQUIRE ACTIVITYID command, in the current unit of work.

To be canceled successfully, an activity must be in INITIAL or DORMANT mode. CICS tries to cancel activities synchronously. However, if one or more descendant activities of the activity to be canceled are inaccessible (due, for example, to the failure of a communications link):

- The subtree of descendant activities is canceled asynchronously.
- The activity to be canceled is placed in CANCELLING mode.

The completion event associated with a canceled activity is not deleted from the parent's event pool. On normal completion of this command, the activity still exists, and can be reset and run again, if necessary.

When an acquired activity is canceled, its parent is reactivated because of the firing of the canceled activity's completion event.

Processes

The only process a program can cancel is the one it has acquired in the current unit of work. If it does so, it cannot acquire another process within the current unit of work.

To be canceled successfully, a process must be in INITIAL, DORMANT, or COMPLETE mode.

CICS tries to cancel the process synchronously, in the way described for activities.

CHANGE PASSWORD

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

CHANGE PASSWORD

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

Conditions: INVREQ, NOTAUTH, USERIDERR

Description

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.

Attention: 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.

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.

Conditions

INVREQ

RESP2 values:

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

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

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

Default action: terminate the task abnormally.

USERIDERR

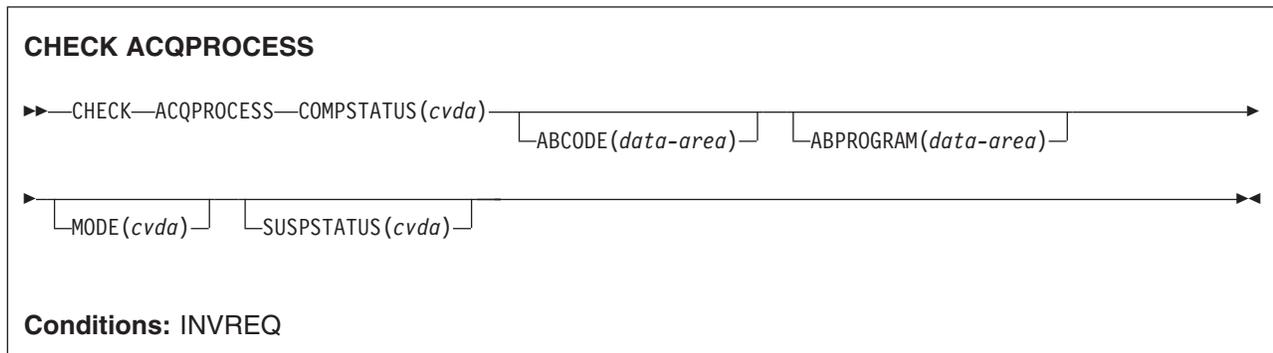
RESP2 values:

- 8** The USERID is not known to the external security manager.

Default action: terminate the task abnormally.

CHECK ACQPROCESS

Check the completion status of a BTS process.



Description

CHECK ACQPROCESS returns the completion status of the currently-acquired BTS process. Typically, it is used to check the success of a previous RUN ACQPROCESS or LINK ACQPROCESS command. It allows the requestor to discover whether the process completed successfully, or whether, for example, it needs to be reactivated in order to complete its processing.

The only process a program can check is the one that it has acquired in the current unit of work—see the *CICS Business Transaction Services* manual.

The RESP and RESP2 options on this command reflect whether the command is understood by CICS—for example, PROCESSERR occurs if the process is not currently acquired by the requestor.

The COMPSTATUS option returns a CVDA value indicating the completion status of the process's root activity—for example, NORMAL is returned if the root activity has successfully completed all its processing steps, while INCOMPLETE is returned if it has returned from an activation but needs to be reattached in order to complete its processing.

Options

ABCODE(*data-area*)

returns, if the process's root activity terminated abnormally, the 4-character abend code.

ABPROGRAM(*data-area*)

returns, if the process's root activity terminated abnormally, the 8-character name of the program that was in control at the time of the abend.

ACQPROCESS

specifies that the process that is currently acquired by the requestor is to be checked.

COMPSTATUS(*cvda*)

indicates the completion status of the process. CVDA values are:

ABEND

The program that implements the process's root activity abended. Any children of the root activity have been canceled.

FORCED

The process was forced to complete—for example, it was canceled with a CANCEL ACQPROCESS command.

INCOMPLETE

The process is incomplete. This could mean:

- That it has not yet been run
- That it has returned from one or more activations but needs to be reattached in order to complete all its processing steps
- That it is currently active.

NORMAL

The process completed successfully.

MODE (cvda)

indicates the processing state of the process. CVDA values are:

ACTIVE

An activation of the process is running.

CANCELLING

CICS is waiting to cancel the process. A CANCEL ACQPROCESS command has been issued, but CICS cannot cancel the process immediately because one or more of the root activity's children are inaccessible.

COMPLETE

The process has completed.

DORMANT

The process is waiting for an event to fire its next activation.

INITIAL

No RUN or LINK command has yet been issued against the process.

SUSPSTATUS (cvda)

indicates whether the process is currently suspended. CVDA values are:

SUSPENDED

The process is currently suspended. If a reattachment event occurs, it will not be reactivated.

NOTSUSPENDED

The process is not currently suspended. If a reattachment event occurs, it will be reactivated.

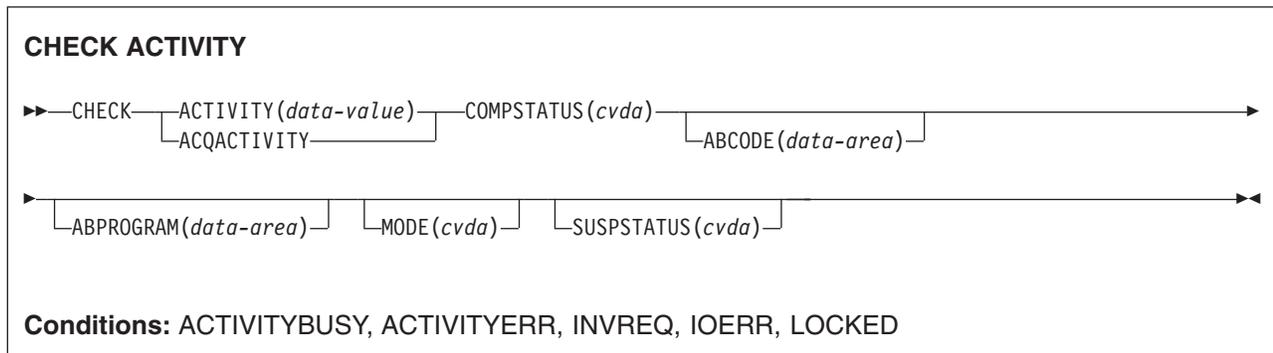
Conditions**INVREQ**

RESP2 values:

- 15** The unit of work that issued the request has not acquired a process.

CHECK ACTIVITY

Check the completion status of a BTS activity.



Description

CHECK ACTIVITY returns the completion status of a BTS activity. Typically, it is used to check the success of a previous RUN ACTIVITY or LINK ACTIVITY command. It allows the requestor to discover whether an activity completed successfully, or whether, for example, it needs to be reactivated in order to complete its processing.

CHECK ACTIVITY can be issued:

1. By a parent activity, to check the completion status of one of its children
2. By a program that has acquired an activity by means of an ACQUIRE ACTIVITYID command.

It can be used to check descendant (not root) activities:

- That have completed
- That have not completed
- That were requested to run asynchronously
- That were requested to run synchronously.

The RESP and RESP2 options on this command reflect whether the command is understood by CICS—for example, ACTIVITYERR occurs if the child named on the ACTIVITY option has not been defined to the parent.

The COMPSTATUS option returns a CVDA value indicating the completion status of the activity—for example, NORMAL is returned if the activity has successfully completed all its processing steps, while INCOMPLETE is returned if it has returned from an activation but needs to be reattached in order to complete its processing.

If this command is issued by a parent activity in respect of one of its children, and the child has completed, on return from the command CICS deletes the child's completion event from the parent's event pool.

For further guidance on the use of the CHECK ACTIVITY command, see the *CICS Business Transaction Services* manual.

Options

ABCODE(data-area)

returns, if the activity terminated abnormally, the 4-character abend code.

ABPROGRAM(data-area)

returns, if the activity terminated abnormally, the 8-character name of the program that was in control at the time of the abend.

ACQACTIVITY

specifies that the activity to be checked is the one that the current unit of work has acquired by means of an ACQUIRE ACTIVITYID command.

ACTIVITY(data-value)

specifies the name (1–16 characters) of the activity to be checked.

Use this option to check the state of a child of the current activity.

COMPSTATUS(cvda)

indicates the completion status of the activity. CVDA values are:

ABEND

The program that implements the activity abended. Any children of the activity have been canceled.

The activity's completion event is deleted from the parent's event pool.

FORCED

The activity was forced to complete—for example, it was canceled with a CANCEL ACTIVITY command.

The activity's completion event is deleted from the parent's event pool.

INCOMPLETE

The named activity is incomplete. This could mean:

- That it has not yet been run
- That it has returned from one or more activations but needs to be reattached in order to complete all its processing steps
- That it is currently active.

The activity's completion event is **not** deleted from the parent's event pool.

NORMAL

The named activity completed successfully.

The activity's completion event is deleted from the parent's event pool.

MODE(cvda)

indicates the processing state of the activity. CVDA values are:

ACTIVE

An activation of the activity is running.

CANCELLING

CICS is waiting to cancel the activity. A CANCEL ACTIVITY command has been issued, but CICS cannot cancel the activity immediately because one or more of the activity's children are inaccessible.

COMPLETE

The activity has completed.

DORMANT

The activity is waiting for an event to fire its next activation.

INITIAL

No RUN or LINK command has yet been issued against the activity; or the activity has been reset by means of a RESET ACTIVITY command.

SUSPSTATUS (cvda)

indicates whether the activity is currently suspended. CVDA values are:

SUSPENDED

The activity is currently suspended. If a reattachment event occurs, it will not be reactivated.

NOTSUSPENDED

The activity is not currently suspended. If a reattachment event occurs, it will be reactivated.

Conditions**ACTIVITYBUSY**

RESP2 values:

- 19** The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8** The activity named in the ACTIVITY option could not be found.

INVREQ

RESP2 values:

- 4** The ACTIVITY option was used to name a child activity, but the command was issued outside the scope of a currently-active activity.
- 24** The ACQACTIVITY option was used, but the unit of work that issued the request has not acquired an activity.

IOERR

RESP2 values:

- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

CHECK TIMER

Check the status of a BTS timer.

CHECK TIMER

▶▶—CHECK—TIMER(*data-value*)—STATUS(*cvda*)—————▶▶

Conditions: INVREQ, IOERR, TIMERERR

Description

CHECK TIMER returns the status of a BTS timer. It allows the requestor to discover whether a timer has expired and, if so, whether it expired normally or whether its expiry was forced by means of a FORCE TIMER command.

On return from this command, if the timer has expired its associated event is deleted from the current activity's event pool.

The only timers a program can check are those owned by the current activity.

Options

STATUS(*cvda*)

indicates the status of the timer. CVDA values are:

EXPIRED

The timer expired normally.

Its associated event is deleted from the current activity's event pool.

FORCED

The timer expired because a FORCE TIMER command was issued against it.

Its associated event is deleted from the current activity's event pool.

UNEXPIRED

The timer has not yet expired.

Its associated event is not deleted from the current activity's event pool.

TIMER(*data-value*)

specifies the name (1–16 characters) of the timer to be checked.

Conditions

INVREQ

RESP2 values:

- 1 The command was issued outside the scope of a currently-active activity.

IOERR

An input/output error has occurred on the repository file.

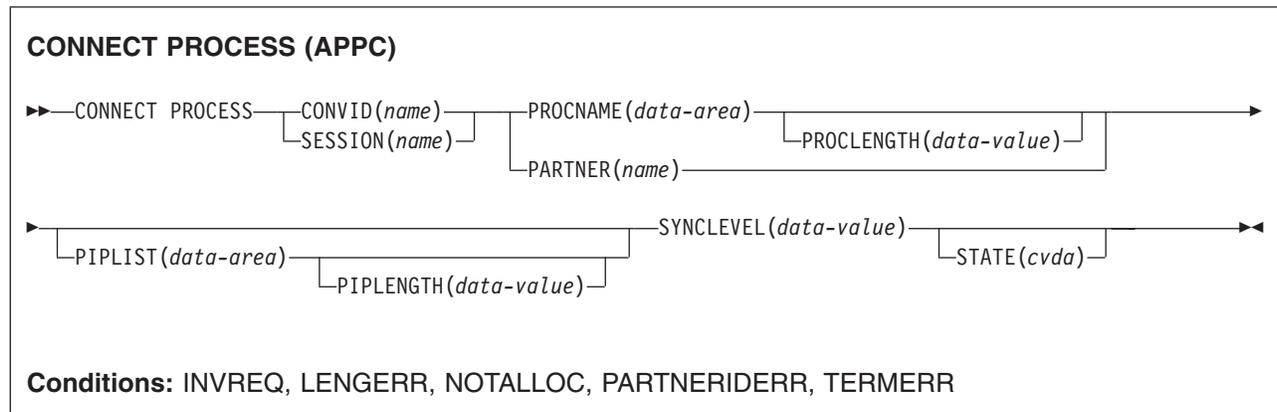
TIMERERR

RESP2 values:

13 The timer specified on the TIMER option does not exist.

CONNECT PROCESS

Initiate APPC mapped conversation.



Description

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.

Options

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character 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.

PIPLENGTH(*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 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

SYNCLEVEL(*data-value*)

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

- 0 None
- 1 Confirm
- 2 Syncpoint

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) 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.

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

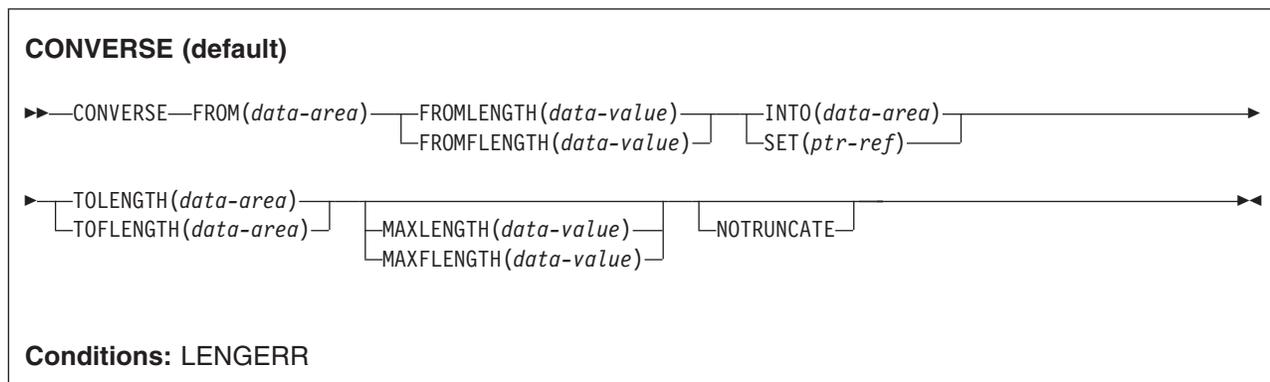
occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

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.

CONVERSE (VTAM default)

Communicate on standard CICS terminal support.

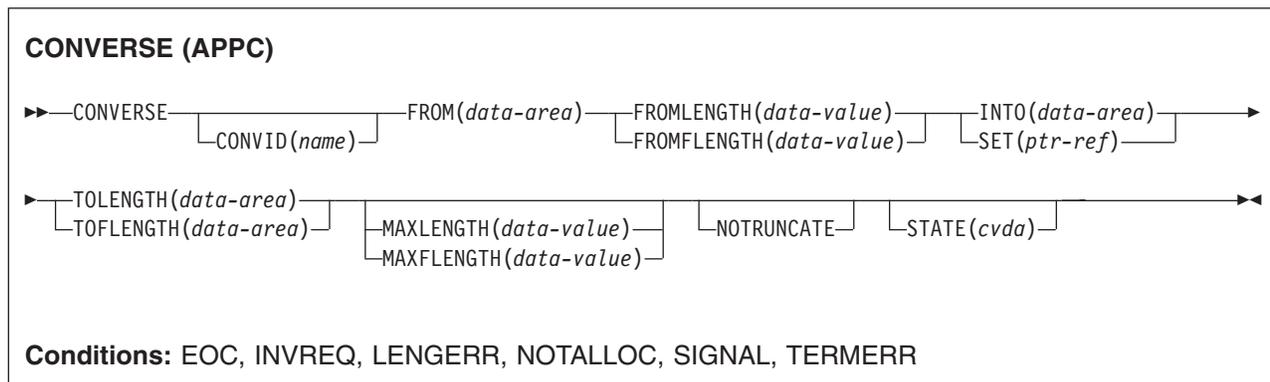


Description

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)

Communicate on an APPC mapped conversation.

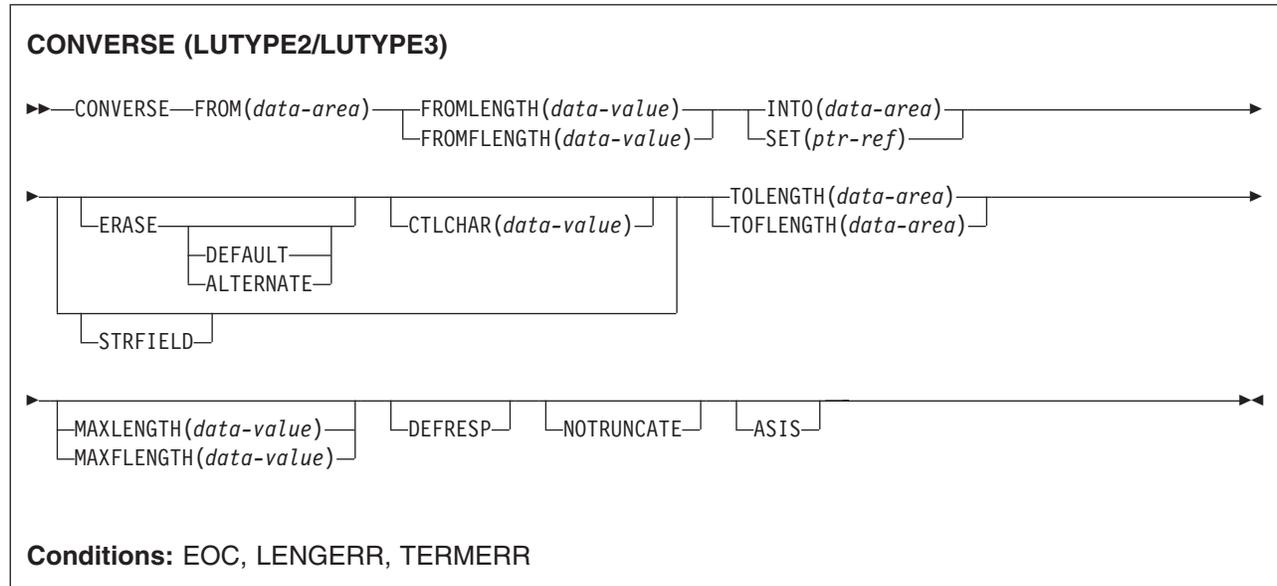


Description

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

CONVERSE (LUTYPE2/LUTYPE3)

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

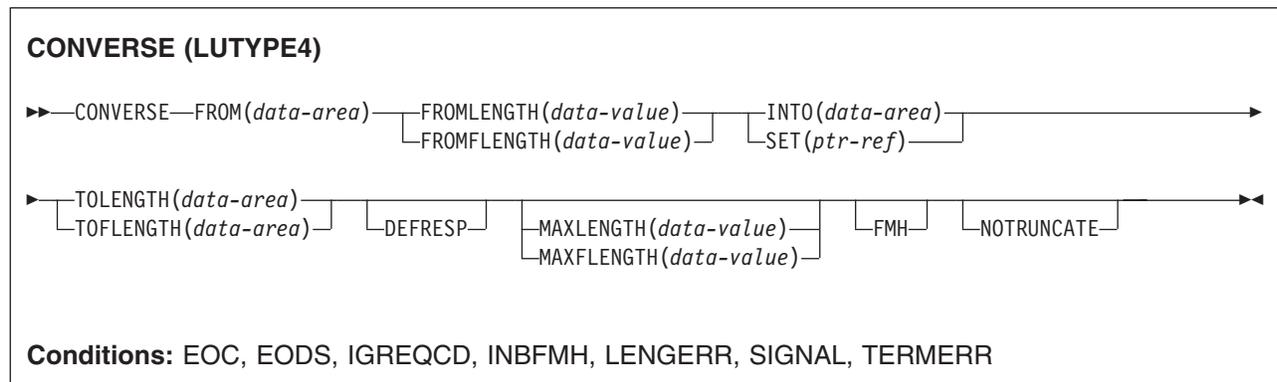


Description

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

CONVERSE (LUTYPE4)

Communicate on an LUTYPE4 logical unit.

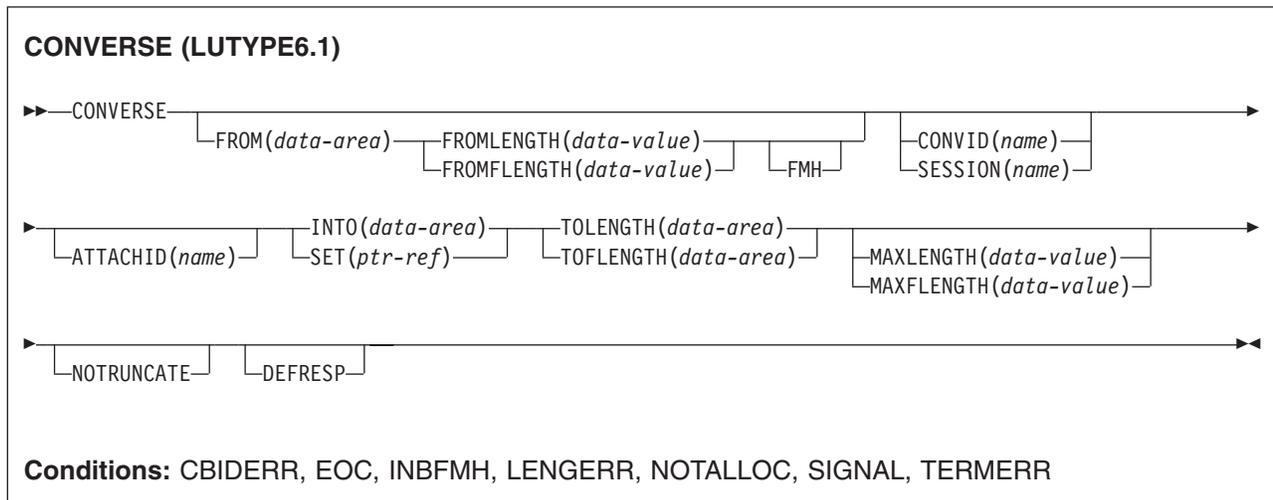


Description

CONVERSE communicates on an LUTYPE4 logical unit.

CONVERSE (LUTYPE6.1)

Communicate on an LUTYPE6.1 logical unit.

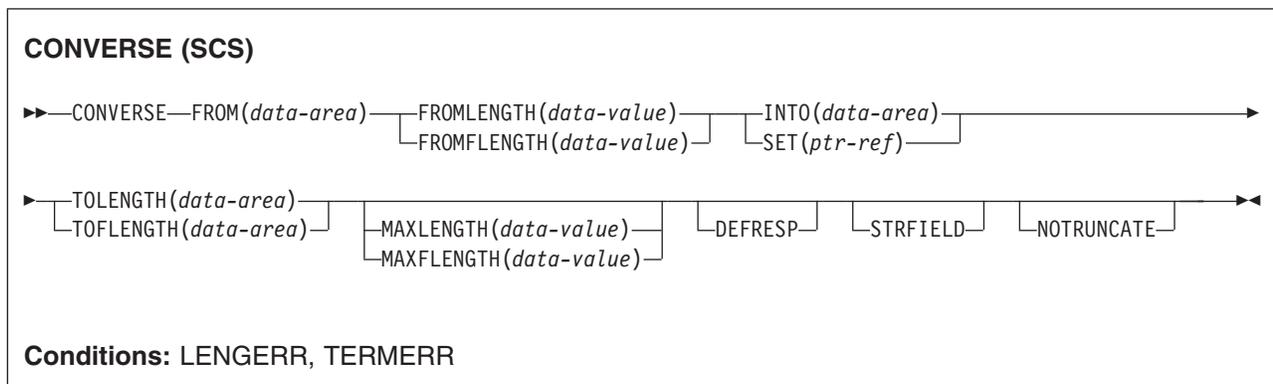


Description

CONVERSE communicates on an LUTYPE6.1 logical unit.

CONVERSE (SCS)

Communicate on a 3270 SCS printer logical unit.



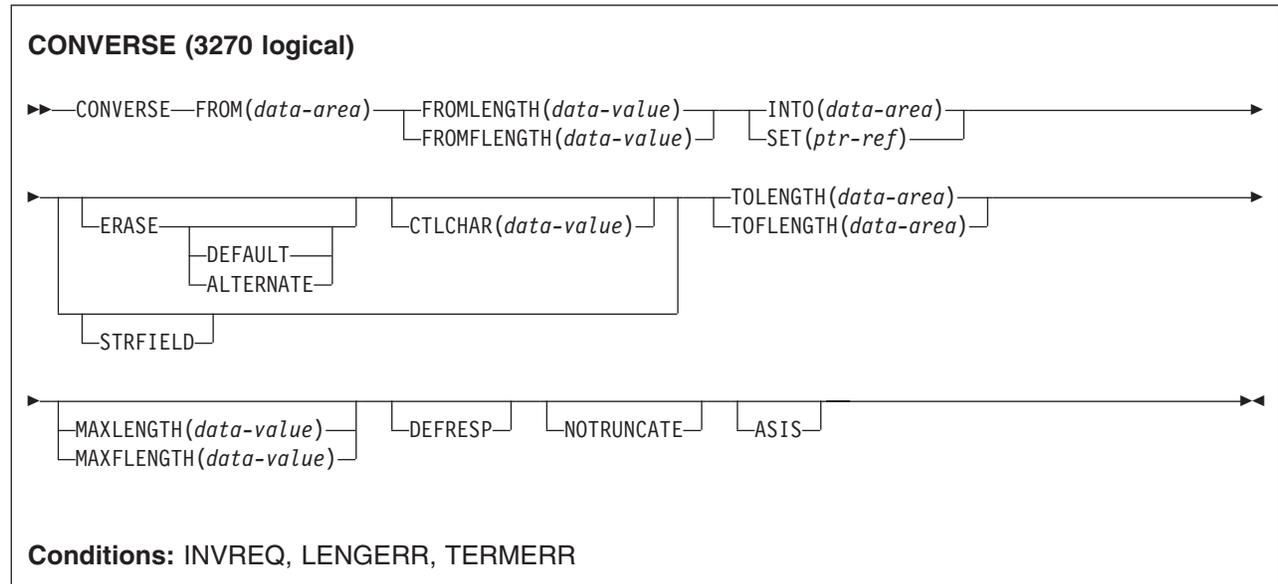
Description

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 (3270 logical)

Communicate on a 3270 logical unit.

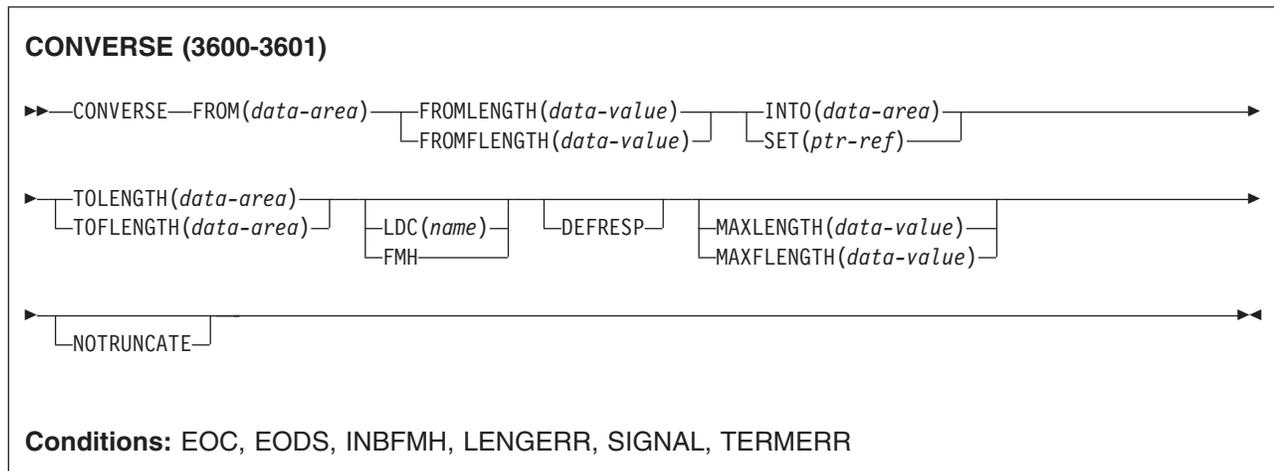


Description

CONVERSE communicates on a 3270 logical unit.

CONVERSE (3600-3601)

Communicate on a 3600 (3601) logical unit.



Description

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 Function Management Header (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. A numeric value (0 through 255) corresponds to each LDC mnemonic for each TCTTE.

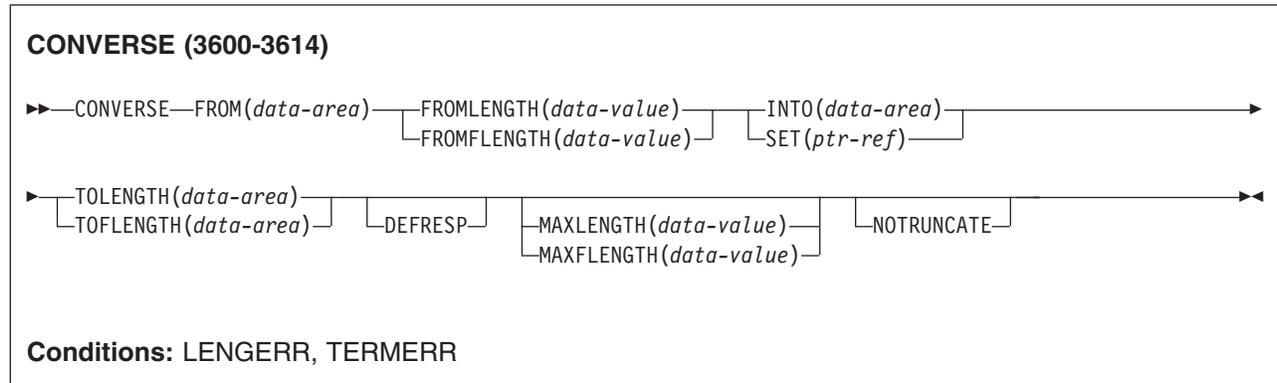
A 3600 device and a logical page size are also associated with an LDC. "LDC" or "LDC value" is used in this book to refer to the code specified by the user; "LDC mnemonic" refers to the 2-character symbol that represents the LDC numeric value.

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)

Communicate on a 3600 (3614) logical unit.



Description

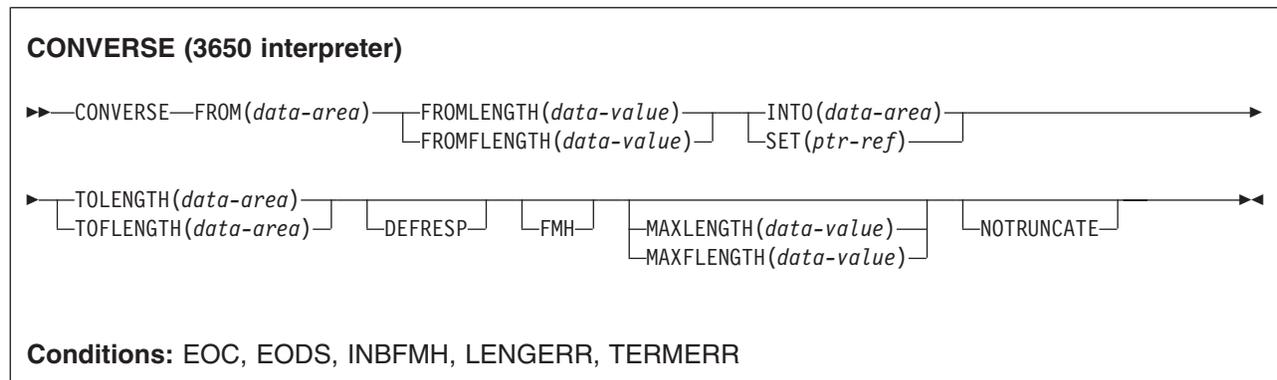
CONVERSE communicates on a 3600 logical unit.

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 *IBM 4700/3600/3630 Guide*.

CONVERSE (3650 interpreter)

Communicate on a 3650 interpreter logical unit.

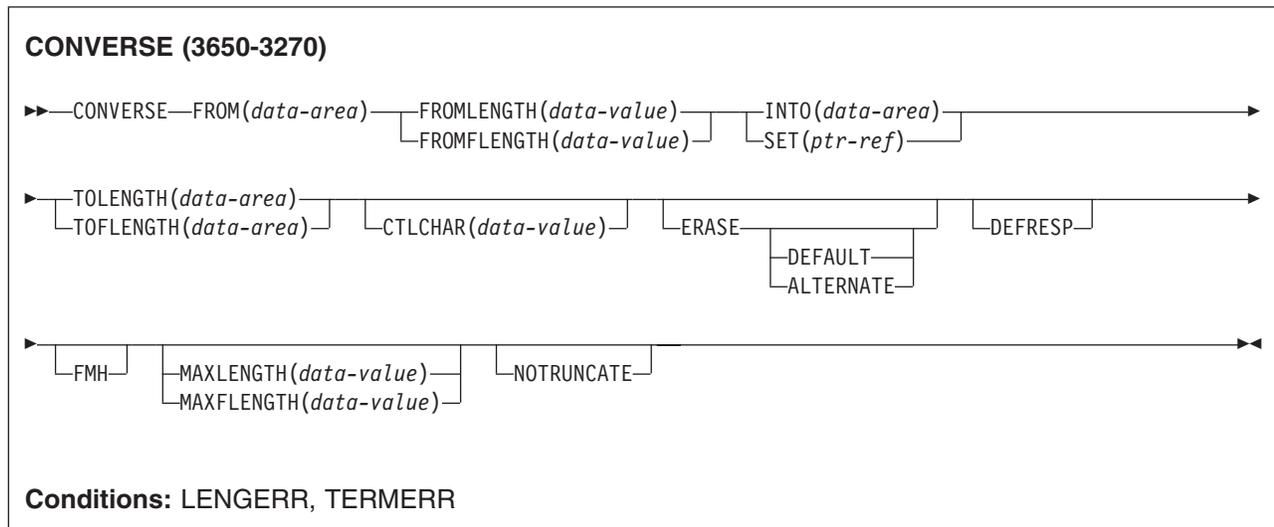


Description

CONVERSE communicates on a 3650 interpreter logical unit.

CONVERSE (3650-3270)

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

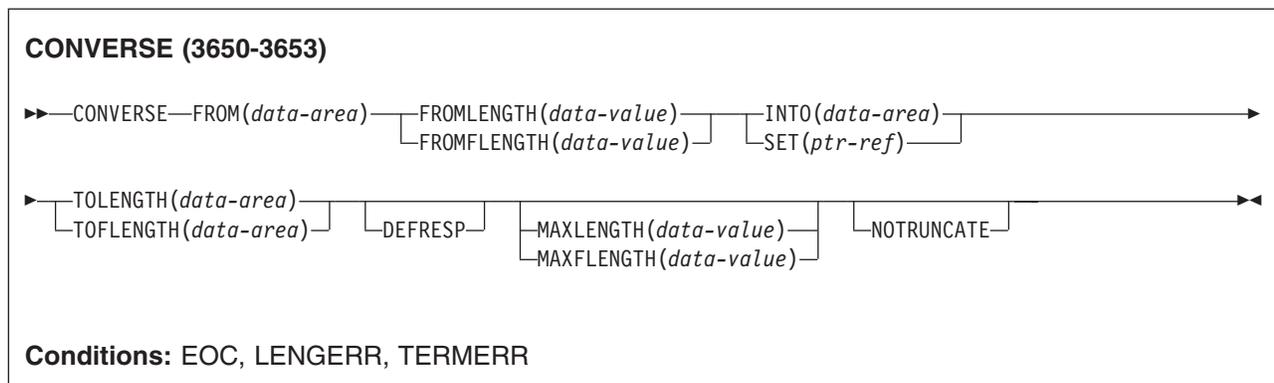


Description

CONVERSE communicates on a 3650 host conversational logical unit.

CONVERSE (3650-3653)

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

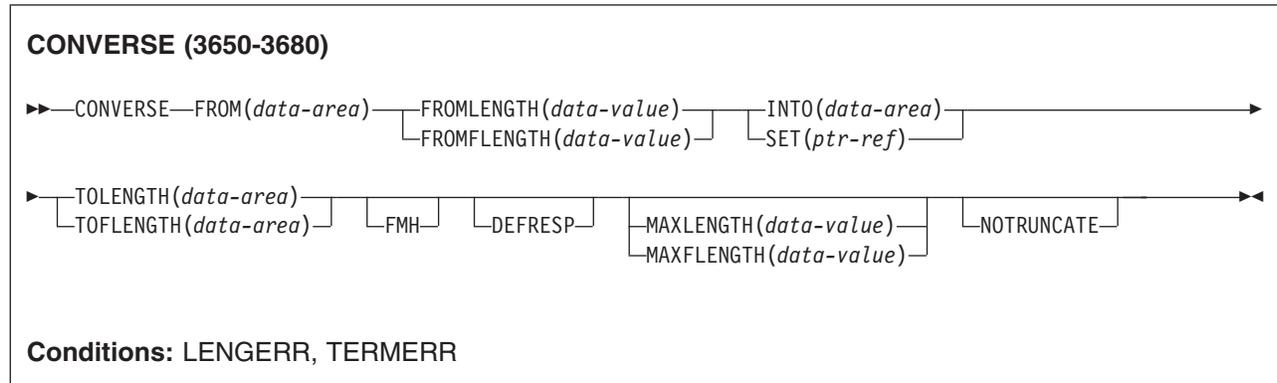


Description

CONVERSE communicates on a 3650 host conversational logical unit.

CONVERSE (3650-3680)

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

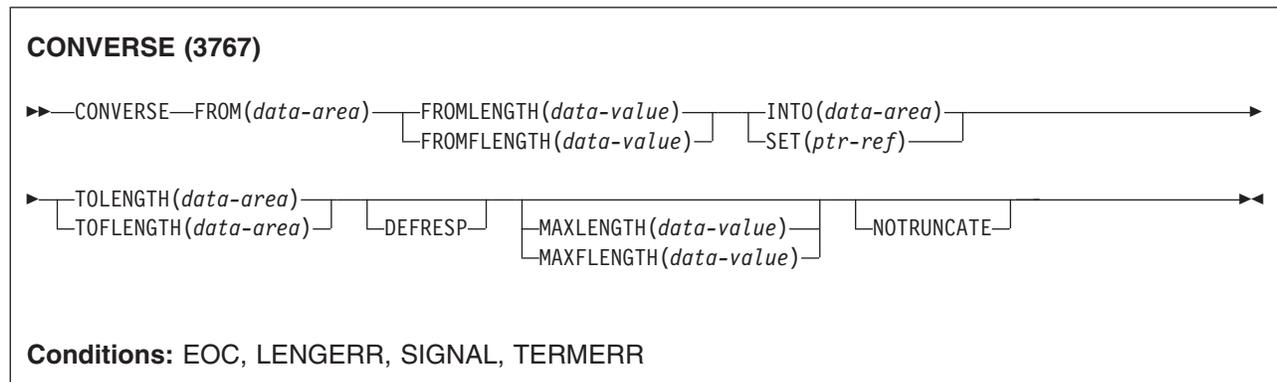


Description

CONVERSE communicates on a 3650 host command processor logical unit.

CONVERSE (3767)

Communicate on a 3767 interactive logical unit.

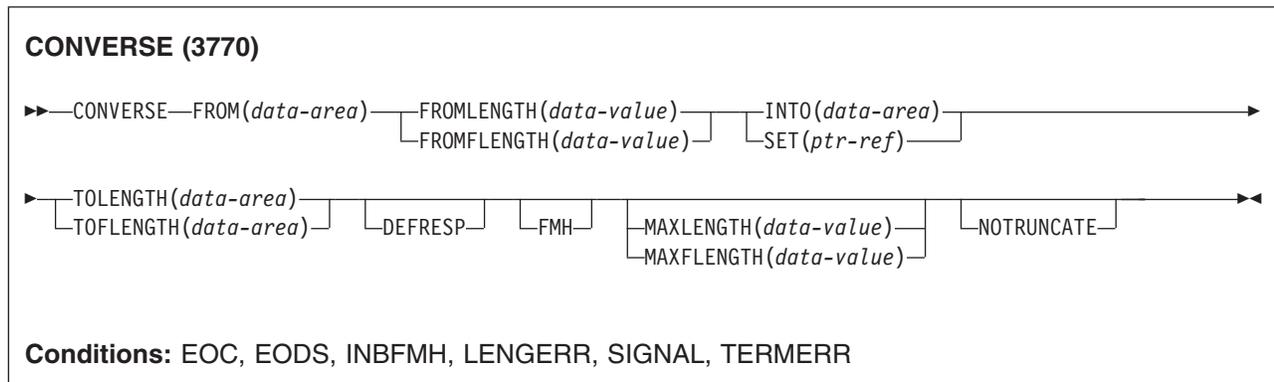


Description

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

CONVERSE (3770)

Communicate on a 3770 batch logical unit.

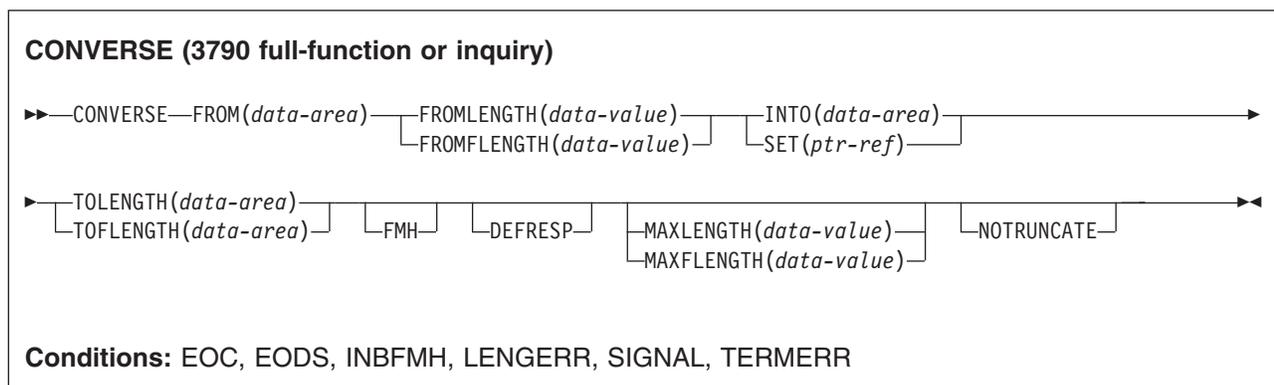


Description

CONVERSE communicates on a 3770 batch logical unit.

CONVERSE (3790 full-function or inquiry)

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

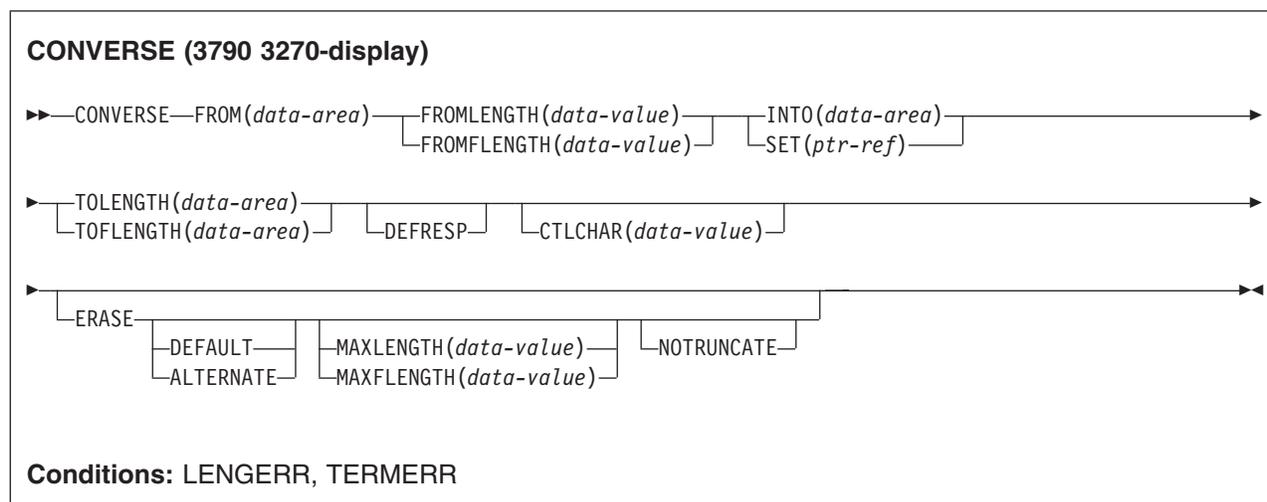


Description

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

CONVERSE (3790 3270-display)

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



Description

CONVERSE communicates on a 3790 logical unit.

CONVERSE: VTAM options

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.

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 RDO 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 in CICS commands" on page 8.

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 defined by a DFHTCT TYPE=LDC macro.

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 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 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 in CICS commands” on page 8.

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.

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

RESP2 values:

200 A distributed program link server application specified the function shipping session (its principal facility)

also occurs (RESP2 not set) 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.

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

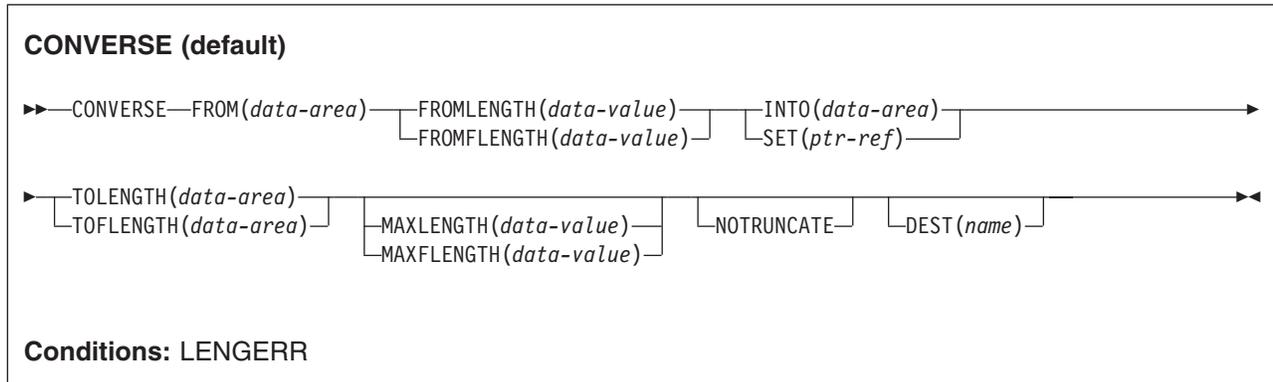
occurs for a terminal or session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

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)

Communicate on standard CICS terminal support.

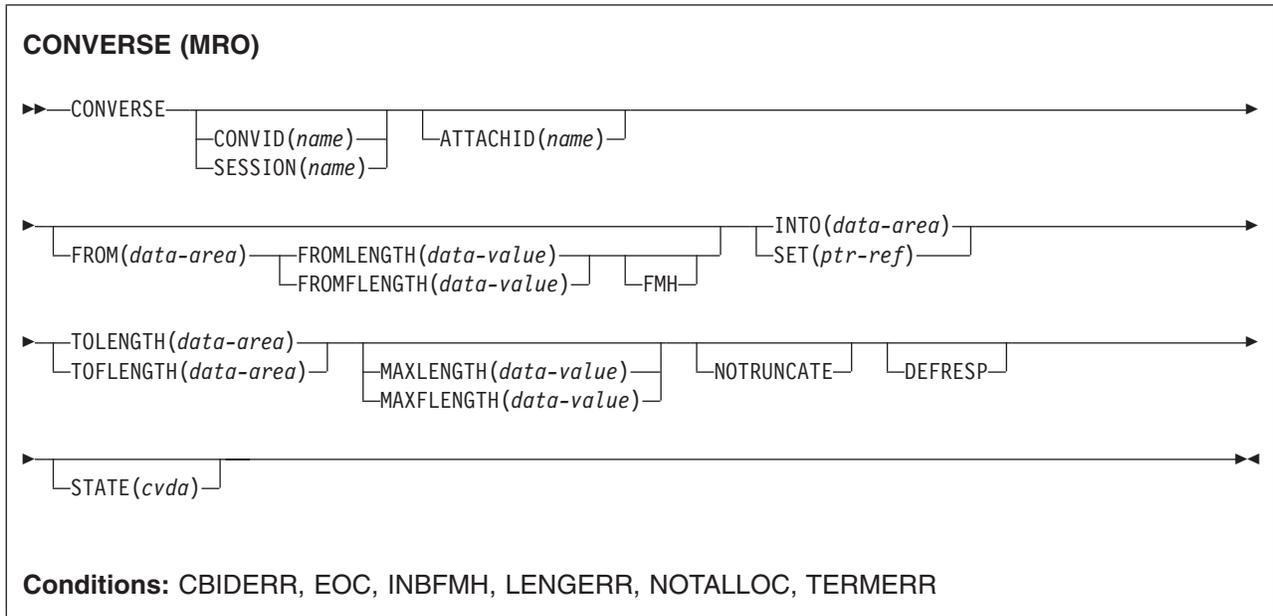


Description

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

CONVERSE (MRO)

Communicate on an MRO session.

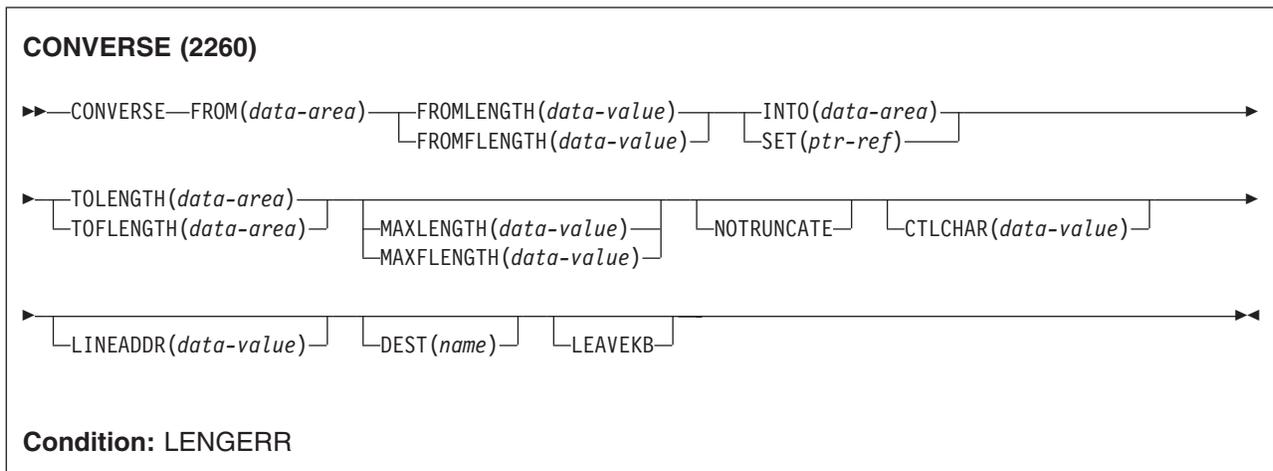


Description

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

CONVERSE (2260)

Communicate on a 2260 or 2265 display station.

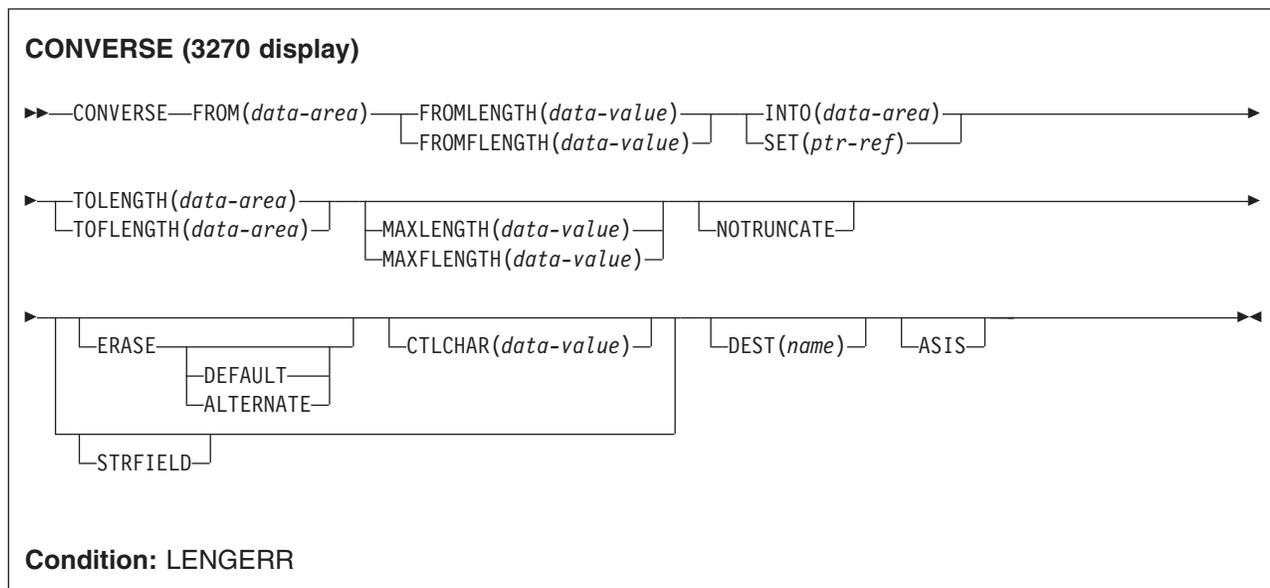


Description

CONVERSE communicates on a 2260 or 2265 display station.

CONVERSE (3270 display)

Communicate on a 3270 information display system.



Description

CONVERSE communicates on a 3270 information display system.

CONVERSE: non-VTAM options

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 EIBRSRCE 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 (WCC) that controls the CONVERSE command. (The WCC is documented in the *IBM 3270 Data Stream Programmer's Reference* manual.) 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.

Note: In CICS TS 3.1, local TCAM terminals are not supported. The only TCAM terminals supported are remote terminals connected to a pre-CICS TS 3.1 terminal-owning region by the DCB (not ACB) interface of 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.

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 RDO 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.

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 in CICS commands” on page 8.

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 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 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 in CICS commands” on page 8.

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

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

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.

CONVERTTIME

Converts an architected date and time stamp string to the ABSTIME format.

CONVERTTIME

►—CONVERTTIME—DATESTRING(*data-area*)—ABSTIME(*data-area*)—►

Conditions: INVREQ, LENGERR

This command is threadsafe.

Description

CONVERTTIME analyzes three different date and time stamp formats which are commonly used on the Internet, and converts them to the ABSTIME (absolute date and time) format.

ABSTIME format gives the time, in packed decimal, since 00:00 on 1 January 1900 (in milliseconds rounded to the nearest hundredth of a second). The FORMATTIME command can be used to change this into other formats.

The architected date and time stamp string formats recognized by the CONVERTTIME command are:

RFC 1123 format

The preferred standard format for date and time stamps for the HTTP protocol, as specified in RFC 1123. An example of a date and time stamp in this format is "Tue, 01 Apr 2003 10:01:02 GMT".

RFC 850 format

An older date and time stamp format for the Internet. An example of a date and time stamp in this format is "Tuesday, 01-Apr-03 10:01:02 GMT".

Important: Because the year has only two digits in this format, CICS uses the assumption that the years are in the range 1970 to 2069. In the example above, CICS would assume that the date of the document was 1 April 2003. Given the date and time stamp "Thursday, 13-Feb-98 15:30:00 GMT", CICS would assume that the date of the document was 13 February 1998. Be aware of this when coding your application, if you think that you could receive date and time stamps in this format.

ASCtime format

A date and time stamp format output from the C ASCtime function. An example of a date and time stamp in this format is "Tue Apr 1 10:01:02 2003".

Options

DATESTRING(*data-area*)

specifies a 64-character data-area to contain the architected date and time stamp string. You can supply a string in any of the formats recognized by the command, and you do not need to specify which format is used. If the date and

#

time stamp string is in the RFC 1123 format, which is always at GMT, the date
and time are converted to local time for the ABSTIME which is returned.

ABSTIME (*data-area*)

specifies a data-area to receive the converted date and time stamp in ABSTIME format. For the format of this data-area, see the description of the ASKTIME command. If the date and time stamp was not in a recognized format, no ABSTIME is returned.

Conditions

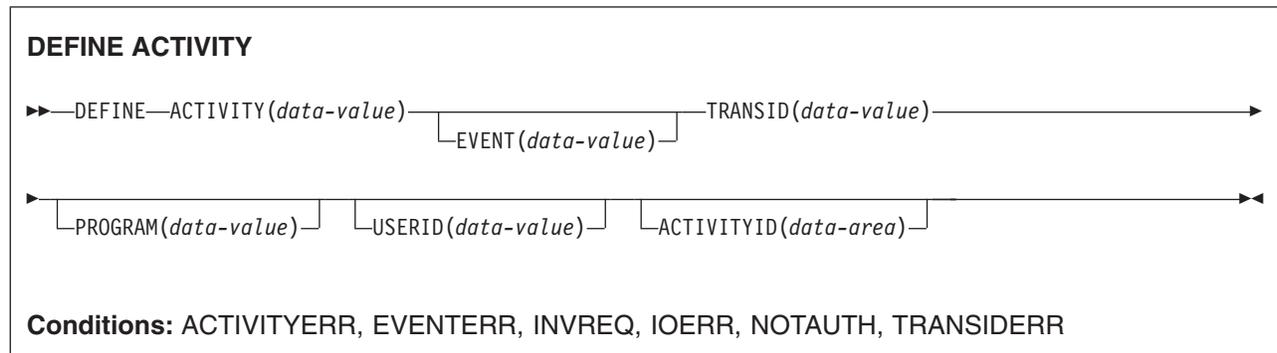
INVREQ

occurs for the following conditions. RESP2 values are:

- 1 Format of date and time stamp string not recognized. (This error can be caused by a year value that has more or less than the correct number of digits for the identified format.)
- 2 Invalid time.
- 3 Invalid month.
- 4 Invalid year (includes years before 1900).
- 5 Invalid day name.
- 6 Invalid day number for month and year specified.
- 7 GMT was not stated (required for RFC 1123 and RFC 850 formats).

DEFINE ACTIVITY

Define a CICS business transaction services activity.



Description

DEFINE ACTIVITY defines an activity to CICS business transaction services. It is used to add a child activity to the current activity.

The name of the program used in the execution of the new activity is taken either from the PROGRAM option, or, if PROGRAM is not specified, from the transaction definition pointed to by the TRANSID option.

The transaction attributes specified on the TRANSID and USERID options take effect when the activity is activated by a RUN command, but *not* if it is activated by a LINK command—see “Context-switching” on page 500.

BTS does not commit the addition of the activity until the requesting transaction has taken a successful syncpoint.

Options

ACTIVITY(data-value)

specifies the name (1–16 characters) of the new activity. The name must not be the name of another child activity of the activity that issues the DEFINE command.

The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _. Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

ACTIVITYID(data-area)

returns the 52-character identifier assigned by CICS to the newly-defined activity. This identifier is unique across the sysplex.

EVENT(data-value)

specifies the name (1–16 characters) of the completion event for the activity. The completion event is sent to the activity's parent when the activity completes.

If EVENT is not specified, the completion event is given the same name as the activity itself.

The acceptable characters are A-Z a-z 0-9 \$ @ # . - and _. Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

PROGRAM(data-value)

specifies the name (1–8 characters) of the program for the activity being defined. If no program is specified, the name is taken from the TRANSID definition.

TRANSID(data-value)

specifies the name (1–4 characters) of the transaction under which the activity is to run, when it is activated by a RUN command.

Note: If the activity is activated by a LINK command, it is run under the TRANSID of the transaction that issues the LINK.

The transaction must be defined in the CICS region in which the process is running.

USERID(data-value)

specifies the userid (1–8 characters) under whose authority the activity is to run, when it is activated by a RUN command.

Note: If the activity is activated by a LINK command, it is run under the userid of the transaction that issues the LINK.

The value of this field is known as the *defined userid*.

If you omit USERID, the defined userid defaults to the userid under which the transaction that issues the DEFINE command is running—we can call this the *command userid*.

If USERID is specified, CICS performs (at define time) a surrogate security check to verify that the command userid is authorized to use the defined userid. Thus, if you specify USERID, you must authorize the command userid as a surrogate user of the defined userid.

Conditions

ACTIVITYERR

RESP2 values:

- 3 The name specified on the ACTIVITY option has already been used to name another child of the current activity.

EVENTERR

RESP2 values:

- 7 The completion event specified on the EVENT option has already been defined to the current activity's event pool.

INVREQ

RESP2 values:

- 4 The DEFINE ACTIVITY command was issued outside the scope of a currently-active activity.
- 17 The activity name specified on the ACTIVITY option, or the event name specified on the EVENT option, is invalid.

IOERR

RESP2 values:

- 29 The repository file is unavailable.

30 An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

101 The user associated with the issuing task is not authorized to access the file associated with the BTS repository data set on which details of the activity are to be stored.

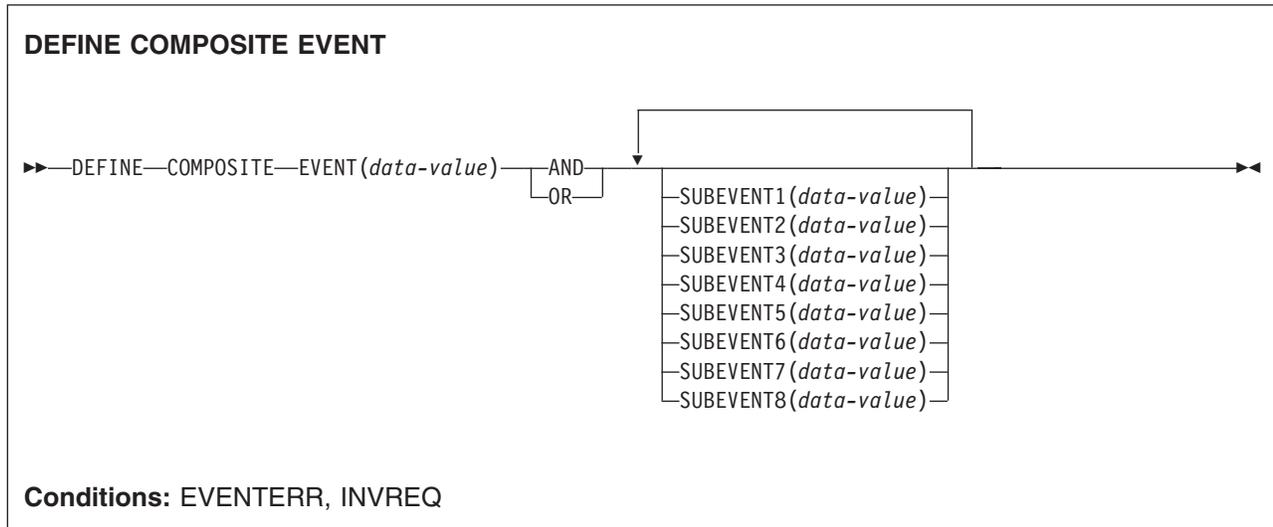
102 The user associated with the issuing task is not authorized as a surrogate of the defined userid specified on the USERID option.

TRANSIDERR

The transaction identifier specified on the TRANSID option cannot be found in the program control table.

DEFINE COMPOSITE EVENT

Define a BTS composite event.



Description

DEFINE COMPOSITE EVENT defines a composite event to BTS. A composite event is formed from zero or more atomic events known as sub-events.

DEFINE COMPOSITE EVENT defines a *predicate*, which is a logical expression involving sub-events. At all times, the composite event's fire status (FIRED or NOTFIRED) reflects the value of the predicate. When the predicate becomes true, the composite event fires; when it becomes false, the composite's fire status reverts to NOTFIRED.

The logical operator that is applied to the sub-events in the composite event's predicate is one of the Boolean operators AND or OR. *AND and OR cannot both be used.*

You can specify up to 8 sub-events to be added to the composite event when the composite is created. If you do not specify any sub-events, the composite event is defined as “empty”—that is, as containing no sub-events.

To add sub-events to a composite event after the composite has been defined, use the ADD SUBEVENT command. There is no limit to the number of sub-events that you can add using ADD SUBEVENT.

Note: The following *cannot* be added as sub-events to a composite event:

- Composite events
- System events
- Sub-events of other composite events
- Input events, if the composite uses the AND operator.

To remove sub-events from a composite event, use the REMOVE SUBEVENT command.

Options

AND

specifies that the Boolean operator to be associated with this composite's predicate is AND. This means that the composite event will fire when *all* of its sub-events have fired.

Note: The fire status of an empty composite event that uses the AND operator is always FIRED (true).

EVENT(data-value)

specifies the name (1–16 characters) of the composite event being defined. The acceptable characters are A-Z a-z 0-9 \$ @ # . - and _. Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

OR specifies that the Boolean operator to be associated with this composite's predicate is OR. This means that the composite event will fire when *any* of its sub-events fires.

Note: The fire status of an empty composite event that uses the OR operator is always NOTFIRED (false).

SUBEVENTn(data-value)

specifies the name (1–16 characters) of a sub-event to be added to the composite event when the composite is created. The acceptable characters are A-Z a-z 0-9 \$ @ # . - and _. Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

You can specify this option up to 8 times; *n* must be in the range 1–8.

The sub-events that you specify must previously have been defined to the current activity by means of DEFINE INPUT EVENT, DEFINE ACTIVITY, or DEFINE TIMER commands. They must not be sub-events of existing composite events.

Conditions

EVENTERR

RESP2 values:

- 6** The event name specified on the EVENT option is invalid.
- 7** The event name specified on the EVENT option has already been defined to this activity.
- 21–28** One or more of the sub-events named on the SUBEVENTn option does not exist. The RESP2 value indicates the first sub-event that does not exist.

INVREQ

RESP2 values:

- 1** The command was issued outside the scope of an activity.
- 31–38** One or more of the sub-events names specified on the SUBEVENTn option is invalid. The RESP2 value indicates the first invalid sub-event name.

DEFINE COUNTER and DEFINE DCOUNTER

Define a named counter.

DEFINE COUNTER

```
DEFINE COUNTER (name) [ POOL (name) ] [ VALUE (data-value) ] [ MINIMUM (data-value) ] [ MAXIMUM (data-value) ]
```

Conditions: INVREQ

DEFINE DCOUNTER

```
DEFINE DCOUNTER (name) [ POOL (name) ] [ VALUE (data-value) ] [ MINIMUM (data-value) ] [ MAXIMUM (data-area) ]
```

Conditions: INVREQ

Description

These counter commands create a new named counter in a named counter pool in the coupling facility.

DEFINE COUNTER creates counters that are handled as fullword *signed* binary values, and DEFINE DCOUNTER creates counters that are handled as doubleword *unsigned* binary values.

Note: Although the CICS API allows you to operate with either fullword (signed) or doubleword (unsigned) binary values, the named counter server stores *all* values as doubleword unsigned values. This can give rise to overflow conditions if, for example, you define a counter with the DCOUNTER command and try to access it using the COUNTER command. You should always access a named counter using commands from the *same command set* that you used to define the counter.

For information about specifying fullword and doubleword variables on these named counter commands, see “CICS command argument values” on page 3.

Options

COUNTER(*name*)

specifies the 16-character name of the named counter to be created. All value fields for this counter are handled as fullword signed binary values. Valid characters for names are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

DCOUNTER(*name*)

specifies the 16-character name of the named counter to be created. All value fields for this counter are handled as doubleword unsigned binary values. Valid characters for names are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

MAXIMUM(*data-value*)

specifies the maximum number for the named counter, using a fullword signed binary value for COUNTER and a doubleword unsigned value for DCOUNTER. This is the maximum number that can be assigned on a GET command, after which the counter must be reset by a REWIND command.

If you omit the MAXIMUM parameter, the named counter is defined with a default maximum of high-values (X'7FFFFFFF' for the signed fullword case, or a doubleword filled with X'FF').

MINIMUM(*data-value*)

specifies the minimum number for the named counter, using a fullword signed binary value for COUNTER and a doubleword unsigned value for DCOUNTER. This is the value to which a named counter is reset as a result of a REWIND command.

If you omit the MINIMUM parameter, the named counter is defined with a default minimum of low-values (a fullword or doubleword filled with X'00').

POOL(*name*)

specifies an 8-character string to be used as a pool selection parameter to select the pool in which the named counter is to be created. The string can be a logical pool name, or the actual pool name.

Valid characters for the pool selector string are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 8 characters, the name must be padded with trailing blanks.

This parameter is optional. If you omit the name of the pool, a pool selector value of 8 blanks is assumed.

If there is no matching entry in the DFHNCOPT options table, CICS uses the name specified on the NCPLDFT system initialization parameter, which specifies the default named counter pool.

For information about generating a named counter options table using the DFHNCO macro, see the *CICS Application Programming Guide*.

VALUE(*data-value*)

specifies the initial number at which the new named counter is to start, using a fullword signed binary value for COUNTER and a doubleword unsigned value for DCOUNTER.

You can specify a number that is equal to, or greater than, the minimum value, up to the maximum value plus 1. If you specify an initial number that is equal to

the maximum value plus 1, the counter is created with the counter-at-limit condition set and it cannot be used until it is rewind.

If you omit both the VALUE and MINIMUM parameters, the named counter is created with an initial value of zero. If you omit VALUE but specify a MINIMUM, the translator issues an error; the VALUE parameter is required if you specify the MINIMUM parameter.

Conditions

INVREQ

RESP2 values:

- 202** Duplicate counter name. A named counter of this name already exists.
- 301** The server has reported an error code that is not understood by the named counter interface. Generally, this is not possible unless the interface load module, DFHNCIF, is at a lower maintenance or release level than the server itself.
- 302** The server cannot create the new named counter because there is not enough space in the named counter pool.
- 303** An unexpected error, such as structure failure or loss of connectivity, has occurred on a macro used to access the coupling facility. Further information can be found in message DFHNC0441 in the CICS job log.
- 304** The pool selection parameter specified in the program cannot be resolved to a valid server name using the current options table.
- 305** The interface is unable to establish a connection to the server for the selected named counter pool. Further information can be found in an AXM system services message (AXMSC n n n) in the CICS job log.
- 306** An abend occurred during server processing of a request. Further information can be found in a message in the CICS job log and the server job log.
- 308** The DFHNCOPT options table module, required for resolving a pool name, could not be loaded.
- 309** During processing of the options table, the named counter interface encountered an unknown entry format. Either the options table is not correctly generated, or the DFHNCIF interface load module is not at the same release level as the options table.
- 310** An options table entry matching the given pool name specified a user exit program, but the user exit program is not link-edited with the options table and cannot be loaded.
- 311** A response from the named counter server to the client region interface module, DFHNCIF, indicates that a system-managed rebuild is in progress but the EXEC CICS interface does not recognize the condition. This means that the CICS region is at CICS TS 2.1 or earlier.
- 403** The POOL parameter contains invalid characters or embedded spaces.
- 404** The COUNTER parameter contains invalid characters or embedded spaces.
- 406** The VALUE parameter is invalid. Initial values cannot be less than the minimum value, and cannot be greater than the maximum value plus 1.
- 407** The MINIMUM or MAXIMUM parameter is invalid. Either the MAXIMUM

parameter specifies a value that is less than the minimum value, or (for COUNTER only) one of the parameters specifies a negative value.

Default action: terminate the task abnormally.

DEFINE INPUT EVENT

Define a BTS input event.

DEFINE EVENT

▶—DEFINE—INPUT—EVENT(*data-value*)————▶

Conditions: EVENTERR, INVREQ

Description

DEFINE INPUT EVENT defines an input event to BTS. Typically, an input event is passed to an activity by its parent, causing the activity to be activated. (Sometimes, however, the input event originates from outside the process.)

Most events fire on the completion of something, such as an activity or a specified time interval. An input event is different in that it fires after a RUN command that names it is issued.

An activity defines an input event in order to receive notification (via the INPUTEVENT option of the RUN or LINK ACTIVITY commands) of why it has been activated.

Note: System events such as DFHINITIAL are a special type of input event. They are recognized by all activities and do not need to be defined.

Options

EVENT(*data-value*)

specifies the name (1–16 characters) of the input event being defined. The acceptable characters are A-Z a-z 0-9 \$ @ # . - and _. Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

Conditions

EVENTERR

RESP2 values:

- 6** The event name specified on the EVENT option is invalid.
- 7** The event name specified on the EVENT option has already been defined to this activity.

INVREQ

RESP2 values:

- 1** The command was issued outside the scope of an activity.

DEFINE PROCESS

Define a CICS business transaction services process.

DEFINE PROCESS

```
▶▶—DEFINE—PROCESS(data-value)—PROCESSTYPE(data-value)—TRANSID(data-value)————▶▶
▶
┌PROGRAM(data-value)└┬USERID(data-value)└┬NOCHECK└▶▶
```

Conditions: INVREQ, IOERR, NOTAUTH, PROCESSERR, TRANSIDERR

Description

DEFINE PROCESS defines a BTS process. It:

- Adds a new process (for example, a new instance of a business transaction) to the CICS business transaction services system
- Creates the process's root activity.

The name of the program used in the execution of the new process is taken either from the PROGRAM option, or, if PROGRAM is not specified, from the transaction definition pointed to by the TRANSID option.

The transaction attributes specified on the TRANSID and USERID options take effect when the process is activated by a RUN command, but *not* if it is activated by a LINK command—see “RUN” on page 500.

BTS does not commit the addition of the process until the requesting transaction has taken a successful syncpoint.

Options

NOCHECK

specifies that no record is to be written to the repository data set to reserve the name of the process.

Note that the process name must be unique in the repository—see the PROCESS and PROCESSTYPE options—and that BTS does not commit the addition of the process until the requesting transaction has taken a successful syncpoint.

You can use this option to improve BTS performance by removing the write to the repository and its associated logging. However, if you do so be aware that the error of specifying a non-unique process name no longer causes a PROCESSERR condition to be returned on the DEFINE PROCESS command. The error may not be discovered until much later—when syncpoint occurs—making it much harder to debug.

PROCESS(*data-value*)

specifies a name (1–36 characters) to identify the new process (business transaction instance). The name must be unique within the BTS repository data set on which details of the process are to be stored—see the PROCESSTYPE option. For example, it is valid to issue a DEFINE command on which the

PROCESS option specifies a name that is currently in use by another process, *provided that* the PROCESSTYPE option maps to a different underlying repository data set from that on which the first process is defined.

The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are also permitted.

If the name is specified as a literal string that is less than 36 characters long, it is padded with trailing blanks up to 36 characters. If the name is specified as a variable whose value is less than 36 characters long, no padding occurs.

PROCESSTYPE(data-value)

specifies the type (1–8 characters) of the new process.

Each process-type maps to a VSAM data set (the repository), on which information about processes of the named type is stored. That is, information about the state of a process (and of its constituent activities) is stored on the repository associated with the process-type to which it belongs. Records for multiple process-types can be stored on the same repository data set.

You can categorize your processes by assigning them to different process-types.

PROGRAM(data-value)

specifies the name (1–8 characters) of the program for the process being added. If no program is specified, the name is taken from the TRANSID definition.

TRANSID(data-value)

specifies the name (1–4 characters) of the transaction under which the process is to run when it is activated by a RUN command.

Note: If the process is activated by a LINK command, it is run under the TRANSID of the transaction that issues the LINK.

The transaction must be defined in the CICS region in which the DEFINE PROCESS command is executed.

USERID(data-value)

specifies the userid (1–8 characters) under whose authority the process is to run when it is activated by a RUN command.

Note: If the process is activated by a LINK command, it is run under the userid of the transaction that issues the LINK.

The value of this field is known as the *defined userid*.

If you omit USERID, the defined userid defaults to the userid under which the transaction that issues the DEFINE command is running—we can call this the *command userid*.

If USERID is specified, CICS performs (at define time) a surrogate security check to verify that the command userid is authorized to use the defined userid. Thus, if you specify USERID, you must authorize the command userid as a surrogate user of the defined userid.

Conditions

INVREQ

RESP2 values:

12 The installed PROCESSTYPE is not enabled.

- 22** The unit of work that issued the DEFINE PROCESS command has already acquired an activity.

IOERR

RESP2 values:

- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

- 101** The user associated with the issuing task is not authorized to access the file associated with the BTS repository data set on which details of the process are to be stored.
- 102** The user associated with the issuing task is not authorized as a surrogate of the defined userid specified on the USERID option.

PROCESSERR

RESP2 values:

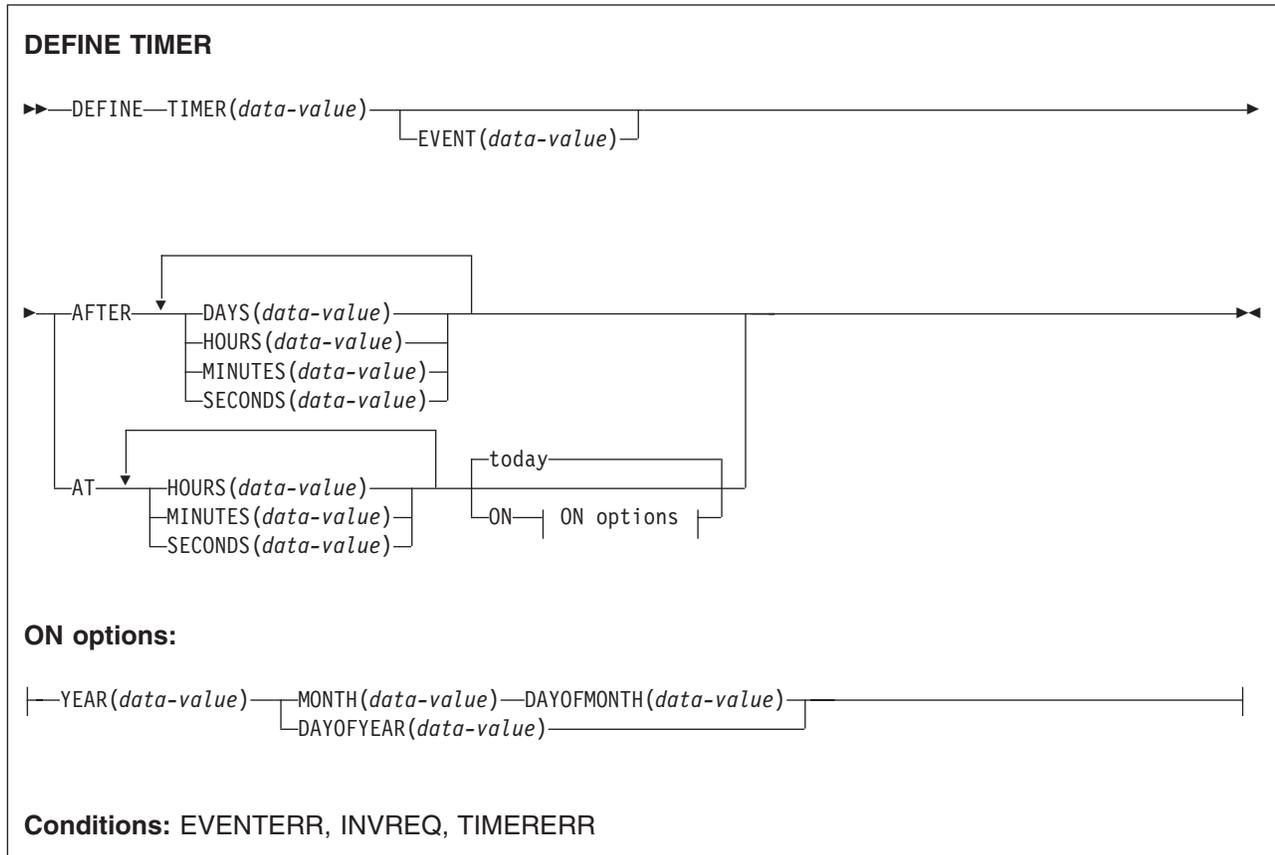
- 2** The process name specified on the PROCESS option is already in use on the BTS repository data set associated with the PROCESSTYPE option.
- 9** The process-type specified on the PROCESSTYPE option could not be found.
- 16** The process name specified on the PROCESS option contains an invalid character or characters.

TRANSIDERR

The transaction identifier specified on the TRANSID option cannot be found in the program control table.

DEFINE TIMER

Define a BTS timer.



Description

DEFINE TIMER defines a BTS timer which will expire after a specified interval, or at a specified time and date. When a timer is defined, an associated event is also defined, in the event pool of the current activity. The name of the associated event defaults to the name of the timer. When the timer expires, its associated event fires.

Note:

1. All dates and times refer to local time.
2. A timer that specifies a time and date that has already passed expires immediately. Similarly, if the requested interval is zero, the timer expires immediately.

Options

AFTER

specifies the interval of time that is to elapse before the timer is to expire.

You must specify one or more of DAYS(0–999), HOURS(0–23), MINUTES(0–59), and SECONDS(0–59). For example, HOURS(1) SECONDS(3) means one hour and three seconds (the minutes default to zero).

AT specifies the time at which the timer is to expire.

You must specify one or more of HOURS(0–23), MINUTES(0–59), and SECONDS(0–59). For example:

- HOURS(1) means 1 a.m.
- HOURS(15) MINUTES(15) means 3:15 p.m.
- MINUTES(15) means 0:15 a.m.

DAYOFMONTH(*data-value*)

specifies, as a fullword binary value in the range 1–31, the day-of-the-month on which the timer is to expire.

DAYOFYEAR(*data-value*)

specifies, as a fullword binary value in the range 1–366, the day-of-the-year on which the timer is to expire. For example, DAYOFYEAR(1) specifies 1st January.

DAYS(*data-value*)

specifies a fullword binary value in the range 0–999. This is a suboption of the AFTER option. For its use and meaning, see AFTER.

The default value is zero.

EVENT(**data-value**)

specifies the name (1–16 characters) of the event to be associated with the timer. The acceptable characters are A-Z a-z 0-9 \$ @ # . - and _. Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

The default event name is the name of the timer.

HOURS(*data-value*)

specifies a fullword binary value in the range 0–23. This is a suboption of the AFTER and AT options. For its use and meaning, see these options.

The default value is zero.

MINUTES(*data-value*)

specifies a fullword binary value in the range 0–59. This is a suboption of the AFTER and AT options. For its use and meaning, see these options.

The default value is zero.

MONTH(*data-value*)

specifies, as a fullword binary value in the range 1–12, the month in which the timer is to expire.

ON specifies the date at which the timer is to expire, as a combination of the YEAR, MONTH, DAYOFMONTH, and DAYOFYEAR options.

If the ON option is not specified, the default date is today.

SECONDS(*data-value*)

specifies a fullword binary value in the range 0–59. This is a suboption of the AFTER and AT options. For its use and meaning, see these options.

The default value is zero.

TIMER(*data-value*)

specifies the name (1–16 characters) of the timer. The acceptable characters are A-Z a-z 0-9 \$ @ # . - and _. Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

YEAR(*data-value*)

specifies, as a fullword binary value in the range 0–2040, the year in which the timer is to expire.

Conditions

EVENTERR

RESP2 values:

- 6 The event name specified on the EVENT option is invalid.
- 7 The event name specified on the EVENT option (or the default event name taken from the timer name) has already been defined to this activity.

INVREQ

RESP2 values:

- 1 The command was issued outside the scope of a currently-active activity.
- 11 An invalid interval was specified.
- 12 An invalid date or time was specified.

TIMERERR

RESP2 values:

- 14 The timer name specified on the TIMER option is invalid.
- 15 The timer name specified on the TIMER option has already been defined to this activity.

Examples

```
DEFINE TIMER() AT HOURS(15)
```

defines a timer that will expire at 3 p.m. today (or immediately if the local time is already later than 3 p.m.).

```
DEFINE TIMER() AT HOURS(15) ON YEAR(2001) MONTH(11) DAYOFMONTH(3)
```

defines a timer that will expire at 3 p.m. on 3rd November 2001.

```
DEFINE TIMER() AT HOURS(15) ON YEAR(2001) DAYOFYEAR(32)
```

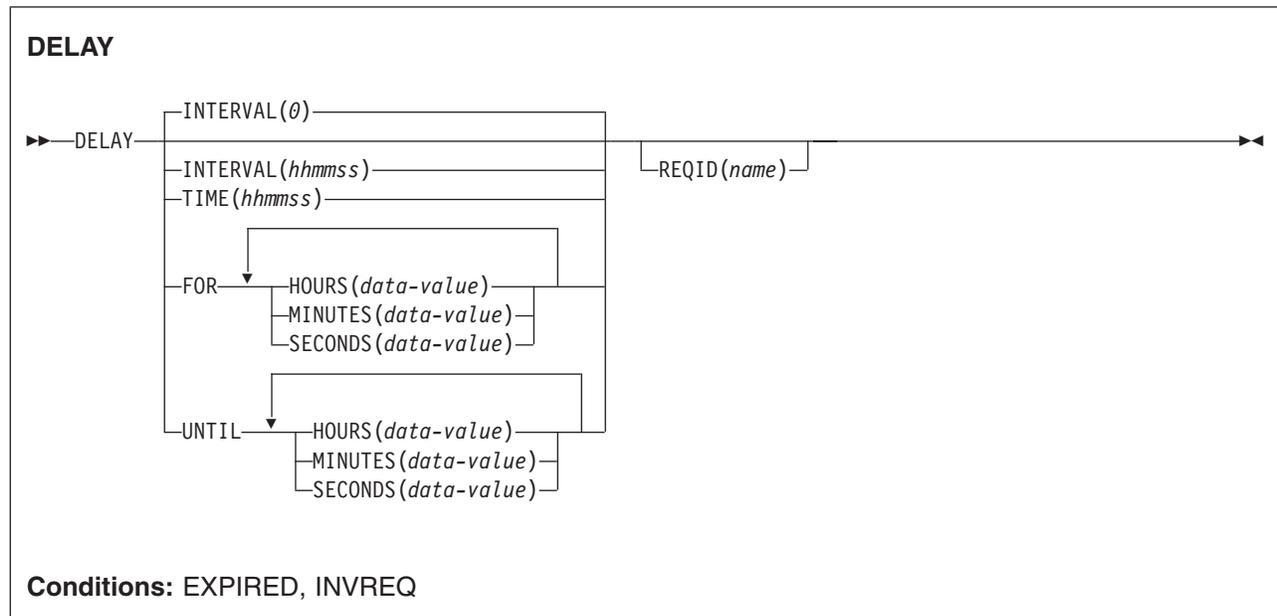
defines a timer that will expire at 3 p.m. on 1st February 2001.

```
DEFINE TIMER() AT HOURS(8) ON YEAR(1997) MONTH(1) DAYOFMONTH(1)
```

defines a timer that expires immediately.

DELAY

Delay the processing of a task.



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 Application Programming Guide* for more information about transaction affinities.

Description

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).

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.

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 Application Programming Guide*.

UNTIL

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

Conditions

EXPIRED

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

Default action: ignore the condition.

INVREQ

RESP2 values:

4 Hours are out of range.

5 Minutes are out of range.

6 Seconds are out of range.

also occurs (RESP2 not set) if the DELAY command is not valid for processing by CICS.

Default action: terminate the task abnormally.

Examples

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:

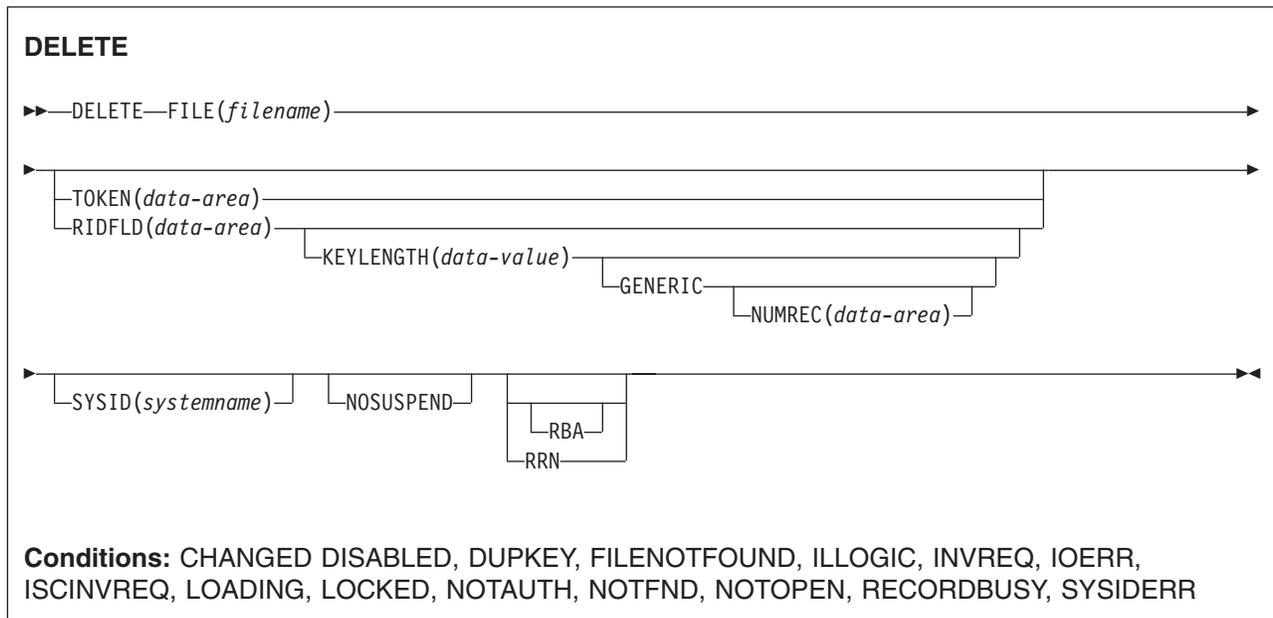
```
EXEC CICS DELAY  
  TIME(124500)  
  REQID('UNICODE')
```

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(0–359 999). HOURS(1) means one hour. MINUTES(62) means one hour and two minutes. SECONDS(3723) means one hour, two minutes, and three seconds.

DELETE

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



Description

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 or a BDAM file. **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 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.

When this command is used to delete records from a coupling facility data table, the update is made only to the data table in the coupling facility.

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 (*VSAM KSDS only*)

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.

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.

For remote files, the KEYLENGTH can be specified in the FILE definition. If KEYLENGTH is not defined there, and is not specified in the application program, and the key is longer than 4 characters, the default value is 4.

NOSUSPEND (*RLS only*)

specifies that the request is not to wait if VSAM is holding an active lock against the record, including records locked as the result of a DEADLOCK.

NUMREC(*data-area*) (*VSAM KSDS only*)

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

RBA

(VSAM KSDS 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 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
- Coupling facility data tables
- Any files opened in RLS access mode
- KSDS files that are capable of holding more than 4GB of data

RIDFLD(*data-area*)

specifies the record identification 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 or coupling facility data tables.

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

RRN (*VSAM RRDS only*)

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-area*)

Specifies, as a fullword binary value, a unique identifier for this DELETE request. Use this to associate the delete request with a record returned on a previous READ UPDATE or BROWSE for UPDATE request. The value to use is the value returned in the TOKEN held by the earlier READ UPDATE or BROWSE for UPDATE request.

TOKEN can be function shipped. However, if a request specifying TOKEN is function shipped to a member of the CICS family of products that does not recognize this option, the request fails.

Conditions

CHANGED

RESP2 values:

- 109** A DELETE command (without RIDFLD) is issued for a file that is a defined as a coupling facility data table using the contention update model and the record has been changed since the application program read it for update. To perform the DELETE successfully, repeat the read for update to get the latest version of the record, and try the DELETE again.

Default action: terminate the task abnormally.

DISABLED

RESP2 values:

- 50** A file is disabled. 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 DELETE follows any read with the UPDATE option.

Default action: terminate the task abnormally.

DUPKEY

RESP2 values:

- 140** A record is accessed by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key follows.

Default action: terminate the task abnormally.

FILENOTFOUND

RESP2 values:

- 1** The file name referred to in the FILE option cannot be found in the file resource definition.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values:

- 110** A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the EXEC interface block; refer to “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

- 20** Delete operations are not allowed according to the file entry specification in the FCT.
- 21** A DELETE command is issued for a file referring to a VSAM ESDS.
- 22** A generic delete is issued for a file that is not a VSAM KSDS.
- 25** 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.
- 26** 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.
- 27** A DELETE command is issued for a file referring to a BDAM data set.
- 31** A DELETE command without the RIDFLD option is issued for a file for which no previous READ UPDATE command has been issued.
- 42** The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than zero.
- 44** The DELETE command does not conform to the format of DELETE for a user-maintained or coupling facility data table; for example if RBA was specified.
- 47** A DELETE instruction includes a token whose value cannot be matched against any token in use for an existing read for UPDATE request.
- 51** A DELETE command specifying the RBA keyword is issued against a KSDS file that is being accessed in RLS mode. RLS does not support relative byte address (RBA) access to KSDS files.
- 55** NOSUSPEND is specified for a non-RLS file.
- 56** An attempt to update a recoverable coupling facility data table has failed because the current unit of work has already updated 1024 recoverable coupling facility data tables. You cannot update more than 1024 recoverable coupling facility data tables within a unit of work

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 120** There is an I/O error during the file control operation. 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 a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

(Further information is available in the EXEC interface block; refer to “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

ISCVREQ

RESP2 values:

- 70** The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LOADING

RESP2 values:

- 104** A delete request is issued for a user-maintained data table that is currently being loaded. A user-maintained data table cannot be modified during loading.

LOADING is also returned for a coupling facility data table if the delete request is for a key that is not yet loaded. A coupling facility data table can be modified during loading, but only if the requested key is within the range of those records already loaded.

The LOADING response can also be returned for a coupling facility data table that has failed during loading. For more information about what happens if the load for a coupling facility data table fails, see the description of the XD TLC global user exit in the *CICS Customization Guide*.

If your application programs encounter the LOADING condition persistently or too frequently, check that this is not caused by conflicting file definitions that reference the same data set.

Default action: terminate the task abnormally.

LOCKED

RESP2 values:

- 106** An attempt is made to delete a record specifying the RIDFLD, but a *retained* lock exists against this key (see “Retained and active locks” on page 134). If the request specifies the GENERIC keyword, all possible records are deleted, but the locked records remain. The number of records deleted is returned by NUMREC.

The LOCKED condition can also occur for a DELETE request to a recoverable CFDT that uses the locking model, if the record being read is locked by a retained lock. See the *CICS Recovery and Restart Guide* for information about investigating retained locks on records in a coupling facility data table.

Default action: abend the task with code AEX8.

NOTAUTH

RESP2 values:

- 101** A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

80 An attempt to delete a record based on the search argument provided is unsuccessful.

For user-maintained data and coupling facility 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.

This does not mean that there is no such record in the source data set (if the table was created from one); 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 data table.

For coupling facility data tables, this condition can also occur when a DELETE command (without a RIDFLD) is issued to a coupling facility data table using the contention model, and the record has been deleted since it was read for update.

Default action: terminate the task abnormally.

NOTOPEN

RESP2 values:

60 NOTOPEN (RESP2 60) is returned for one of the following reasons:

- 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. You can also make CLOSED, UNENABLED the initial state, by specifying STATUS(UNENABLED) and OPENTIME(FIRSTREF) on the FILE resource definition.
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.
- A DELETE command is issued against a data set that is quiesced, or is being quiesced, as a result of a SET DSNAME QUIESCED or IMMQUIESCED command.
- The requested file is CLOSED and ENABLED, so CICS has tried to open the file as part of executing the request. This file open has failed for some reason. You should examine the console for messages that explain why the file open has been unsuccessful.

This condition does not occur if the request is made to a CLOSED, DISABLED file. In this case, the DISABLED condition occurs.

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

Default action: terminate the task abnormally.

RECORDBUSY

RESP2 values:

107 The NOSUSPEND keyword is specified for the deletion of a record that is locked by a VSAM *active* lock (see “Retained and active locks” on page 134).

If the request specifies the GENERIC keyword, all possible records are deleted except for the locked records which remain. The number of records deleted is returned by NUMREC.

Default action: abend the task with code AEX9.

SYSIDERR

RESP2 values:

- 130** 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.
- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The DELETE is issued against a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

Retained and active locks

RECORDBUSY refers to active locks and LOCKED refers to retained locks:

- DELETE requests for records that have *retained* locks are always rejected with a LOCKED response.
- DELETE requests for records that have *active* locks wait for the lock to be released, except when the NOSUSPEND keyword is specified, in which case CICS returns the RECORDBUSY response.

Examples

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 ACTIVITY

Delete a BTS child activity.

DELETE

▶▶—DELETE—ACTIVITY(*data-value*)—▶▶

Conditions: ACTIVITYBUSY, ACTIVITYERR, INVREQ, IOERR, LOCKED

Description

DELETE ACTIVITY removes a child activity from the BTS repository data set on which it is defined. The child activity's completion event is removed from the parent's event pool. Any descendants of the child activity are also deleted.

The activity to be deleted must be a child of the activity that issues the DELETE command. To be eligible for deletion, the child activity must be in one of the following processing states (modes):

- COMPLETE—completed normally, abnormally, or previously canceled.
- INITIAL—not yet run, or reset by means of a RESET ACTIVITY command.

For a description of all possible processing states, see the *CICS Business Transaction Services* manual.

Note: A child activity that is not deleted explicitly by means of a DELETE ACTIVITY command is deleted automatically by CICS when its parent completes.

Options

ACTIVITY(*data-value*)

specifies the name (1–16 characters) of the child activity to be deleted.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19** The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8** The activity named on the ACTIVITY option could not be found.
- 14** The child activity named on the ACTIVITY option is not in COMPLETE or INITIAL mode, and is therefore ineligible for deletion.

INVREQ

RESP2 values:

- 4** The DELETE ACTIVITY command was issued outside the scope of a currently-active activity.

IOERR

RESP2 values:

29 The repository file is unavailable.

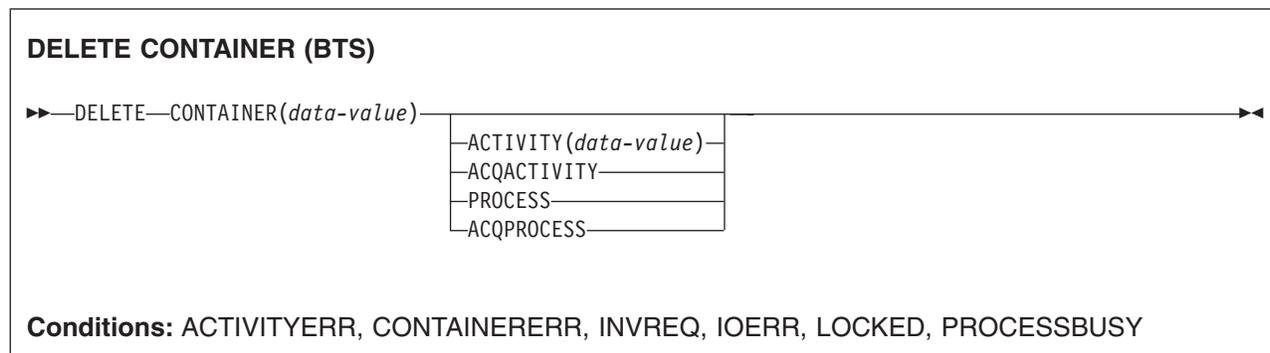
30 An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

DELETE CONTAINER (BTS)

Delete a named BTS data-container.



Description

DELETE CONTAINER (BTS) deletes a BTS data-container and discards any data that it contains.

The container is identified by name and by the process or activity for which it is a container—the process or activity that “owns” it. The activity that owns the container can be identified:

- Explicitly, by specifying one of the PROCESS- or ACTIVITY-related options.
- Implicitly, by omitting the PROCESS- and ACTIVITY-related options. If these are omitted, the current activity is implied.

Note: Process containers can be deleted only by the root activity or by a program that has acquired the process.

Options

ACQACTIVITY

specifies either of the following:

- If the program that issues the command has acquired a process, that the container is owned by the root activity of that process.
- Otherwise, that the container is owned by the activity that the program has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the container is owned by the process that the program that issues the command has acquired in the current unit of work.

ACTIVITY(*data-value*)

specifies the name (1–16 characters) of the activity that owns the container. This must be a child of the current activity.

CONTAINER(*data-value*)

specifies the name (1–16 characters) of the container to be deleted.

PROCESS

specifies that the container to be deleted is owned by the current process—that is, the process that the program that issues the command is executing on behalf of.

Conditions

ACTIVITYERR

RESP2 values:

- 8 The activity named on the ACTIVITY option could not be found.

CONTAINERERR

RESP2 values:

- 10 The container named on the CONTAINER option could not be found.
- 26 The process container named on the CONTAINER option is read-only. (Process containers can be deleted only by the root activity or by a program that has acquired the process.)

INVREQ

RESP2 values:

- 4 The command was issued outside the scope of a currently-active activity.
- 15 The ACQPROCESS option was used, but the unit of work that issued the request has not acquired a process.
- 24 The ACQACTIVITY option was used, but the unit of work that issued the request has not acquired an activity.
- 25 The PROCESS option was used, but the command was issued outside the scope of a currently-active process.

IOERR

RESP2 values:

- 30 An input/output error has occurred on the repository file.
- 31 The record on the repository file is in use.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

PROCESSBUSY

RESP2 values:

- 13 The request could not be satisfied because the process record is locked by another task.

DELETE CONTAINER (CHANNEL)

Delete a named channel container.

DELETE CONTAINER (CHANNEL)

►►—DELETE—CONTAINER(*data-value*)—┐
 └CHANNEL(*data-value*)┘◄◄

Conditions: CHANNELERR, CONTAINERERR, INVREQ

This command is threadsafe.

Description

DELETE CONTAINER (CHANNEL) deletes a container from a channel and discards any data that it contains.

The container is identified by name and by the channel for which it is a container—the channel that “owns” it. The channel that owns the container can be identified:

- Explicitly, by specifying the CHANNEL option.
- Implicitly, by omitting the CHANNEL option. If this is omitted, the current channel is implied.

Options

CHANNEL(*data-value*)

specifies the name (1–16 characters) of the channel that owns the container.

CONTAINER(*data-value*)

specifies the name (1–16 characters) of the container to be deleted.

Conditions

CHANNELERR

RESP2 values:

- 2** The channel specified on the CHANNEL option could not be found.

CONTAINERERR

RESP2 values:

- 10** The container named on the CONTAINER option could not be found.

INVREQ

RESP2 values:

- 4** The command was issued outside the scope of a currently-active channel.

- 30** You cannot delete a CICS-defined read only container.

DELETE COUNTER and DELETE DCOUNTER

Delete the named counter

DELETE COUNTER

►►—DELETE COUNTER(*name*)—┐
 └POOL(*name*)—┘

Conditions: INVREQ

DELETE DCOUNTER

►►—DELETE DCOUNTER(*name*)—┐
 └POOL(*name*)—┘

Conditions: INVREQ

Description

These commands delete the named counter from the specified pool. COUNTER operates on a fullword counter, and DCOUNTER operates on a doubleword counter.

Options

COUNTER(*name*)

specifies the name of the fullword counter to be deleted. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

DCOUNTER(*name*)

specifies the name of the doubleword counter to be deleted. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

POOL(*poolname*)

specifies an 8-character string to be used as a pool selection parameter to select the pool in which the named counter resides. The string can be a logical pool name, or the actual pool name.

Valid characters for the pool selector string are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 8 characters, the name must be padded with trailing blanks.

This parameter is optional. If you omit the name of the pool, a pool selector value of 8 blanks is assumed.

If there is no matching entry in the DFHNCOPT options table, CICS uses the name specified on the NCPLDFT system initialization parameter, which specifies the default named counter pool.

For information about generating a named counter options table using the DFHNCO macro, see the *CICS Application Programming Guide*.

Conditions

INVREQ

RESP2 values:

- 201** Named counter not found.
- 301** The server has reported an error code that is not understood by the named counter interface. Generally, this is not possible unless the interface load module, DFHNCIF, is at a lower maintenance or release level than the server itself.
- 303** An unexpected error, such as structure failure or loss of connectivity, has occurred on a macro used to access the coupling facility. Further information can be found in message DFHNC0441 in the CICS job log.
- 304** The pool selection parameter specified in the program cannot be resolved to a valid server name using the current options table.
- 305** The interface is unable to establish a connection to the server for the selected named counter pool. Further information can be found in an AXM services message (AXMSCnnnn) in the CICS job log.
- 306** An abend occurred during server processing of a request. Further information can be found in a message in the CICS job log and the server job log.
- 308** The DFHNCOPT options table module, required for resolving a pool name, could not be loaded.
- 309** During processing of the options table, the named counter interface encountered an unknown entry format. Either the options table is not correctly generated, or the DFHNCIF interface load module is not at the same release level as the options table.
- 310** An options table entry matching the given pool name specified a user exit program, but the user exit program is not link-edited with the options table and cannot be loaded.
- 311** A response from the named counter server to the client region interface module, DFHNCIF, indicates that a system-managed rebuild is in progress but the EXEC CICS interface does not recognize the condition. This means that the CICS region is at CICS TS 2.1 or earlier.
- 403** The pool selection parameter contains characters that are not allowed, or embedded spaces.

Default action: terminate the task abnormally.

DELETE EVENT

Delete a BTS event.

DELETE EVENT

▶▶—DELETE—EVENT(*data-value*)—▶▶

Conditions: EVENTERR, INVREQ

Description

DELETE EVENT deletes a BTS event that is no longer needed. The event is removed from the current activity's event pool. An event can be deleted whether it has fired or not.

DELETE EVENT can be used to delete only the following types of event:

- Input
- Composite.

DELETE EVENT *cannot* be used to delete:

- Activity completion events. These are implicitly deleted when a response from the completed activity is acknowledged by a CHECK ACTIVITY command issued by the activity's parent; or when a DELETE ACTIVITY command is issued.
- Timer events. These are implicitly deleted when the expiry of the associated timer is acknowledged by a CHECK TIMER command; or when a DELETE TIMER command is issued.
- System events.

Note:

1. If the event to be deleted is included in the predicate of a composite event, it is removed from the predicate's Boolean expression. The fire status of the composite event (FIRED or NOTFIRED) is re-evaluated.
2. Deleting a composite event has no effect on its sub-events.

Options

EVENT(*data-value*)

specifies the name (1–16 characters) of the event to be deleted.

Conditions

EVENTERR

RESP2 values:

- 4 The event specified on the EVENT option is not recognized by BTS.

INVREQ

RESP2 values:

- 1 The command was issued outside the scope of an activity.

- 2 The event specified on the EVENT option cannot be deleted because it is a system, timer, or activity completion event.

DELETE TIMER

Delete a BTS timer.

DELETE TIMER

▶—DELETE—TIMER(*data-value*)—▶

Conditions: INVREQ, TIMERERR

Description

DELETE TIMER deletes a BTS timer. If an event is associated with the timer, the event is also deleted and removed from the current activity's event pool. (There will be no event associated with the timer if the timer has expired and a CHECK TIMER command has been issued.)

The only timers a program can delete are those owned by the current activity. A timer can be deleted whether it has expired or not.

Options

TIMER(*data-value*)

specifies the name (1–16 characters) of the timer to be deleted.

Conditions

INVREQ

RESP2 values:

- 1 The command was issued outside the scope of a currently-active activity.

TIMERERR

RESP2 values:

- 13 The timer specified on the TIMER option does not exist.

DELETEQ TD

Delete all transient data.

DELETEQ TD

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

Conditions: DISABLED, INVREQ, ISCVREQ, LOCKED, NOTAUTH, QIDERR, SYSIDERR

Description

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.

Options

QUEUE(*name*)

specifies the symbolic name (1–4 alphanumeric characters) of the queue to be deleted. The named queue must be defined to CICS.

If SYSID is specified, the queue is assumed to be on a remote system, irrespective of how it is defined. Otherwise the resource definition is used to find out whether the queue 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.

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.

ISCVREQ

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

Default action: terminate the task abnormally.

LOCKED

occurs when the request cannot be performed because use of the queue has been restricted owing to a unit of work failing in-doubt. This can happen on any request for a logically-recoverable queue defined with WAIT(YES) and WAITACTION(REJECT) in the TDQUEUE resource definition.

Specify WAIT(YES) and WAITACTION(Queue) in the TDQueue resource definition if you want the transaction to wait.

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 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 TS

Delete a temporary storage queue.

DELETEQ TS

►►—DELETEQ TS—
└─┬─ QUEUE (*name*)
└─┬─ QNAME (*name*)
└─┬─ SYSID (*systemname*)
└─►►

Conditions: INVREQ, ISCINVREQ, LOCKED, NOTAUTH, QIDERR, SYSIDERR

This command is threadsafe.

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 Application Programming Guide* for more information about transaction affinities.

Description

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.

Options

QUEUE (*name*)

specifies the symbolic name (1–8 characters) of the queue to be deleted. The name may not consist solely of binary zeros and must be unique within the CICS system. If the name has less than 8 characters, you must still use an 8-character field, padded with blanks if necessary.

QNAME (*name*)

an alternative to QUEUE, QNAME specifies the symbolic name (1–16 characters) of the queue to be deleted. The name may not consist solely of binary zeros and must be unique within the CICS system. If the name has less than 16 characters, you must still use a 16-character field, padded with blanks if necessary.

SYSID (*systemname*)

(remote and shared queues only) specifies the system name (1–4 characters) identifying the remote system or shared queue pool to which the request is directed.

Conditions

INVREQ

occurs in either of the following situations:

- the queue was created by CICS internal code.

- the queue name specified consists solely of binary zeroes.

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.

LOCKED

occurs when the request cannot be performed because use of the queue has been restricted owing to a unit of work failing in-doubt.

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

Default action: terminate the task abnormally.

SYSIDERR

occurs in any of the following situations:

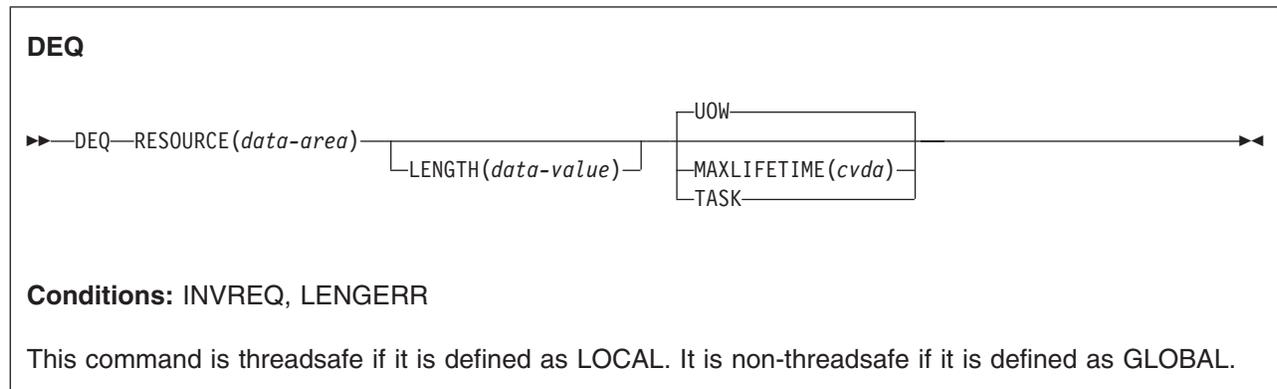
- When the SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION)
- When the link to the remote system is closed
- When the CICS region in which the temporary storage command is executed fails to connect to the TS server managing the TS pool that supports the referenced temporary storage queue. (For example, this can happen if the CICS region is not authorized to access the temporary storage server).

SYSIDERR can also occur if the temporary storage server has not been started, or because the server has failed (or been stopped) while CICS continues executing.

Default action: terminate the task abnormally.

DEQ

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



Note for dynamic transaction routing: Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing, unless the name specified in RESOURCE matches the name specified in an installed ENQMODEL resource definition, that has sysplex-wide scope. See the *CICS Application Programming Guide* for more information about transaction affinities.

Description

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.

When an EXEC CICS DEQ (or ENQ) command is issued for a resource whose name matches that of an installed ENQMODEL resource definition, CICS checks the value of the ENQSCOPE attribute to determine whether the scope is local or sysplex-wide. If the ENQSCOPE attribute is left blank (the default value), CICS treats the DEQ as local to the issuing CICS region. If no ENQMODEL matches the resource name, the scope of the DEQ command is local. See the *CICS Resource Definition Guide* for more information about the ENQMODEL resource definition.

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. CVDA values are:

UOW The enqueue was acquired with a duration of a unit of work. This is the default value.

Note: For compatibility with previous releases of CICS, a CVDA value of LUW is also supported.

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.

Conditions

INVREQ

RESP2 values:

2 The MAXLIFETIME option is set with an incorrect CVDA.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

1 The value you specified for the LENGTH option is outside the range 1 through 255.

Default action: terminate the task abnormally.

Examples

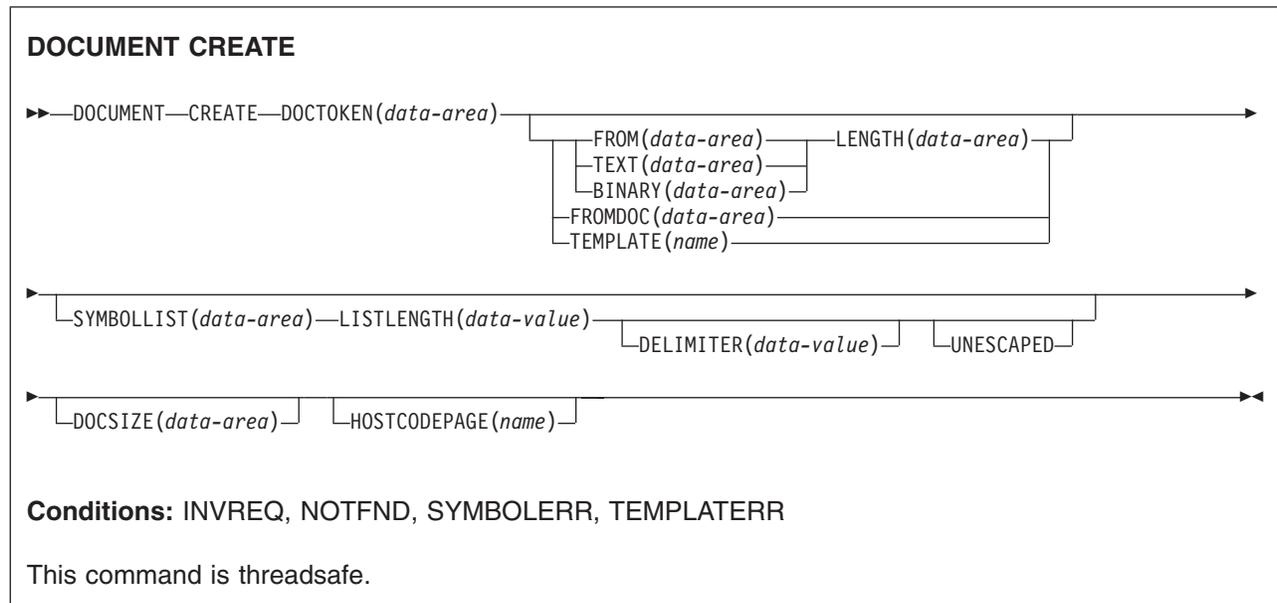
The following examples show how to dequeue from a resource:

```
EXEC CICS DEQ
      RESOURCE(RESNAME)

EXEC CICS DEQ
      RESOURCE(SOCSECNO)
      LENGTH(9)
```

DOCUMENT CREATE

Create a document.



Description

DOCUMENT CREATE signals the start of the document creation process. The document being created can be an empty document, or it can be based on an existing document, a template, or data contained in an application buffer.

Options

BINARY(*data-area*)

specifies a buffer of data which is to be used as the contents of the new document being created. The data is copied unchanged to the document content and no attempt is made to parse the data for symbol substitution. The purpose of the BINARY option is to allow the application to insert blocks of data that must not undergo conversion to a client code page when the data is sent. Use the LENGTH option to specify the length of this buffer.

DELIMITER(*data-value*)

specifies an optional 1-byte value used to delimit symbol name-value pairs in the SYMBOLLIST buffer. If this option is not specified, the value defaults to an ampersand. There are several disallowed DELIMITER values, all of which cause an INVREQ condition if used. The disallowed values are:

- null (binary X'00')
- shift in (binary X'0E')
- shift out (binary X'0F')
- space (binary X'40')
- plus sign (binary X'4E')
- colon (binary X'7A')
- equals (binary X'7E')
- percent sign (binary X'6C')

- backslash (binary X'E0')

If this option is used, the application must ensure that the DELIMITER does not appear in any symbol value in the SYMBOLLIST buffer. For this reason, the application should not use alphanumeric and other printable characters as the DELIMITER value.

DOCSIZE(data-area)

specifies a binary fullword area that will be updated with the current size of the document in bytes. This is the maximum size of the buffer needed to contain a copy of the document when a RETRIEVE command is issued.

DOCTOKEN(data-area)

specifies a data area to contain the symbolic name of the document. The area must be 16 bytes in length and will be set to a CICS-generated name by which the document can be referred to in later commands.

FROM(data-area)

specifies that data supplied by the application is to be used to create the contents of the new document. The data content could be a template or a document which was created and retrieved. If the data is a template, symbol substitution will take place where the symbols exist in the symbol table. If the data is a previously retrieved document, the conversion and bookmark tags which were inserted during retrieval will be removed from the content and stored in the internal format required by the API commands. Note that symbol substitution will not be attempted on any unresolved symbols contained in a retrieved document. Use the LENGTH option to specify the length of this buffer.

FROMDOC(data-area)

specifies the symbolic name (see the **DOCTOKEN** option) of a document whose contents are to be copied to the new document being created. All bookmark and conversion tags are copied to the new document. The symbol table will be not be copied.

HOSTCODEPAGE(name)

specifies the name of the host codepage that the data being added is encoded in. This option applies to the TEXT, SYMBOL and TEMPLATE options only. The name must be eight characters long; if it is shorter than eight characters it must be padded on the right with blanks.

LENGTH(data-value)

specifies the length, as a fullword binary value, of the buffer containing the TEXT, BINARY or FROM data.

LISTLENGTH(data-value)

specifies the length, as a fullword binary value, of the symbol list.

SYMBOLLIST(data-area)

specifies a buffer which contains a symbol list. Use the LISTLENGTH option to specify the length of this buffer. A symbol list is a character string consisting of one or more symbol definitions separated by ampersands. Each symbol definition consists of a name, an equals sign, and a value. By default, symbols in the symbol list are separated by the & character, but you can override this by using the DELIMITER keyword to specify a different symbol separator. Here is an example of a symbol list:

```
applid=IYCQ&jobname=test
```

The following rules apply when setting symbols using a SYMBOLLIST:

- The name is case-sensitive. It may only contain uppercase and lowercase letters, numbers, and the special characters dollar ('\$'), underscore ('_'),

hyphen ('-'), number sign ('#'), period ('.') and at sign ('@'). The name is case-sensitive, so uppercase letters are regarded as different from lowercase letters. Unlike the symbols in the template, the names in the symbol list have neither an ampersand at the beginning, nor a semicolon at the end. For example, the symbol &mytitle; in the template corresponds to the name *mytitle* in the symbol list.

- The values in the symbol list can contain any characters. However, special coding is required if you need to include the following characters in symbol values in the symbol list:
 - The character that you have used as the symbol separator (which defaults to an ampersand, but can be overridden by use of the DELIMITER option).
 - The plus sign and the percent sign.

You can use the percent sign, followed by two characters that are hexadecimal digits (that is, 0–9, a-f, and A-F), to include characters such as these that have a special meaning. When the value is put into the symbol table, a percent sign and the two hexadecimal digits following it are interpreted as the EBCDIC equivalent of the single ASCII character denoted by the two digits. %2B produces a plus sign, %25 produces a percent sign, and %26 produces an ampersand. If the characters following the percent sign are not two valid hexadecimal digits, the percent sign and the following characters are put into the symbol table as they appear in the symbol list.

If you prefer not to use this special coding, you can specify the UNESCAPED option. When you specify this option, no conversion takes place, and the symbol values are put into the symbol table exactly as you entered them. However, the UNESCAPED option does not allow you to include the character that you have used as the symbol separator within a symbol value in a symbol list. If you want to use the UNESCAPED option, choose a symbol separator that will never be used within a symbol value.

- If you want to include spaces in a value, CICS allows you to use the space character, a plus sign, or an escape sequence (%20). However, you **cannot** use a plus sign or escape sequence when the UNESCAPED option is used. In this case, you must only use a space character to indicate a space.

TEMPLATE (name)

specifies the 48-byte name of a template. The template must be defined to CICS using RDO. If the name is shorter than 48 bytes, it must be padded on the right with blanks.

Note: If you insert a template before the symbols contained in it are set, the symbols will never be substituted. This can occur if you create a document from a template without specifying a symbol list.

TEXT (data-area)

specifies a buffer of data which is to be used as the contents of the new document being created. The data is copied unchanged to the document content and no attempt is made to parse the data for symbol substitution. The data will be marked as requiring conversion to the client code page when the document is sent. Use the LENGTH option to specify the length of this buffer.

UNESCAPED

prevents CICS from unescaping symbol values contained in the SYMBOLLIST buffer. If this option is used, plus signs are not converted to spaces, and sequences such %2B are not converted to single byte values.

The UNESCAPED option does not allow you to include the character that you have used as the symbol separator within a symbol value in a symbol list. If you want to use the UNESCAPED option, choose a symbol separator that will never be used within a symbol value.

Conditions

INVREQ

RESP2 values are:

- 1 The retrieved document specified on the FROM option is not in a valid RETRIEVE format.

NOTFND

RESP2 values:

- 2 The document specified on the FROMDOC option could not be found or was named incorrectly.
- 3 The template specified on the TEMPLATE option could not be found or was named incorrectly.
- 7 The host codepage specified on the HOSTCODEPAGE option could not be found or was named incorrectly.
- 8 The value specified for DELIMITER is not valid.

SYMBOLERR

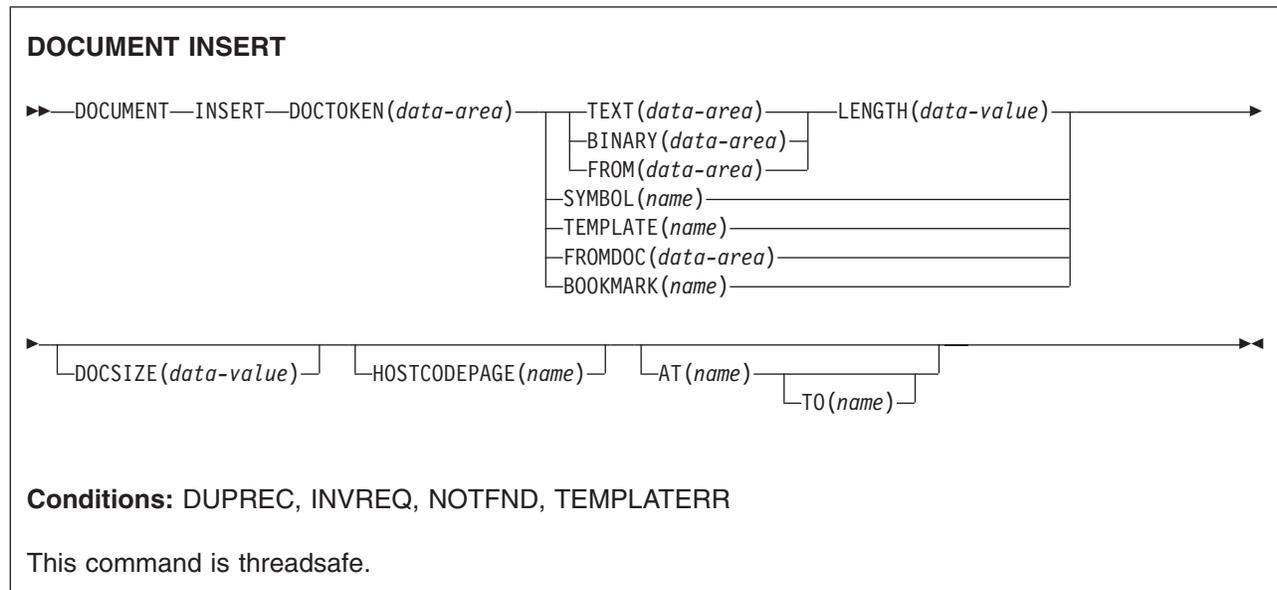
a symbol specified in the symbol list does not conform to the naming rules for symbols. RESP2 contains the offset of the symbol in the list.

TEMPLATERR

an invalid #set, #include or #echo command has been encountered while processing the supplied template data. RESP2 contains the offset of the invalid command.

DOCUMENT INSERT

Insert document objects.



Description

DOCUMENT INSERT allows the application to insert document objects at insertion points within the document. The insertion points (bookmarks) define relative positions within the document. Bookmarks must be defined before being referenced. Data is always inserted after the position identified by the bookmark.

Options

AT(*name*)

specifies the 16-byte symbolic name of a bookmark which identifies the position of the insertion point in the document. Data is inserted after the bookmark, and any data following the bookmark is shifted down. The application can use a combination of the AT and TO options to perform an overlay operation. If the AT operand is not specified, the data is inserted at the end of the document. A pre-defined bookmark of TOP is provided to allow the application to insert data at the beginning of the document.

BINARY(*data-area*)

specifies a buffer of data to be inserted into the document. The data is copied unchanged to the insertion point in the document, and no attempt is made to parse the data for symbol substitution. The BINARY option allows the application to insert blocks of data that must not undergo conversion to a client code page when the data is sent. Use the LENGTH option to specify the length of this buffer.

BOOKMARK(*name*)

specifies a bookmark to be inserted into the document. A bookmark is a

symbolic name which identifies an insertion point in the document. The name can be up to 16 characters in length, and must not contain any imbedded spaces.

DOCSIZE(*data-value*)

specifies a binary fullword area to be updated with the current size of the document in bytes. This is the maximum size of the buffer needed to contain a copy of the document when a RETRIEVE command is issued.

DOCTOKEN(*data-area*)

specifies the 16-byte symbolic name of the document into which data is to be inserted.

FROM(*data-area*)

specifies that a buffer of data supplied by the application is to be inserted into the document. The data content can be a template or a document that was previously created and retrieved. If the data is a template, symbol substitution takes place where the symbols exist in the symbol table. If the data is a previously retrieved document, the conversion and bookmark tags which were inserted during the retrieval will be removed from the content and stored in the internal form required by the API commands. Note that symbol substitution will not be attempted on any unresolved symbols contained in a retrieved document. Use the LENGTH option to specify the length of this buffer.

FROMDOC(*data-area*)

specifies the symbolic name of a document (see the DOCTOKEN option) whose contents are copied to the insertion point of the target document. All bookmarks and conversion tags are copied to the target document. The symbol table is not copied.

HOSTCODEPAGE(*name*)

specifies the symbolic name (see the DOCTOKEN option) of the host codepage that the data being added is encoded in. This option applies to the TEXT, SYMBOL and TEMPLATE options only. The name must be eight characters long; if it is shorter than eight characters, it must be padded on the right with blanks.

LENGTH(*data-value*)

specifies the length, as a fullword binary value, of the buffer containing the TEXT, BINARY or FROM data.

When the DOCUMENT INSERT command follows a DOCUMENT RETRIEVE command, without the use of the DATAONLY option, and the retrieved document is being inserted using the FROM option, the LENGTH specified must be equal to the length of the retrieved document.

SYMBOL(*name*)

specifies the 32-byte name of a symbol in the symbol table. The data associated with the symbol in the symbol table is inserted, but not the symbol itself. Note that when data associated with a symbol has been inserted into a document, you cannot change that data in the document that is being composed. If you set a different value for the symbol, the new value will be used the next time that symbol is inserted into a document. Your change will not affect the value that has already been inserted into the document.

TEMPLATE(*name*)

specifies the 48-byte name of a template. The template must be defined to CICS using RDO. If the name is less than 48 bytes, it must be padded on the right with blanks. The current values of any symbols are substituted into the template.

Note: When a template containing symbols has been inserted into a document, you cannot change the substituted values of those symbols in the document that is being composed. If you set different values for the symbols, the new values will be used the next time that the template is inserted into a document. Your changes will not affect the values that have already been inserted into the document.

TEXT (*data-area*)

specifies a buffer of data to be inserted into the document. The data is copied unchanged to the insertion point in the document, and no attempt is made to parse the data for symbol substitution. When the document is sent, it is marked as requiring conversion to the client code page. Use the LENGTH option to specify the length of this buffer.

TO (*name*)

specifies the symbolic name of a bookmark identifying the end position of an overlay operation. Data between the bookmarks identified by the AT and TO operands is deleted, and new data is inserted in its place. It is possible to delete data between two bookmarks by specifying a null string on the TEXT or BINARY option with a LENGTH of zero.

Conditions

DUPREC

the bookmark has already been defined.

INVREQ

RESP2 values are:

- 0 The bookmark specified on the TO option appears before the bookmark specified on the AT bookmark.
- 1 The retrieved document specified on the FROM option is not in a valid RETRIEVE format.
- 2 The bookmark name on the BOOKMARK option is invalid.

NOTFND

one of the following documents or templates could not be found, or its name was incorrect.

RESP2 values:

- 1 The document specified on the DOCUMENT option.
- 2 The document specified on the FROMDOC option.
- 3 The template specified on the TEMPLATE option.
- 4 The document specified on the SYMBOL option.
- 5 The document specified on the AT option.
- 6 The document specified on the TO option.
- 7 The document specified on the HOSTCODEPAGE option.

TEMPLATERR

an invalid #set, #include or #echo command has been encountered while processing the supplied template data. RESP2 contains either a zero (if the maximum of 32 levels of embedded templates is exceeded), or the offset of the invalid command.

MAXLENGTH(*data-value*)

specifies the length, as a fullword binary value, of the maximum amount of data the buffer can receive.

Conditions**LENGERR**

RESP2 values:

- 1 MAXLENGTH is less than or equal to zero. The document is truncated.
- 2 The length of the receiving buffer is zero, or is too short to contain the document contents. The document is truncated.

NOTFND

RESP2 values:

- 1 The document has not been created, or the name is incorrectly specified.
- 7 The specified client codepage can not be found.

- percent sign (binary X'6C')
- backslash (binary X'E0')

If this option is used, the application must ensure that the DELIMITER does not appear in any symbol value in the SYMBOLLIST buffer. For this reason, the application should not use alphanumeric and other printable characters as the DELIMITER value.

DOCTOKEN (*data-area*)

specifies the 16-byte symbolic name of the document that owns the symbol table.

LENGTH (*data-value*)

specifies the length, as a fullword binary value, of the buffer containing the data value associated with the symbol, or the length of the buffer containing the symbol list when the SYMBOLLIST option is used.

SYMBOL (*name*)

specifies the name of the symbol that is to be added to the table. The name can be 1 to 32 characters in length with no embedded spaces. The name of the symbol must contain only uppercase and lowercase letters, numbers and the special characters dollar ('\$'), underscore ('_'), hyphen ('-'), number sign ('#'), period ('.') and at sign ('@'). The name is case-sensitive, so uppercase letters are regarded as different from lowercase letters. If you want to define more than one symbol in the same command, use the SYMBOLLIST option instead.

SYMBOLLIST (*data-area*)

specifies a buffer which contains a symbol list. Use the LISTLENGTH option to specify the length of this buffer. A symbol list is a character string consisting of one or more symbol definitions separated by ampersands. Each symbol definition consists of a name, an equals sign, and a value. By default, symbols in the symbol list are separated by the & character, but you can override this by using the DELIMITER keyword to specify a different symbol separator. Here is an example of a symbol list:

```
applid=IYCQ&jobname=test
```

The following rules apply when setting symbols using a SYMBOLLIST:

- The name is case-sensitive. It may only contain uppercase and lowercase letters, numbers, and the special characters dollar ('\$'), underscore ('_'), hyphen ('-'), number sign ('#'), period ('.') and at sign ('@'). The name is case-sensitive, so uppercase letters are regarded as different from lowercase letters. Unlike the symbols in the template, the names in the symbol list have neither an ampersand at the beginning, nor a semicolon at the end. For example, the symbol &mytitle; in the template corresponds to the name *mytitle* in the symbol list.
- The values in the symbol list can contain any characters. However, special coding is required if you need to include the following characters in symbol values in the symbol list:
 - The character that you have used as the symbol separator (which defaults to an ampersand, but can be overridden by use of the DELIMITER option).
 - The plus sign and the percent sign.

You can use the percent sign, followed by two characters that are hexadecimal digits (that is, 0–9, a-f, and A-F), to include characters such as these that have a special meaning. When the value is put into the symbol table, a percent sign and the two hexadecimal digits following it are interpreted as the EBCDIC equivalent of the single ASCII character denoted

by the two digits. %2B produces a plus sign, %25 produces a percent sign, and %26 produces an ampersand. If the characters following the percent sign are not two valid hexadecimal digits, the percent sign and the following characters are put into the symbol table as they appear in the symbol list. If the UNESCAPED option is used, no conversion takes place, and all the characters are put into the symbol table as they appear in the symbol list.

- If you want to include spaces in a value, CICS allows you to use the space character, a plus sign, or an escape sequence (%20). However, you **cannot** use a plus sign or escape sequence to indicate a space character when the UNESCAPED option is used. In this case, you must only use a space character to indicate a space.

UNESCAPED

prevents CICS from unescaping symbol values contained in the SYMBOLLIST buffer. If this option is used, plus signs are not converted to spaces, and sequences such %2B are not converted to single byte values.

The UNESCAPED option does not allow you to include the character that you have used as the symbol separator within a symbol value in a symbol list. If you want to use the UNESCAPED option, choose a symbol separator that will never be used within a symbol value. Alternatively, you can use the SYMBOL and VALUE options to specify symbol values that contain the character you have used as the symbol separator, because the symbol separator has no special meaning when used in the VALUE option.

VALUE (*data-area*)

specifies an area containing the value to be associated with the SYMBOL.

The rules for including spaces in a symbol value in a symbol list also apply to the VALUE option: you can use a simple space character or a plus sign, unless the UNESCAPED option has been specified, in which case you must use a space character. Also, the special coding that is required to include a plus sign or percent sign in symbol lists is similarly required in the VALUE option, unless the UNESCAPED option has been specified. However, ampersands, or any other character that you have specified as a symbol separator for the symbol list, have no special significance when used in the VALUE option.

Conditions

INVREQ

RESP2 values:

- 8** The value specified for DELIMITER is not valid.

NOTFND

RESP2 values:

- 1** The document has not been created or the name is incorrectly specified.

SYMBOLERR

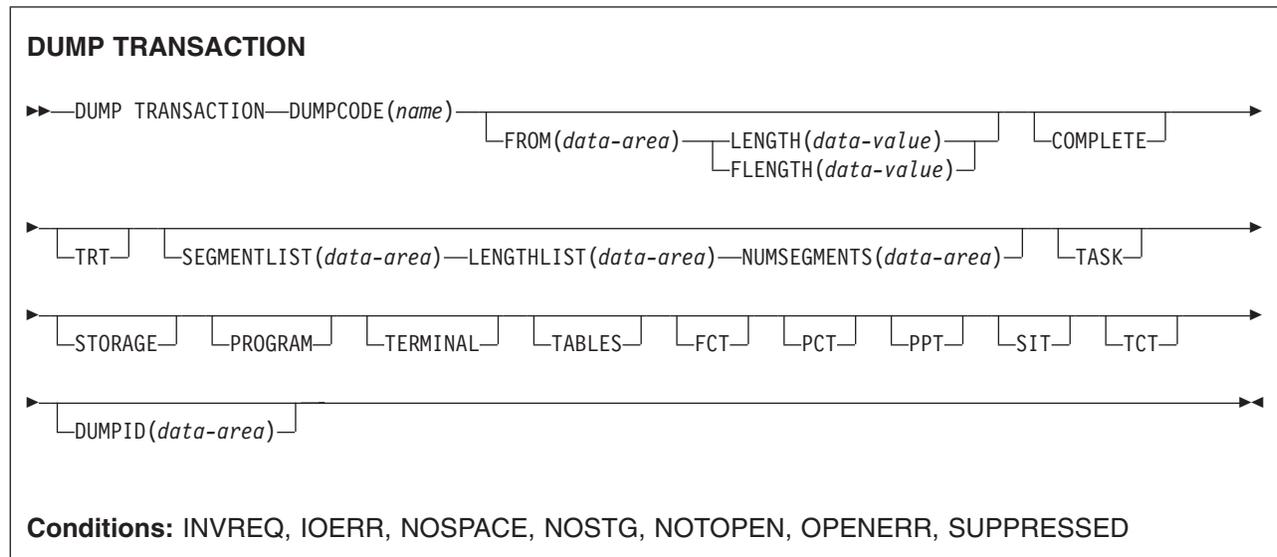
a symbol name is invalid. RESP2 values:

- 0** SYMBOLLIST was not used.

offset RESP2 contains the offset of the invalid symbol in the list.

DUMP TRANSACTION

Request a transaction dump.



Description

DUMP TRANSACTION dumps all, a series, or any single main storage area related to a task, any or all of the CICS tables (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.

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 Problem Determination Guide*.

Options

COMPLETE

dumps all main storage areas related to a task, all the CICS tables, and the DL/I control blocks.

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 dump as if you specified TASK, but without the DL/I control blocks.

DUMPID (*data-area*)

returns a 6- to 9-character dump identifier generated for this particular dump. The format of the identifier is *xxxx/yyyy*, where *xxxx* represents the **dump run number**, *yyyy* is the **dump count**, and the slash (/) symbol is a separator character. The dump identifier is generated as follows:

Dump run number

A number in the range 1 to 9999. (Leading zeros are not used for this number, which is why the dump id can vary from 6 to 9 characters.) The dump run number begins at 1 when you first start CICS with a newly-initialized local catalog, and is incremented by 1 each time you restart CICS. The dump run number is saved in the local catalog when you perform a normal shutdown, but is reset if you start CICS with a START=INITIAL or START=COLD system initialization parameter.

Dump count

A number in the range 0001 through 9999. (Leading zeros are required in the dump id.) This is the number assigned to the dump in this run of CICS, starting at 0001 for the first dump, and incremented by 1 with each dump taken.

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 16 777 215 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).
- If TRAN is specified for the TRTRAN TY SIT parameter, only the trace entries associated with the current task are formatted. If TRTRAN TY=ALL is specified, the entire internal trace table is formatted. This applies only 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 in CICS commands" on page 8.

LENGTH and FLENGTH are mutually exclusive.

LENGTHLIST (*data-area*)

specifies a list of 32-bit 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

umps 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

umps 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

umps the 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
- 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.

Conditions

INVREQ

RESP2 values:

- 13** An incorrect DUMPCODE is specified. DUMPCODE contains unprintable characters, or leading or imbedded blanks.

The dump is produced but no entry is added to the system dump table.

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 9** The SDUMP process is not authorized.

- 10** An error occurred during system dumping.

- 13** The CICS routine issuing the SDUMP is unable to establish a recovery routine (FESTAE).

Default action: terminate the task abnormally.

NOSPACE

RESP2 values:

4 The transaction dump is incomplete due to lack of space.

Default action: terminate the task abnormally.

NOSTG

RESP2 values:

5 CICS has run out of working storage.

Default action: terminate the task abnormally.

NOTOPEN

RESP2 values:

6 The current CICS dump data set is not open.

Default action: terminate the task abnormally.

OPENERR

RESP2 values:

7 There is an error on opening, closing, or writing to the current CICS dump routine.

Default action: terminate the task abnormally.

SUPPRESSED

RESP2 values:

1 The transaction dump is suppressed by MAXIMUM in table.

2 The transaction dump is suppressed by NOTRANDUMP in table.

3 The transaction dump is suppressed by a user exit program.

Default action: terminate the task abnormally.

Examples

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);

```

ENDBR

End browse of a file.

ENDBR

```

▶▶ ENDBR FILE(filename)
    ┌──REQID(data-value)──┐ ┌──SYSID(systemname)──┐

```

Conditions: FILENOTFOUND, ILLOGIC, INVREQ, IOERR, ISCINVREQ, NOTAUTH, SYSIDERR

Description

ENDBR ends a browse on a file or data table on a local or a remote CICS region.

The UPDATE option is available within browse so we recommend that you use this because otherwise you would need to issue an ENDBR command before using READ UPDATE to avoid self deadlock abends. We recommend issuing an ENDBR before syncpoint for similar reasons.

If STARTBR was not successful, you need not issue 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.

Conditions

FILENOTFOUND

RESP2 values:

1 The name referred to in the FILE option cannot be found in the FCT.

Default action: terminate the task abnormally.

ILLOGIC (VSAM)

RESP2 values:

110 A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the EXEC interface block; refer to “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

35 The REQID, if any, does not match that of any successful STARTBR command.

Default action: terminate the task abnormally.

IOERR

RESP2 values:

120 There is an I/O error during the file control operation. 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; see “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

ISCVREQ

RESP2 values:

70 The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

101 A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

SYSIDERR

RESP2 values:

130 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.

ENDBROWSE ACTIVITY

End a browse of the child activities of a BTS activity, or of the descendant activities
of a BTS process. Although this command appears in the Application Programming
set of topics, to use it you must specify the system programming (SP) parameter in
the EXEC statement of the translate step of your compile job. See Technote
1265081 for further information.

ENDBROWSE ACTIVITY

▶—ENDBROWSE—ACTIVITY—BROWSETOKEN(*data-value*)—▶

Conditions: ILLOGIC, TOKENERR

Description

ENDBROWSE ACTIVITY ends a browse of the child activities of a BTS activity (or of the descendant activities of a BTS process), and invalidates the browse token.

Options

BROWSETOKEN(*data-value*)

specifies, as a fullword binary value, the browse token to be deleted.

Conditions

ILLOGIC

RESP2 values:

- 1 The value specified in the BROWSETOKEN option matches a current browse token, but not one that is being used for an activity browse.

TOKENERR

RESP2 values:

- 3 The browse token is not valid.

ENDBROWSE CONTAINER

#

End a browse of the containers associated with a channel, or with a BTS activity or process. Although this command appears in the Application Programming set of topics, to use it you must specify the system programming (SP) parameter in the EXEC statement of the translate step of your compile job. See Technote 1265081 for further information.

ENDBROWSE CONTAINER

▶—ENDBROWSE—CONTAINER—BROWSETOKEN(*data-value*)—▶

Conditions: ILLOGIC, TOKENERR

Description

ENDBROWSE CONTAINER ends a browse of the containers associated with a channel, or with a BTS activity or process, and invalidates the browse token.

Options

BROWSETOKEN(*data-value*)

specifies, as a fullword binary value, the browse token to be deleted.

Conditions

ILLOGIC

RESP2 values:

- 1 The value specified in the BROWSETOKEN option matches a current browse token, but not one that is being used for a browse of containers.

TOKENERR

RESP2 values:

- 3 The browse token is not valid.

ENDBROWSE EVENT

End a browse of the events known to a BTS activity. Although this command
appears in the Application Programming set of topics, to use it you must specify the
system programming (SP) parameter in the EXEC statement of the translate step of
your compile job. See Technote 1265081 for further information.

ENDBROWSE EVENT

▶▶—ENDBROWSE—EVENT—BROWSETOKEN(*data-value*)—————▶◀

Conditions: TOKENERR

Description

ENDBROWSE EVENT ends a browse of the events that are within the scope of a BTS activity, and invalidates the browse token.

Options

BROWSETOKEN(*data-value*)

specifies, as a fullword binary value, the browse token to be deleted.

Conditions

TOKENERR

RESP2 values:

3 The browse token is not valid.

ENDBROWSE PROCESS

End a browse of processes of a specified type within the CICS business transaction
services system. Although this command appears in the Application Programming
set of topics, to use it you must specify the system programming (SP) parameter in
the EXEC statement of the translate step of your compile job. See Technote
1265081 for further information.

ENDBROWSE PROCESS

▶—ENDBROWSE—PROCESS—BROWSETOKEN(*data-value*)—————▶

Conditions: ILLOGIC, TOKENERR

Description

ENDBROWSE PROCESS ends a browse of the processes of a specified type within the CICS business transaction services system, and invalidates the browse token.

Options

BROWSETOKEN(data-value)

specifies, as a fullword binary value, the browse token to be deleted.

Conditions

ILLOGIC

RESP2 values:

- 1 The value specified in the BROWSETOKEN option matches a current browse token, but not one that is being used for a process browse.

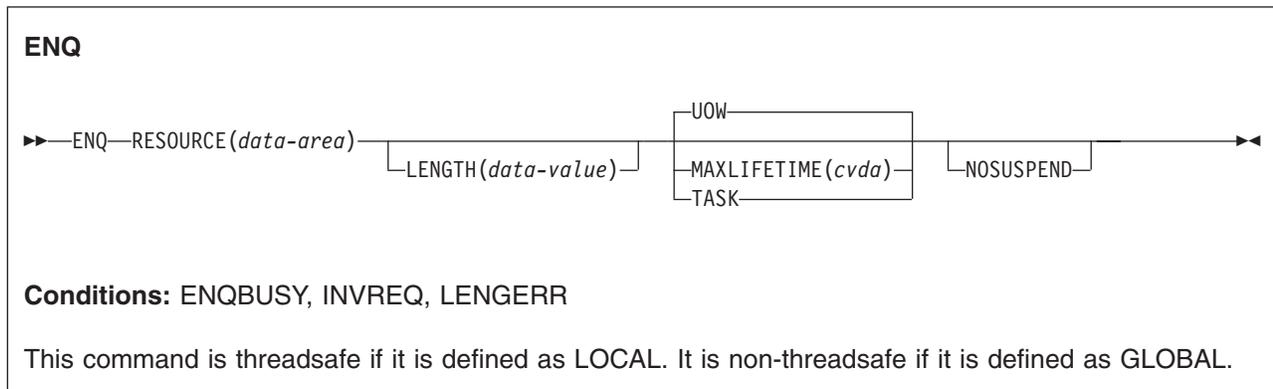
TOKENERR

RESP2 values:

- 3 The browse token is not valid.

ENQ

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



Description

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 UOW forces the dequeue at the end of a unit of work (UOW). Option TASK forces the dequeue at the end of a task. If there are several units of work in a task, the enqueue carries over the UOWs.

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.
- 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. LENGTH is required; the presence of the LENGTH option tells CICS to enqueue on the contents of the data value.

When an EXEC CICS ENQ (or DEQ) command is issued for a resource whose name matches that of an installed ENQMODEL resource definition, CICS checks the value of the ENQSCOPE attribute to determine whether the scope is local or sysplex-wide. If the ENQSCOPE attribute is left blank (the default value), CICS treats the ENQ as local to the issuing CICS region. If no ENQMODEL matches the resource name, the scope of the ENQ command is local. See the *CICS Resource Definition Guide* for more information about the ENQMODEL resource definition.

Resource unavailability

If a resource is not available when ENQ is issued, 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, as it is also raised if you have an active HANDLE condition. This allows the application program to handle the case of resource unavailability (by HANDLE CONDITION ENQBUSY) without waiting for the resource to become available.

Options

LENGTH(*data-value*)

specifies as a halfword binary value the length of the resource to be enqueued on. The value must be in the range 1 through 255; otherwise, the 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. 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. CVDA values are:

UOW The duration of the ENQ is a unit of work. Examples are a syncpoint rollback or syncpoint, if the application does not issue a DEQ before the unit of work ends. This is the default value.

Note: For compatibility with previous releases of CICS/ESA, a CVDA value of LUW is also supported.

TASK The duration of the ENQ is a task. The enqueue carries over the units of work 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(UOW) 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:
or

```
EXEC CICS ENQ RESOURCE(RESNAME) UOW
```

```
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.

Note, however, that if a HANDLE CONDITION for ENQBUSY is active when the command is executed, action, control is passed to the user label supplied in the HANDLE CONDITION. This takes precedence over the NOSUSPEND option but is, of course, negated by either NOHANDLE or RESP.

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.

Conditions

ENQBUSY

occurs when an ENQ command specifies a resource that is unavailable and the NOSUSPEND option is specified, or there is an active HANDLE CONDITION ENQBUSY.

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

RESP2 values: CVDA values are:

- 2 The MAXLIFETIME option is set with an incorrect CVDA.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 1 The value specified for the LENGTH option is outside the range 1 through 255.

Default action: terminate the task abnormally.

Examples

Two tasks, enqueueing on the same resource and specifying a data area that is the resource, 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)
```

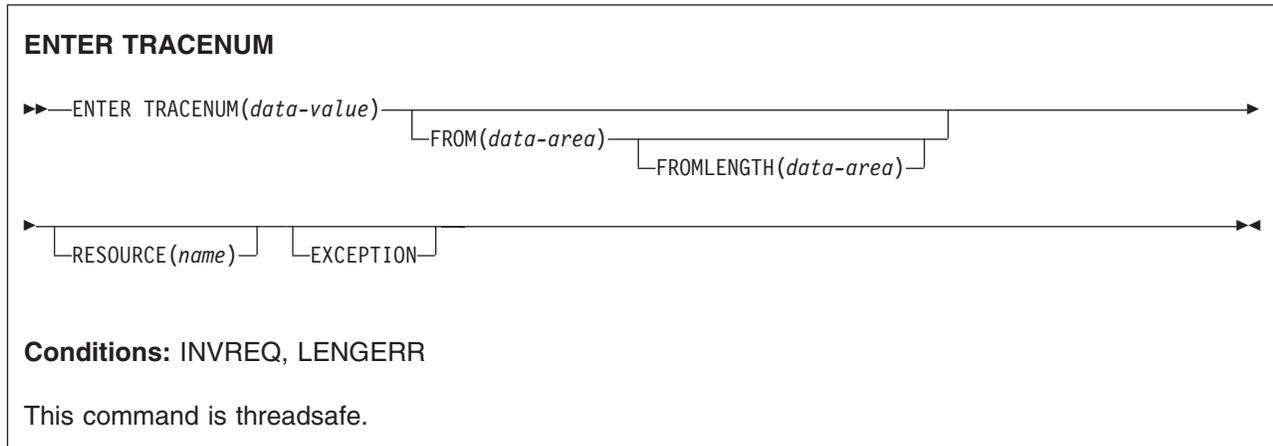
Two tasks, enqueueing on the same resource and specifying a data area that contains a unique argument, can refer to the same location or to different locations, but the contents of the locations must be the same. The length must be supplied in the LENGTH option.

```
EXEC CICS ENQ  
  RESOURCE(SOCSECNO)  
  LENGTH(9)
```

The two methods cannot be combined. If one task uses the LENGTH option, and
the other task does not, CICS regards the enqueues with and without the LENGTH
option as different types of enqueues, and the tasks are not serialized.

ENTER TRACENUM

Write a trace entry.



Description

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 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 *CICS Problem Determination Guide*.

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 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.

Conditions

INVREQ

RESP2 values:

- 1 TRACENUM is outside the range 0 through 199.
- 2 There is no valid trace destination.
- 3 The user trace flag is set OFF and EXCEPTION has not been specified.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 4 FROMLENGTH is outside the range 0 through 4000.

Default action: terminate the task abnormally.

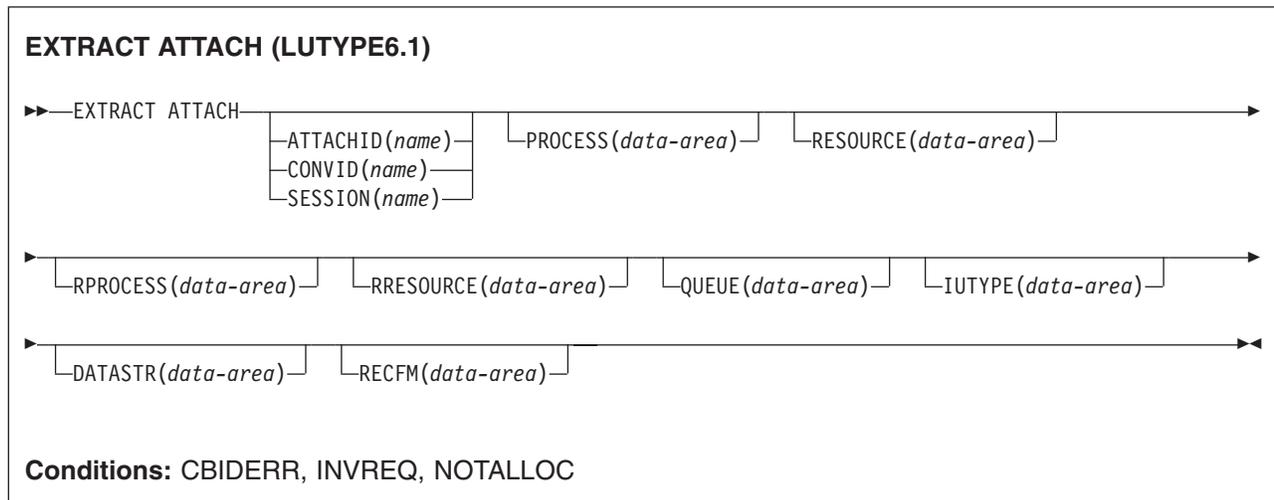
Examples

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.
```

EXTRACT ATTACH (LUTYPE6.1)

Retrieve values from an LUTYPE6.1 attach header.



Description

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 Function Management Header (FMH), EIBATT must have been set during a RECEIVE or CONVERSE command.

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.

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.

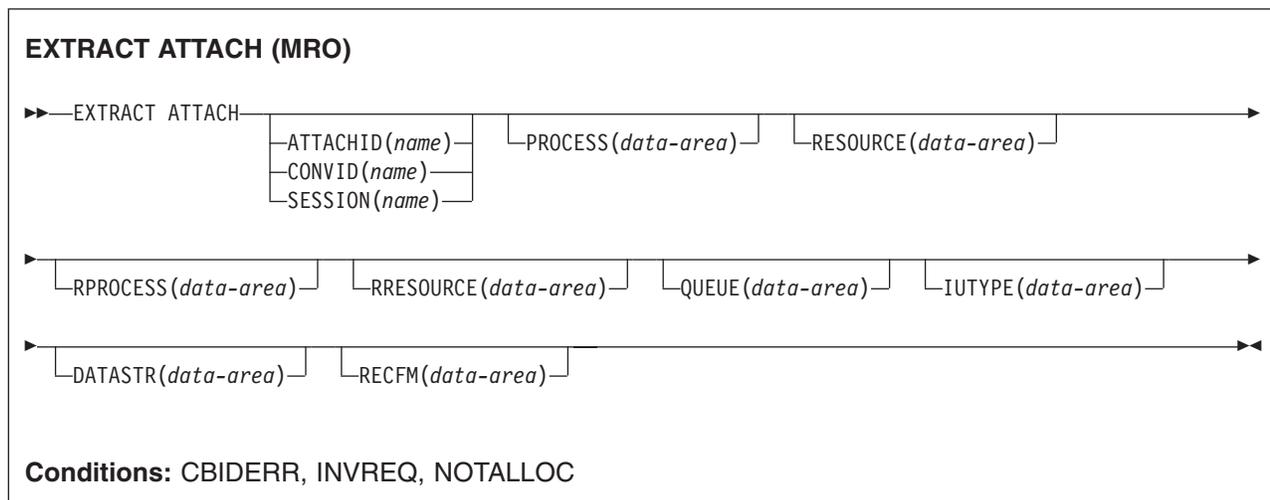
NOTALLOC

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

Default action: terminate the task abnormally.

EXTRACT ATTACH (MRO)

Retrieve values from an MRO attach header.



Description

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 Function Management Header (FMH), EIBATT must have been set during a RECEIVE or CONVERSE command.

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

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.

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.

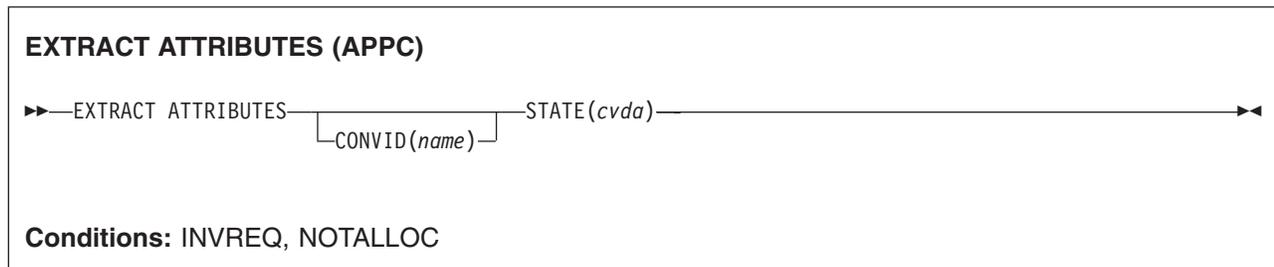
NOTALLOC

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

Default action: terminate the task abnormally.

EXTRACT ATTRIBUTES (APPC)

Obtain the state of the APPC conversation.



Description

EXTRACT ATTRIBUTES extracts conversation state information for APPC mapped conversations.

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

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application explicitly, or implicitly by default, specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) in any 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.)

Default action: terminate the task abnormally.

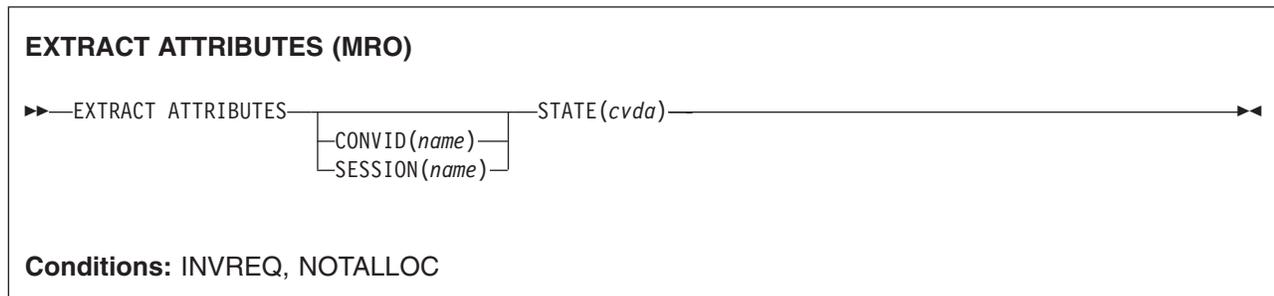
NOTALLOC

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

Default action: terminate the task abnormally.

EXTRACT ATTRIBUTES (MRO)

Extract attributes from an MRO conversation.



Description

EXTRACT ATTRIBUTES (MRO) extracts conversation state information for MRO conversations.

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
- SYNCSEND

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application explicitly, or implicitly by default, specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) in any of the following situations:

- An incorrect command has been issued for the terminal or LU in use.

Default action: terminate the task abnormally.

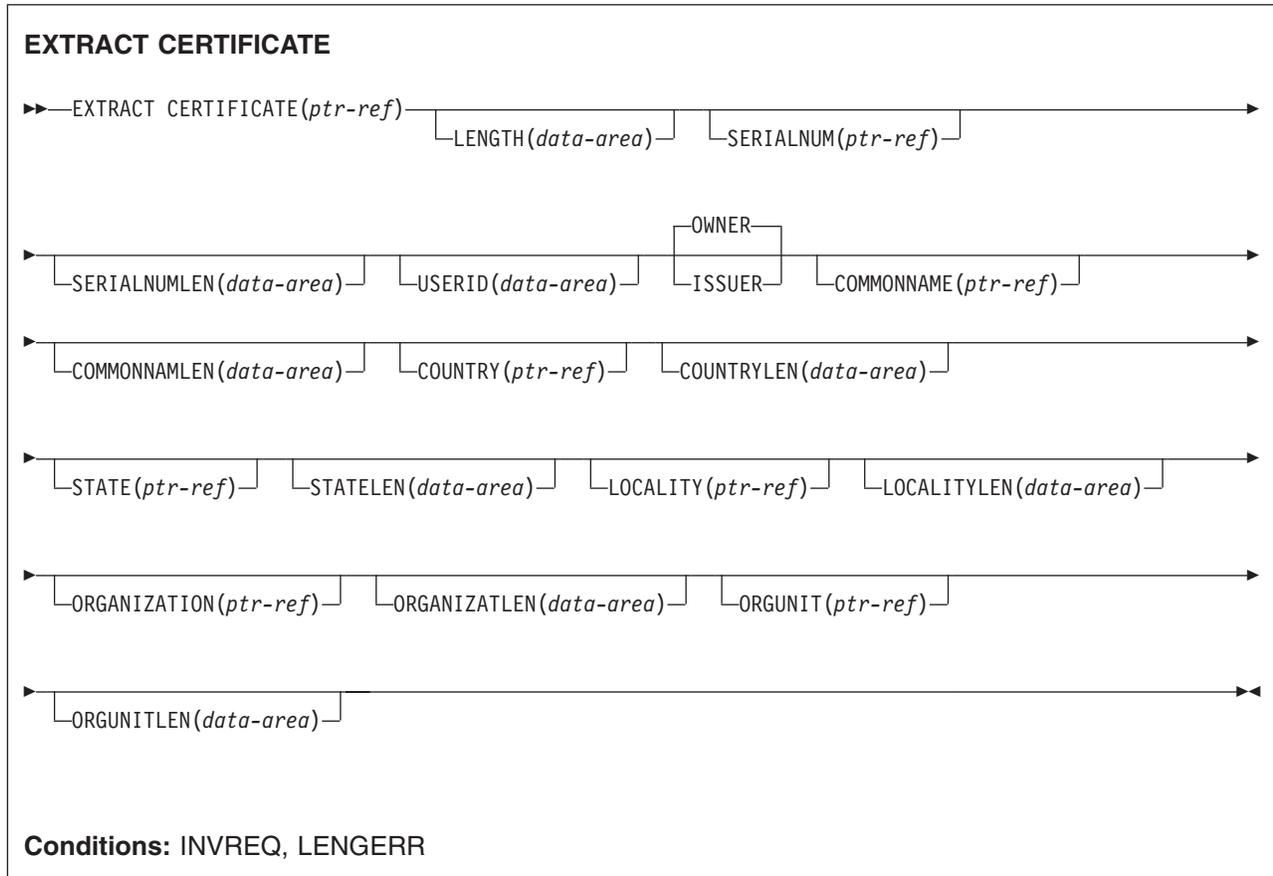
NOTALLOC

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

Default action: terminate the task abnormally.

EXTRACT CERTIFICATE

Obtain information from the client certificate received over a TCP/IP service that specified client authentication.



Description

EXTRACT CERTIFICATE allows the application to obtain information from the X.509 certificate that was received from a client during a Secure Sockets Layer (SSL) handshake over a TCPIP SERVICE that specified SSL(CLIENTAUTH). The certificate contains fields that identify the owner (or subject) of the certificate, and fields that identify the Certificate Authority that issued the certificate. You can select the fields that you require by specifying the OWNER or ISSUER option. You cannot retrieve both OWNER and ISSUER fields with one command.

Options

CERTIFICATE(ptr-ref)

specifies a pointer reference to be set to the address of the full binary certificate received from the client. The pointer reference is valid until the next CICS command or the end of task.

COMMONNAME(ptr-ref)

specifies a pointer reference to be set to the common name from the client certificate. The pointer reference is valid until the next CICS command or the end of task.

COMMONNAMLEN(data-area)

specifies a fullword binary data area to be set to the length of the common name from the client certificate.

COUNTRY(ptr-ref)

specifies a pointer reference to be set to the address of the country from the client certificate. The pointer reference is valid until the next CICS command or the end of task.

COUNTRYLEN(data-area)

specifies a fullword binary data area to be set to the length of the country from the client certificate.

ISSUER

indicates that the values returned by this command refer to the Certificate Authority that issued this certificate.

LENGTH(data-area)

specifies a fullword binary data area to be set to the length of the body of the client certificate.

LOCALITY(ptr-ref)

specifies a pointer reference to be set to the address of the locality from the client certificate. The pointer reference is valid until the next CICS command or the end of task.

LOCALITYLEN(data-area)

specifies a fullword binary data area to be set to the length of the locality from the client certificate.

ORGANIZATION(ptr-ref)

specifies a pointer reference to be set to the address of the organization from the client certificate. The pointer reference is valid until the next CICS command or the end of task.

ORGANIZATLEN(data-area)

specifies a fullword binary data area to be set to the length of the organization from the client certificate.

ORGUNIT(ptr-ref)

specifies a pointer reference to be set to the address of the organization unit from the client certificate. The pointer reference is valid until the next CICS command or the end of task.

ORGUNITLEN(data-area)

specifies a fullword binary data area to be set to the length of the organization unit from the client certificate.

OWNER

indicates that the values returned by this command refer to the owner of the certificate.

SERIALNUM(ptr-ref)

specifies a pointer reference to be set to the address of the serial number of the certificate assigned by the certificate issuer. The pointer reference is valid until the next CICS command or the end of task.

SERIALNUMLEN(data-area)

specifies a fullword binary data area to be set to the length of the serial number.

STATE(ptr-ref)

specifies a pointer reference to be set to the address of the state or province from the client certificate. The pointer reference is valid until the next CICS command or the end of task.

STATELEN(data-area)

specifies a fullword binary data area to be set to the length of the state or province from the client certificate.

USERID(data-area)

specifies an 8-byte field to be set to the user ID connected with the client certificate.

#

Conditions**INVREQ**

occurs for the following conditions:

- the command is being issued in a non-CICS Web Interface application.
- the command is being issued for a non-HTTP request.
- if an error occurs retrieving the certificate data from CICS intermediate storage.

LENGERR

the string being extracted is longer than the length specified for one of the options.

EXTRACT LOGONMSG

Access VTAM logon data.

EXTRACT LOGONMSG

▶▶—EXTRACT LOGONMSG—INTO(*data-area*)—LENGTH(*data-area*)—▶▶
 └──SET(*ptr-ref*)──┘

Condition: NOTALLOC

Description

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.

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.

Conditions

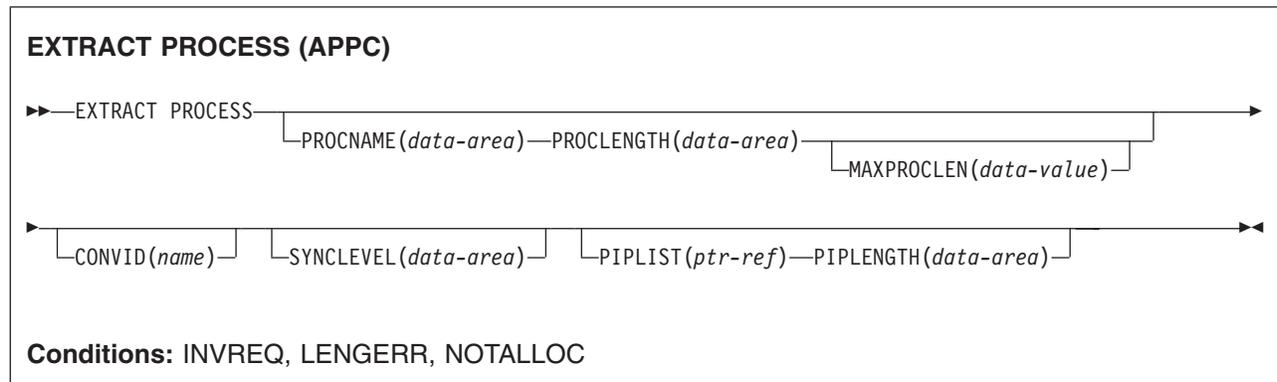
NOTALLOC

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

Default action: terminate the task abnormally.

EXTRACT PROCESS

Retrieve values from APPC conversation attach header.



Description

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.

Options

CONVID(*name*)

identifies the conversation to which the command relates. The 4-character name identifies the token representing the principal session (EIBTRMID).

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

If CONVID and SESSION are both omitted, the principal facility for the task is used by default.

MAXPROCLEN(*data-value*)

specifies the buffer length of PROCNAME. If MAXPROCLEN is not specified, the buffer is assumed to have 32 bytes.

PIPLENGTH(*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 Intercommunication Guide*.

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) 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.

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.

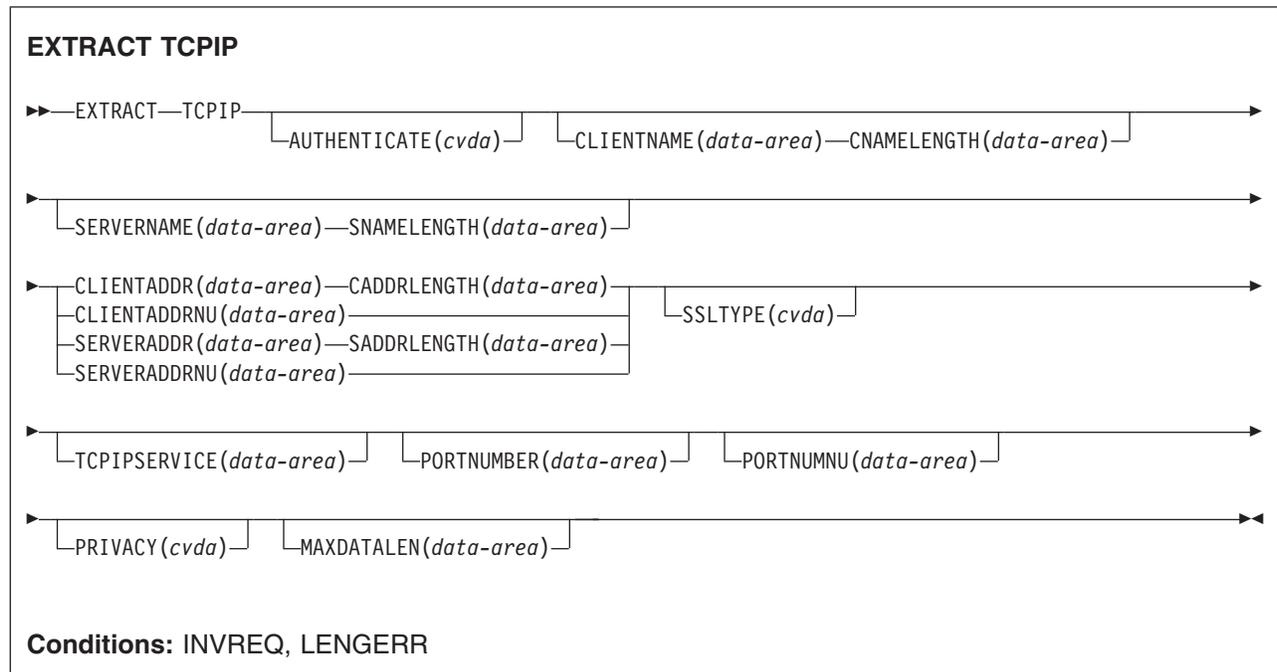
NOTALLOC

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

Default action: terminate the task abnormally.

EXTRACT TCPIP

Obtain information about TCPIP characteristics of the current transaction.



Options

AUTHENTICATE (cvda)

returns a CVDA indicating the authentication requested for the client using this transaction. Possible values are:

- ASSERTED
- AUTOAUTH
- AUTOREGISTER
- BASICAUTH
- CERTIFICAUTH
- NOAUTHENTIC

CADDRLENGTH (data-area)

specifies the length of the buffer supplied on the CLIENTADDR option, and is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

CLIENTADDR (data-area)

specifies a buffer to contain the client's TCP/IP address.

CLIENTADDRNU (data-area)

specifies a fullword binary field containing the client's TCP/IP address in binary form.

CLIENTNAME(data-area)

specifies a buffer to contain the client's name as known by the Domain Name Server.

CNAMELENGTH(data-area)

specifies the length of the buffer supplied on the CLIENTNAME option, and is set to the actual length of the data returned to the application, or zero if the client's name is not known to the domain name server. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

MAXDATALEN(data-area)

specifies a fullword binary field to contain the setting for the maximum length of data that can be received by CICS as an HTTP server.

PRIVACY(cvda)

returns a CVDA indicating the level of SSL encryption used between the transaction and its client for an inbound IIOF request. Possible values are:

- NOTSUPPORTED
- REQUIRED
- SUPPORTED

PORTNUMBER(data-area)

Specifies a 5-character field to contain the port number associated with this transaction in character form. This is the port on which the incoming data that initiated this transaction was received.

PORTNUMNU(data-area)

Fullword field to contain the port number associated with this transaction in binary form. This is the port on which the incoming data that initiated this transaction was received.

SADDRLENGTH(data-area)

specifies the length of the buffer supplied on the SERVERADDR option, and is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

SERVERADDR(data-area)

specifies a buffer to contain the server's TCP/IP address in dotted decimal character form (nnn.nnn.nnn.nnn).

SERVERADDRNU(data-area)

specifies a fullword binary field containing the server's TCP/IP address in binary form.

SERVERNAME(data-area)

specifies a buffer to contain the server's name as known by the Domain Name Server.

SNAMELENGTH(data-area)

specifies the length of the buffer supplied on the SERVERNAME option, and is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

SSLTYPE(CVDA)

Returns a CVDA indicating whether the Secure Sockets Layer (SSL) is being used to secure communications for this transaction. Possible values are:

- SSL
- NOSSL

- CLIENTAUTH

TCPIPSERVICE(data-area)

An 8-byte field to contain the name of the TCPIPSERVICE associated with this transaction.

Conditions**INVREQ**

RESP2 values:

- 2** An invalid socket response was received.
- 5** The command was issued from a non-TCPIP application.

LENGERR

RESP2 values:

- 1** CLIENTADDR, SERVERADDR, CLIENTNAME or SERVERNAME is specified, but the relevant length field is either not specified, or it is less than or equal to zero.
- 3** CLIENTADDR is too small to contain the string extracted.
- 4** SERVERADDR is too small to contain the string extracted.
- 6** CLIENTNAME is too small to contain the string extracted.
- 7** SERVERNAME is too small to contain the string extracted.

EXTRACT TCT

Convert an 8-character name to a 4-character name on an LUTYPE6.1 logical unit.

EXTRACT TCT

▶▶—EXTRACT TCT—NETNAME(*name*)—
 —SYSID(*systemname*)—
 —TERMID(*data-area*)—▶▶

Condition: INVREQ, NOTALLOC

Description

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.

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.

Conditions

INVREQ

occurs if NETNAME is not valid.

Default action: terminate the task abnormally.

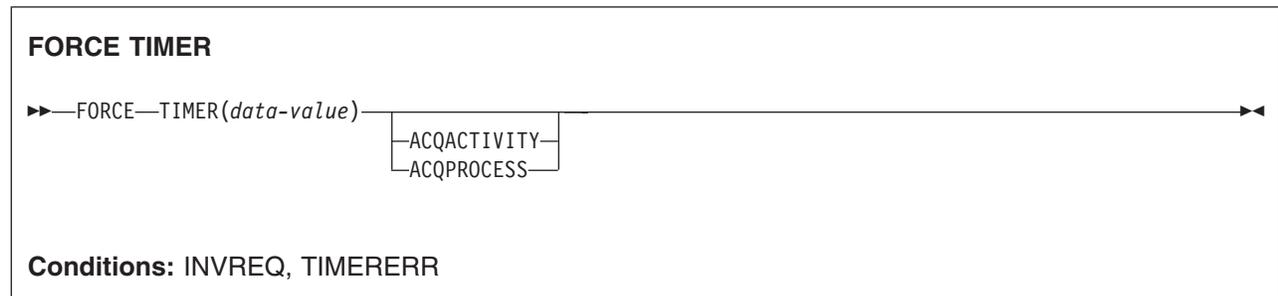
NOTALLOC

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

Default action: terminate the task abnormally.

FORCE TIMER

Force the early expiry of a BTS timer.



Description

FORCE TIMER forces a BTS timer that has not yet expired to expire immediately. This causes the event associated with the timer to fire.

If the timer has already expired, the command has no effect.

The activity that owns the timer can be identified:

- Explicitly, by specifying either the ACQPROCESS or ACQACTIVITY option.
- Implicitly, by omitting the ACQPROCESS and ACQACTIVITY options. If these are omitted, the current activity is implied.

Options

ACQACTIVITY

specifies either of the following:

- If the program that issues the command has acquired a process, that the timer is owned by the root activity of that process.
- Otherwise, that the timer is owned by the activity that the program has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the timer is owned by the process that the program that issues the command has acquired in the current unit of work.

TIMER(*data-value*)

specifies the name (1–16 characters) of the timer to be forced.

Conditions

INVREQ

RESP2 values:

- | | |
|----|--|
| 1 | The command was issued outside the scope of a currently-active activity. |
| 16 | The ACQPROCESS option was specified, but there is no acquired process. |
| 17 | The ACQACTIVITY option was specified, but there is no acquired activity. |

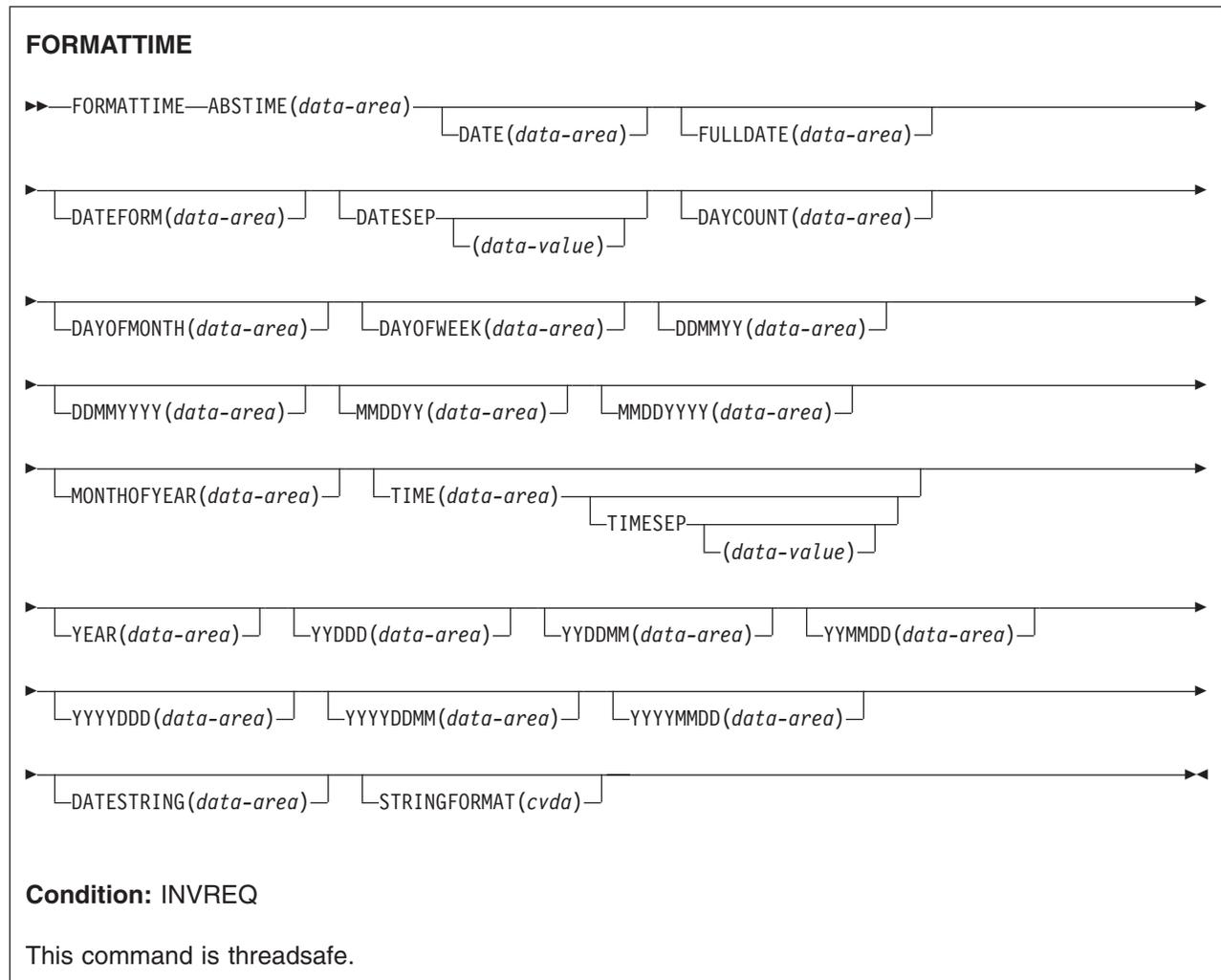
TIMERERR

RESP2 values:

- 13** The timer named on the TIMER option does not exist.

FORMATTIME

Transform absolute date and time into a specified format.



Description

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.

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. 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.

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.

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. 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. 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.

DATESTRING(*data-area*)

specifies the 64-character user field where CICS is to return the architected date and time stamp string in the format specified by the STRINGFORMAT option. If STRINGFORMAT is not specified, the default format provided is the RFC 1123 format (RFC1123).

Note: If you are using the DATESTRING option, run the ASKTIME ABSTIME command first to obtain a value for the ABSTIME option. If the value for the ABSTIME option is from any other source, the architected date and time stamp string which is returned by the FORMATTIME command might be incorrect.

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, 21/10/98). 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.

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). 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.

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.

STRINGFORMAT(*cvda*)

specifies the format for the architected date and time stamp string returned in DATESTRING. The only CVDA value available at present is:

RFC1123

specifies the RFC 1123 format, which is suitable for use on the Internet. This date and time stamp string contains the day, date, and 24-hour clock time at GMT, for example "Tue, 01 Apr 2003 10:01:02 GMT".

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.

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.

YYMDD(*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). 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.

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 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.

YYYYMDD(*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.

Conditions

INVREQ

RESP2 values:

- 1 The ABSTIME value is less than zero or not in packed-decimal format.

Default action: terminate the task abnormally.

Examples

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 FORMATIME ABSTIME(utime)
          DATESEP('-') DDMMYY(date)
          TIME(time) TIMESEP
```

This gives the values 06-12-89 for “date” and 19:01:05 for “time”.

FREE

Return a terminal or logical unit.

FREE

▶▶—FREE—▶▶

Condition: NOTALLOC

Description

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.

Conditions

NOTALLOC

occurs if the task is not associated with the terminal.

Default action: terminate the task abnormally.

- SYNCSEND

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) 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.

Default action: terminate the task abnormally.

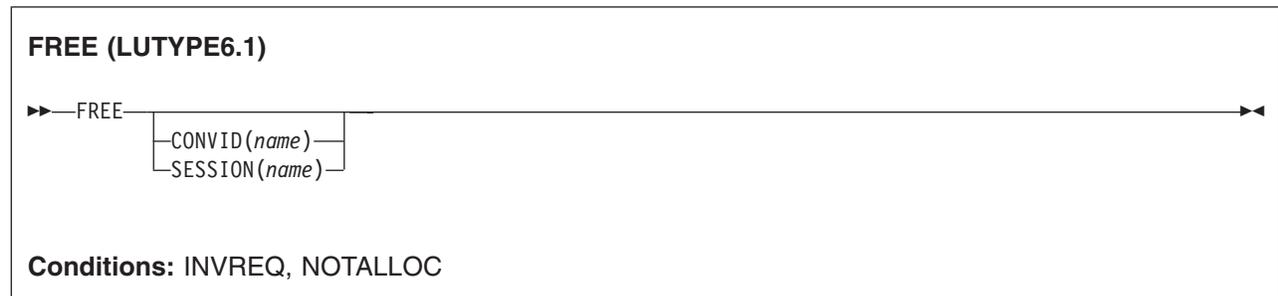
NOTALLOC

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)

Return LUTYPE6.1 sessions to CICS.



Description

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.

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.

Conditions

INVREQ

occurs if the session specified in the command was allocated for a basic (unmapped) APPC conversation.

(See also EIBRCODE in “EXEC interface block” on page 745.)

Default action: terminate the task abnormally.

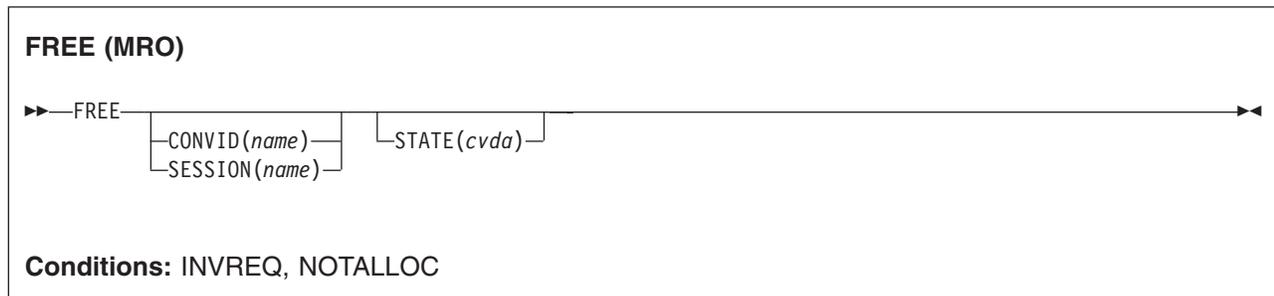
NOTALLOC

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

Default action: terminate the task abnormally.

FREE (MRO)

Return MRO sessions to CICS.



Description

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.

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

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 “EXEC interface block” on page 745.)

Default action: terminate the task abnormally.

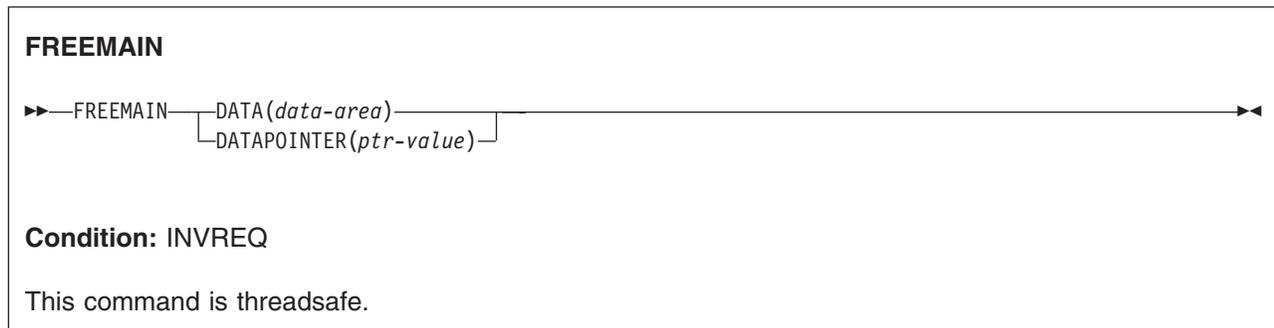
NOTALLOC

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

Default action: terminate the task abnormally.

FREEMAIN

Release main storage acquired by a GETMAIN command.



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 Application Programming Guide* for more information about transaction affinities.

Description

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.

Options

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 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 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.

Conditions

INVREQ

RESP2 values:

- 1 The storage specified by the DATA or DATAPOINTER parameter is not storage acquired by a GETMAIN command.
- 2 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.

Default action: terminate the task abnormally.

Example: 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.
```

Alternatively, the previous COBOL example could free the storage by the following command:

```
EXEC CICS FREEMAIN DATAPOINTER(AREA-POINTER)
END-EXEC.
```

Example: 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;
}

```

Example: 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);

```

Example: 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:

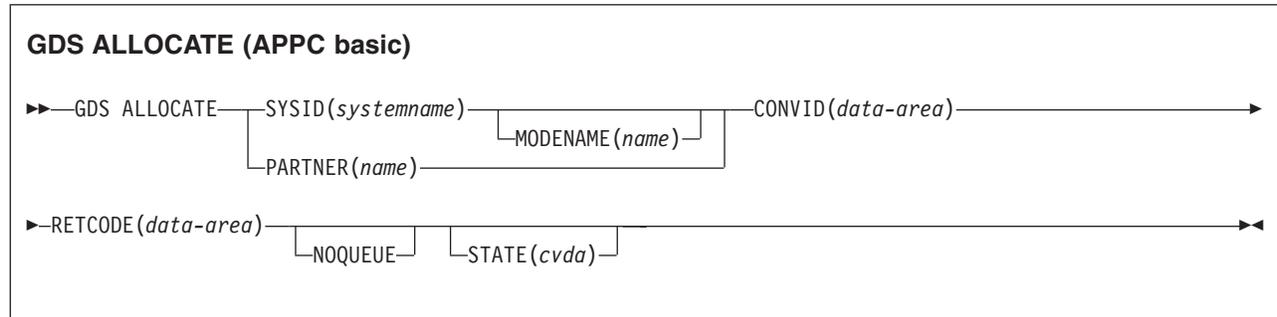
```

WORKAREA  DS  CL100
.
          EXEC CICS GETMAIN SET(9) LENGTH(100)
          USING  WORKAREA,9
.
.
          DROP  9
.
          EXEC CICS FREEMAIN DATAPOINTER(9)

```

GDS ALLOCATE

Acquire a session to a remote system for use by APPC basic conversation (assembler-language and C programs only).



Description

GDS ALLOCATE acquires a session to a remote system.

The return code is given in RETCODE (see Table 1 on page 220). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 (eight 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 (shown in Table 1) 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.

Table 1. GDS ALLOCATE return codes (continued)

RETCODE (hexadecimal)	Description
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

Get identifier of principal facility in use by APPC basic conversation (assembler-language and C programs only).

GDS ASSIGN (APPC basic)



Description

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 Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 (shown in Table 2) is to be moved.

Table 2. GDS ASSIGN return codes

RETCODE (hexadecimal)	Description
03 00	Principal facility is not APPC.
03 04	Principal facility is not basic.
04	No terminal principal facility exists.

PIPLIST (*data-area*)

specifies the PIP data that is to be sent to the remote process. See the *CICS 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. 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 (shown in Table 3) 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

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.

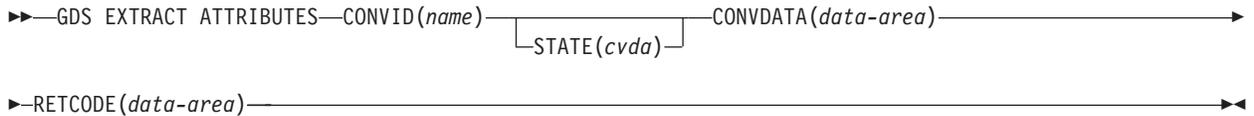
Table 3. GDS CONNECT PROCESS return codes (continued)

RETCODE (hexadecimal)	Description
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

Access state information on an APPC basic conversation (assembler-language and C programs only).

GDS EXTRACT ATTRIBUTES (APPC basic)



Description

GDS EXTRACT ATTRIBUTES accesses state information about an APPC basic conversation.

The return code is given in RETCODE (see Table 4 on page 227). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 Distributed Transaction Programming Guide*.

RETCODE(*data-area*)

specifies the 6-byte application data area into which return code information (shown in Table 4 on page 227) 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

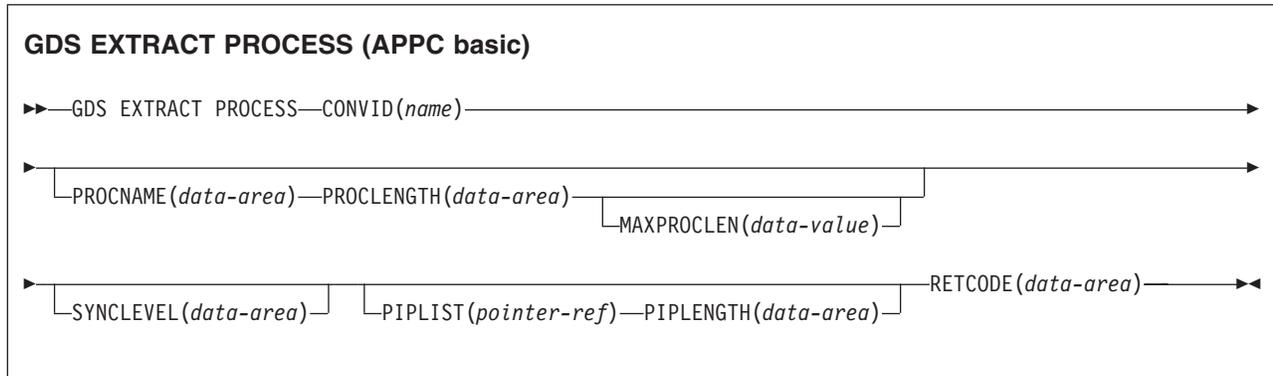
- PENDRECEIVE
- RECEIVE
- ROLLBACK
- SEND
- SYNCFREE
- SYNCRECEIVE
- SYNCSEND

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

Retrieve values from an APPC basic conversation (assembler-language and C programs only).



Description

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 on page 229). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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.

PIPELENGTH(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 (shown in Table 5) 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

Return an APPC session to CICS (assembler-language and C programs only).

GDS FREE (APPC basic)

```
▶▶—GDS FREE—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—▶▶
```

Description

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 on page 231). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 6 on page 231) 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
- SYNCSEND

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

Terminate APPC basic conversation abnormally (assembler-language and C programs only).

GDS ISSUE ABEND (APPC basic)

```
►►—GDS ISSUE ABEND—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—◄◄
```

Description

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 on page 233). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 7 on page 233) 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

Issue synchronization request on APPC basic conversation (assembler-language and C programs only).

GDS ISSUE CONFIRMATION (APPC basic)

```
►►—GDS ISSUE CONFIRMATION—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—————►►
◄◄—————◄◄
└─STATE(cvda)—┘
```

Description

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 on page 235). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 8 on page 235) 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 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

Inform APPC basic conversation partner of error (assembler-language and C programs only).

GDS ISSUE ERROR (APPC basic)

▶▶—GDS ISSUE ERROR—CONVID(*name*)—CONVDATA(*data-area*)—RETCODE(*data-area*)—STATE(*cvda*)—▶▶

Description

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 Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 9 on page 237) 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

Issue first flow of syncpoint request on APPC basic conversation (assembler-language and C programs only).

GDS ISSUE PREPARE (APPC basic)

```
▶▶—GDS ISSUE PREPARE—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—▶▶
```

Description

GDS ISSUE PREPARE issues first flow of syncpoint request.

The return code is given in RETCODE (see Table 10 on page 239). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 10 on page 239) 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

Request change of direction from sending transaction APPC basic conversation (assembler-language and C programs only).

GDS ISSUE SIGNAL (APPC basic)

►►—GDS ISSUE SIGNAL—CONVID(*name*)—CONVDATA(*data-area*)—RETCODE(*data-area*)—STATE(*cvda*)—◄◄

Description

GDS ISSUE SIGNAL requests a change of direction.

The return code is given in RETCODE (see Table 11 on page 241). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 11 on page 241) 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

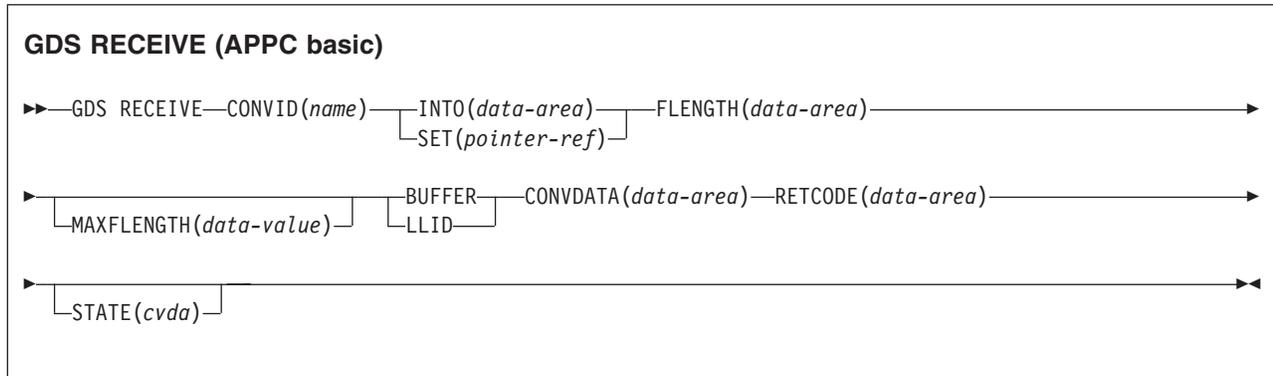
- ROLLBACK
- SEND
- SYNCFREE
- SYNCRECEIVE
- SYNCSEND

Table 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

Receive data on an APPC basic conversation (assembler-language and C programs only).



Description

GDS RECEIVE receives data and indicators from a partner transaction.

The return code is given in RETCODE (see Table 12 on page 244). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 12 on page 244) 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.

STATE(*cvda*)

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

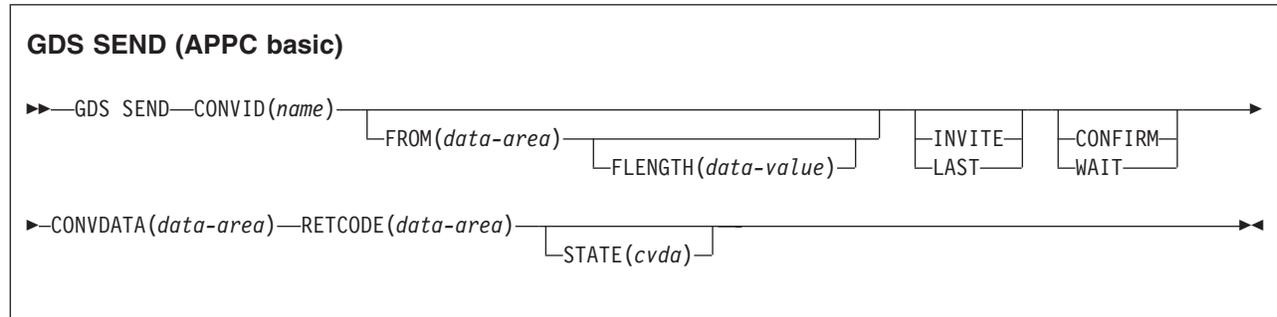
- ALLOCATED
- CONFFREE
- CONFRECEIVE
- CONFSEND
- FREE
- PENDFREE
- PENDRECEIVE
- RECEIVE
- ROLLBACK
- SEND
- SYNCFREE
- SYNCRECEIVE
- SYNCSSEND

Table 12. GDS RECEIVE 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 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

Send data on an APPC basic conversation (assembler-language and C programs only).



Description

GDS SEND sends data.

The return code is given in RETCODE (see Table 13 on page 247). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 13 on page 247) 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

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.

Table 13. GDS SEND 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 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

Ensure accumulated data transmitted on an APPC conversation (assembler-language and C programs only).

GDS WAIT (APPC basic)

```
▶▶—GDS WAIT—CONVID(name)—CONVDATA(data-area)—RETCODE(data-area)—STATE(cvda)—▶▶
```

Description

GDS WAIT ensures that the accumulated data has been sent.

The return code is given in RETCODE (see Table 14 on page 249). For a list of return code values, see the *CICS Distributed Transaction Programming Guide*. EXEC CICS conditions are never raised on GDS commands.

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 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 (shown in Table 14 on page 249) 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

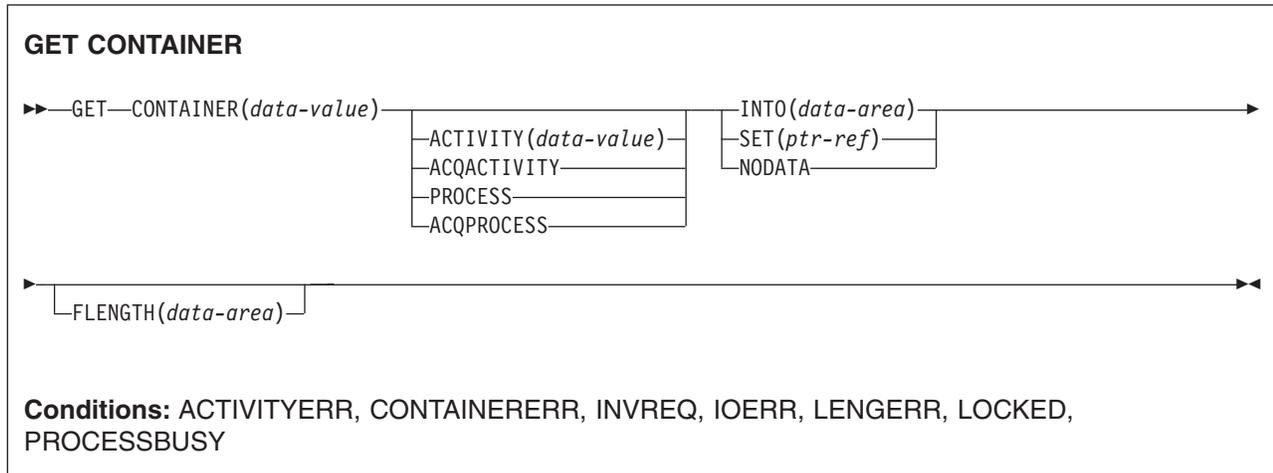
- ROLLBACK
- SEND
- SYNCFREE
- SYNCRECEIVE
- SYNCSEND

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.

GET CONTAINER (BTS)

Retrieve data from a named BTS data-container.



Description

GET CONTAINER reads the data associated with a specified BTS activity or process into working storage.

The container which holds the data is identified by name and by the process or activity for which it is a container—the process or activity that “owns” it. The activity that owns the container can be identified:

- Explicitly, by specifying one of the PROCESS- or ACTIVITY-related options.
- Implicitly, by omitting the PROCESS- and ACTIVITY-related options. If these are omitted, the current activity is implied.

See also “PUT CONTAINER (BTS)” on page 377 and “MOVE CONTAINER (BTS)” on page 363.

Options

ACQACTIVITY

specifies either of the following:

- If the program that issues the command has acquired a process, that the container is owned by the root activity of that process.
- Otherwise, that the container is owned by the activity that the program has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the container is owned by the process that the program that issues the command has acquired in the current unit of work.

ACTIVITY(*data-value*)

specifies the name (1–16 characters) of the activity that owns the container. This must be a child of the current activity.

CONTAINER(*data-value*)

specifies the name (1–16 characters) of the container that holds the data to be retrieved.

FLENGTH(data-area)

As an input field, FLENGTH specifies, as a fullword binary value, the length of the data to be read. As an output field, FLENGTH returns the length of the data in the container. Whether FLENGTH is an input or an output field depends on which of the INTO, SET, or NODATA options you specify.

INTO option specified

FLENGTH is both an input and an output field.

On input, FLENGTH specifies the maximum length of the 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.

FLENGTH need not be specified if the length can be generated by the compiler from the INTO variable. If you specify both INTO and FLENGTH, FLENGTH specifies the maximum length of the data that the program accepts.

On **output** (that is, on completion of the retrieval operation) CICS sets the data area, if specified, to the actual length of the data in the container.

SET or NODATA option specified

FLENGTH is an output-only field. It must be specified and specified as a data-area.

On completion of the retrieval operation, the data area is set to the actual length of the data in the container.

INTO(data-area)

specifies an area of working storage into which the retrieved data is to be placed.

NODATA

specifies that no data is to be retrieved. Use this option to discover the length of the data in the container (returned in FLENGTH).

PROCESS

specifies that the container to be retrieved is owned by the current process—that is, the process that the program that issues the command is executing on behalf of.

SET(ptr-ref)

specifies a data area in which the address of the retrieved data is returned. The data area is maintained by CICS until a subsequent GET CONTAINER command with the SET option is issued by the task, or until the task ends.

If your application needs to keep the data it should move it into its own storage.

Conditions**ACTIVITYERR**

RESP2 values:

8 The activity named on the ACTIVITY option could not be found.

CONTAINERERR

RESP2 values:

10 The container named on the CONTAINER option could not be found.

INVREQ

RESP2 values:

- 2 The INTOCCSID option was specified without the CHANNEL option, and there is no current channel (because the program that issued the command was not passed one.) INTOCCSID is valid only on GET CONTAINER commands that specify (explicitly or implicitly) a channel. It is not valid on GET CONTAINER (BTS) commands.
- 4 The command was issued outside the scope of a currently-active activity.
- 15 The ACQPROCESS option was used, but the unit of work that issued the request has not acquired a process.
- 24 The ACQACTIVITY option was used, but the unit of work that issued the request has not acquired an activity.
- 25 The PROCESS option was used, but the command was issued outside the scope of a currently-active process.

IOERR

RESP2 values:

- 30 An input/output error has occurred on the repository file.
- 31 The record on the repository file is in use.

LENGERR

RESP2 values:

- 11 The length of the program area is not the same as the length of the data in the container. If the area is smaller, the data is truncated to fit into it.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

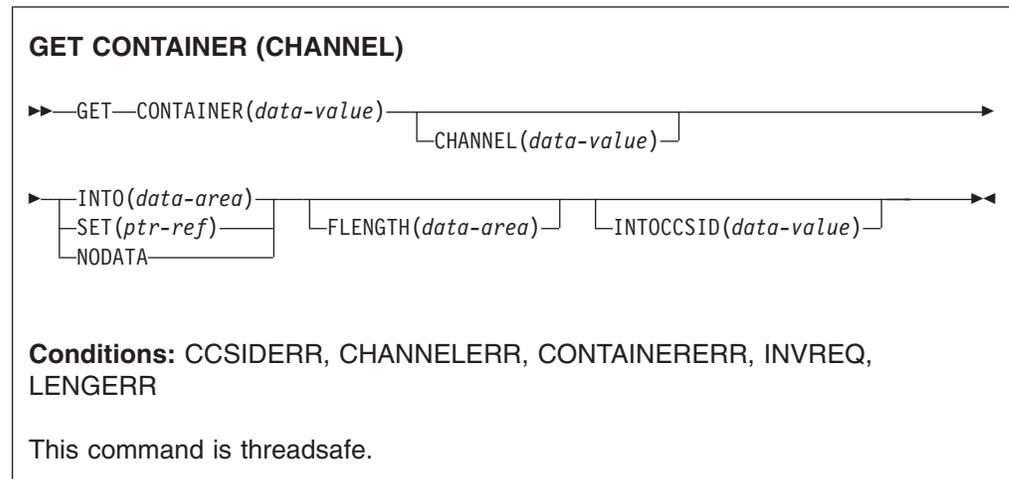
PROCESSBUSY

RESP2 values:

- 13 The request could not be satisfied because the process record is locked by another task.

GET CONTAINER (CHANNEL)

Retrieve data from a named channel container.



Description

GET CONTAINER (CHANNEL) reads the data associated with a specified channel container.

The container which holds the data is identified by name and by the channel for which it is a container—the channel that “owns” it. The channel that owns the container can be identified:

- Explicitly, by specifying the CHANNEL option.
- Implicitly, by omitting the CHANNEL option. If this is omitted, the current channel is implied.

Options

CHANNEL(data-value)

specifies the name (1–16 characters) of the channel that owns the container.

CONTAINER(data-value)

specifies the name (1–16 characters) of the container that holds the data to be retrieved.

FLENGTH(data-area)

As an input field, FLENGTH specifies, as a fullword binary value, the length of the data to be read. As an output field, FLENGTH returns the length of the data in the container. Whether FLENGTH is an input or an output field depends on which of the INTO, SET, or NODATA options you specify.

INTO option specified

FLENGTH is both an input and an output field.

On input, FLENGTH specifies the maximum length of the 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.

FLENGTH need not be specified if the length can be generated by the compiler from the INTO variable. If you specify both INTO and FLENGTH, FLENGTH specifies the maximum length of the data that the program accepts.

On **output** (that is, on completion of the retrieval operation) CICS sets the data area, if specified, to the actual length of the data in the container. If the container holds character data that has been converted from one CCSID to another, this is the length of the data *after conversion*.

SET or NODATA option specified

FLENGTH is an output-only field. It must be specified and must be specified as a data-area.

On completion of the retrieval operation, the data area is set to the actual length of the data in the container. If the container holds character data that has been converted from one CCSID to another, this is the length of the data *after conversion*.

INTO(data-area)

specifies the data area into which the retrieved data is to be placed.

INTOCCSID(data-value)

specifies, as a fullword binary number, the Coded Character Set Identifier (CCSID) into which the character data in the container is to be converted. For CICS Transaction Server for z/OS applications, this is typically an EBCDIC CCSID. (However, it is possible to specify an ASCII CCSID if, for example, you want to retrieve ASCII data without it being automatically converted to EBCDIC.)

If INTOCCSID is not specified, its value defaults to the CCSID of the region. The default CCSID of the region is specified on the **LOCALCCSID** system initialization parameter.

Only character data can be converted, and only then if a DATATYPE of CHAR was specified on the **PUT CONTAINER** command used to place the data in the container.

For more information about data conversion with channels, see the *CICS Application Programming Guide*.

For an explanation of CCSIDs, and a list of the CCSIDs supported by CICS, see the *CICS Family: Communicating from CICS on System/390* manual.

NODATA

specifies that no data is to be retrieved. Use this option to discover the length of the data in the container (returned in FLENGTH).

The length of character data may change if data conversion takes place. Therefore, if character data is to be converted into any CCSID *other than that of this region*, when you specify NODATA you should also specify INTOCCSID. This ensures that the correct length of the converted data is returned in FLENGTH.

SET(ptr-ref)

specifies a data area in which the address of the retrieved data is returned.

The data area is maintained by CICS until any of the following occurs:

- A subsequent GET CONTAINER command with the SET option, for the same container in the same channel, is issued by any program that can access this storage.
- The container is deleted by a DELETE CONTAINER command.

- The container is moved by a MOVE CONTAINER command.
- The channel goes out of program scope.

Beware of linking to other programs that might issue one of the above commands.

Do *not* issue a FREEMAIN command to release this storage.

If your application needs to keep the data it should move it into its own storage.

Conditions

CCSIDERR

RESP2 values:

- 1 The CCSID specified on the INTOCCSID option is outside the range of valid CCSID values.
- 2 The CCSID specified on the INTOCCSID option and the CCSID of the channel are an unsupported combination.
- 3 The data was created with a data-type of BIT. Code page conversion is not possible. The data was returned without any code page conversion.
- 4 One or more characters could not be converted. The character has been replaced by a blank in the converted data.
- 5 There was an internal error in the code page conversion of a container.

CHANNELERR

RESP2 values:

- 2 The channel specified on the CHANNEL option could not be found.

CONTAINERERR

RESP2 values:

- 10 The container named on the CONTAINER option could not be found.

INVREQ

RESP2 values:

- 2 The INTOCCSID option was specified without the CHANNEL option, and there is no current channel (because the program that issued the command was not passed one.) INTOCCSID is valid only on GET CONTAINER commands that specify (explicitly or implicitly) a channel.
- 4 The CHANNEL option was not specified, there is no current channel (because the program that issued the command was not passed one), and the command was issued outside the scope of a currently-active BTS activity.

LENGERR

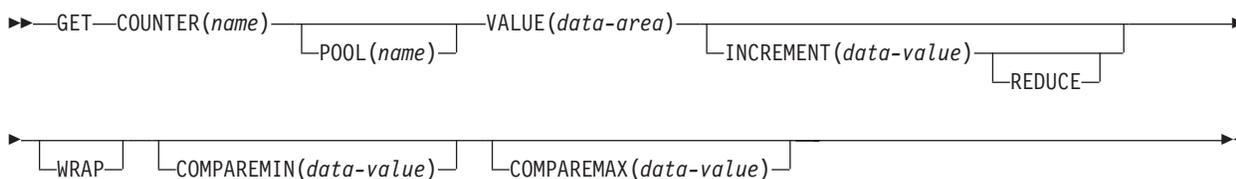
RESP2 values:

- 11 The length of the program area is shorter than the length of the data in the container. When the area is smaller, the data is truncated to fit into it.

GET COUNTER and GET DCOUNTER

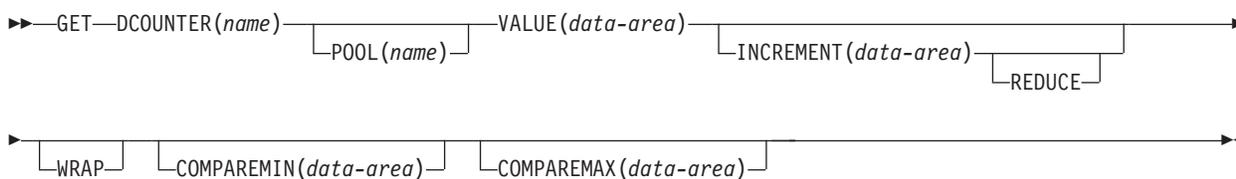
Get the next number from the named counter.

GET COUNTER



Conditions: INVREQ, LENGERR, SUPPRESSED

GET DCOUNTER



Conditions: INVREQ, SUPPRESSED

Description

These counter commands obtain, from the named counter server, the current number from the named counter in the specified pool, and updates the current number by the default, or by a specified, increment. The default increment is 1. COUNTER operates with fullword *signed* binary values, and DCOUNTER operates with doubleword *unsigned* binary values.

You can use the COMPAREMAX and COMPAREMIN options to obtain a number only if it falls within a specified range, or is above or below a specified value.

For information about specifying fullword and doubleword variables on these named counter commands, see “CICS command argument values” on page 3.

Options

COMPAREMAX(data-value)

specifies, as a fullword signed binary value (doubleword unsigned binary value for DCOUNTER), a value to be compared with the named counter's current value, and makes the result of the GET command conditional on the comparison:

- If the current value to be assigned is less than, or equal to, the value specified on the COMPAREMAX parameter, the current value is returned, with response normal
- If the current value is greater than the specified value, CICS returns an exception condition.

The value you specify on the COMPAREMAX parameter can be less than the value on the COMPAREMIN parameter, in which case the current value is considered to be in range if it satisfies the COMPAREMIN *or* the COMPAREMAX comparison. In the normal case where the COMPAREMIN value is less than the COMPAREMAX value, the current value must satisfy both comparisons (that is, it must be greater than, or equal to, the COMPAREMIN value, *and* be less than, or equal to the COMPAREMAX value).

COMPAREMIN (*data-value*)

specifies, as a fullword signed binary value (doubleword unsigned binary value for DCOUNTER), a value to be compared with the named counter's current value, and makes the result of the GET command conditional on the comparison:

- If the current value to be assigned is equal to, or greater than, the value specified on the COMPAREMIN parameter, the CICS returns the current value, with response normal
- If the current value is less than the specified value, CICS returns an exception condition.

Note: The value you specify on the COMPAREMIN parameter can be greater than the value on the COMPAREMAX parameter. See the COMPAREMAX parameter for the effect of this.

COUNTER (*name*)

specifies the name of the fullword counter from which the current number is to be assigned to the application program. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

DCOUNTER (*name*)

specifies the name of the doubleword counter from which the current number is to be assigned to the application program. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

INCREMENT (*data-value*)

specifies, as a fullword signed binary value (doubleword unsigned binary value for DCOUNTER), an increment by which the named counter is to be updated, instead of the default value of 1. The counter is incremented after the current number has been assigned.

Specifying an increment to override the default increment of 1 enables the application program to obtain exclusive use of more than 1 number for each call. For example, if you want to obtain exclusive use of a block of 20 numbers, specify INCREMENT(20).

See the description of the REDUCE and WRAP options for the effect of specifying an increment when the counter is at, or near, the maximum value.

POOL (*poolname*)

specifies an 8-character string to be used as a pool selection parameter to select the pool in which the named counter resides. The string can be a logical pool name, or the actual pool name.

Valid characters for the pool selector string are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 8 characters, the name must be padded with trailing blanks.

This parameter is optional. If you omit the name of the pool, a pool selector value of 8 blanks is assumed.

If there is no matching entry in the DFHNCOPT options table, CICS uses the name specified on the NCPLDFT system initialization parameter, which specifies the default named counter pool.

For information about generating a named counter options table using the DFHNCO macro, see the *CICS Application Programming Guide*.

REDUCE

specifies that you want the named counter server to reduce the specified increment if the range of numbers remaining to be assigned is too small.

The range of numbers is too small if the difference between the current value and the maximum value plus 1 is less than the specified increment, in which case:

- If you specify REDUCE, the INCREMENT parameter value is reduced and the GET request succeeds. In this case, the GET command has reserved a range of numbers less than that specified by the INCREMENT parameter, and the current value is updated to the maximum value plus 1.
- If you do not specify the REDUCE option, the result depends on whether or not you specify the WRAP option. If the REDUCE and WRAP options are both omitted, the request fails with the counter-at-limit error (SUPPRESSED, RESP2=101), but the current number is not changed. For example, if a request specifies an INCREMENT parameter value of 15 when the current number is 199 990 and the counter maximum number is defined as 199 999, the GET command fails because updating the counter by the specified increment would cause the current number to exceed 200 000.

VALUE (*data-area*)

specifies the data area (fullword signed data-area for COUNTER, and doubleword unsigned data-area for DCOUNTER) into which CICS returns the current number, obtained from the named counter server for the specified pool.

WRAP

specifies that you want the named counter server automatically to perform a rewind of the named counter if it is in a counter-at-limit condition, avoiding the error condition that would otherwise result.

If it finds that the named counter is in the counter-at-limit condition, or that the increment specified without the REDUCE option would cause the condition, the server:

- Resets the current value of the named counter equal to the minimum value defined for the counter
- Returns the new current value to the application program, with DFHRESP(NORMAL)
- Updates the current value by the required increment ready for the next request.

If you omit the WRAP option, and the counter-at-limit condition has been reached, CICS returns SUPPRESSED, RESP2=101.

Conditions

INVREQ

RESP2 values:

- 201** Named counter not found.
- 301** The server has reported an error code that is not understood by the named counter interface. Generally, this is not possible unless the interface load module, DFHNCIF, is at a lower maintenance or release level than the server itself.
- 303** An unexpected error, such as structure failure or loss of connectivity, has occurred on a macro used to access the coupling facility. Further information can be found in message DFHNC0441 in the CICS job log.
- 304** The pool selection parameter specified in the program cannot be resolved to a valid server name using the current options table.
- 305** The interface is unable to establish a connection to the server for the selected named counter pool. Further information can be found in an AXM services message (AXMSC $nnnn$) in the CICS job log.
- 306** An abend occurred during server processing of a request. Further information can be found in a message in the CICS job log and the server job log.
- 308** The DFHNCOPT options table module, required for resolving a pool name, could not be loaded.
- 309** During processing of the options table, the named counter interface encountered an unknown entry format. Either the options table is not correctly generated, or the DFHNCIF interface load module is not at the same release level as the options table.
- 310** An options table entry matching the given pool name specified a user exit program, but the user exit program is not link-edited with the options table and cannot be loaded.
- 311** A response from the named counter server to the client region interface module, DFHNCIF, indicates that a system-managed rebuild is in progress but the EXEC CICS interface does not recognize the condition. This means that the CICS region is at CICS TS 2.1 or earlier.
- 403** The POOL parameter contains invalid characters or embedded spaces.
- 404** The COUNTER parameter contains invalid characters or embedded spaces.
- 406** The INCREMENT value is invalid. The value specified cannot be greater than the total range of the counter ((maximum value - minimum value) + 1).

Default action: terminate the task abnormally.

LENGERR

LENGERR occurs for COUNTER commands only and does not apply to DCOUNTER requests. It occurs when a counter that was defined by a DCOUNTER command or by the CALL interface has a value which is too large to be correctly represented as a fullword signed binary value (that is, the counter uses more than 31 bits).

In each of the three cases of overflow, the named counter server completes the operation, and returns a warning response to CICS, which CICS returns to your

application program as the RESP2 value. The data area contains the low-order 32 bits returned from the named counter server, which could be a negative number.

RESP2 values:

- 001** The current value that the server has attempted to return in the VALUE data area has overflowed into the high-order (sign) bit (that is, the value returned is negative).
- 002** The current value is too large for a fullword data area by only 1 bit. In this case, the overflow value is exactly 1.
- 003** The current value is too large for a fullword data area by a value greater than 1.

Default action: terminate the task abnormally.

SUPPRESSED

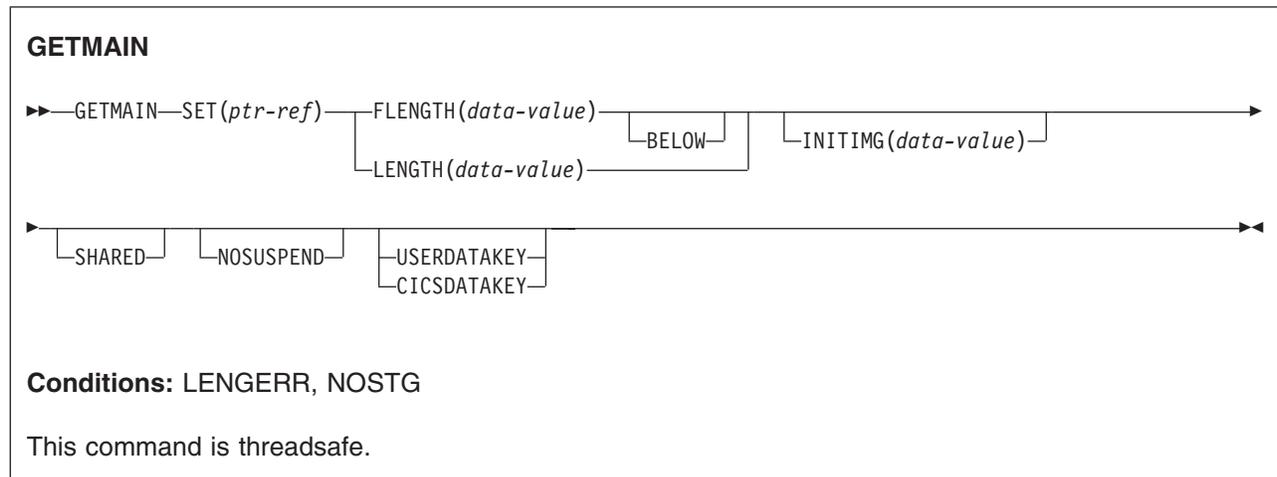
RESP2 values:

- 101** The maximum value for the named counter has already been assigned and the counter is in the 'counter-at-limit' condition. No more counter numbers can be assigned until the named counter has been reset, either by a REWIND command, or by specifying the WRAP option on the GET command.
- 103** The current value of the named counter is:
 - not within the range specified by the COMPAREMAX and COMPAREMIN parameters, when both are specified
 - greater than the COMPAREMAX parameter or less than the COMPAREMIN parameter, when only one option is specified.

Default action: terminate the task abnormally.

GETMAIN

Get main storage.



Note for dynamic transaction routing: Using GETMAIN with SHARED could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS Application Programming Guide* for more information about transaction affinities.

Description

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

In most cases, CICS obtains the storage from a DSA. If the FLENGTH option is specified alone, and the addressing mode of the requesting program is 31-bit, CICS obtains the storage from an EDSA.

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 CICSDATAKEY option on the GETMAIN command
- The TASKDATAKEY option on the RDO TRANSACTION resource definition under which the requesting program is running if the USERDATAKEY or CICSDATAKEY option is omitted
- The SHARED option on the GETMAIN command

The data-key option on the GETMAIN command overrides the TASKDATAKEY option on the RDO TRANSACTION resource definition. The effect of the data-key options is summarized in the following table:

Table 15. Data-key options on GETMAIN command

No data-key option	USERDATAKEY specified	CICSDATAKEY specified
Determined by TASKDATAKEY on transaction definition	User-key storage. From UDSA or EUDSA if SHARED option is not specified, or from SDSA or ESDSA if SHARED option is specified.	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.

Specifying CICSDATAKEY ensures that the requesting program obtains CICS-key storage from a CICS DSA, even if TASKDATAKEY(USER) is specified on the RDO TRANSACTION resource definition.

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.

Note: If the program is running under a task defined with TASKDATAKEY(USER) on the transaction resource definition, do not explicitly use FREEMAIN but allow the storage to be freed as part of the task termination.

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 65 520 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.

Note, however, that if a HANDLE CONDITION for NOSTG is active when the command is executed, control is passed to the user label supplied in the HANDLE CONDITION. This takes precedence over the NOSUSPEND option but is, of course, negated by either NOHANDLE or RESP.

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.

Conditions

LENGERR

RESP2 values:

- 1** 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. See the discussion about DSAs in CICS storage allocation.

Also occurs if the LENGTH value is zero.

Default action: terminate the task abnormally.

NOSTG

RESP2 values:

- 2** The storage requested is more than is currently available in the target DSA. See the discussion about DSAs in CICS storage allocation.

Default action: ignore the condition. An active HANDLE CONDITION NOSTG also raises this condition.

Examples

The following 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 RDO 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.

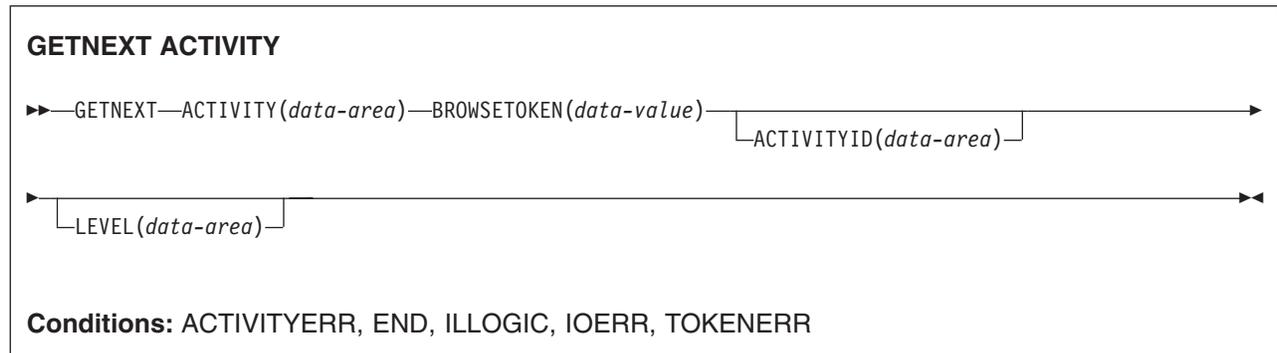
The following example shows how to get a 2048-byte area from CICS-key storage above the 16MB line (regardless of the TASKDATAKEY option specified on the transaction resource definition), and initialize it to spaces:

```
EXEC CICS GETMAIN SET(PTR)
          FLENGTH(2048)
          INITIMG(BLANK)
          CICSDATAKEY
```

GETNEXT ACTIVITY

#

Browse the child activities of a BTS activity, or the descendant activities of a BTS process. Although this command appears in the Application Programming set of topics, to use it you must specify the system programming (SP) parameter in the EXEC statement of the translate step of your compile job. See Technote 1265081 for further information.



Description

GETNEXT ACTIVITY returns either:

- The name and identifier of the next child activity of a BTS activity (if the PROCESS and PROCESSTYPE options were omitted from the STARTBROWSE ACTIVITY command)
- The name and identifier of the next descendant activity of a BTS process (if the PROCESS and PROCESSTYPE options were specified on the STARTBROWSE ACTIVITY command).

You can use the INQUIRE ACTIVITYID command to query the identified activity.

Options

ACTIVITYID(*data-area*)

returns the 52-character identifier of the next activity.

ACTIVITY(*data-area*)

returns the 16-character name of the next activity.

BROWSETOKEN(*data-value*)

specifies, as a fullword binary value, a browse token returned on a previous STARTBROWSE ACTIVITY command.

LEVEL(*data-area*)

returns a fullword value indicating the depth in the activity-tree at which the next activity lies.

On a browse of the descendant activities of a process, a value of '0' indicates the root activity, '1' a child of the root activity, '2' a grandchild of the root activity, and so on.

On a browse of the child activities of an activity, the value returned is always 0.

Conditions

ACTIVITYERR

RESP2 values:

- 19 The request timed out. It may be that another task using this activity-record has been prevented from ending.

END

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 The value specified in the BROWSETOKEN option matches a current browse token, but not one that is being used for an activity browse.

IOERR

RESP2 values:

- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

TOKENERR

RESP2 values:

- 3 The browse token is not valid.

GETNEXT CONTAINER

#

Browse the containers associated with a channel, or with a BTS activity or process. Although this command appears in the Application Programming set of topics, to use it you must specify the system programming (SP) parameter in the EXEC statement of the translate step of your compile job. See Technote 1265081 for further information.

GETNEXT CONTAINER

▶—GETNEXT—CONTAINER(*data-area*)—BROWSETOKEN(*data-value*)—◀

Conditions: END, ILLOGIC, TOKENERR

Description

GETNEXT CONTAINER returns the name of the next container associated with a channel, or with a BTS activity or process. You can use the INQUIRE CONTAINER command to query the returned container.

Note:

1. You can use successive GETNEXT CONTAINER commands to retrieve the names of all the channel's or activity's containers that existed at the time the STARTBROWSE CONTAINER command was executed. However, the names of any containers that are deleted after the STARTBROWSE and before they have been returned by a GETNEXT are not returned.
2. The names of any containers that are created on (or moved to) this channel or activity after the STARTBROWSE command is executed may or may not be returned.
3. The order in which containers are returned is undefined.

#

Options

BROWSETOKEN(*data-value*)

specifies, as a fullword binary value, a browse token returned on a previous STARTBROWSE CONTAINER command.

CONTAINER(*data-area*)

returns the 16-character name of the next data-container.

Conditions

END

RESP2 values:

- 2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

- 1 The value specified in the BROWSETOKEN option matches a current browse token, but not one that is being used for a browse of containers.

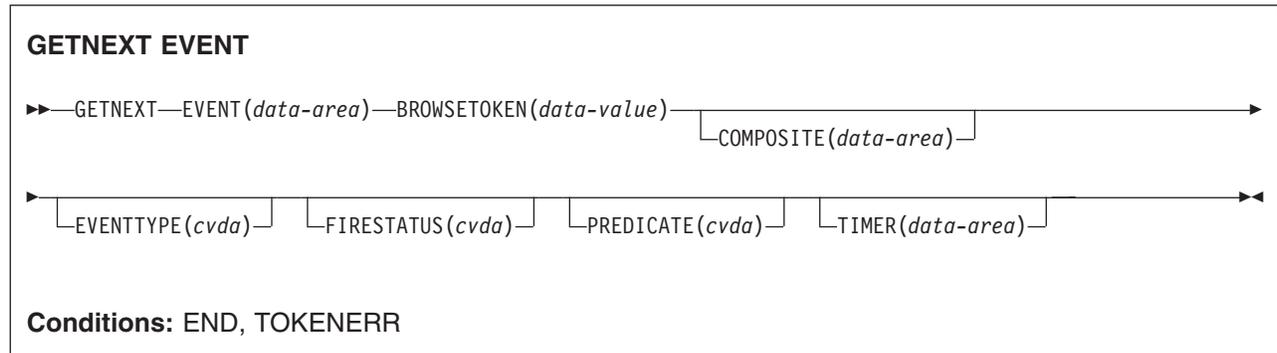
TOKENERR

RESP2 values:

- 3** The browse token is not valid.

GETNEXT EVENT

Browse the events known to a BTS activity. Although this command appears in the
Application Programming set of topics, to use it you must specify the system
programming (SP) parameter in the EXEC statement of the translate step of your
compile job. See Technote 1265081 for further information.



Description

GETNEXT EVENT returns the attributes of the next event, or sub-event, that is within the scope of a BTS activity.

Options

BROWSETOKEN (data-value)

specifies, as a fullword binary value, a browse token returned on a previous STARTBROWSE EVENT command.

COMPOSITE (data-area)

returns, if the named event is a sub-event, the 16-character name of the composite event that it is part of.

EVENT (data-area)

returns the 16-character name of the next event. This may be:

- An atomic event. An atomic event returned on this command may or may not be a sub-event.
- A composite event.
- A system event.

EVENTTYPE (cvda)

indicates the type of the named event. CVDA values are:

ACTIVITY

Activity completion

COMPOSITE

Composite

INPUT

Input

SYSTEM

System

TIMER

Timer.

FIRESTATUS(cvda)

indicates the state of the named event. CVDA values are:

FIRED The event has fired normally.

NOTFIRED

The event has not fired.

PREDICATE(cvda)

indicates, if the named event is composite, the Boolean operator applied to its predicate. CVDA values are:

AND The Boolean operator applied to the predicate is AND.

OR The Boolean operator applied to the predicate is OR.

TIMER(data-area)

returns, if the named event is a timer event, the 16-character name of its associated timer.

Conditions**END**

RESP2 values:

2 There are no more resource definitions of this type.

TOKENERR

RESP2 values:

3 The browse token is not valid.

GETNEXT PROCESS

#

Browse all processes of a specified type within the CICS business transaction services system. Although this command appears in the Application Programming set of topics, to use it you must specify the system programming (SP) parameter in the EXEC statement of the translate step of your compile job. See Technote 1265081 for further information.

GETNEXT PROCESS

▶ GETNEXT PROCESS(*data-area*) BROWSETOKEN(*data-value*)
└─ACTIVITYID(*data-area*)─┘

Conditions: END, ILLOGIC, IOERR, PROCESSERR, TOKENERR

Description

GETNEXT PROCESS returns the name of the next process of a specified type within the CICS business transaction services system.

Options

ACTIVITYID(*data-area*)

returns the 52-character identifier of the next process's root activity.

BROWSETOKEN(*data-value*)

specifies, as a fullword binary value, a browse token returned on a previous STARTBROWSE PROCESS command.

PROCESS(*data-area*)

returns the 36-character name of the next process.

Conditions

END

RESP2 values:

2 There are no more resource definitions of this type.

ILLOGIC

RESP2 values:

1 The value specified in the BROWSETOKEN option matches a current browse token, but not one that is being used for a process browse.

IOERR

RESP2 values:

30 An input/output error has occurred on the repository file.

PROCESSERR

RESP2 values:

13 The request timed out. It may be that another task using this process-record has been prevented from ending.

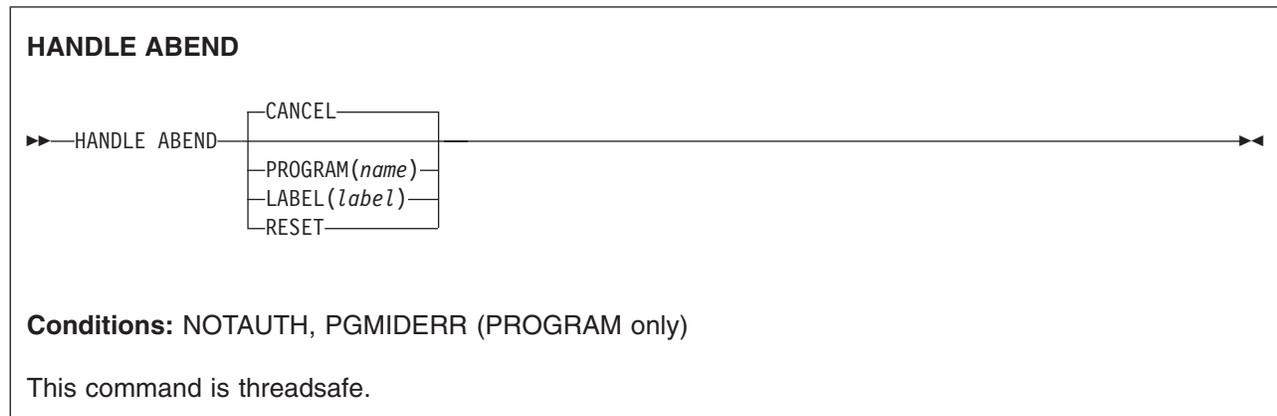
TOKENERR

RESP2 values:

3 The browse token is not valid.

HANDLE ABEND

Handle an abnormal termination exit.



Description

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 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.

Assembler

R15: Abend label. R0-14: Contents at the point when the last EXEC CICS command was issued.

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. Where more than one application program was involved in the task, note that the COMMAREA that is passed to the abend exit is the COMMAREA of

#

the program that issued the HANDLE ABEND command. This is not necessarily the COMMAREA of the program in which the abend actually occurred.

If a current channel exists, it will be accessible from the specified PROGRAM.

Options

CANCEL

specifies that a previously established exit at the logical level of the application program in control is to be canceled. The CANCEL option is the default setting for the HANDLE ABEND command.

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.

Conditions

NOTAUTH

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

Default action: terminate the task abnormally.

PGMIDERR

RESP2 values:

- 1 The program has no entry in the PPT and autoinstall for programs is not active.
- 2 The program is disabled.
- 9 The installed program definition is for a remote program.

Default action: terminate the task abnormally.

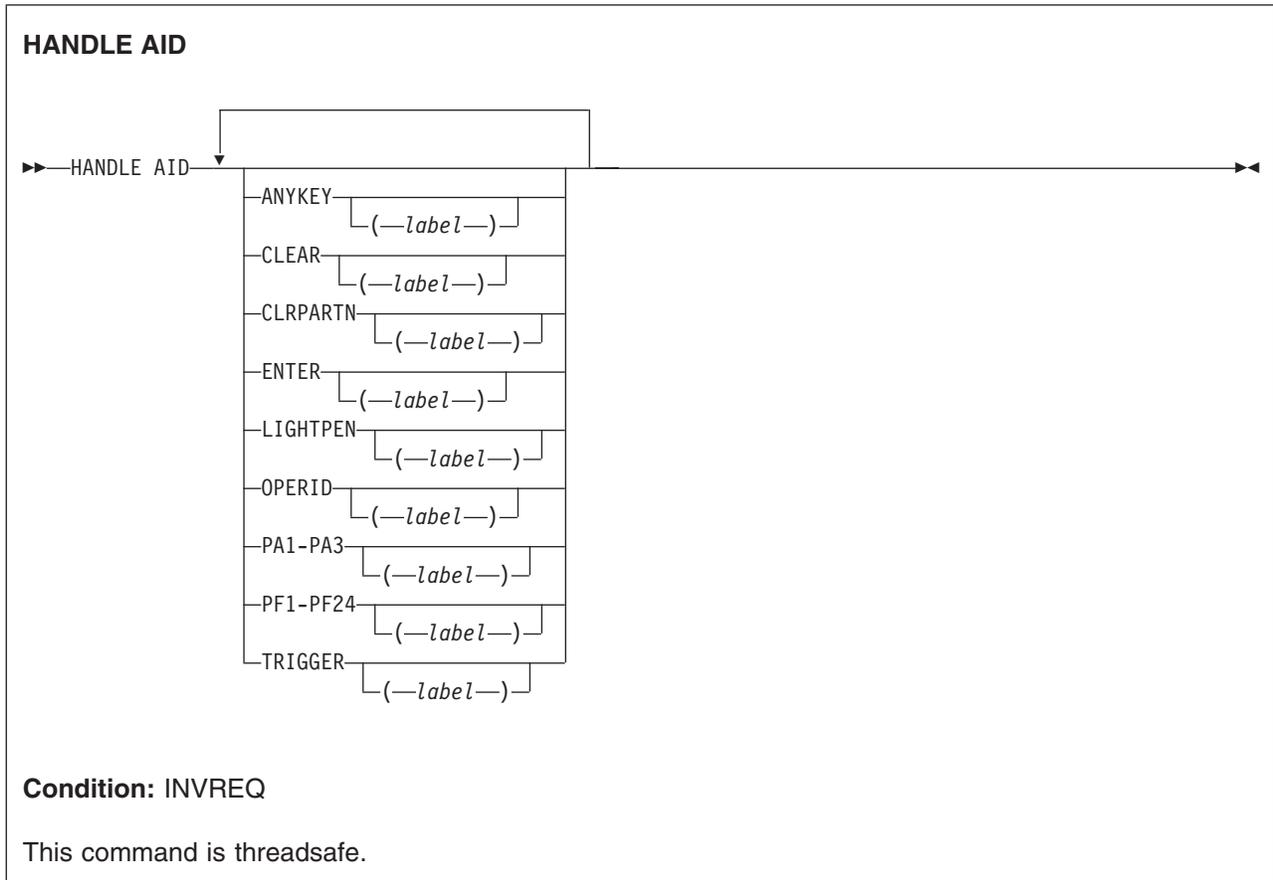
Examples

The following example shows how to establish a program as an exit:

```
EXEC CICS HANDLE ABEND  
PROGRAM('EXITPGM')
```

HANDLE AID

Handle attention identifiers (AIDs).



Description

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)

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 “BMS-related constants” on page 779.

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.

Conditions

INVREQ

RESP2 values:

200 The command was issued by a distributed program link server application.

Default action: terminate the task abnormally.

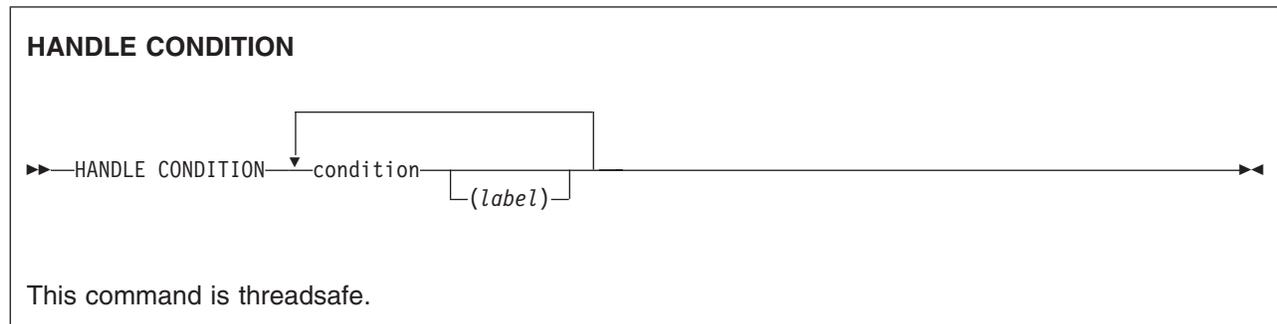
Examples

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
```

HANDLE CONDITION

Handle conditions.



Description

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.

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.

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 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, when a branch to a label is caused
by a condition, the registers in the application program are restored to their values
in the program at the point where the command that caused the condition 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.

Options

condition(*label*)

specifies the name of the condition; “label” specifies the location within the program to be branched to if the condition occurs.

For more information about the conditions, see “EXEC interface block” on page 745.

Examples

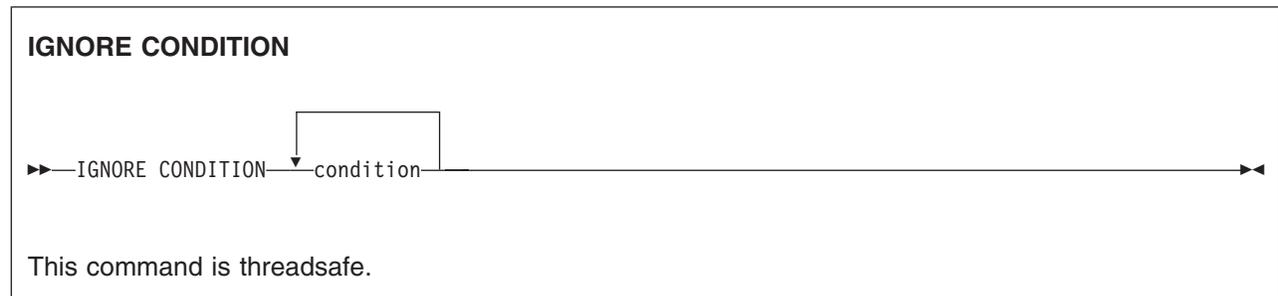
The following example shows you how to handle conditions, such as DUPREC, LENGERR, and so on, that can occur when you use a WRITE command to add a record to a data set.

Suppose that you want DUPREC to be handled as a special case; that you want default 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
```

IGNORE CONDITION

Ignore conditions.



Description

IGNORE CONDITION is not supported for C programs.

For information about the conditions, see “EXEC interface block” on page 745.

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 include more than sixteen conditions in the same command; the conditions must be separated by at least one space. You may specify additional conditions in further IGNORE CONDITION commands.

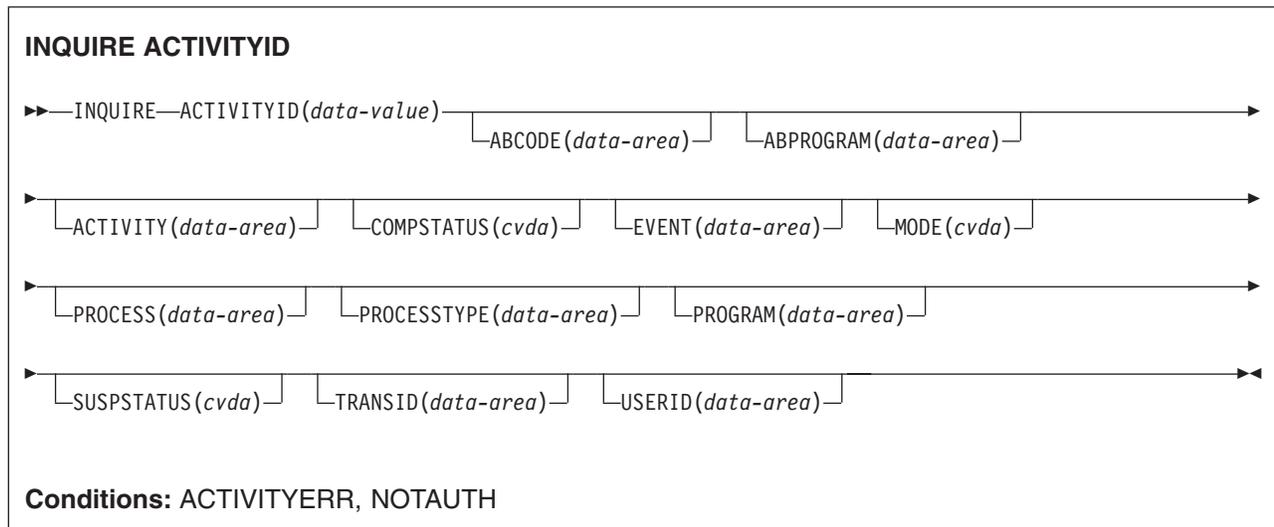
Options

condition

specifies the name of the condition to be ignored.

INQUIRE ACTIVITYID

Retrieve the attributes of a BTS activity.



Description

INQUIRE ACTIVITYID returns the attributes of a specified BTS activity.

You can use this command to get details of an activity whose identifier has been retrieved during a browse operation.

Options

ABCODE (data-area)

returns, if the activity terminated abnormally, the 4-character abend code.

ABPROGRAM (data-area)

returns, if the activity terminated abnormally, the 8-character name of the program that was in control at the time of the abend.

ACTIVITY (data-area)

returns the 16-character name of the activity being queried.

ACTIVITYID (data-value)

specifies the identifier (1–52 characters) of the activity to be queried. (Typically, the activity identifier will have been retrieved by a GETNEXT ACTIVITY command, during an activity browse.)

COMPSTATUS (cvda)

indicates the completion status of the activity. CVDA values are:

ABEND

The program that implements the activity abended. Any children of the activity have been canceled.

FORCED

The activity was forced to complete—for example, it was canceled with a CANCEL ACTIVITY command.

INCOMPLETE

The named activity is incomplete. This could mean:

- That it has not yet been run
- That it has returned from one or more activations but needs to be reattached in order to complete all its processing steps
- That it is currently active.

NORMAL

The named activity completed successfully.

EVENT(data-area)

returns the 16-character name of the completion event that is sent to the requestor of this activity when the activity completes asynchronously with the requestor.

MODE(cvda)

indicates the current state (mode) of the activity. CVDA values are:

ACTIVE

An activation of the activity is running.

CANCELLING

CICS is waiting to cancel the activity. A CANCEL ACTIVITY command has been issued, but CICS cannot cancel the activity immediately because one or more of the activity's children are inaccessible.

No further operations on the activity are permitted until it has been canceled.

COMPLETE

The activity has completed, either successfully or unsuccessfully. The value returned on the COMPSTATUS option tells you how it completed.

DORMANT

The activity is waiting for an event to fire its next activation.

INITIAL

No RUN or LINK command has yet been issued against the activity; or the activity has been reset by means of a RESET ACTIVITY command.

PROCESS(data-area)

returns the 36-character name of the process to which this activity belongs.

PROCESSTYPE(data-area)

returns the 8-character name of the process-type to which the process that contains this activity belongs.

PROGRAM(data-area)

returns the 8-character name of the program that executes when this activity is run.

SUSPSTATUS(cvda)

indicates whether the activity is currently suspended. CVDA values are:

SUSPENDED

The activity is currently suspended. If a reattachment event occurs, it will not be reactivated.

NOTSUSPENDED

The activity is not currently suspended. If a reattachment event occurs, it will be reactivated.

TRANSID(data-area)

returns the 4-character transaction identifier under which this activity runs.

USERID(data-area)

returns the 8-character identifier of the user under whose authority this activity runs.

Conditions**ACTIVITYERR**

RESP2 values:

- 1** The activity identifier specified on the ACTIVITYID option does not relate to any activity that is within the scope of this task.
- 19** The request timed out. It may be that another task using this activity-record has been prevented from ending.
- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

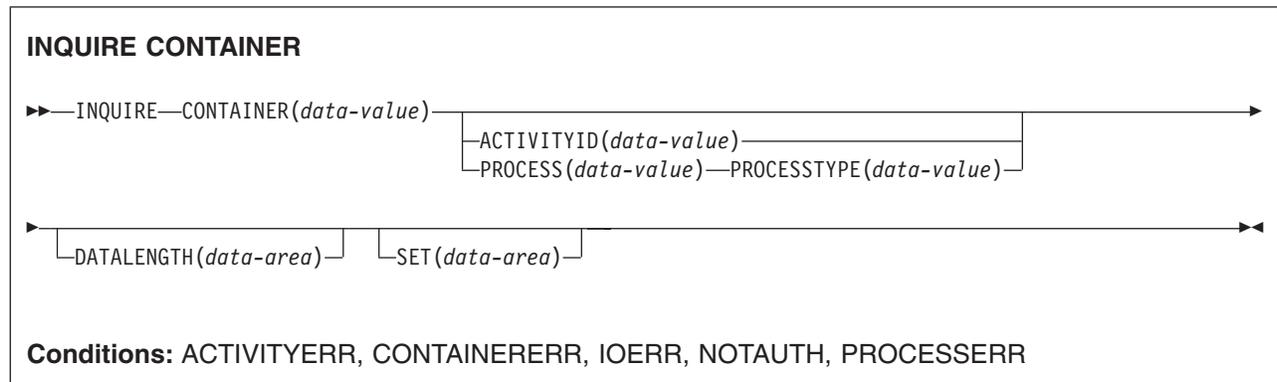
NOTAUTH

RESP2 values:

- 101** The user associated with the issuing task is not authorized to access this resource in the way requested.

INQUIRE CONTAINER

Retrieve the attributes of a BTS data-container.



Description

INQUIRE CONTAINER returns a pointer to the contents of a named BTS data-container, plus the length of the data.

To inquire upon a container associated with the current activity, omit the ACTIVITYID and PROCESS options.

To inquire upon a container associated with another activity, specify the ACTIVITYID option. (The activity identifier specified on the ACTIVITYID option may, for example, have been returned on a GETNEXT ACTIVITY command during a browse operation.)

To inquire upon a process container (including one associated with the *current* process), specify the PROCESS and PROCESSTYPE options.

Note:

1. Inquiring on a container of the current activity returns details of the in-storage version, rather than the committed version on the repository. This means that it's possible to see:
 - Containers that are not yet on the repository
 - Container contents that differ from those on the repository.
2. Inquiring on a container not owned by the current activity returns details of the committed version on the repository. However, the read of the repository record is “dirty”—the record is not locked. So, if the record is being updated by another task, it's possible for the returned data to be unreliable.

Options

ACTIVITYID(data-value)

specifies the identifier (1–52 characters) of the activity which the data-container is associated with.

If both this and the process options are omitted, the current activity is assumed.

CONTAINER(data-value)

specifies the name (1–16 characters) of the data-container being inquired upon.

DATALENGTH(data-area)

returns the fullword length of the data contained in the named data-container.

PROCESS(data-value)

specifies the name (1–36 characters) of the process which the data-container is associated with.

If both this and the ACTIVITYID option are omitted, the current activity is assumed.

PROCESSTYPE(data-value)

specifies the process-type (1–8 characters) of the process named in the PROCESS option.

SET(data-area)

returns a pointer to the contents of the data-container.

Conditions**ACTIVITYERR**

RESP2 values:

- 2** The activity indicated by the ACTIVITYID option could not be found.
- 3** Because neither the ACTIVITYID nor the PROCESS options were specified, an inquiry on the current activity was implied—but there is no current activity associated with the request.
- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

CONTAINERERR

RESP2 values:

- 1** The container specified on the CONTAINER option could not be found.

IOERR

RESP2 values:

- 30** An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

- 101** The user associated with the issuing task is not authorized to access this resource in the way requested.

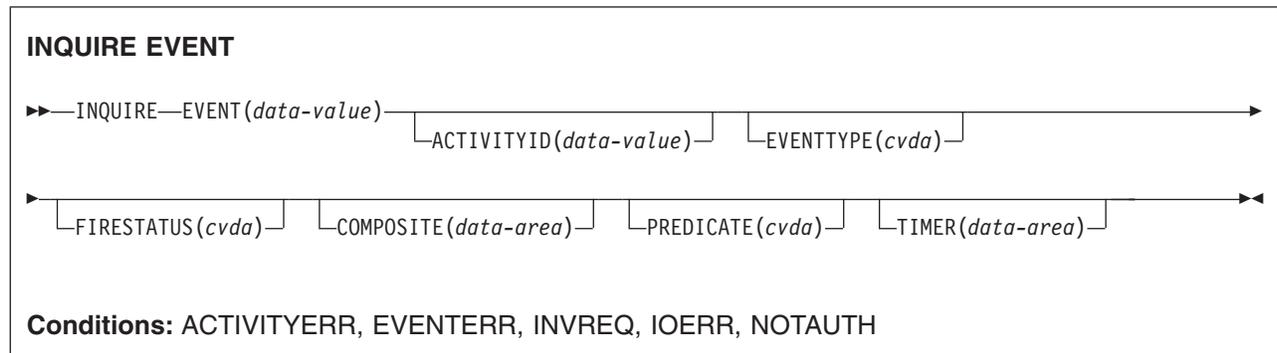
PROCESSERR

RESP2 values:

- 2** The process-type specified on the PROCESSTYPE option could not be found.
- 4** The process specified on the PROCESS option could not be found.
- 13** The request timed out. It may be that another task using this process-record has been prevented from ending.

INQUIRE EVENT

Retrieve the attributes of a BTS event.



Description

INQUIRE EVENT returns the attributes of a named BTS event.

To inquire upon an event associated with the current activity, omit the ACTIVITYID option. To inquire upon an event associated with another activity, specify the ACTIVITYID option. (The activity identifier specified on the ACTIVITYID option may, for example, have been returned on a GETNEXT ACTIVITY command during a browse operation.)

Options

ACTIVITYID (data-value)

specifies the identifier (1–52 characters) of the activity which the event is associated with.

If this option is omitted, the current activity is assumed.

COMPOSITE (data-area)

returns, if the named event is a sub-event, the 16-character name of the composite event that it is part of.

EVENT (data-value)

specifies the name (1–16 characters) of the event being inquired upon.

EVENTTYPE (cvda)

indicates the type of the named event. CVDA values are:

ACTIVITY

Activity completion

COMPOSITE

Composite

INPUT Input

SYSTEM

System

TIMER

Timer.

FIRESTATUS (cvda)

indicates the state of the named event. CVDA values are:

FIRED The event has fired normally.

NOTFIRED

The event has not fired.

PREDICATE(cvda)

indicates, if the named event is composite, the Boolean operator applied to its predicate. CVDA values are:

AND The Boolean operator applied to the predicate is AND.

OR The Boolean operator applied to the predicate is OR.

TIMER(data-area)

returns, if the named event is a timer event, the 16-character name of the timer.

Conditions

ACTIVITYERR

RESP2 values:

3 The activity indicated by the ACTIVITYID option could not be found.

29 The repository file is unavailable.

30 An input/output error has occurred on the repository file.

EVENTERR

RESP2 values:

1 The event specified on the EVENT option could not be found.

INVREQ

RESP2 values:

1 There is no current activity within the scope of this task.

IOERR

RESP2 values:

30 An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

101 The user associated with the issuing task is not authorized to access this resource in the way requested.

INQUIRE PROCESS

Retrieve the attributes of a BTS process.

INQUIRE PROCESS

▶—INQUIRE—PROCESS(*data-value*)—PROCESSTYPE(*data-value*)—
└─ACTIVITYID(*data-area*)—▶

Conditions: ILLOGIC, NOTAUTH, PROCESSERR

Description

INQUIRE PROCESS returns the attributes of a named BTS process. It can be used, for example, to obtain the identifier of the root activity of a process, in order to start a browse of the root activity's child activities, containers, or events.

Options

ACTIVITYID(*data-area*)

returns the 52-character identifier of the root activity of the process that is being queried.

PROCESS(*data-value*)

specifies the name (1–36 characters) of the process to be queried.

PROCESSTYPE(*data-value*)

specifies the process-type (1–8 characters) of the process to be queried.

Conditions

ILLOGIC

RESP2 values:

- 1 A browse of this resource type is already in progress.

NOTAUTH

RESP2 values:

- 101 The user associated with the issuing task is not authorized to access this resource in the way requested.

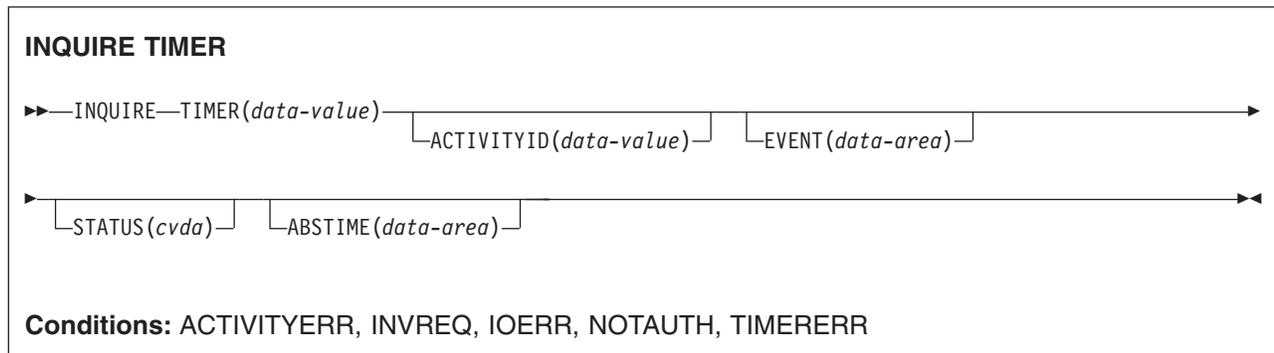
PROCESSERR

RESP2 values:

- 1 The process specified on the PROCESS option could not be found.
- 4 The process-type specified on the PROCESSTYPE option could not be found.

INQUIRE TIMER

Retrieve the attributes of a BTS timer.



Description

INQUIRE TIMER returns the attributes of a named BTS timer.

To inquire upon a timer associated with the current activity, omit the ACTIVITYID option. To inquire upon a timer associated with another activity, specify the ACTIVITYID option. (The activity identifier specified on the ACTIVITYID option may, for example, have been returned on a GETNEXT ACTIVITY command during a browse operation.)

Options

ABSTIME(data-area)

returns, in packed decimal format, the time at which the timer will expire, expressed in milliseconds since 00:00 on 1 January 1900 (rounded to the nearest hundredth of a second).

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

ACTIVITYID(data-value)

specifies the identifier (1–52 characters) of the activity which the timer is associated with.

If this option is omitted, the current activity is assumed.

EVENT(data-area)

returns the 16-character name of the event (if any) associated with the timer.

STATUS(cvda)

indicates the state of the timer. CVDA values are:

EXPIRED

The timer expired normally.

FORCED

Expiry of the timer was forced by means of a FORCE TIMER command.

UNEXPIRED

The timer has not yet expired.

TIMER(data-value)

specifies the name (1–16 characters) of the timer.

Conditions

ACTIVITYERR

RESP2 values:

- 3** The activity indicated by the ACTIVITYID option could not be found.
- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

INVREQ

RESP2 values:

- 1** The command was issued outside the scope of a currently—active activity.

IOERR

RESP2 values:

- 30** An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

- 101** The user associated with the issuing task is not authorized to access this resource in the way requested.

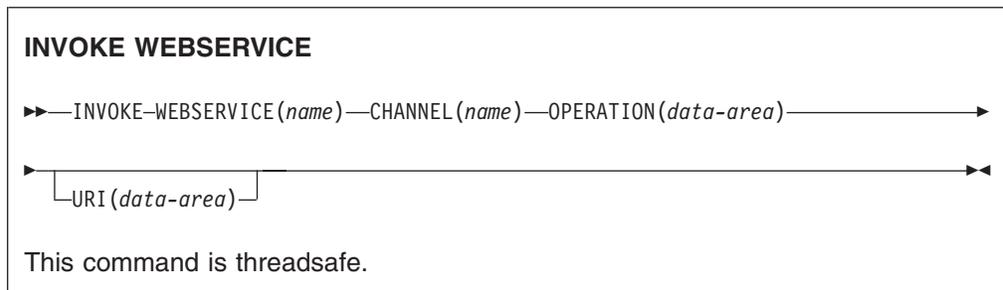
TIMERERR

RESP2 values:

- 1** The timer specified on the TIMER option could not be found.

INVOKE WEBSERVICE

This command invokes a Web service from a CICS application. The command specifies the name of a WEBSERVICE resource, which contains information about the service to be invoked.



Description

INVOKE WEBSERVICE enables a CICS application to act as a Web service requester.

The INVOKE WEBSERVICE command drives the XWBOPEN user exit, which can
make the connection to the server go through a proxy server, if required.

Options

CHANNEL(*name*)

specifies the name of the channel used to pass the containers that hold the data mapped by the application data structure. On return, the same channel holds the response from the Web service, again mapped by the application data structure. The name of the channel can be up to 16 characters. If *name* is a variable, and it contains a name that is less than 16 characters, then the variable must be padded with trailing blanks.

OPERATION(*data-area*)

specifies a data area containing the name of the operation that is to be invoked. The name of the operation is contained in the WSDL for the target Web service. The data area must be 255 characters long; if the operation name is less than 255 characters, then the data area must be padded with trailing blanks.

URI(*data-area*)

specifies a data area containing the URI of the Web service to be invoked. If specified, this option supersedes any URI specified in the WEBSERVICE resource definition. If this option is omitted, then the WEBSERVICE resource definition must include either a provider URI or a provider application name. The data area must be 255 characters long; if the URI is less than 255 characters, then the data area must be padded with trailing blanks.

WEBSERVICE(*name*)

specifies the name of the WEBSERVICE resource that defines the Web service to be invoked. The WEBSERVICE resource specifies the location of the Web service description, and the Web service binding file that CICS uses when it communicates with the Web service. The name of the WEBSERVICE can be up to 32 characters. If *name* is a variable, and it contains a name that is less than 32 characters, then the variable must be padded with trailing blanks.

Conditions

INVREQ

RESP2 values:

- 1 The name specified for the CHANNEL option contains an illegal character or combination of characters.
- 2 The name specified for the OPERATION option contains an illegal character or combination of characters.
- 3 The Web service binding file associated with the WEBSERVICE is invalid.
- 4 The value specified for the URI contained an illegal character or combination of characters.
- 5 The PIPELINE used by the WEBSERVICE is defined as a service requester pipeline but is invoked in a service provider or *vice versa*.
- # 6 The invoked WEBSERVICE returned a SOAP fault. The description of the fault is available in its XML format in the container DFHWS-BODY.
- #
- 7 The URI option was not specified on the command, and the WEBSERVICE definition does not specify a URI or a program name.
- 8 The WEBSERVICE is not in service
- 9 The container **DFHWS-DATA** does not have the correct DATATYPE. For this container, a DATATYPE of BIT must be specified.
- 10 The PIPELINE used by the WEBSERVICE is not enabled.
- 11 CICS could not link to the program specified in the WEBSERVICE definition.
- 12 The containers that the command expects were not on the correct channel.
- 13 The application data structure contains invalid data that cannot be converted to a SOAP request. No further information is available.
- # 14 A conversion error occurred when CICS attempted to convert between the application data structure and the SOAP message. Either the application data structure contains invalid data that cannot be converted to a SOAP request, or data in the SOAP response message cannot be converted into the application's data structure. Some possible causes of this condition are:
 - # • A value in the SOAP response message is larger than the corresponding field in the application's data structure.
 - # • When building the SOAP request, the Web services binding file indicates that a data field contains packed decimal or zoned decimal data, and the contents of the field are invalid for this data type.
- 15 An unhandled error has occurred in the pipeline. Information about the error is in container DFHERROR.
- # 16 A locally optimized Web service has abended. The underlying unit of work has been backed out.
- #
- # 17 A remote Web service request did not return a response message.
- # 101 The container **DFHWS-BODY** does not have the correct DATATYPE. For this container, a DATATYPE of CHAR must be specified.
- #

- # **103** The container **DFHWS-BODY** contains no data.
- # **104** Either the container **DFHREQUEST** or the container **DFHWS-BODY** is missing.
- # **105** A fault was built within the service requester **PIPELINE** used by the **WEBSERVICE**, either while the request was being sent, or while the response was being processed. This condition could indicate that a header processing program has issued a fault.
- # **106** Either the generated SOAP request message was not well formed, or the SOAP response message was not well formed. This condition could indicate that the XML parser returned a fatal error code.
- # **107** Either the generated SOAP request message was not a valid SOAP message, or the SOAP response message was not a valid SOAP message.

NOTFND

RESP2 values:

- 1** The Web service binding file associated with the **WEBSERVICE** specifies the name of a SOAP message parsing program supplied by another product, but the parsing program could not be found.
- 2** The **CHANNEL** specified could not be located.
- 3** The **OPERATION** specified was not in the Web service binding file.
- 4** The **WEBSERVICE** specified could not be located.
- 5** A **CONTAINER** specified in the Web service binding file could not be located.

TIMEDOUT

- # **62** Timeout on socket receive.

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) in any of the following situations:

- ISSUE ABEND is used on any conversation other than an EXEC CICS APPC mapped conversation.

Default action: terminate the task abnormally.

NOTALLOC

occurs if the specified CONVID value relates to a conversation that is not owned by the application.

Default action: terminate the task abnormally.

TERMERR

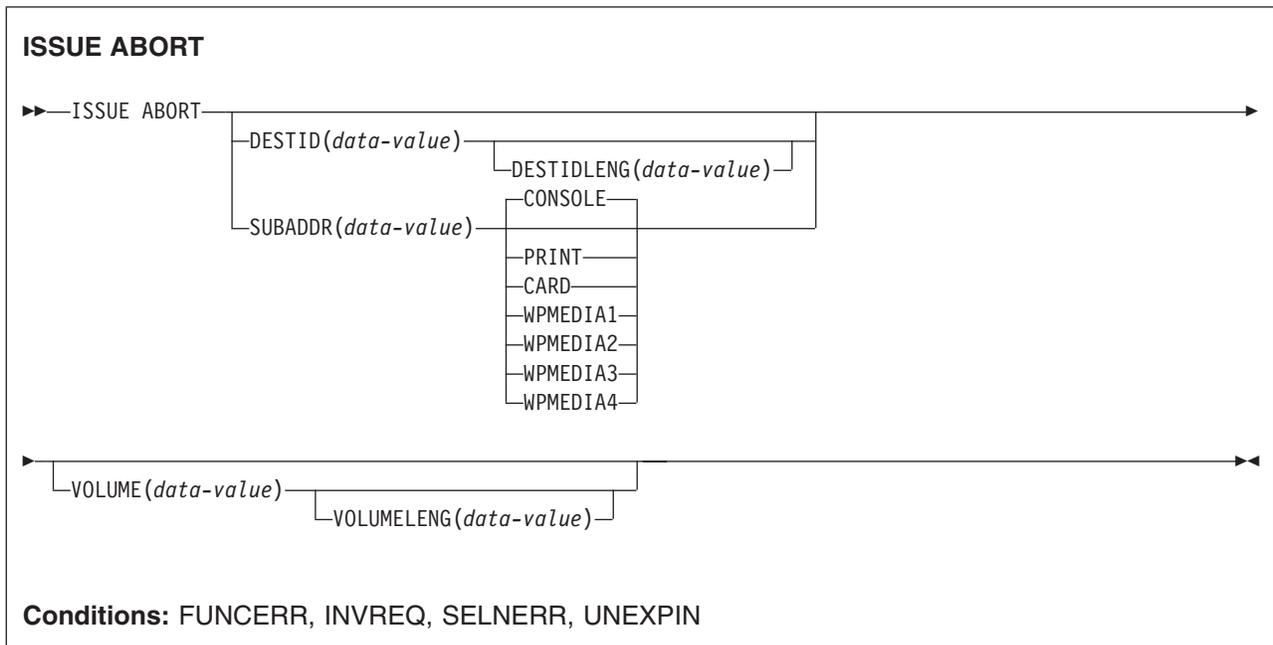
occurs for a session-related error. Any action on that conversation other than a FREE command causes an ATCV abend.

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

End processing of a data set abnormally.



Description

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.

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.

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

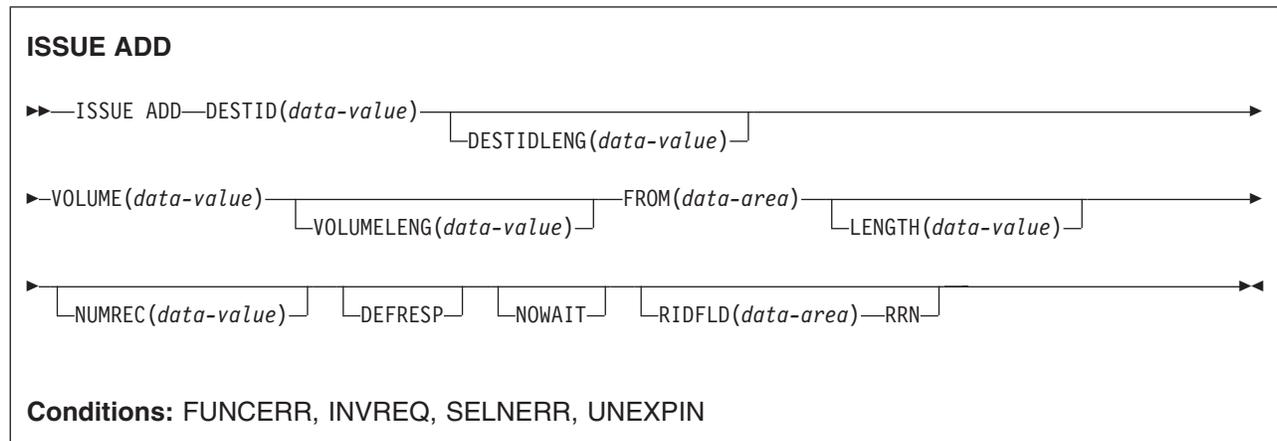
UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE ADD

Add a record to a data set.



Description

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).

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 in CICS commands” on page 8.

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.

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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

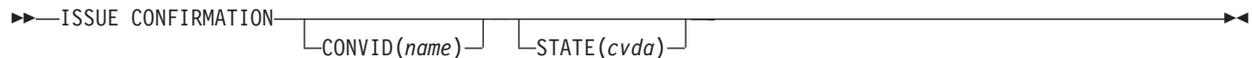
occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE CONFIRMATION

Issue a positive response to a SEND CONFIRM on an APPC mapped conversation.

ISSUE CONFIRMATION (APPC)



Conditions: INVREQ, NOTALLOC, SIGNAL, TERMERR

Description

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.

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 facility (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 both CONVID and SESSION are 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

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session on the CONVID option.

also occurs (RESP2 not set) in any of the following situations:

- ISSUE CONFIRMATION is used on a conversation that is either of the following:
 - Sync level 0
 - Not APPC mapped

Default action: terminate the task abnormally.

NOTALLOC

occurs if the specified CONVID value relates to a conversation that 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 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 logical)

Copy data from 3270 logical unit.

ISSUE COPY (3270 logical)

►—ISSUE COPY—TERMINID(*name*)—┐ CTLCHAR(*data-value*) ┐ WAIT ┐—►

Conditions: LENGERR, NOTALLOC, TERMERR

Description

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.

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.

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.

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 DISCONNECT (default)

Terminate a session between CICS and a logical unit or terminal.

ISSUE DISCONNECT (default)

▶—ISSUE DISCONNECT—▶

Conditions: SIGNAL, TERMERR

Description

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
- 2260 or 2265 display station
- 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
- 3767/3770 interactive logical unit
- 3770 batch logical unit
- 3790 logical units

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 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 (LUTYPE6.1)

Disconnect an LUTYPE6.1 logical unit.

ISSUE DISCONNECT (LUTYPE6.1)

▶—ISSUE DISCONNECT—┐
 └SESSION(*name*)┘

Conditions: NOTALLOC, TERMERR

Description

ISSUE DISCONNECT disconnects the unit, if DISCREQ=YES is set in the TYPETERM resource definition.

Options

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.

Conditions

NOTALLOC

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

Default action: terminate the task abnormally.

TERMERR

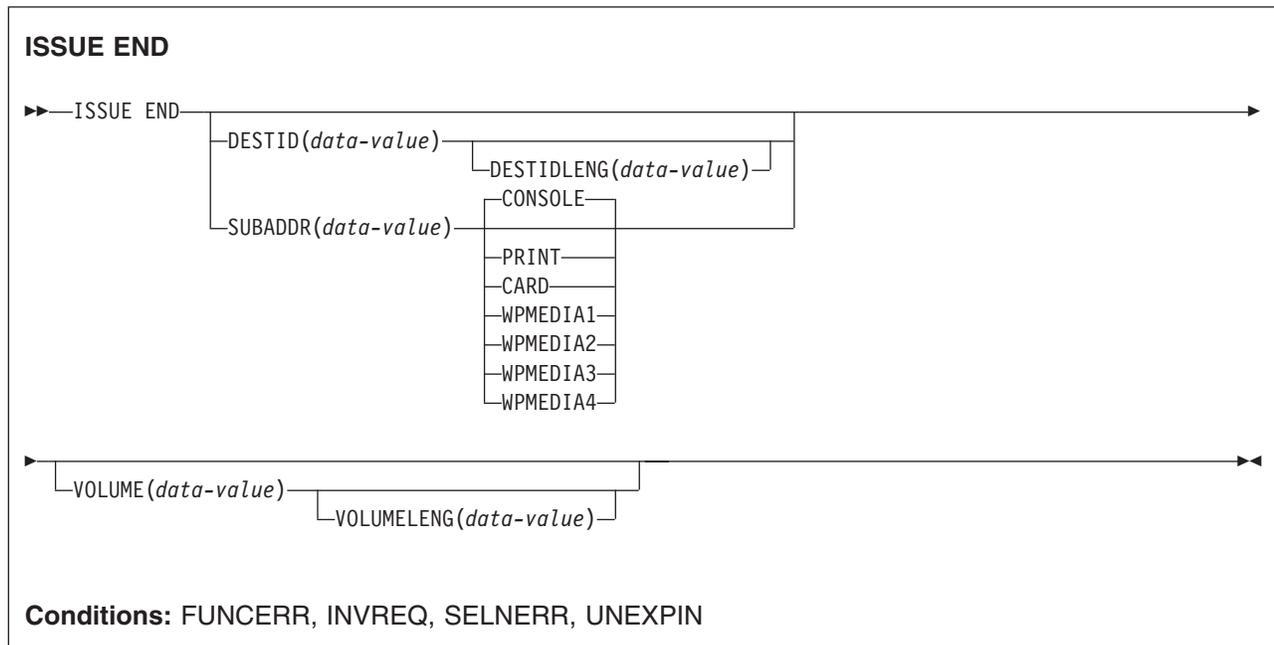
occurs for a terminal-related error, such as a session failure.

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 with abend code ATNI.

ISSUE END

End processing of a data set.



Description

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.

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.

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.

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

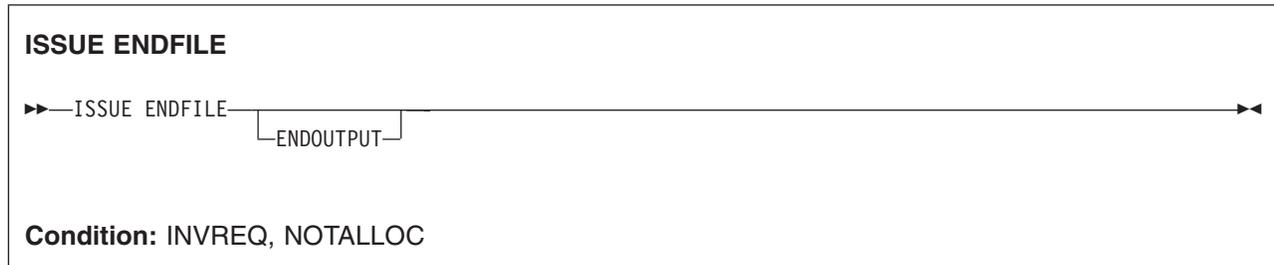
UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE ENDFILE

Indicate the end-of-file condition to the 3740 data entry system.



Description

ISSUE ENDFILE indicates the end-of-file condition to the 3740.

Options

ENDOUTPUT

indicates the end-of-output condition as well as end of file.

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application attempted to send on its function shipping session, its principal facility.

Default action: terminate the task abnormally.

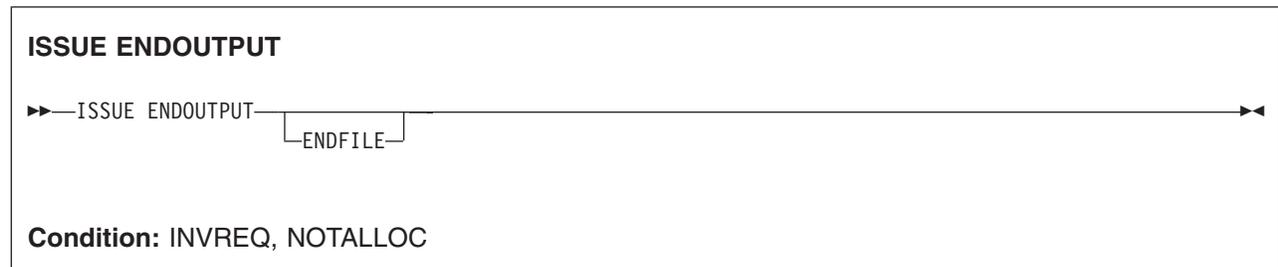
NOTALLOC

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

Default action: terminate the task abnormally.

ISSUE ENDOUTPUT

Indicate the end-of-output condition to the 3740 data entry system.



Description

ISSUE ENDOUTPUT indicates the end-of-output condition to the 3740.

Options

ENDFILE

indicates the end-of-file condition as well as end of output.

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application attempted to send on its function shipping session, its principal facility.

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

Send end-of-data-set function management header to the 3650 interpreter logical unit.

ISSUE EODS

▶—ISSUE EODS—▶

Conditions: INVREQ, NOTALLOC, TERMERR

Description

ISSUE EODS issues the end-of-data-set management header.

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application attempted to send on its function shipping session, its principal facility.

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

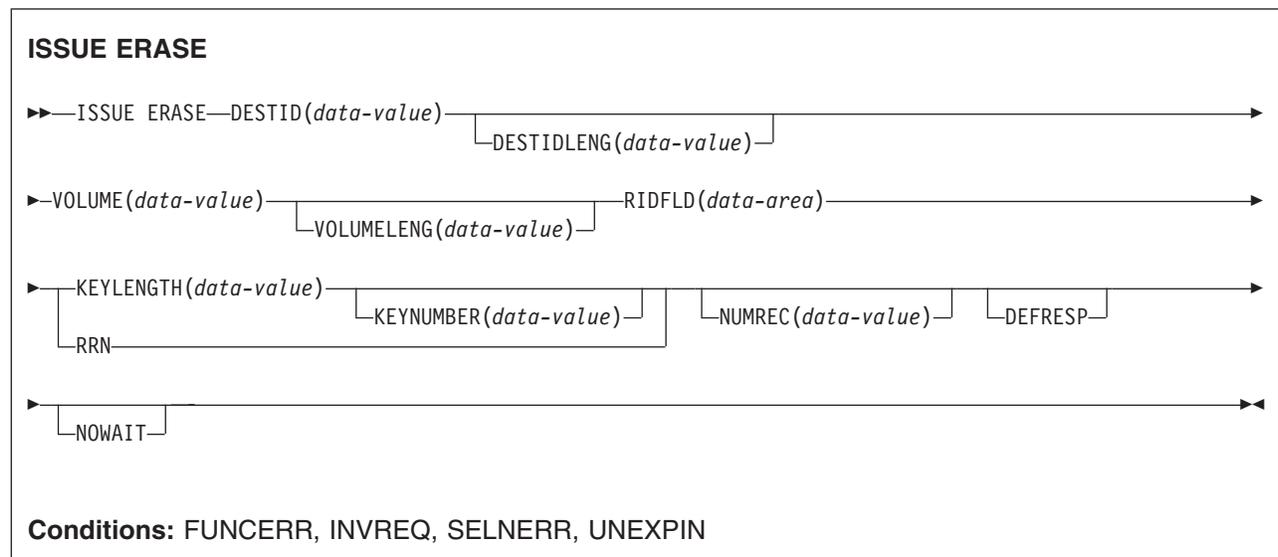
occurs for a terminal-related error, such as a session failure.

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 with abend code ATNI.

ISSUE ERASE

Delete a record from a data set.



Description

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.

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.

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.

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

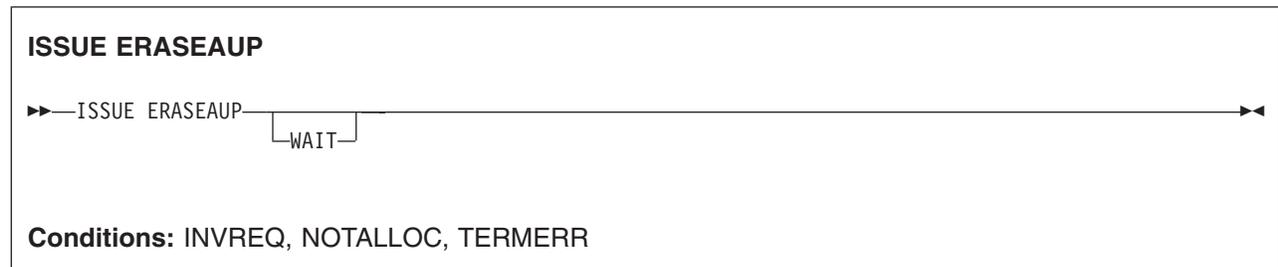
UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE ERASEAUP

Erase all unprotected fields of a 3270 buffer.



Description

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 logical unit
- 3650 host conversational (3270) logical unit
- 3790 (3270-display) logical unit
- 3790 (3270-printer) logical unit

Options

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.

Conditions

INVREQ

RESP2 values:

- 200** A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID 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.

TERMERR

occurs if there is 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 ERROR

Inform APPC mapped conversation partner of error.

ISSUE ERROR (APPC)

►—ISSUE ERROR—┐┐
└CONVID(*name*)┘└STATE(*cvda*)┘

Conditions: INVREQ, NOTALLOC, SIGNAL, TERMERR

Description

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.

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 facility (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 both CONVID and SESSION are 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

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session on the CONVID option.

also occurs (RESP2 not set) 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.

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.

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

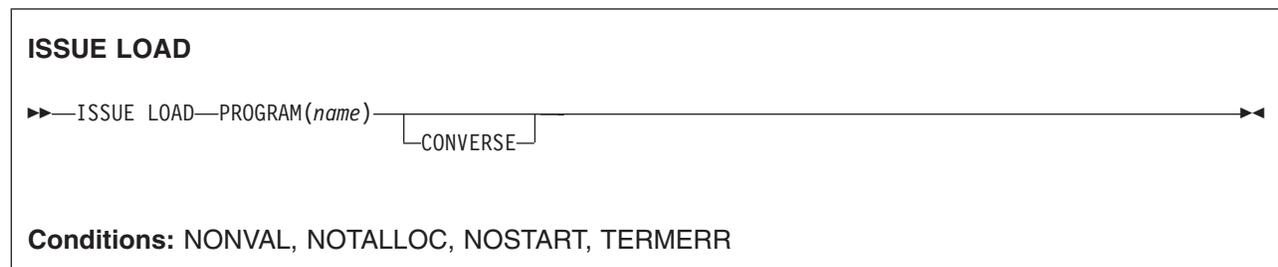
occurs for a session-related error. Any action on that conversation other than a FREE command causes an ATCV abend.

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

Specify the name of a program on 3650 interpreter logical unit.



Description

ISSUE LOAD specifies the name of the 3650 application program that is to be loaded.

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.

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 with abend code ATNI.

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

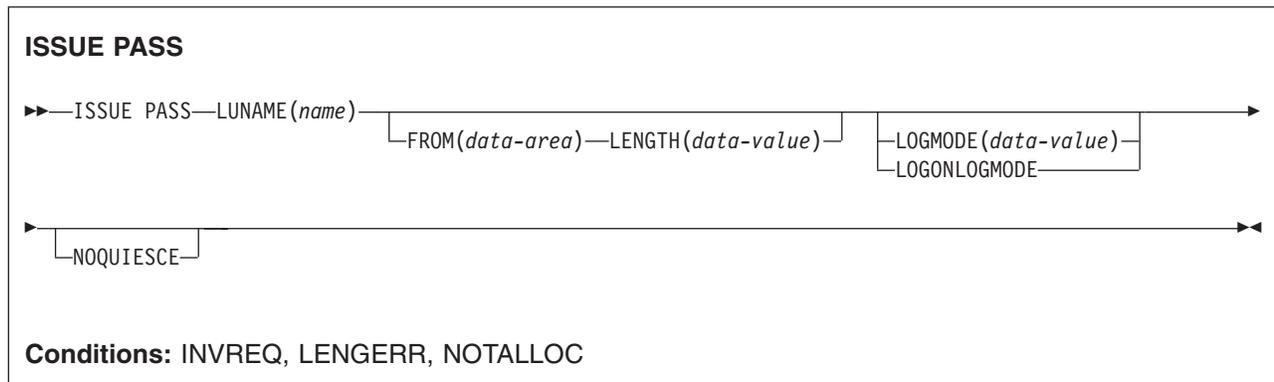
UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE PASS

VTAM application routing.



Description

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 VTAM APPL macro for the CICS terminal-owning system that issues it, with DISCREQ=YES or RELREQ=YES in the RDO TYPETERM resource definition 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=NOTIFYINONOTIFY 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 Customization Guide*.

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*)

specifies the name (1–8 characters) of the VTAM logon mode table entry used by VTAM to establish the new session.

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 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.

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

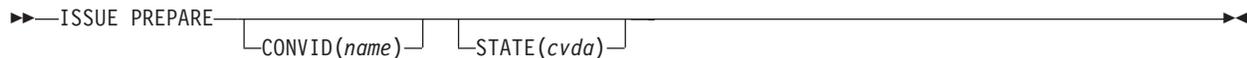
occurs if the facility specified in the command is not owned by the application.

Default action: terminate the task abnormally.

ISSUE PREPARE

Issue the first flow of a syncpoint request on an APPC mapped conversation.

ISSUE PREPARE (APPC)



Conditions: INVREQ, NOTALLOC, TERMERR

Description

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.

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 facility (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 both CONVID and SESSION are 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

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) 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.

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

Print displayed data on first available printer.

ISSUE PRINT

▶—ISSUE PRINT—▶

Conditions: INVREQ, NOTALLOC, TERMERR

Description

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 logical unit or a 3650 host conversational (3270) logical unit, the printer must be defined by the PRINTER or ALTPRINTER options on the RDO TERMINAL resource definition, 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 specifying DEVICE=LUTYPE2 and PRINTADAPTER=YES on the RDO TYPETERM resource definition.
- 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.

Conditions

INVREQ

RESP2 values:

- 200** A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID 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.

TERMERR

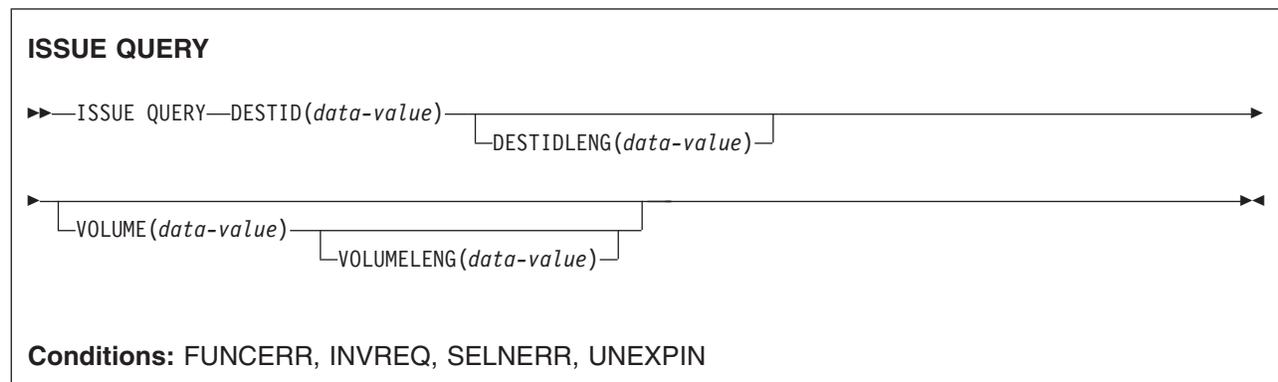
occurs for a terminal-related error.

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 QUERY

Interrogate a data set.



Description

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.

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 (1–6 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.

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

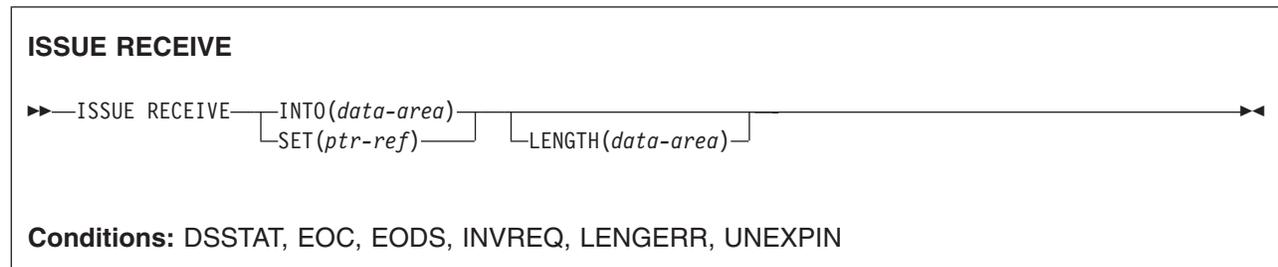
UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE RECEIVE

Read a record from a data set.



Description

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).

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.

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.

EODS

occurs when the end of the data set is encountered.

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

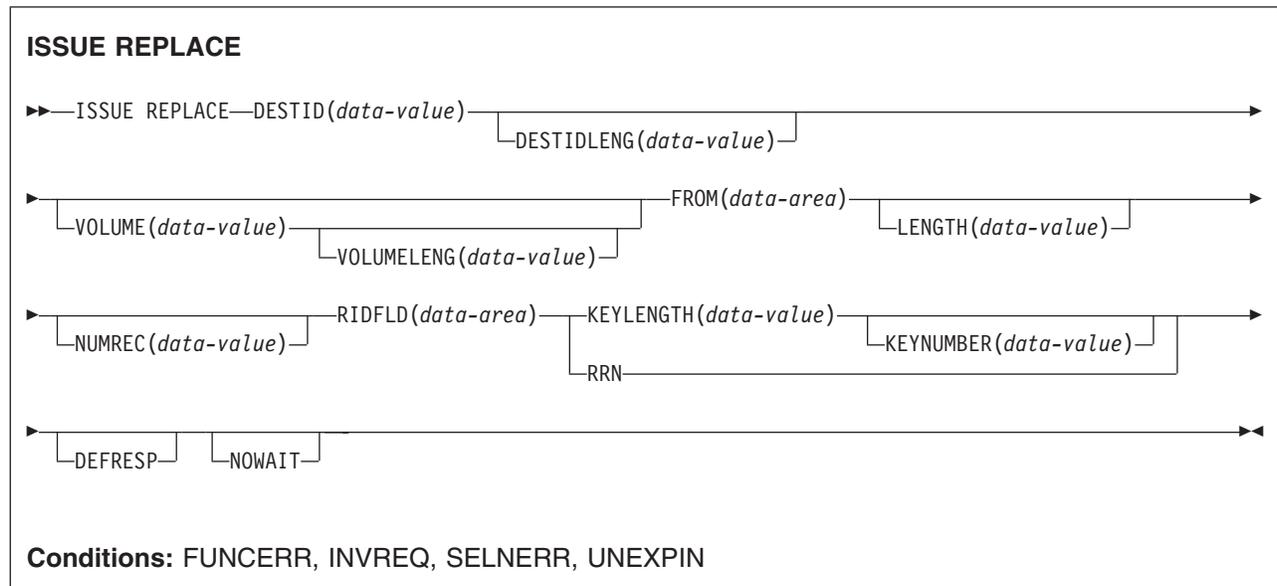
UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE REPLACE

Update a record in a data set.



Description

ISSUE REPLACE updates (replaces) a record in either a relative (addressed direct) or an indexed (keyed direct) data set in an outboard controller.

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.

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 (1–6 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.

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

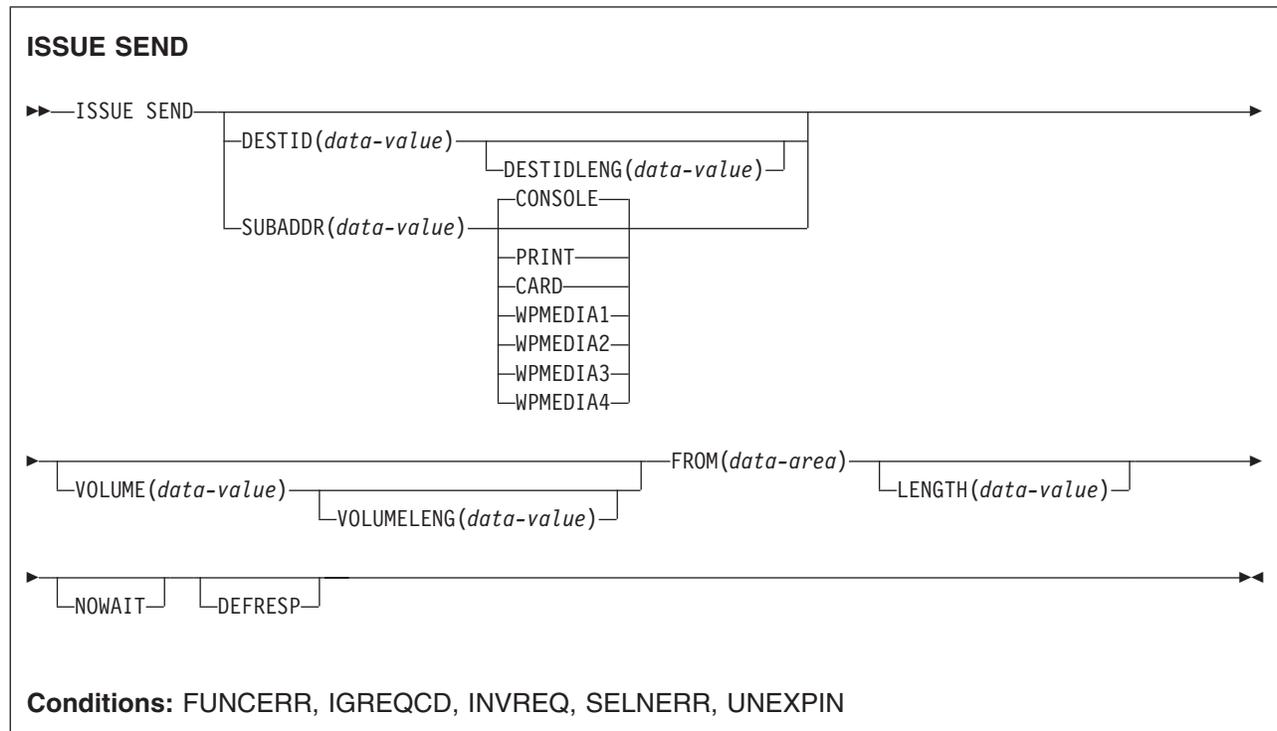
ISSUE RESET

Relinquish use of a telecommunication line.

This command is supported for compatibility with earlier releases of CICS. It is superseded by the `ISSUE DISCONNECT` command, which you are recommended to use instead.

ISSUE SEND

Send data to a named data set or to a selected medium.



Description

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.

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 (1–6 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.

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
RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

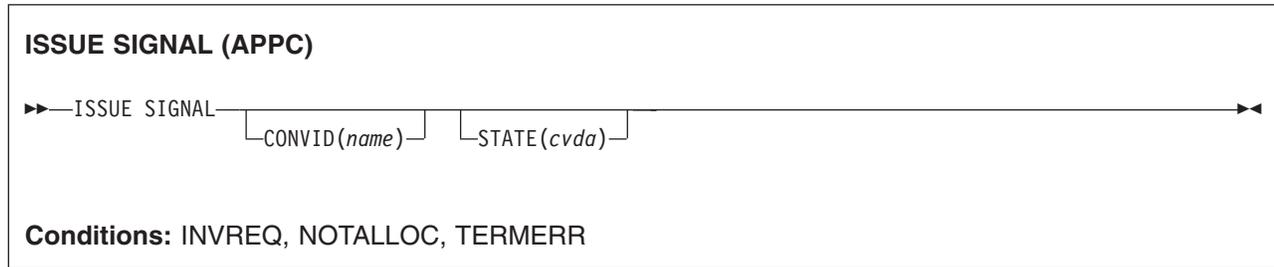
UNEXPIN

occurs when some unexpected or unrecognized information is received from the outboard controller.

Default action: terminate the task abnormally.

ISSUE SIGNAL (APPC)

Request change of direction from sending transaction on an APPC mapped conversation.



Description

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.

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 facility (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 both CONVID and SESSION are 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

Conditions

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) 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.

Default action: terminate the task abnormally.

NOTALLOC

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

Default action: terminate the task abnormally.

TERMERR

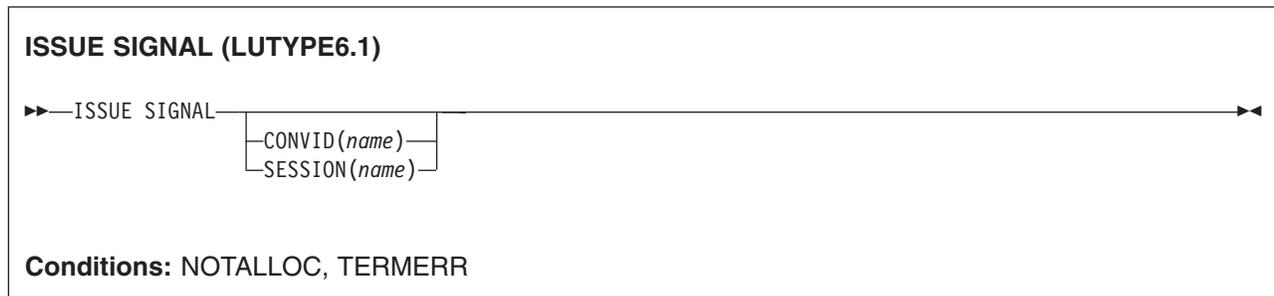
occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

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)

Request change of direction from sending transaction on LUTYPE6.1 conversation.



Description

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.

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 facility (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.

Conditions

NOTALLOC

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

Default action: terminate the task abnormally.

TERMERR

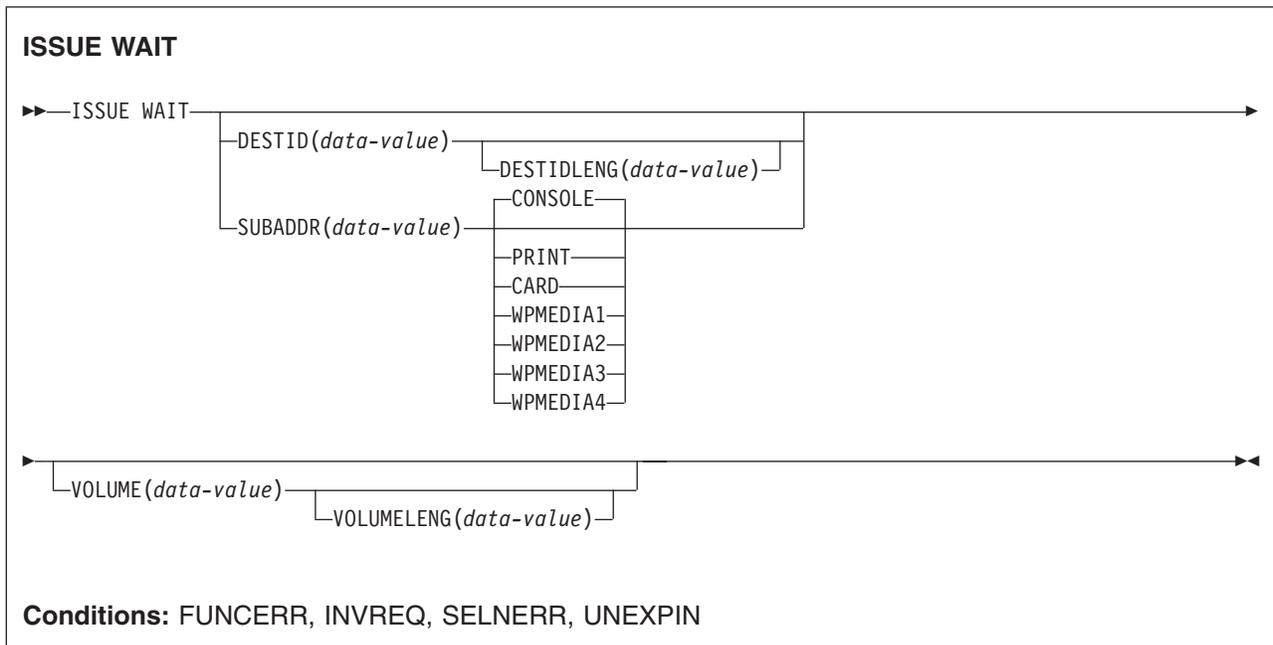
occurs for a session-related error. Any action on that conversation other than a FREE causes an ATCV abend.

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 WAIT

Wait for an operation to be completed.



Description

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.

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.

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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.

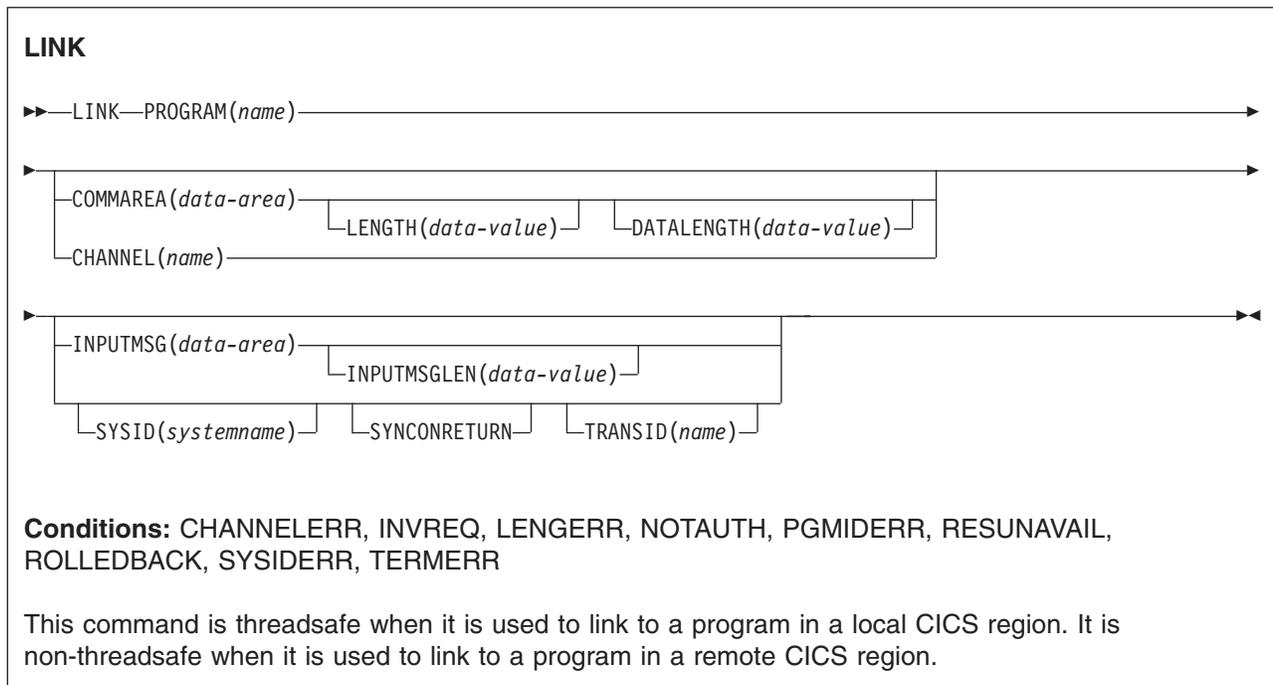
JOURNAL

Create a journal record.

This command is supported for compatibility with earlier releases of CICS. It is superseded by the `WRITE JOURNALNAME` command, which you are recommended to use instead.

LINK

Link to another program expecting return. The external CICS interface (EXCI) provides a LINK command that performs all six commands of the interface in one invocation. See the *CICS External Interfaces Guide* for information about the EXCI.



Description

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 this is a local definition, and the linked-to program is not already in main storage, CICS loads it.

In some circumstances, the linked-to program may reside on another CICS region—see “Distributed program link” on page 343.

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 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 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 Application Programming Guide* for further details

about the relationship between LINK and HANDLE ABEND.

Distributed program link

In any of the following cases, the link is a *distributed program link* (DPL):

- You specify a remote region name on the SYSID option (with or without the associated TRANSID and SYNCONRETURN options).
- The REMOTESYSTEM option on the installed PROGRAM definition ² specifies the name of a remote region.
- The installed program definition specifies DYNAMIC(YES)—or there is no installed program definition—and the dynamic routing program routes the link request to a remote region.

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 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 “API restrictions for distributed program link” on page 774.

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.

You cannot use DPL to link to the CICS master terminal program, DFHEMTA, or to the RDO program, DFHEDAP. The addresses passed as parameters to DFHEMTA and DFHEDAP are valid only in the region that issues the EXEC CICS LINK command, which means you cannot route a DFHEMTA or DFHEDAP request to a remote CICS.

2. By “installed program definition” we mean a program definition that has been installed statically, by means of autoinstall, or by an EXEC CICS CREATE command.

Important: For examples of the use of the LINK command when the linked program is remote, see the *CICS Application Programming Guide*. For information about writing a dynamic routing program, see the *CICS Customization Guide*.

Options

CHANNEL (*name*)

specifies the name (1–16 characters) of a channel that is to be made available to the invoked program. The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

Channel names are always in EBCDIC. The allowable set of characters for channel names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if containers are to be shipped between regions, the characters used in naming them should be restricted to A-Z a-z 0-9 & : = , ; < > . - and _ .

The program that issues the LINK command may:

- Have created the channel by means of one or more PUT CONTAINER CHANNEL commands
- Specify its current channel, by name
- Name a non-existent channel, in which case a new, empty, channel is created

COMMAREA (*data-area*)

specifies a communication area that is to be made available to the invoked program. In this option the data area is passed, and you must give it the name DFHCOMMAREA in the receiving program. (See the section about passing data to other programs in the *CICS 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. For a remote LINK request, 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.

The value of DATALENGTH is only checked when the LINK request is remote or dynamic. It is not checked for static local links.

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 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.

In any of the following cases, the linked-to program is a server program in a remote region:

- The SYSID option specifies a remote region.
- The REMOTESYSTEM option on the installed PROGRAM definition ³ specifies the name of a remote region.
- The installed program definition specifies DYNAMIC(YES)—or there is no installed program definition—and the dynamic routing program routes the link request to a remote region.

Note the use of quotes:

PROGX is in quotes because it is the program name.

```
EXEC CICS LINK PROGRAM('PROGX')
```

DAREA is not in quotes because it is the name of a data area that contains the

```
EXEC CICS LINK PROGRAM(DAREA)
```

actual program name.

Note: When linking to a CICS 3270 program that is to be executed under the Link3270 bridge mechanism, the PROGRAM name must be DFHL3270, not the name of the target 3270 program.

SYNCONRETURN

specifies that the server region named on the SYSID option is to take a syncpoint on successful completion of the server program.

Changes to recoverable resources made by the server program are committed or rolled-back independently of changes to recoverable resources made by the client program issuing the LINK request, or changes made by the server in any subsequent LINK.

- The NORMAL condition is returned if changes to recoverable resources are committed before return from the server program.
- The ROLLEDBACK condition is returned if changes to recoverable resources are rolled back before return from the server program.
- The TERMERR condition is raised following failure of the communications link or the system in which the server program is executing. The client program is responsible for handling the condition and ensuring that data consistency is restored.

3. By “installed program definition” we mean a program definition that has been installed statically or dynamically by means of autoinstall, or by an EXEC CICS CREATE command.

Synconreturn is only applicable to remote LINKs, it is ignored if the LINK is local.

SYSID(*systemname*)

specifies the system name of a CICS server region to where the program link request is to be routed.

If you do not specify a remote system name in the SYSID option or omit the SYSID option, CICS uses the REMOTESYSTEM attribute defined in the installed PROGRAM definition. If you specify a local system name in the SYSID option or the REMOTESYSTEM attribute, CICS ignores the name.

A remote system name specified on the SYSID option takes priority over any remote system name specified on the PROGRAM resource definition or returned by the dynamic routing program.

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, reference is made to PROGRAM resource definitions held locally if the installed PROGRAM definition specifies remote attribute DYNAMIC(YES). Otherwise, 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.

Conditions

CHANNELERR

RESP2 values:

- 1 The name specified on the CHANNEL option contains an illegal character or combination of characters.

INVREQ

RESP2 values:

- 8 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.
- 14 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. (That is, a unit of work (UOW) 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.
- 15 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.
- 16 The TRANSID specified is all blanks.
- 17 The TRANSID supplied by the dynamic routing program is all blanks.
- 19 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.

#

- 30 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.
- 44 A LINK has been attempted to a Java program, but the JVMpool is disabled.
- 45 A LINK has been attempted to a Java program, but the JVM profile cannot be found.
- 46 A LINK has been attempted to a Java program, but the JVM profile is not valid.
- 47 A LINK has been attempted to a Java program, but the system properties file cannot be found.
- 48 A LINK has been attempted to a Java program, but the user class cannot be found.
- 49 The shared class cache is STOPPED and autostart is disabled, so a Java program requesting use of the shared class cache cannot be executed.
- 50 The Language Environment options specified in DFHJVMRO are too long.

Default action: terminate the task abnormally.

Note: RESP2 values are not returned to the client for conditions occurring in a DPL server program.

LENGERR

RESP2 values:

- 11 The COMMAREA length is less than 0 or greater than 32767.
- 12 The length specified on the DATALENGTH option is a negative value.
- 13 The length specified on the DATALENGTH option is greater than the length specified on the LENGTH option.
- 26 The COMMAREA address is zero, but the COMMAREA length is non zero.
- 27 The INPUTMSG length is less than 0 or greater than 32767.

also occurs (RESP2 not set) 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.

Default action: terminate the task abnormally.

Note: RESP2 values are not returned to the client for conditions occurring in a DPL server program.

NOTAUTH

RESP2 values:

- 101 A resource security check has failed on PROGRAM(name).

Default action: terminate the task abnormally.

PGMIDERR

RESP2 values:

- 1 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.

- 2 A program is disabled.
- 3 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.

- 21 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.
- 22 The model returned by the program autoinstall control program was not defined in the PPT table, or was not enabled.
- 23 The program autoinstall control program returned invalid data.
- 24 Define for the program failed due to autoinstall returning an invalid program name or definition.
- 25 The dynamic routing program rejected the link request.

Default action: terminate the task abnormally.

Note: RESP2 values are not returned to the client for conditions occurring in a DPL server program.

RESUNAVAIL

RESP2 values:

- 0 A resource required by the linked-to program is unavailable on the target region. The RESUNAVAIL condition applies to dynamically-routed distributed program link (DPL) requests.

RESUNAVAIL is returned on the EXEC CICS LINK command *executed by the mirror in the target region*, if an XPCERES global user exit program indicates that a required resource is unavailable on the target region. It is not returned to the application.

Default action: reinvoke the dynamic routing program for route selection failure.

ROLLEDBACK

RESP2 values:

- 29 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 unit of work, are backed out.

Default action: terminate the task abnormally.

SYSIDERR

RESP2 values:

- 18 The SYSID specified cannot be found in the intersystem table.

20 The remote system specified by SYSID is an LUTYPE6.1-connected system. Distributed program link requests are not supported on LUTYPE6.1 connections.

Note:

1. There is no local queuing in the event of a SYSIDERR.
2. RESP2 values are not returned for conditions occurring on DPL requests.

21 The CHANNEL option was used and the LINK request was shipped or routed to a remote system which doesn't support it. (MRO connections only)

28 The remote system specified by SYSID is not in service. This response can also indicate that the transaction has not been defined on the remote system.

29 The remote system specified by SYSID is in service, but there are no sessions available, and the dynamic routing program has chosen not to queue the link request.

31 The request to allocate a session to the remote system has been rejected.

32 The queue of allocate requests for sessions to the remote system has been purged.

Default action: terminate the task abnormally.

TERMERR

RESP2 values:

17 An irrecoverable error occurs during the conversation with the mirror (for example, if the session fails, or if the server region fails).

Default action: terminate the task abnormally.

#

If SYNCONRETURN was not specified on the LINK then the client program must decide whether to abend or rollback on receipt of this condition.

Note: RESP2 values are not returned to the client for conditions occurring in a DPL server program.

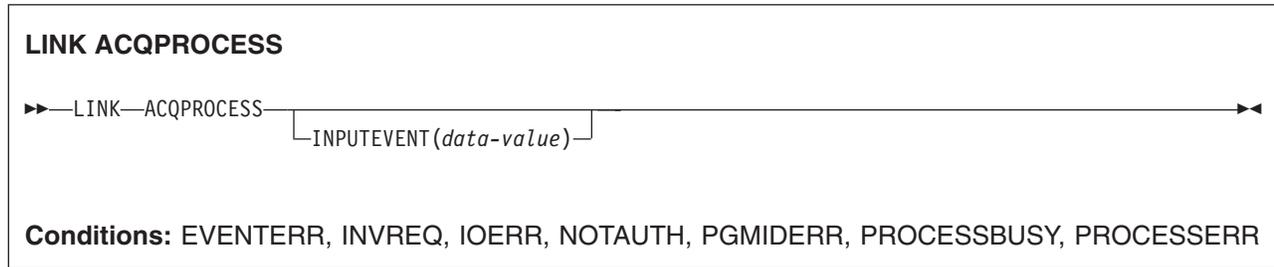
Examples

The following example shows how to request a link to an application program called PROGNAME:

```
EXEC CICS LINK PROGRAM(PROGNAME)
           COMMAREA(COMA) LENGTH(LENA)
           DATALENGTH(LENI) SYSID('CONX')
```

LINK ACQPROCESS

Execute a CICS business transaction services process synchronously without context-switching.



Description

LINK ACQPROCESS executes the CICS business transaction services process currently acquired by the requestor. The process is executed synchronously with the requestor, with no context-switching.

The only process that a program can link is the one that it has acquired in the current unit of work. (Note, however, that if the program is running as the activation of an activity, it must use a RUN, not a LINK, command to activate the process it has acquired.) See the *CICS Business Transaction Services* manual.

To check the response from the process, the CHECK ACQPROCESS command must be used. This is because the response to the request to activate the process does not contain any information about the success or failure of the process itself—only about the success or failure of the request to activate it. Typically, the CHECK command is issued immediately after the LINK command.

LINK ACQPROCESS causes BTS to invoke the process's root activity and send it an input event. If the root activity is in its initial state—that is, if this is the first time it is to be run—CICS sends it the DFHINITIAL system event. If the root activity is not in its initial state, the input event must be specified on the INPUTEVENT option.

No context-switching

When an process is activated by a LINK ACQPROCESS command, it is invoked synchronously with the requestor and:

- In the same unit of work as the requestor
- With the transaction attributes (TRANSID and USERID) of the requesting transaction.

In other words, there is no context-switch. To invoke a process synchronously *with* context-switching—that is, in a separate UOW from that of the requesting transaction and with the TRANSID and USERID attributes specified on its DEFINE PROCESS command—use the RUN ACQPROCESS SYNCHRONOUS command.

Note: A context-switch always occurs when a process is run asynchronously.

If performance is more important than failure isolation, recoverability, and security, use LINK ACQPROCESS rather than RUN ACQPROCESS SYNCHRONOUS.

Options

ACQPROCESS

specifies that the process currently acquired by the requestor is to be run.

INPUTEVENT(*data-value*)

specifies the name (1–16 characters) of the event that causes the process to be attached.

You *must not* specify this option if the process's root activity is in its initial state; that is, if this is the first time the process is to be run. In this case, CICS sends the root activity the DFHINITIAL system event.

You *must* specify this option if the root activity is not in its initial state; that is, if it has been activated before.

If you specify INPUTEVENT, for the LINK command to be successful the root activity must have defined the named event as an input event.

Conditions

EVENTERR

RESP2 values:

- 7 The event named on the INPUTEVENT option has not been defined by the root activity of the process to be run as an input event; or its fire status is FIRED.

INVREQ

RESP2 values:

- 15 The task that issued the LINK command has not defined or acquired a process.
- 23 The process is suspended, and therefore cannot be run synchronously.
- 40 The program that implements the process to be run is remote.
- 44 A LINK has been attempted to a Java program, but the JVM pool is disabled.
- 45 A LINK has been attempted to a Java program, but the JVM profile cannot be found.
- 46 A LINK has been attempted to a Java program, but the JVM profile is not valid.
- 47 A LINK has been attempted to a Java program, but the system properties file cannot be found.
- 48 A LINK has been attempted to a Java program, but the user class cannot be found.
- 49 The shared class cache is STOPPED and autostart is disabled, so a Java program requesting use of the shared class cache cannot be executed.

IOERR

RESP2 values:

- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

- 101** The user associated with the issuing task is not authorized to run the process.

PGMIDERR

RESP2 values:

- 1** A program has no entry in the PPT and either program autoinstall was switched off, or the program autoinstall user program indicated that the program should not be autoinstalled.

- 2** A program is disabled.

- 3** 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.

- 21** The program autoinstall user program failed either because the program autoinstall user program is incorrect, incorrectly defined, or as a result of an abend in the program autoinstall user program. Program autoinstall is disabled and message DFHPG0202 or DFHPG0203 written to the CSPL.

- 22** The model returned by the program autoinstall user program was not defined in the PPT table, or was not enabled.

- 23** The program autoinstall user program returned invalid data.

- 24** Define for the program failed due to autoinstall returning an invalid program name or definition.

PROCESSBUSY

RESP2 values:

- 13** The request timed out. It may be that another task using this process-record has been prevented from ending.

PROCESSERR

RESP2 values:

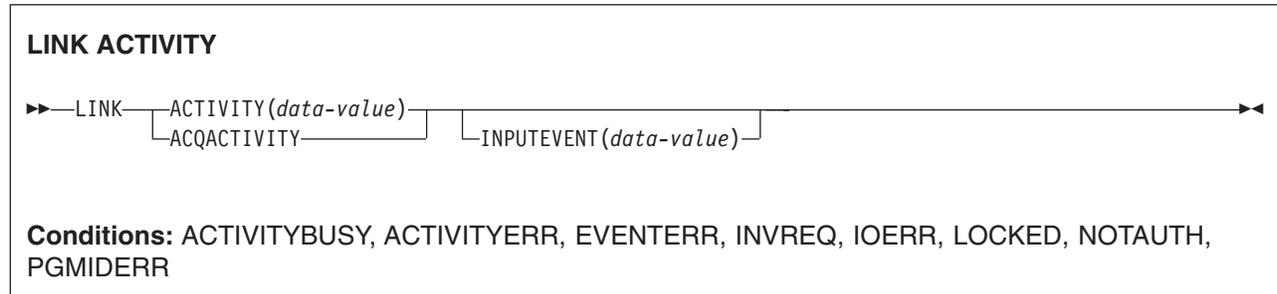
- 6** Another process is current. That is, the program that issued the LINK command cannot link to the process it has acquired because it is itself running as an activation of a process.

- 9** The process-type could not be found.

- 14** The root activity of the process to be run is not in INITIAL or DORMANT mode.

LINK ACTIVITY

Execute a CICS business transaction services activity synchronously without context-switching.



Description

LINK ACTIVITY executes a CICS business transaction services activity synchronously with the requestor, with no context-switching. The activity must previously have been defined to BTS.

LINK ACTIVITY causes BTS to invoke the activity and send it an input event. If the activity is in its initial state—that is, if this is the first time it is to be run, or if it has been reset by a RESET ACTIVITY command—CICS sends it the DFHINITIAL system event. If the activity is not in its initial state, the input event must be specified on the INPUTEVENT option.

The only activities a program can link to are as follows:

- If it is running as the activation of an activity, its own child activities. It can link to several of its child activities within the same unit of work.
- The activity it has acquired, by means of an ACQUIRE ACTIVITYID command, in the current unit of work. (Note, however, that if the program is running as the activation of an activity, it must use a RUN, not a LINK, command to activate the activity it has acquired.)

To check the response from the activity, the CHECK ACTIVITY command must be used. This is because the response to the request to activate the activity does not contain any information about the success or failure of the activity itself—only about the success or failure of the request to activate it. Typically, the CHECK command is issued immediately after the LINK command.

No context-switching

When an activity is activated by a LINK ACTIVITY command, it is invoked synchronously with the requestor and:

- In the same unit of work as the requestor
- With the transaction attributes (TRANSID and USERID) of the requesting transaction.

In other words, there is no **context-switch**. To invoke an activity synchronously *with* context-switching—that is, in a separate UOW from that of the requesting transaction and with the TRANSID and USERID attributes specified on its DEFINE ACTIVITY command—use the RUN ACTIVITY SYNCHRONOUS command.

Note: A context-switch always occurs when an activity is run asynchronously.

If performance is more important than failure isolation, recoverability, and security, use LINK ACTIVITY rather than RUN ACTIVITY SYNCHRONOUS.

Options

ACQACTIVITY

specifies that the activity to be run is the one that the current unit of work has acquired by means of an ACQUIRE ACTIVITYID command.

ACTIVITY(data-value)

specifies the name (1–16 characters) of the activity to be run. The name must be that of a child of the current activity.

INPUTEVENT(data-value)

specifies the name (1–16 characters) of the event that causes the activity to be attached.

You *must not* specify this option if the activity is in its initial state; that is, if this is the first time it is to be run, or if it has been reset by a RESET ACTIVITY command. In this case, CICS sends the activity the DFHINITIAL system event.

You *must* specify this option if the activity is not in its initial state; that is, if it has been activated before, and has not been reset by a RESET ACTIVITY command.

If you specify INPUTEVENT, for the LINK command to be successful the activity to be attached must have defined the named event as an input event.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19** The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8** The activity named on the ACTIVITY option could not be found.
- 14** The target activity is not in the correct mode to process the specified event option. If the INPUTEVENT option was not specified, the activity must be in INITIAL mode. If the INPUTEVENT option was specified, the activity must be in DORMANT mode.

EVENTERR

RESP2 values:

- 7** The event named on the INPUTEVENT option has not been defined by the activity to be run as an input event; or its fire status is FIRED.

INVREQ

RESP2 values:

- 4** The ACTIVITY option was used to name a child activity, but the command was issued outside the scope of a currently-active activity.
- 21** The activity is suspended, and therefore cannot be run synchronously.
- 24** The ACQACTIVITY option was used, but the issuing task has not acquired an activity.

- 40 The program that implements the activity is remote.
- 44 A LINK has been attempted to a Java program, but the JVM pool is disabled.
- 45 A LINK has been attempted to a Java program, but the JVM profile cannot be found.
- 46 A LINK has been attempted to a Java program, but the JVM profile is not valid.
- 47 A LINK has been attempted to a Java program, but the system properties file cannot be found.
- 48 A LINK has been attempted to a Java program, but the user class cannot be found.
- 49 The shared class cache is STOPPED and autostart is disabled, so a Java program requesting use of the shared class cache cannot be executed.

IOERR

RESP2 values:

- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

NOTAUTH

RESP2 values:

- 101 The user associated with the issuing task is not authorized to run the activity.

PGMIDERR

RESP2 values:

- 1 A program has no entry in the PPT and either program autoinstall was switched off, or the program autoinstall user program indicated that the program should not be autoinstalled.
- 2 A program is disabled.
- 3 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.
- 21 The program autoinstall user program failed either because the program autoinstall user program is incorrect, incorrectly defined, or as a result of an abend in the program autoinstall user program. Program autoinstall is disabled and message DFHPG0202 or DFHPG0203 written to the CSPL.
- 22 The model returned by the program autoinstall user program was not defined in the PPT table, or was not enabled.

- 23 The program autoinstall user program returned invalid data.
- 24 Define for the program failed due to autoinstall returning an invalid program name or definition.

LOAD

Load a program from the CICS DFHRPL concatenation library into main storage.

LOAD

►►—LOAD—PROGRAM(*name*)—┐SET(*ptr-ref*)┐┐LENGTH(*data-area*)┐┐FLENGTH(*data-area*)┐┐ENTRY(*ptr-ref*)┐┐HOLD┐►►

Conditions: INVREQ, LENGERR, NOTAUTH, PGMIDERR

This command is threadsafe.

Note for 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 Application Programming Guide* for more information about transaction affinities.

Description

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 Application Programming Guide* for further details about maps.) Using LOAD can reduce system overhead.

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.

Conditions

INVREQ

RESP2 values:

- 30** 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.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 19** LENGTH is used and the length of the loaded program is not less than 32KB.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

- 101** A resource security check has failed on PROGRAM(name).

Default action: terminate the task abnormally.

PGMIDERR

RESP2 values:

- 1** 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.
- 2** A program is disabled.
- 3** 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.

- 9** The installed program definition is for a remote program.
- 21** 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.
- 22** The model returned by the program autoinstall control program was not defined in the PPT table, or was not enabled.
- 23** The program autoinstall control program returned invalid data.
- 24** Define for the program failed due to autoinstall returning an invalid program name or definition.
- 42** An attempt has been made to LOAD a JVM program. This is invalid because Java byte codes programs are not managed by CICS Loader.

Default action: terminate the task abnormally.

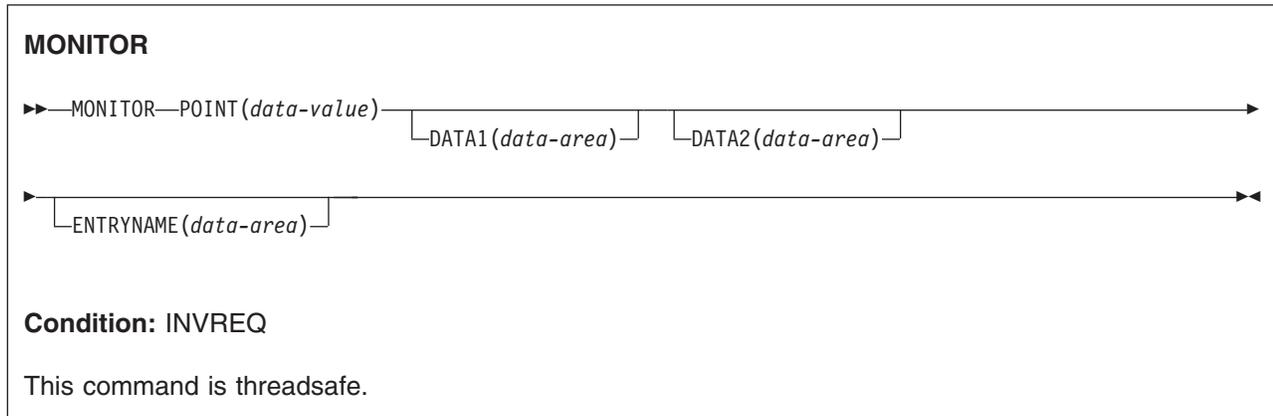
Examples

The following example shows how to load a user-prepared table called TB1:

```
EXEC CICS LOAD PROGRAM('TB1') SET(PTR)
```

MONITOR

Code a user event-monitoring point.



Description

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

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 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. The default value depends on the option that you have defined in the EMP definition. If you specify a null value in DATA2, monitoring uses the default value that is specified in the EMP definition. 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. The default value depends on the option that you have defined in the EMP definition. If you specify a null value in DATA2, monitoring uses the default value that is specified in the EMP definition. If DATA2 is not specified, the MOVE operation raises an INVREQ although the operation was successful.

See 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. Specify in the data-area the name of the 8-byte field in your application program that contains the monitoring point entry name.

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.

Conditions

INVREQ

RESP2 values:

- 1 Your POINT value is outside the range 1 through 255.
- 2 Your POINT value is not defined in the MCT.
- 3 Your DATA1 value is not valid.
- 4 Your DATA2 value is not valid.
- 5 You did not specify DATA1 for an MCT operation that required it.
- 6 You did not specify DATA2 for an MCT operation that required it.

Default action: terminate the task abnormally.

Examples

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 3 on page 362 gives examples of MONITOR commands (and of the MCT entries you need for them). See the *CICS Customization Guide* for more information about monitoring.

Note:

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)

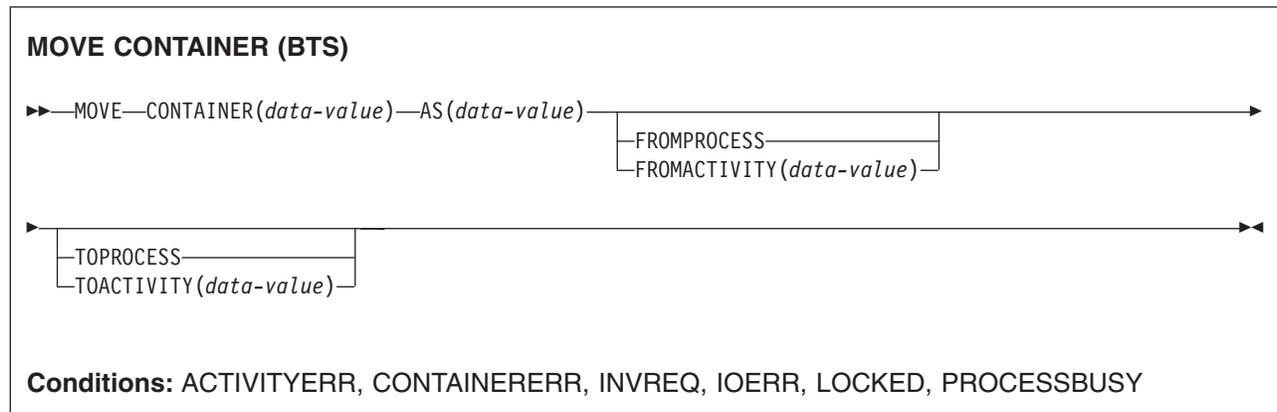
2:
EXEC CICS MONITOR
    POINT(12)
    ENTRYNAME(ENTRY3)
    needing: DFHMCT TYPE=EMP,
             CLASS=PERFORM,
             ID=(ENTRY3.12),
             PERFORM=PCLOCK(1)

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 3. Examples of coding user EMPs

MOVE CONTAINER (BTS)

Move a BTS data-container (and its contents) from one activity to another.



Description

MOVE CONTAINER (BTS) moves a data-container (and its contents) from one BTS activity to another. After the move, the source container is destroyed.

The source and target containers are identified by name and by the activities that own them. The activity that owns the source container can be identified:

- Explicitly, by specifying the FROMPROCESS or FROMACTIVITY option.
- Implicitly, by omitting the FROMPROCESS and FROMACTIVITY options. If these are omitted, the current activity is implied.

Similarly, the activity that owns the target container can be identified:

- Explicitly, by specifying the TOPROCESS or TOACTIVITY option.
- Implicitly, by omitting the TOPROCESS and TOACTIVITY options. If these are omitted, the current activity is implied.

You can move a container:

- From the current activity to a child of the current activity
- From a child of the current activity to the current activity
- From the current activity to the current activity (thus renaming the container)
- From one child of the current activity to another

In addition, *if the current activity is the root activity*, you can move a container:

- From the current process to the current (root) activity
- From the current process to a child of the current activity
- From the current process to the current process (thus renaming the container)
- From the current activity to the current process
- From a child of the current activity to the current process

You can use MOVE CONTAINER, instead of GET CONTAINER and PUT CONTAINER, as a more efficient way of transferring data between activities—for an explanation, see the *CICS Business Transaction Services* manual.

Note:

1. If the source container does not exist, an error occurs.

2. If the target container does not already exist, it is created. If the target container already exists, its previous contents are overwritten.
3. You cannot move containers from one process to another. Both the source and target containers must be within the scope of the current process.
4. Only the root activity can specify a process-container as the source or target of a MOVE CONTAINER command.

A process's containers are *not* the same as its root activity's containers.

See also “GET CONTAINER (BTS)” on page 250 and “PUT CONTAINER (BTS)” on page 377.

Options

AS(*data-value*)

specifies the name (1–16 characters) of the target container. If the target container already exists, its contents are overwritten.

CONTAINER(*data-value*)

specifies the name (1–16 characters) of the source container that is to be moved.

FROMACTIVITY(*data-value*)

specifies the name (1–16 characters) of the activity that owns the source container. If specified, this option must name a child of the current activity (or the current activity itself).

FROMPROCESS

specifies that the source container is owned by the current process—that is, the process that the program that issues the command is executing on behalf of.

TOACTIVITY(*data-value*)

specifies the name (1–16 characters) of the activity that owns the target container. If specified, this option must name a child of the current activity (or the current activity itself).

TOPROCESS

specifies that the target container is owned by the current process—that is, the process that the program that issues the command is executing on behalf of.

Conditions

ACTIVITYERR

RESP2 values:

- 8** The activity named on the FROMACTIVITY or TOACTIVITY option could not be found.

CONTAINERERR

RESP2 values:

- 10** The container named on the CONTAINER option could not be found.
- 26** The process container named on the CONTAINER option is read-only.

INVREQ

RESP2 values:

- 4** The command was issued outside the scope of a currently-active activity.

- 25** The FROMPROCESS or TOPPROCESS option was used, but the command was issued outside the scope of a currently-active process.

IOERR

RESP2 values:

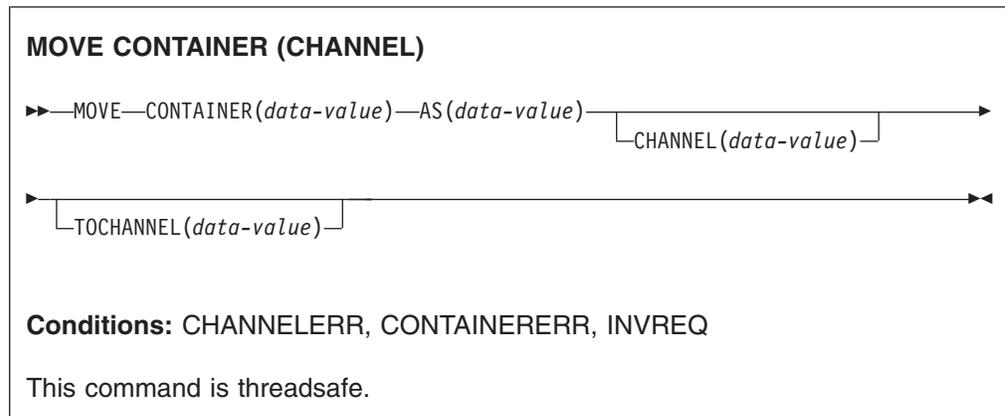
- 30** An input/output error has occurred on the repository file.
- 31** The record on the repository file is in use.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

MOVE CONTAINER (CHANNEL)

Move a container (and its contents) from one channel to another.



Description

MOVE CONTAINER (CHANNEL) moves a container from one channel to another. After the move, the source container no longer exists.

The source and target containers are identified by name and by the channels that own them. The channel that owns the source container can be identified:

- Explicitly, by specifying the CHANNEL option.
- Implicitly, by omitting the CHANNEL option. If this is omitted, the current channel is implied.

Similarly, the channel that owns the target container can be identified:

- Explicitly, by specifying the TOCHANNEL option.
- Implicitly, by omitting the TOCHANNEL option. If this is omitted, the current channel is implied.

You can move a container:

- From one channel to another.
- Within the same channel—for example, from the current channel to the current channel. This has the effect of renaming the container.

You can use MOVE CONTAINER, instead of GET CONTAINER and PUT CONTAINER, as a more efficient way of transferring data between channels.

Note:

1. The source channel must be within the scope of the program that issues the MOVE CONTAINER command.
2. If the target channel does not exist, within the scope of the program that issues the MOVE CONTAINER command, it is created.
3. If the source container does not exist, an error occurs.
4. If the target container does not already exist, it is created. If the target container already exists, its previous contents are overwritten.
5. If you try to overwrite a container with itself, nothing happens. That is, if you specify the same value for the CONTAINER and AS options, and

either omit both the CHANNEL and TOCHANNEL options or give them the same value, so that the same channel is specified, the source container is not changed and not deleted. No error condition is raised.

Options

AS(data-value)

specifies the name (1–16 characters) of the target container. If the target container already exists, its contents are overwritten.

The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

Container names are always in EBCDIC. The allowable set of characters for container names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if containers are to be shipped between regions, the characters used in naming them should be restricted to A-Z a-z 0-9 & : = , ; < > . - and _ .

#

CHANNEL(data-value)

specifies the name (1–16 characters) of the channel that owns the source container. If this option is not specified, the current channel is implied.

CONTAINER(data-value)

specifies the name (1–16 characters) of the source container that is to be moved.

TOCHANNEL(data-value)

specifies the name (1–16 characters) of the channel that owns the target container. If you are specifying a new channel, remember that the acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

channel names are always in EBCDIC. The allowable set of characters for channel names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if channels are to be shipped between regions, the characters used in naming them should be restricted to A-Z a-z 0-9 & : = , ; < > . - and _ .

#

If this option is not specified, the current channel is implied.

Conditions

CHANNELERR

RESP2 values:

- 1 The name specified on the TOCHANNEL option contains an illegal character or combination of characters.
- 2 The channel specified on the CHANNEL option could not be found.

CONTAINERERR

RESP2 values:

- 10 The container named on the CONTAINER option could not be found.
- 18 The name specified on the AS option contains an illegal character or combination of characters.

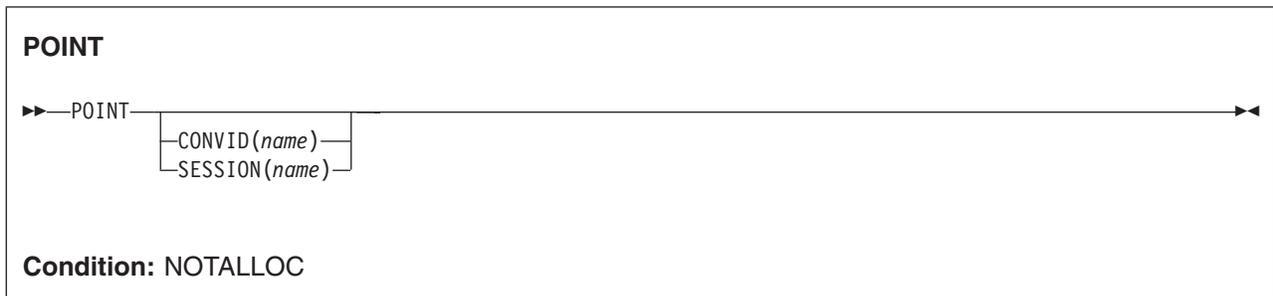
INVREQ

RESP2 values:

- 4** The CHANNEL or TOCHANNEL option (or both) was not specified, there is no current channel (because the program that issued the command was not passed one), and the command was issued outside the scope of a currently-active BTS activity.
- 30** You cannot move a CICS-defined readonly container.
- 31** You cannot move a container to (that is, overwrite) an existing, CICS-defined, readonly container.

POINT

Get information about an LUTYPE6.1 logical unit.



Description

POINT gets information about a named facility, such as whether it owns the given facility.

This command can be used on an MRO session.

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.

Conditions

NOTALLOC

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

Default action: terminate the task abnormally.

POP HANDLE

Restore the stack.

POP HANDLE

►—POP HANDLE—◄

Condition: INVREQ

This command is threadsafe.

Description

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 effects are NOT inherited by the linked-to program, but CICS will search preceding logical levels for a HANDLE ABEND exit. See *CICS 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.

Conditions

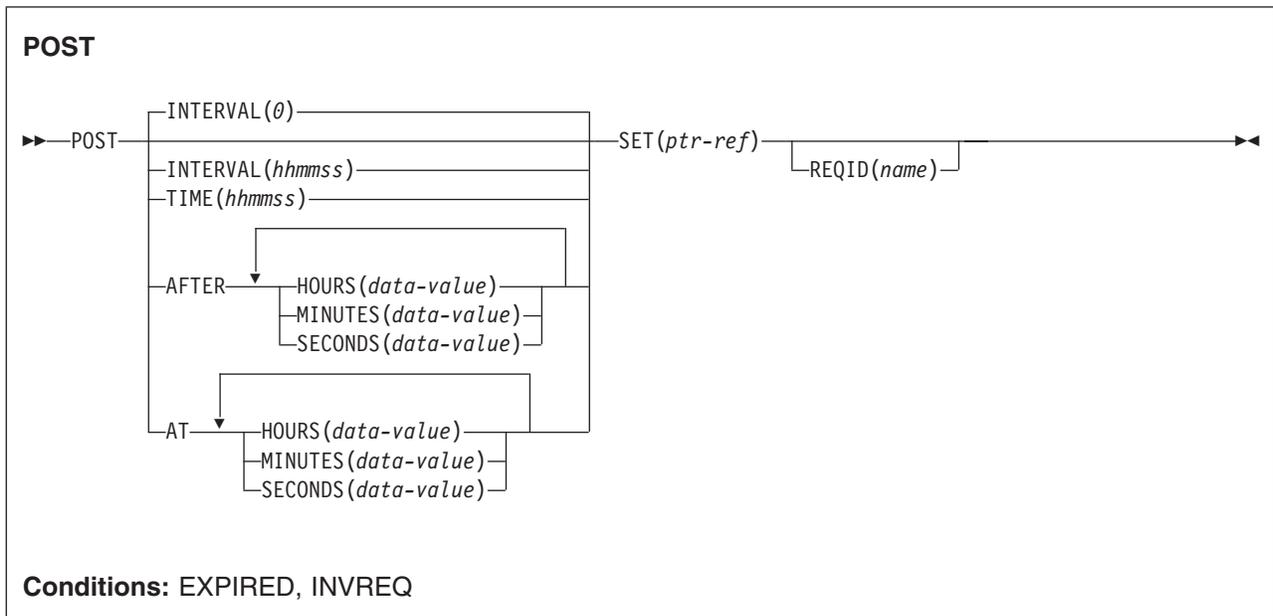
INVREQ

occurs if no matching PUSH HANDLE command has been executed at the current link level.

Default action: terminate the task abnormally.

POST

Request notification when a specified time has expired.



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 Application Programming Guide* for more information about transaction affinities.

Description

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. 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 or POST command.
- The task issues a subsequent START command naming a transaction in the local system. (A START command that names a transaction on a remote system does not affect the event set up by the POST command, unless the transaction is defined with LOCALQ set to YES and local queuing is performed.)
- 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 or POST command, or a START command naming a transaction in the local system, 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).

Options

AFTER

specifies the interval of time to elapse.

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(0–359 999). 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 expiring. For the ways to enter the time, see the AFTER option.

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.

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. This is a suboption of the AFTER and AT options. For its use and meaning, see the AFTER option.

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 0–359 999 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.

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 Application Programming Guide*.

Conditions

EXPIRED

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

Default action: ignore the condition.

INVREQ

RESP2 values:

- 4 Hours are out of range.
- 5 Minutes are out of range.
- 6 Seconds are out of range.

also occurs (RESP2 not set) in any of the following situations:

- The POST command is not valid for processing by CICS.

Default action: terminate the task abnormally.

Examples

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 ask to be notified 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)
```

PURGE MESSAGE

Discontinue building a BMS logical message.

PURGE MESSAGE

▶—PURGE MESSAGE—◀

Conditions: Full BMS: INVREQ, TSIOERR

Description

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 “BMS macros” on page 784 for map definition macros.

PURGE MESSAGE is only available on full-function BMS. For further information about BMS, see the *CICS Application Programming Guide*.

Conditions

INVREQ

RESP2 values:

200 The command was called in a distributed program link server program.

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

Suspend the stack.

PUSH HANDLE



This command is threadsafe.

Description

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.

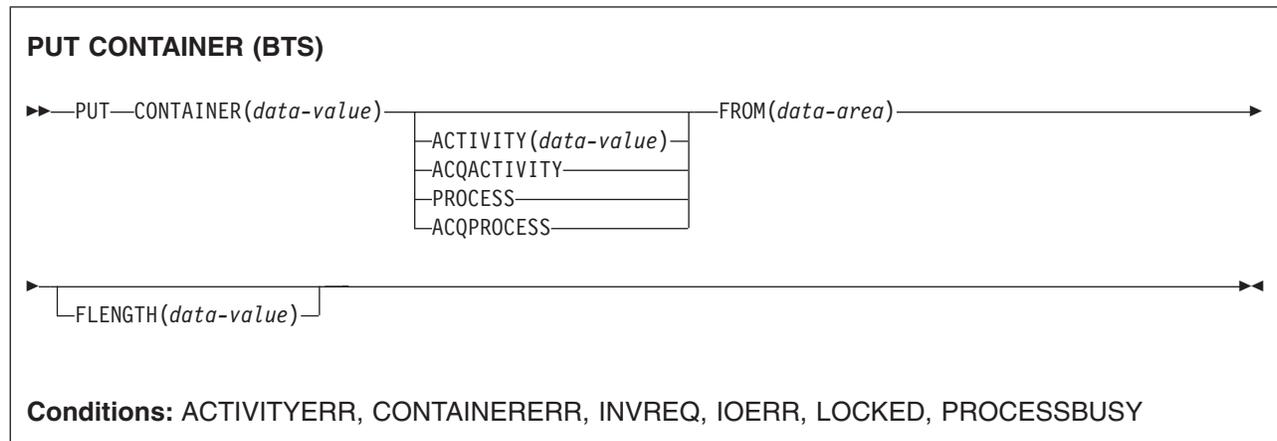
Note: When a CICS program uses EXEC CICS LINK to call another CICS program, the HANDLE CONDITION options are NOT inherited by the linked-to program, but CICS will search preceding logical levels for a HANDLE ABEND exit. See *CICS Application Programming Guide* for further details about the relationship 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.

PUT CONTAINER (BTS)

Save data in a named BTS data-container.



Description

PUT CONTAINER (BTS) saves data and places it in a container associated with a specified BTS activity or process.

The container is identified by name. The process or activity that owns the container can be identified:

- Explicitly, by specifying one of the PROCESS- or ACTIVITY-related options.
- Implicitly, by omitting the PROCESS- and ACTIVITY-related options. If these are omitted, the current activity is implied.

Note:

1. There is no limit to the number of containers that can be associated with an activity.
2. Different activities can own identically-named containers—these are different containers.
3. If the named container does not already exist, it is created. If the named container already exists, its previous contents are overwritten.
4. Containers owned by a process (*process-containers*) can be read by every activity in the process. However, they can be updated only by the root activity, or by a program that has acquired the process.

A process's containers are *not* the same as its root activity's containers.

See also “GET CONTAINER (BTS)” on page 250 and “MOVE CONTAINER (BTS)” on page 363.

Options

ACQACTIVITY

specifies either of the following:

- If the program that issues the command has acquired a process, that the container is owned by the root activity of that process.
- Otherwise, that the container is owned by the activity that the program has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the container is owned by the process that the program that issues the command has acquired in the current unit of work.

ACTIVITY(data-value)

specifies the name (1–16 characters) of the activity that owns the container. This must be a child of the current activity.

CONTAINER(data-value)

specifies the name (1–16 characters) of the container into which data is to be placed.

The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

FLENGTH(data-value)

specifies, as a fullword binary value, the length of the data area from which data is to be read.

FROM(data-area)

specifies an area of working storage from which the data to be saved is to be read.

PROCESS

specifies that the container into which data is to be placed is owned by the current process—that is, the process that the program that issues the command is executing on behalf of.

Conditions**ACTIVITYERR**

RESP2 values:

- 8** The activity named on the ACTIVITY option could not be found.

CONTAINERERR

RESP2 values:

- 10** The container named on the CONTAINER option could not be found.
- 18** The name specified on the CONTAINER option contains an illegal character or combination of characters.
- 26** The process container named on the CONTAINER option is read-only.

INVREQ

RESP2 values:

- 1** The DATATYPE option was specified without the CHANNEL option, and there is no current channel (because the program that issued the command was not passed one.) DATATYPE is valid only on PUT CONTAINER commands that specify (explicitly or implicitly) a channel. It is not valid on PUT CONTAINER (BTS) commands.
- 2** The FROMCCSID option was specified without the CHANNEL option, and there is no current channel (because the program that issued the command was not passed one.) FROMCCSID is valid only on PUT CONTAINER commands that specify (explicitly or implicitly) a channel. It is not valid on PUT CONTAINER (BTS) commands.
- 4** The command was issued outside the scope of a currently-active activity.

- 15 The ACQPROCESS option was used, but the unit of work that issued the request has not acquired a process.
- 24 The ACQACTIVITY option was used, but the unit of work that issued the request has not acquired an activity.
- 25 The PROCESS option was used, but the command was issued outside the scope of a currently-active process.

IOERR

RESP2 values:

- 30 An input/output error has occurred on the repository file.
- 31 The record on the repository file is in use.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

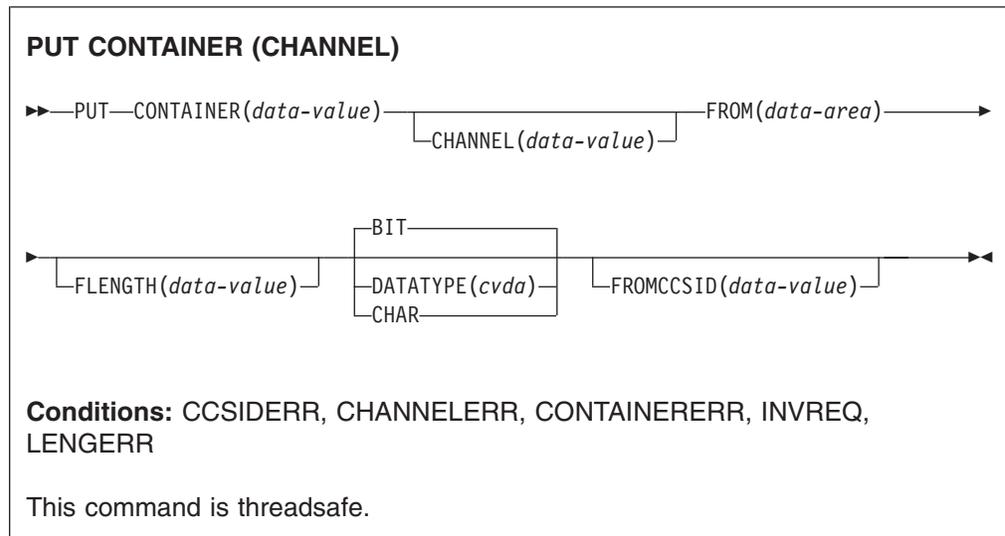
PROCESSBUSY

RESP2 values:

- 13 The request could not be satisfied because the process record is locked by another task.

PUT CONTAINER (CHANNEL)

Place data in a named channel container.



Description

PUT CONTAINER (CHANNEL) places data in a container associated with a specified channel.

The container is identified by name. The channel that owns the container can be identified:

- Explicitly, by specifying the CHANNEL option.
- Implicitly, by omitting the CHANNEL option. If this is omitted, the current channel is implied.

Note:

1. There is no limit to the number of containers that can be associated with a channel.
2. The size of individual containers is limited only by the amount of storage available.

CAUTION:

Take care not to create so many large containers that you limit the amount of storage available to other applications.

3. If the named container does not already exist, it is created. If the named container already exists, its previous contents are overwritten.
4. If the named channel does not already exist, it is created.

Options

CHANNEL(data-value)

specifies the name (1–16 characters) of the channel that owns the container. The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

#

Channel names are always in EBCDIC. The allowable set of characters for channel names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if channels are to be shipped between regions, the characters used in naming them should be restricted to A-Z a-z 0-9 & : = , ; < > . - and _.

CONTAINER(data-value)

specifies the name (1–16 characters) of the container into which data is to be placed.

The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

Do not use container names beginning with “DFH”, unless requested to do so by CICS.

Container names are always in EBCDIC. The allowable set of characters for container names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if containers are to be shipped between regions, the characters used in naming them should be restricted to A-Z 0-9 & : = , ; < > . - and _.

DATATYPE(cvda)

specifies the type of data to be put into the container. This option applies only to *new* containers. If the container already exists its data-type was established when it was created and cannot be changed. CVDA values are:

BIT Bit data. The data in the container cannot be converted. This is the default value, unless FROMCCSID is specified.

CHAR Character data. The data in the container is converted (if necessary) to the code page of the application that *created* the channel. If the channel was created by a client application on an ASCII-based system, this will be an ASCII code page. If it was created by a CICS Transaction Server for z/OS application, it will be an EBCDIC code page. Conversion is only necessary if the client and server programs run on different platforms.

All the data in a container is converted as if it were a single character string. For SBCS code pages, a structure consisting of several character fields is equivalent to a single-byte character string. However, for DBCS code pages this is not the case. If you use DBCS code pages, to ensure that data conversion works correctly you must put each character string into a separate container.

#

For CHAR containers, the data is stored in the Coded Character Set Identifier (CCSID) specified on the original PUT CONTAINER command that created the container. If the FROMCCSID option was not specified on the original PUT CONTAINER command, the data is stored in the region's default CCSID (or, for CICS-created channels, in the CCSID of the channel). The data on all future PUT CONTAINER CHANNEL commands for this container is converted into this same CCSID. If you want to avoid this, the application program should delete the existing container before issuing the new PUT CONTAINER command, thus recreating the container.

A DATATYPE of CHAR must be specified if the container contains character data *and* the channel will be passed from CICS Transaction Server for z/OS to

an ASCII system. If the container contains binary data, or the channel will not be passed to an ASCII system, DATATYPE is an optional parameter.

It is not possible to change the data-type of an existing container by means of a PUT CONTAINER command. For example, if a container is created with a data-type of BIT and a subsequent PUT CONTAINER command specifies a data-type of CHAR, for the same container, an INVREQ condition is raised. If you do need to replace an existing container by one of a different data-type, you must first explicitly delete the existing container.

For more information about data conversion with channels, see the *CICS Application Programming Guide*.

FLENGTH(data-value)

specifies, as a fullword binary value, the length of the data area from which data is to be read.

FROM(data-area)

specifies the data area from which the data is written to the container.

FROMCCSID(data-value)

specifies, as a fullword decimal number, the current Coded Character Set Identifier (CCSID) of the character data to be put into the container. For CICS Transaction Server for z/OS applications, this is typically an EBCDIC CCSID. (However, it is possible to specify an ASCII CCSID, if you want to pass ASCII data.)

FROMCCSID is effective only on the PUT CONTAINER command that creates
the container. This is because, for CHAR containers, the data is stored in the
CCSID specified on the original PUT CONTAINER command that created the
container. If you want to use a different CCSID, the application program should
delete the existing container before issuing the new PUT CONTAINER
command, thus recreating the container.

If FROMCCSID is specified, DATATYPE(DFHVALUE(CHAR)) is implied.

If FROMCCSID is not specified, its value defaults to the CCSID of the region
(or, for CICS-created channels, the CCSID of the channel). The default CCSID
of the region is specified on the LOCALCCSID system initialization parameter.

For an explanation of CCSIDs, and a list of the CCSIDs supported by CICS, see the *CICS Family: Communicating from CICS on System/390*[®] manual.

Conditions

CCSIDERR

RESP2 values:

- 1 The CCSID specified on the FROMCCSID option is outside the range of valid CCSID values.
- 2 The CCSID specified on the FROMCCSID option and the CCSID of the container are an unsupported combination. The CCSID of the container is the value that was specified, or defaulted, on the first PUT CONTAINER command for this container. The first time each invalid combination is used, CICS issues error message DFHAP0802, which contains the pair of CCSIDs.

4 One or more characters could not be converted. Each unconverted
character has been replaced by a blank in the converted data. This
error can occur only when the target of the PUT is an existing
container.

- 5 There was an internal error in the code page conversion of a container. This error can occur only when the target of the PUT is an existing, CICS-created, container.

CHANNELERR

RESP2 values:

- 1 The name specified on the CHANNEL option contains an illegal character or combination of characters.

CONTAINERERR

RESP2 values:

- 18 The name specified on the CONTAINER option contains an illegal character or combination of characters.

INVREQ

RESP2 values:

- 1 The DATATYPE option was specified without the CHANNEL option, and there is no current channel (because the program that issued the command was not passed one.) DATATYPE is valid only on PUT CONTAINER commands that specify (explicitly or implicitly) a channel.
- 2 The FROMCCSID option was specified without the CHANNEL option, and there is no current channel (because the program that issued the command was not passed one.) FROMCCSID is valid only on PUT CONTAINER commands that specify (explicitly or implicitly) a channel.
- 4 The CHANNEL option was not specified, there is no current channel (because the program that issued the command was not passed one), and the command was issued>outside the scope of a currently-active BTS activity.
- 30 You tried to write to a CICS-defined read only container.
- 32 A CVDA value other than CHAR or BIT was specified for DATATYPE.
- 33 An attempt was made to change the data-type of an existing container.

LENGERR

RESP2 values:

- 1 A negative number was specified on the FLENGTH option.

QUERY COUNTER and QUERY DCOUNTER

Query a named counter.

QUERY COUNTER

►► QUERY COUNTER(*name*) ————
 └─POOL(*name*)─┘ └─VALUE(*data-area*)─┘ └─MINIMUM(*data-area*)─┘
 └─MAXIMUM(*data-area*)─┘

Conditions: INVREQ, LENGERR

QUERY DCOUNTER

►► QUERY DCOUNTER(*name*) ————
 └─POOL(*name*)─┘ └─VALUE(*data-area*)─┘ └─MINIMUM(*data-area*)─┘
 └─MAXIMUM(*data-area*)─┘

Conditions: INVREQ

Description

These counter commands return the current, maximum, and minimum values for the named counter. COUNTER operates on fullword named counters and DCOUNTER operates on doubleword named counters.

For information about specifying fullword and doubleword variables on these named counter commands, see “CICS command argument values” on page 3.

Options

COUNTER(*name*)

specifies the 16-character name of the fullword counter being queried. Valid characters for names are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

DCOUNTER(*name*)

specifies the 16-character name of the doubleword counter being queried. Valid characters for names are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

MAXIMUM(*data-area*)

specifies the data area in which CICS is to return the maximum number for the

named counter. CICS returns a fullword signed binary value for the COUNTER command and a doubleword unsigned binary value for the DCOUNTER command

MINIMUM(*data-area*)

specifies the data area in which CICS is to return the minimum number for the named counter. CICS returns a fullword signed binary value for the COUNTER command and a doubleword unsigned binary value for the DCOUNTER command

POOL(*poolname*)

specifies the name of the pool in which the named counter resides.

Valid characters for the pool selector string are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 8 characters, the name must be padded with trailing blanks.

This parameter is optional. If you omit the name of the pool, a pool selector value of 8 blanks is assumed.

If there is no matching entry in the DFHNCOPT options table, CICS uses the name specified on the NCPLDFT system initialization parameter, which specifies the default named counter pool.

For information about generating a named counter options table using the DFHNCO macro, see the *CICS Application Programming Guide*.

VALUE(*data-area*)

specifies the data area in which CICS is to return the current value for the named counter. CICS returns a fullword signed binary value for the COUNTER command and a doubleword unsigned binary value for the DCOUNTER command.

Note that, if the named counter is in the counter-at-limit condition, CICS does not return an exception condition. In this case, CICS returns a normal response with a value that is 1 greater than the maximum value specified or assumed for the counter, using unsigned addition. If the maximum value is the largest positive number that can be held in a signed fullword, the value returned by QUERY COUNTER for a counter-at-limit condition is the largest negative number.

Conditions

INVREQ

RESP2 values:

- 201** Named counter not found.
- 301** The server has reported an error code that is not understood by the named counter interface. Generally, this is not possible unless the interface load module, DFHNCIF, is at a lower maintenance or release level than the server itself.
- 303** An unexpected error, such as structure failure or loss of connectivity, has occurred on a macro used to access the coupling facility. Further information can be found in message DFHNC0441 in the CICS job log.
- 304** The pool selection parameter specified in the program cannot be resolved to a valid server name using the current options table.
- 305** The interface is unable to establish a connection to the server for the selected named counter pool. Further information can be found in an AXM services message (AXMSC $nnnn$) in the CICS job log.

- 306 An abend occurred during server processing of a request. Further information can be found in a message in the CICS job log and the server job log.
- 308 The DFHNCOPT options table module, required for resolving a pool name, could not be loaded.
- 309 During processing of the options table, the named counter interface encountered an unknown entry format. Either the options table is not correctly generated, or the DFHNCIF interface load module is not at the same release level as the options table.
- 310 An options table entry matching the given pool name specified a user exit program, but the user exit program is not link-edited with the options table and cannot be loaded.
- 311 A response from the named counter server to the client region interface module, DFHNCIF, indicates that a system-managed rebuild is in progress but the EXEC CICS interface does not recognize the condition. This means that the CICS region is at CICS TS 2.1 or earlier.
- 403 The POOL parameter contains invalid characters or embedded spaces.
- 404 The COUNTER parameter contains invalid characters or embedded spaces.

Default action: terminate the task abnormally.

LENGERR

LENGERR occurs for COUNTER commands only and does not apply to DCOUNTER requests. It occurs when a counter that was defined by a DCOUNTER command or by the CALL interface has a value which is too large to be correctly represented as a fullword signed binary value (that is, the counter uses more than 31 bits).

In each of the three cases of overflow, the named counter server completes the operation, and returns a warning response to CICS, which CICS returns to your application program as the RESP2 value. The data area contains the low-order 32 bits returned from the named counter server, which could be a negative number.

RESP2 values:

- 001 The current value that the server has attempted to return in one of the data areas has overflowed into the high-order (sign) bit (that is, the value returned is negative).

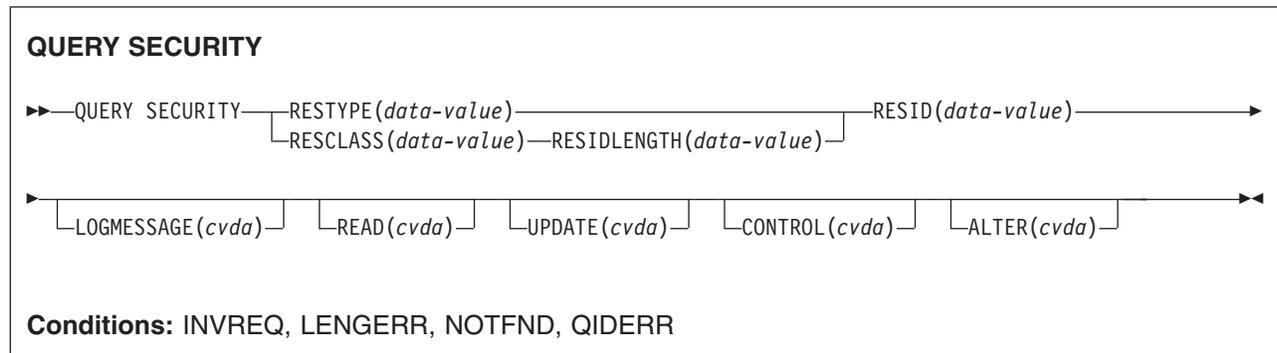
Note: LENGERR with RESP2=001 cannot occur for a named counter that is in the counter-at-limit condition. If the counter-at-limit condition has been reached, the value (which could be negative) is returned with a normal response.

- 002 A value is too large for a fullword data area by only 1 bit. In this case, the overflow value is exactly 1.
- 003 A value is too large for a fullword data area by a value greater than 1.

Default action: terminate the task abnormally.

QUERY SECURITY

To query the security authorization of the user.



Description

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 *CICS RACF Security Guide*.

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 command 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, unless you are using the COBOL3 translator option in which case the maximum length is 160 characters).

Note: RESID refers to a CICS-defined resource only when RESTYPE('SPCOMMAND') is specified, otherwise it refers to a user-defined resource. For a list of the CICS RESID values that you can use when RESTYPE('SPCOMMAND') is specified, see *CICS RACF Security Guide*.

Note that the actual resource checked depends on whether RESCLASS or RESTYPE is specified in the command and whether prefixing is active (SECPRFX=YES or SECPRFX=*prefix* specified as a system initialization parameter).

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. Otherwise the resource checked is the RESID data-value prefixed with either the CICS region userid (if SECPRFX=YES), or another prefix (if SECPRFX=*prefix*).

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:

DB2ENTRY

FILE
JOURNALNAME
JOURNALNUM²
PROGRAM
PSB
SPCOMMAND¹
TDQUEUE
TRANSACTION
TRANSATTACH
TSQUEUE [8 byte TS queue names]
TSQNAME [16 byte TS queue names]
¹ See *CICS RACF Security Guide*.

² Supported for compatibility with earlier releases.

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.

Conditions**INVREQ**

RESP2 values:

- 7** The *cvda* value is not valid for the LOGMESSAGE.
- 9** The RESID is invalid or filled with blanks.
- 10** The external security manager (ESM) is inactive or not present.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 6** The RESIDLENGTH value is not valid, that is, not in the range 1 through 246.

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

- 1** The RESID is not valid.
- 2** The RESTYPE is not valid.
- 3** The RESID value for RESTYPE (SPCOMMAND) not valid.
- 5** The RESCLASS is not defined to the external security manager (ESM).

8 The resource is not protected. 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

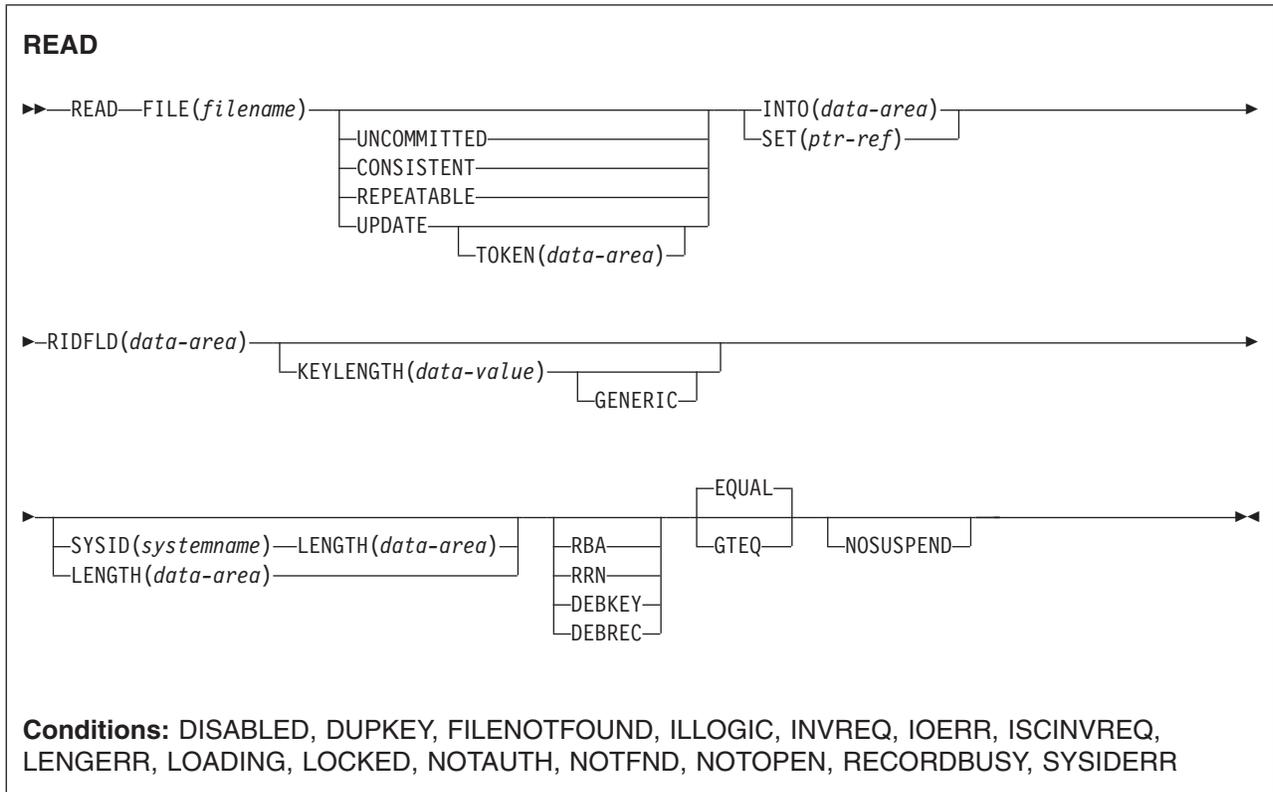
RESP2 values:

1 An indirect queue name associated with the given RESID is not found.

Default action: terminate the task abnormally.

READ

Read a record from a file.



Description

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.

Note: You cannot specify an RBA for VSAM data sets greater than 4GB (that is, defined with the extended format / extended addressability attribute). The only VSAM data sets greater than 4GB supported by CICS are KSDS, and then only if they are accessed by key. CICS does not support ESDS or RRDS data sets defined with the extended attribute.

Data table considerations

When the READ command reads a CICS-maintained data table, a READ request with UPDATE or RBA is always satisfied by a call to VSAM. A full key read that is neither a generic read nor a READ UPDATE, is satisfied by a reference to the data table if possible. 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).

If you carry out a generic read (using the GENERIC option) on a CICS-maintained
data table, and CICS returns a NOTFND condition because the record is not found
in the table, CICS clears the INTO() and RIDFLD() areas to ensure that an incorrect
record is not returned. This behavior optimizes performance, but it differs from the
behavior for a generic read of a VSAM file, when the INTO() and RIDFLD() areas
are left unchanged in the event of a NOTFND condition. When you convert a VSAM
file to a CICS-maintained data table, ensure that any applications that carry out
generic reads of the data take appropriate action if a NOTFND condition is returned
and the INTO() and RIDFLD() areas are cleared.

When the READ 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.

When the READ command reads a coupling-facility data table, only the data table is accessed, even if the table is initially loaded from a VSAM source data set.

If a file that refers to a user-maintained or coupling facility data table is defined with RLSACCESS(YES), the RLS-specific API options CONSISTENT, NOSUSPEND, and REPEATABLE are not supported.

Reading files accessed in RLS mode

When a file is accessed in RLS mode, non-update read requests can specify one of the read integrity options: UNCOMMITTED, CONSISTENT, or REPEATABLE.

If none of these keywords is specified, CICS uses the value specified on the READINTEG parameter of the FILE resource definition, for which the default is UNCOMMITTED.

If you want to use the level of read integrity specified in the READINTEG keyword of the FILE definition, and then you need to change from using a local file to a remote file, or if you change the location of a remote file, ensure that:

- The remote file-owning region supports the read integrity options.
- The FILE definition in the remote system specifies:
 - RLS mode
 - The correct read integrity values for your application.

READ requests that specify the UPDATE keyword, or a CONSISTENT or REPEATABLE read integrity option, (either explicitly or implicitly in the FILE definition) return the LOCKED condition if they reference a record that has a retained lock. The key of a locked record is not returned to the application program. Thus, if an application program specifies GTEQ or GENERIC on the READ request it cannot tell which record key is locked.

If a request specifying read integrity is function-shipped to a member of the CICS family of products that does not support read integrity, the request fails:

- If an ISC link is used, the request receives an ATNI abend.
- If an MRO link is used, the request receives an AXF8 abend.

The AXF8 abend code indicates that your program has attempted to function-ship a request specifying file control options to a remote CICS region that does not support these options.

Retained and active locks

RECORDBUSY refers to active locks and LOCKED refers to retained locks:

These locks affect READ requests which acquire locks; that is, update requests and requests with read integrity. These are the kinds of READ requests which are referred to in the following bullets. Other READ requests are unaffected by the presence of retained or active locks.

- READ requests for records that have *retained* locks are always rejected with a LOCKED response.
- READ requests for records that have *active* locks wait for the lock to be released, except when the NOSUSPEND keyword is specified, in which case CICS returns the RECORDBUSY response.

Options

CONSISTENT (RLS on1y)

The record is read with a level of read integrity provided by a VSAM shared lock that lasts for the duration of the request.

If the record is being modified by another task, which therefore holds an exclusive lock, the READ request waits until the update is complete (unless NOSUSPEND is also specified) as follows:

- For a READ request against a non-recoverable file, the READ completes as soon as any VSAM request performing the update completes.
- For a READ request against a recoverable file, the READ request completes when the updating task completes its next syncpoint or rollback.

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

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, 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.

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.

INTO(*data-area*)

specifies the data area into which the record retrieved from the data set is to be written.

When INTO is 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 C.

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.

For remote files, the KEYLENGTH can be specified in the FILE definition. If KEYLENGTH is not defined there, and is not specified in the application program, and the key is longer than 4 characters, the default value is 4.

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.

This option must be specified if SYSID is specified.

If the file is on a remote system, the LENGTH parameter need not be set here but must be set in the file resource definition

If the file is on a local system, the LENGTH parameter must be set for variable-length records, using the INTO option, but not for fixed-length records. It is, however, advisable to specify the length for fixed-length records 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 COBOL, 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.

If you specify the SET option, you do not need to specify the LENGTH option.

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 before truncation.

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.

NOSUSPEND (RLS on1y)

The request does not wait if the record is locked by VSAM with an active lock, including records locked as the result of a DEADLOCK.

RBA

(VSAM KSDS or ESDS base data sets, or CICS-maintained data tables only, but not paths) specifies that the record identification field specified in the RIDFLD option contains a relative byte address. This option should be used only 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
- Coupling facility data tables
- KSDS and ESDS files that hold more than 4GB
- Any KSDS file opened in RLS access mode

REPEATABLE (RLS on1y)

The record is read with a level of read integrity provided by a VSAM shared lock that lasts for the duration of the unit of work in which the read request is issued.

If the record is being modified by another task, which therefore holds an exclusive lock, the READ request waits until the update is complete (unless NOSUSPEND is also specified) as follows:

- For a recoverable file, the READ request completes when the updating transaction completes its next syncpoint or rollback.
- For a non-recoverable file, the READ completes as soon as the VSAM request performing the update completes.

After the READ request has completed, the record remains locked to the task that issued the READ. Other tasks may continue to read the record but no other task is allowed to update the record until the task that issued the READ performs its next syncpoint or rollback.

RIDFLD(*data-area*)

specifies the record identification 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.

See 'Identifying BDAM records' and 'VSAM record identification' in the *CICS Application Programming Guide* for more information about defining the record identification field.

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.

If the DUPKEY condition occurs in assembler language the specified register has not been set. The specified register 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.

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-area*)

specifies, as a fullword binary value, a unique identifier for this READ UPDATE request. This is an output value returned by file control to the requesting task, for use in associating a subsequent REWRITE or DELETE (or UNLOCK) request with the record returned on this READ UPDATE request.

TOKEN can be function shipped. However, if a request specifying TOKEN is function shipped to a member of the CICS family of products that does not recognize this keyword, the request fails.

Note: If you specify TOKEN it implies update.

UNCOMMITTED

The record is read without read integrity.

The current value of the record, as known to VSAM, is returned. No attempt is made to serialize this read request with any concurrent update activity for the same record. The record may be in the process of being updated by another task, and the record data may change later if that update is subsequently backed out.

UPDATE

specifies that the record is to be obtained for updating or (for VSAM and data tables) deletion. If this option is omitted, a read-only operation is assumed.

UPDATE guarantees read integrity. The mechanism used to ensure data integrity depends on the type of file resource:

- For a VSAM file accessed in RLS mode, the record to be updated is locked by the SMSVSAM server.
- For a VSAM file accessed in non-RLS mode, the record to be updated is locked by CICS and, in addition, the control interval containing the record is held in exclusive control by VSAM.
- For a VSAM file accessed in non-RLS mode, and log(UNDO), CICS holds a record lock until the task syncpoints.
- For a BDAM file, the record to be updated is held in exclusive control by BDAM.
- For a user-maintained data table, the record to be updated is locked by CICS.
- For a CICS-maintained data table, the record to be updated is locked by CICS and, in addition, the control interval containing the record is held in exclusive control by VSAM. The VSAM control interval lock is required because changes to the data table are reflected in the source data set, which is accessed in non-RLS mode.
- For a coupling facility data table using the locking model, the record to be updated is locked by the coupling facility data table server.
- For a coupling facility data table using the contention model, records are not locked, enabling the records to be read for update by more than one task. If a record read for update by one task is then changed by another, the first task is notified, when it issues a REWRITE or DELETE command, by the CHANGED exception condition. If a record read for update by one task is then deleted by another, the first task is notified, when it issues a REWRITE or DELETE command, by the NOTFND condition.

If another task has issued a READ REPEATABLE request against the same record, your READ UPDATE is made to wait until that task reaches SYNCPOINT (unless you issued NOSUSPEND).

Conditions

DISABLED

RESP2 values:

50 A file is disabled because it was initially defined as disabled and has not since been enabled.

A file is disabled because it has been disabled by a SET FILE or a CEMT SET FILE command.

Default action: terminate the task abnormally.

DUPKEY

RESP2 values: (VSAM)

140 A record is retrieved by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key follows

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

RESP2 values:

- 1** The file name supplied in the FILE option cannot be found in the FCT.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values: (VSAM)

- 110** A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the description of the EXEC interface block, "EXEC interface block" on page 745.)

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

RESP2 values:

- 20** READ is not allowed according to the file entry specification in the FCT.
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.
- 25** 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.
- 26** 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.
- 28** Following a READ UPDATE command without TOKEN, another READ UPDATE without TOKEN was issued against the same file without an intervening REWRITE, DELETE without RIDFLD specified, UNLOCK, or SYNCPOINT command. This condition may in some cases be raised despite the fact that the first READ UPDATE was not successful, for example because it timed out.
- 40** A BDAM key conversion error occurred.
- 42** The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than zero.
- 44** The command does not conform to the format of READ for a user-maintained or coupling facility data table; for example, if RBA is specified.
- 51** A READ to a KSDS file that is being accessed in RLS mode specifies the RBA keyword. RLS mode does not support relative byte address access to KSDS data sets.
- 52** CONSISTENT is specified on a READ request to a non-RLS mode file, or to a data table that is specified with RLSACCESS(YES). CONSISTENT is not allowed if the file refers to a data table, even if the file definition specifies RLSACCESS(YES).

- 53 REPEATABLE is specified on a READ request to a non-RLS mode file or to a data table that is specified with RLSACCESS(YES). REPEATABLE is not allowed if the file refers to a data table, even if the file definition specifies RLSACCESS(YES).
- 55 NOSUSPEND is specified on a READ request to a non-RLS mode file or to a data table that is specified with RLSACCESS(YES). NOSUSPEND is not allowed if the file refers to a data table, even if the file definition specifies RLSACCESS(YES).
- 56 An attempt to update a recoverable coupling facility data table has failed because the current unit of work has already updated 1024 recoverable coupling facility data tables. You cannot update more than 1024 recoverable coupling facility data tables within a unit of work

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 120 There is an I/O error during the READ operation. 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.

For a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

(Further information is available in the EXEC interface block, "EXEC interface block" on page 745.)

Default action: terminate the task abnormally.

ISCINVREQ

RESP2 values:

- 70 The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 10 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.

- 11 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.

- 13 An incorrect length is specified for a file with fixed-length records.

Default action: terminate the task abnormally.

LOADING

RESP2 values:

104 The request cannot be satisfied because it is issued against a data table that is still being loaded. The condition can be raised for one of the following reasons:

- The READ specifies a record that has not yet been loaded into a coupling facility data table. Records can be read or modified while a CFDT is loading only if the requested key is within the range of those records already loaded.

The LOADING response can also be returned for a coupling facility data table that has failed during loading. For more information about what happens if the load for a coupling facility data table fails, see the description of the XDTLC global user exit in the *CICS Customization Guide*.

- The READ specifies the UPDATE option for a user-maintained data table. A user-maintained data table cannot be modified during loading.
- The READ specifies the GENERIC or GTEQ options for a user-maintained data table. While a UMT is being loaded, you can use read requests with precise keys only.

If your application programs encounter the LOADING condition persistently or too frequently, check that this is not caused by conflicting file definitions that reference the same data set.

Default action: terminate the task abnormally.

LOCKED

RESP2 values:

106 An attempt is being made to read a record either specifying the UPDATE keyword, or specifying (explicitly or implicitly) CONSISTENT or REPEATABLE, but the record is locked by a retained lock (see “Retained and active locks” on page 393).

The LOCKED condition can also occur for a READ UPDATE request to a recoverable CFDT that uses the locking model, if the record being read is locked by a retained lock. See the *CICS Recovery and Restart Guide* for information about investigating retained locks on records in a coupling facility data table.

Default action: abend the task with code AEX8.

NOTAUTH

RESP2 values:

101 A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

80 An attempt to retrieve a record based on the search argument provided is unsuccessful. For 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. This does not mean that there is no such record in the source data set (if the table was created from one); 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 data table. For remote files, this condition occurs if an attempt

to read a record is made without keylength specified either in the application or the file definition, and the actual key is longer than 4 characters.

Default action: terminate the task abnormally.

NOTOPEN

RESP2 values:

- 60** NOTOPEN (RESP2 60) is returned for one of the following reasons:
- 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. You can also make CLOSED, UNENABLED the initial state, by specifying STATUS(UNENABLED) and OPENTIME(FIRSTREF) on the FILE resource definition. (For BDAM files, you use the FILSTAT parameter of DFHFCT TYPE=FILE.)
 - The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.
 - A READ command is issued against a data set that is quiesced, or is being quiesced, as a result of a SET DSNAME QUIESCED or IMMQUIESCED command.
 - The requested file is CLOSED and ENABLED, so CICS has tried to open the file as part of executing the request. This file open has failed for some reason. You should examine the console for messages that explain why the file open has been unsuccessful.

This condition does not occur if the request is made to a CLOSED, DISABLED file. In this case, the DISABLED condition occurs.

Default action: terminate the task abnormally.

RECORDBUSY

RESP2 values:

- 107** The NOSUSPEND keyword is specified and the record is locked by an active lock (see “Retained and active locks” on page 393).

Default action: abend the task with code AEX9.

SYSIDERR

RESP2 values:

- 130** 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.
- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The READ is issued for a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

Examples

The following example shows you how to read a record from a file named MASTER into a specified data area:

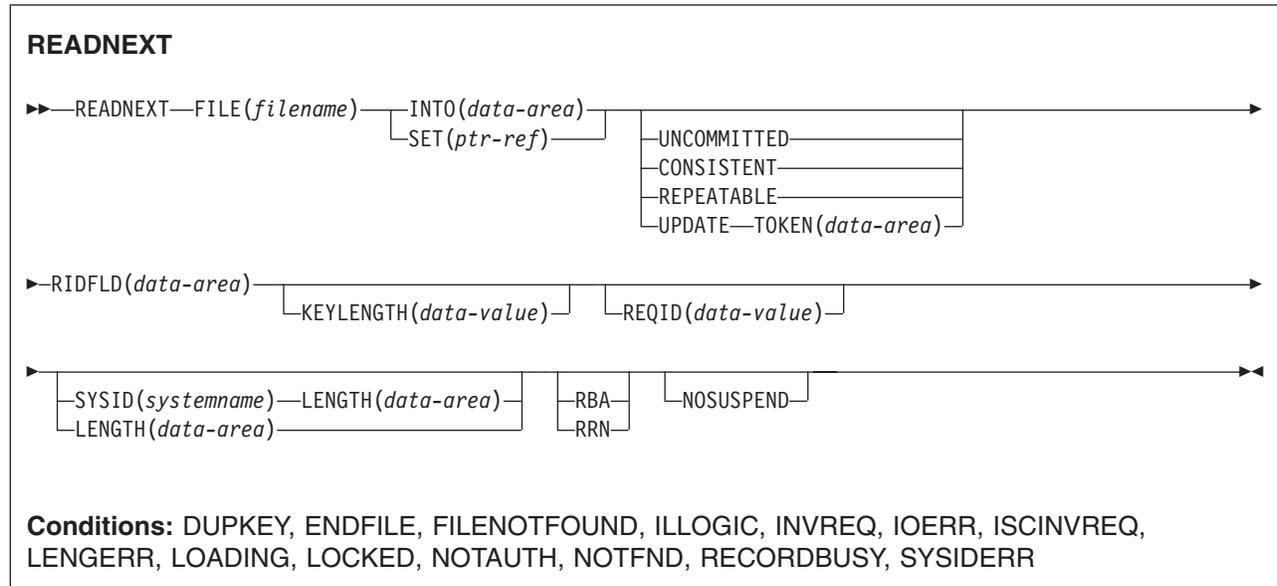
```
EXEC CICS READ  
  INTO(RECORD)  
  FILE('MASTER')  
  RIDFLD(ACCTNO)
```

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
```

READNEXT

Read next record during a browse of a file.



Description

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.

Note: You cannot specify an RBA for VSAM data sets greater than 4GB (that is, defined with the extended format / extended addressability attribute). The only VSAM data sets greater than 4GB supported by CICS are KSDS, and then only if they are accessed by key. CICS does not support ESDS or RRDS data sets defined with the extended attribute.

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.

It may, except for BDAM, 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 'last' key value, is treated as though the RIDFLD value has been modified, and results in a reposition (as above).

Reading files accessed in RLS mode

For files accessed in RLS mode, you can include the UPDATE keyword on the READNEXT request to update some records during the browse. If you specify UPDATE you must also specify TOKEN. You can then update the record by issuing a DELETE or REWRITE command specifying the TOKEN returned on the browse function.

Note: TOKEN without the UPDATE keyword implies UPDATE.

Use of the UPDATE option is subject to the following rules:

- You can specify UPDATE on a READNEXT command only if the file is accessed in RLS mode. If you specify UPDATE for a file accessed in non-RLS mode, CICS returns INVREQ.
- You can specify UPDATE on READNEXT, but not on the STARTBR or RESETBR commands.
- You can intermix UPDATE and non-update requests within the same browse.
- CICS does not preserve the UPDATE option for you from one READNEXT command to the next.

CICS supports only one TOKEN in a browse sequence, and the TOKEN value on each READNEXT invalidates the previous value.

Locks for UPDATE

Specifying UPDATE on a READNEXT acquires an exclusive lock. The duration of these exclusive locks within a browse depends on the action your application program takes:

- If you decide to DELETE or REWRITE the last record acquired by a READNEXT UPDATE in a browse, using the associated token, the lock remains active as follows:
 - If the file is recoverable, the lock is released at completion of the next syncpoint or rollback.
 - If the file is non-recoverable, the lock will be released by the time ENDBR has completed, but might be released earlier.
- If you decide *not* to update the last record read, CICS frees the exclusive lock when your program either issues another READNEXT or READPREV command, or ends the browse.

UNLOCK note

The UNLOCK command does *not* free an exclusive lock held by VSAM against a record acquired by READNEXT UPDATE. An UNLOCK in a browse invalidates the TOKEN returned by the last request.

Locks for read integrity

Specifying one of the read integrity options acquires a shared lock on each READNEXT. The duration of these shared locks with a browse depends on the type of read integrity you specify:

- If you specify CONSISTENT read integrity, the shared lock is held only for the duration of each read request—that is, until the record is returned to your program.
- If you specify REPEATABLE read integrity, the shared lock is held for the duration of the unit of work in which the browse is performed. In this case, your program could acquire a large number of shared locks, which will prevent the granting of exclusive locks for update functions. You should use REPEATABLE read integrity in a browse with caution.

Function shipping READNEXT with UPDATE or read-integrity

If a READNEXT command specifying UPDATE or one of the read-integrity options is function-shipped to a member of the CICS family of products that does not support UPDATE or the read integrity options, the request fails:

- If an ISC link is used, the request receives an ATNI abend.
- If an MRO link is used, the request receives an AXF8 abend.

AXF8 is an abend code, received by the sending side of a function-shipped request. It indicates that an attempt has been made to send a request specifying UPDATE on an MRO link to a CICS region that does not support update or read integrity options.

Read integrity

When a file is accessed in RLS mode, non-update read requests can specify read integrity options: UNCOMMITTED, CONSISTENT, or REPEATABLE.

If you don't specify any of these keywords, CICS uses the value specified on the READINTEG parameter of the FILE resource definition, for which the default is UNCOMMITTED.

If you want to use the level of read integrity specified in the READINTEG keyword of the FILE definition, and then you need to change from using a local file to a remote file, or if you change the location of a remote file, ensure that:

- The remote file-owning region is at the CICS Transaction Server for OS/390®, Version 1 Release 1 (or later) level.
- The FILE definition in the remote system specifies:
 - RLS mode
 - The correct read integrity values for your application.

Retained and active locks

RECORDBUSY refers to active locks, and LOCKED refers to retained locks:

These locks affect READNEXT requests which acquire locks; that is, update requests and requests with read integrity. These are the kinds of READNEXT requests which are referred to in the following bullets. Other READNEXT requests are unaffected by the presence of retained or active locks.

- READNEXT requests for records that have *retained* locks are always rejected with a LOCKED response.
- READNEXT requests for records that have *active* locks wait for the lock to be released, except when the NOSUSPEND keyword is specified, in which case CICS returns the RECORDBUSY response.

Options

CONSISTENT (RLS only)

The record is read with a level of read integrity provided by a VSAM shared lock that lasts for the duration of the request.

If the record is being modified by another task, which therefore holds an exclusive lock, the READNEXT request waits until the update is complete (unless NOSUSPEND is also specified) as follows:

- For a READNEXT request against a non-recoverable file, the READ completes as soon as any VSAM request performing the update completes.
- For a READNEXT request against a recoverable file, the READ request completes when the updating task completes its next syncpoint or rollback.

FILE(*filename*)

specifies the name 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. 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.

For remote files, the KEYLENGTH can be specified in the FILE definition. If KEYLENGTH is not defined there, and is not specified in the application program, and the key is longer than 4 characters, the default value is 4.

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 READNEXT command, the LENGTH parameter contains the actual length of the record.

This option must be specified if SYSID is specified.

If the file is on a remote system, the LENGTH parameter need not be set here but must be set in the file resource definition

If the file is on a local system, the LENGTH parameter must be set for variable-length records, using the INTO option, but not for fixed-length records. It is, however, advisable to specify the length for fixed-length records 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 COBOL, 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.

If you specify the SET option, you need not also specify the LENGTH option.

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 before truncation.

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.

NOSUSPEND (RLS only)

The request does not wait if the record is locked by VSAM with an active lock, including records locked as the result of a DEADLOCK.

RBA

(VSAM KSDS or ESDS base data sets, or CICS-maintained data tables only, but 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
- Coupling facility data tables
- Any KSDS files opened in RLS access mode
- KSDS or ESDS files that hold more than 4GB.

REPEATABLE (RLS only)

The record is read with a level of read integrity provided by a VSAM shared lock that lasts for the duration of the unit of work in which the read request is issued.

If the record is being modified by another task, which therefore holds an exclusive lock, the READNEXT request waits until the update is complete (unless NOSUSPEND is also specified) as follows:

- For a recoverable file, the READNEXT request completes when the updating transaction completes its next syncpoint or rollback.
- For a non-recoverable file, the READNEXT completes as soon as the VSAM request performing the update completes.

After the READNEXT request has completed, the record remains locked to the task that issued the READNEXT. Other tasks may continue to read the record but no other task is allowed to update the record until the task that issued the READNEXT performs its next syncpoint or rollback.

REQID(*data-value*)

specifies as a halfword binary value a unique request identifier for a browse, used to control multiple browse operations on a file. If this option is not specified, a default value of zero is assumed.

RIDFLD(*data-area*)

specifies the record identification 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.

See 'Identifying BDAM records' and 'VSAM record identification' in the *CICS Application Programming Guide* for more information about defining the record identification field.

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 browse (REQID) 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 user-key storage; otherwise it is in CICS-key storage.

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.

TOKEN(*data-area*) (RLS only)

Returns, as a fullword binary value, the request identifier for this READNEXT UPDATE request. This is an output value returned by file control to the requesting task, for use in associating a subsequent REWRITE or DELETE (or UNLOCK) request with the record returned on this READNEXT command.

You must specify the returned TOKEN on a subsequent REWRITE or DELETE command to identify the record being rewritten or deleted. You can also specify the value returned by CICS on the TOKEN option on a subsequent UNLOCK command, to identify the token that is being invalidated.

You must specify TOKEN if you specify UPDATE (but UPDATE is assumed if you specify TOKEN without UPDATE).

CICS supports only one active TOKEN at a time for a given REQID. Thus, a TOKEN value remains valid only until the following READNEXT, READPREV, RESETBR, or ENDBR command for this browse, or until a REWRITE, DELETE, or UNLOCK command (whichever is first).

TOKEN can be function shipped. However, if a request specifying TOKEN is function shipped to a member of the CICS family of products that does not recognize this keyword, the request fails.

UNCOMMITTED

The record is read without read integrity. The current data, as known to VSAM, is returned. No attempt is made to serialize this read request with any

concurrent update activity for the same record. The record may be being updated by another transaction, therefore the value of the record may change later if that update is subsequently backed out.

UPDATE (RLS only)

Specifies that the record is to be obtained for updating or deletion. If this option and TOKEN are both omitted, read only is assumed.

If you specify UPDATE, you must also specify TOKEN.

Conditions

DUPKEY

RESP2 values (VSAM):

140 A record is retrieved by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key follows. 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

RESP2 values:

90 An end-of-file condition is detected during the browse.

Default action: terminate the task abnormally.

FILENOTFOUND

RESP2 values:

1 A file name referred to in the FILE option cannot be found in the FCT.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values (VSAM):

110 A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the EXEC interface block, "EXEC interface block" on page 745.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

20 The FILE definition does not allow updates.

25 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.

26 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.

34 The REQID, if any, does not match that of any successful STARTBR command.

- 37 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.
- 42 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.
- 52 CONSISTENT is specified on a READ request to a non-RLS mode file, or to a data table that is specified with RLSACCESS(YES). CONSISTENT is not allowed if the file refers to a data table, even if the file definition specifies RLSACCESS(YES).
- 53 REPEATABLE is specified on a READ request to a non-RLS mode file, or to a data table that is specified with RLSACCESS(YES). REPEATABLE is not allowed if the file refers to a data table, even if the file definition specifies RLSACCESS(YES).
- 54 UPDATE is not allowed because the file is not a VSAM file that is accessed in RLS mode.
- 55 NOSUSPEND is specified on a READ request to a non-RLS mode file, or to a data table that is specified with RLSACCESS(YES). NOSUSPEND is not allowed if the file refers to a data table, even if the file definition specifies RLSACCESS(YES).

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 120 There is an I/O error during the READNEXT command. 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 a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.
Further information is available in the EXEC interface block, "EXEC interface block" on page 745.

Default action: terminate the task abnormally.

ISCINVREQ

RESP2 values:

- 70 The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 10 Neither the LENGTH nor the SET option is specified for a file with variable-length records or a BDAM file with undefined-format records.
- 11 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.
- 13 An incorrect length is specified for a file with fixed-length records.

Default action: terminate the task abnormally.

LOADING

RESP2 values:

- 104** The read request specifies a record key for a record in a coupling facility data table that is still being loaded, and the key is out of range of the records already loaded. Records can be browsed in a coupling facility data table during loading only if the requested key is within the range of those records already loaded.

The LOADING response can also be returned for a coupling facility data table that has failed during loading. For more information about what happens if the load for a coupling facility data table fails, see the description of the XD TLC global user exit in the *CICS Customization Guide*.

If your application programs encounter the LOADING condition persistently or too frequently, check that this is not caused by conflicting file definitions that reference the same data set.

LOCKED

RESP2 values:

- 106** The read request specifies the UPDATE keyword, or one of the read integrity keywords CONSISTENT or REPEATABLE, or the file resource definition specifies read integrity, but VSAM holds a retained lock against the record (see “Retained and active locks” on page 405).

The key of the locked record is not returned to your application program. If you handle this condition and control is returned to your program, the browse can continue and retrieve the record following the locked record by issuing another READNEXT request.

The LOCKED condition can also occur for a request to a recoverable CFDT that uses the locking model, if the record being read is locked by a retained lock. See the *CICS Recovery and Restart Guide* for information about investigating retained locks on records in a coupling facility data table.

Default action: abend the task with code AEX8.

NOTAUTH

RESP2 values:

- 101** A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

- 80** An attempt to retrieve a record based on the search argument provided is unsuccessful. 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.

RECORDBUSY

RESP2 values:

- 107** NOSUSPEND is specified on the request but VSAM holds an active lock against the record, which would cause the request to wait (see “Retained and active locks” on page 405).

The key of the locked record is not returned to your application program. If you handle this condition and control is returned to your program, the browse can continue and retrieve the record following the locked record by issuing another READNEXT request.

Default action: abend the task with code AEX9.

SYSIDERR

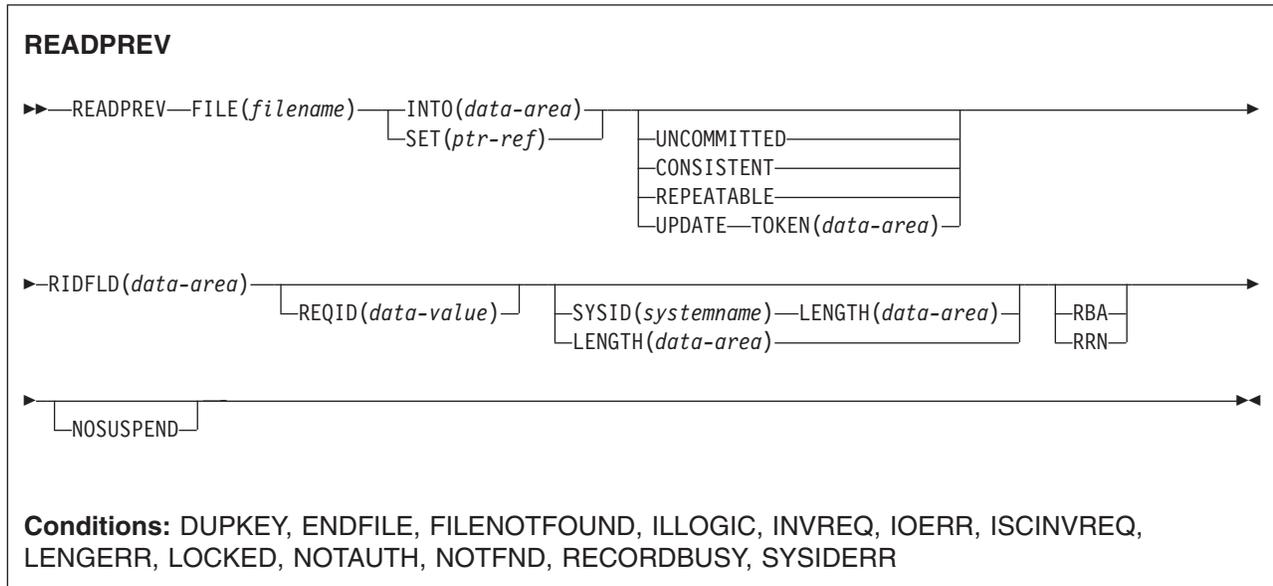
RESP2 values:

- 130** The SYSID option specifies a name that is neither the local CICS region nor a remote system (made known to CICS by defining a CONNECTION). SYSIDERR also occurs when the link to the remote system is closed
- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The READNEXT is issued against a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

READPREV

Read previous record during a file browse; VSAM and data tables only.



Description

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.

Note: You cannot specify an RBA for VSAM data sets greater than 4GB (that is, defined with the extended format / extended addressability attribute). The only VSAM data sets greater than 4GB supported by CICS are KSDS, and then only if they are accessed by key. CICS does not support ESDS or RRDS data sets defined with the extended attribute.

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 'last' key value, is treated as though the RIDFLD value had been modified and results in a reposition (as above).

Reading files accessed in RLS mode

For files accessed in RLS mode, you can include the UPDATE keyword on the READPREV request to update some records during the browse. If you specify UPDATE you must also specify TOKEN. You can then update the record by issuing a DELETE or REWRITE command specifying the TOKEN returned on the browse function.

Note: TOKEN without the UPDATE keyword implies UPDATE.

Use of the UPDATE option is subject to the following rules:

- You can specify UPDATE on a READPREV command only if the file is accessed in RLS mode. If you specify UPDATE for a file accessed in non-RLS mode, CICS returns INVREQ.
- You can specify UPDATE on READPREV, but not on the STARTBR or RESETBR commands.
- You can intermix UPDATE and non-update requests within the same browse.
- CICS does not preserve the UPDATE option for you from one READPREV command to the next.

CICS supports only one TOKEN in a browse sequence, and the TOKEN value on each READPREV invalidates the previous value.

Locks for UPDATE

Specifying UPDATE on a READPREV acquires an exclusive lock. The duration of these exclusive locks within a browse depends on the action your application program takes:

- If you decide to DELETE or REWRITE the last record acquired by a READPREV UPDATE in a browse, using the associated token, the lock remains active as follows:
 - If the file is recoverable, the lock is released at completion of the next syncpoint or rollback.
 - If the file is non-recoverable, the lock will be released by the time ENDBR has completed, but might be released earlier.
- If you decide *not* to update the last record read, CICS frees the exclusive lock when your program either issues another READNEXT or READPREV command, or ends the browse.

UNLOCK note

The UNLOCK command does *not* free an exclusive lock held by VSAM against a record acquired by READPREV UPDATE. An UNLOCK in a browse invalidates the TOKEN returned by the last request.

Locks for read integrity

Specifying one of the read integrity options acquires a shared lock on each READPREV. The duration of these shared locks with a browse depends on the type of read integrity you specify:

- If you specify CONSISTENT read integrity, the shared lock is held only for the duration of each read request—that is, until the record is returned to your program.
- If you specify REPEATABLE read integrity, the shared lock is held for the duration of the unit of work in which the browse is performed. In this case, your program could acquire a large number of shared locks, which will prevent the granting of exclusive locks for update functions. You should use REPEATABLE read integrity in a browse with caution.

Function shipping READPREV with UPDATE or read-integrity

If a READPREV command specifying UPDATE or one of the read-integrity options is function-shipped to a member of the CICS family of products that does not support UPDATE or the read integrity options, the request fails:

- If an ISC link is used, the request receives an ATNI abend.
- If an MRO link is used, the request receives an AXF8 abend.

AXF8 is an abend code, received by the sending side of a function-shipped request. It indicates that an attempt has been made to send a request specifying UPDATE on an MRO link to a CICS region that does not support update or read integrity options.

Read integrity

When a file is accessed in RLS mode, non-update read requests can specify read integrity options: UNCOMMITTED, CONSISTENT, or REPEATABLE.

If you don't specify any of these keywords, CICS uses the value specified on the READINTEG parameter of the FILE resource definition, for which the default is UNCOMMITTED.

If you want to use the level of read integrity specified in the READINTEG keyword of the FILE definition, and then you need to change from using a local file to a remote file, or if you change the location of a remote file, ensure that:

- The remote file-owning region is at the CICS Transaction Server for OS/390, Version 1 Release 1 (or later) level.
- The FILE definition in the remote system specifies:
 - RLS mode
 - The correct read integrity values for your application.

Retained and active locks

RECORDBUSY refers to active locks and LOCKED refers to retained locks:

These locks affect READPREV requests which acquire locks; that is, update requests and requests with read integrity. These are the kinds of READPREV requests which are referred to in the following bullets. Other READPREV requests are unaffected by the presence of retained or active locks.

- READPREV requests for records that have *retained* locks are always rejected with a LOCKED response.
- READPREV requests for records that have *active* locks wait for the lock to be released, except when the NOSUSPEND keyword is specified, in which case CICS returns the RECORDBUSY response.

Options

CONSISTENT (RLS only)

The record is read with a level of read integrity provided by a VSAM shared lock that lasts for the duration of the request.

If the record is being modified by another task, which therefore holds an exclusive lock, the READPREV request waits until the update is complete (unless NOSUSPEND is also specified) as follows:

- For a READPREV request against a non-recoverable file, the READPREV completes as soon as any VSAM request performing the update completes.
- For a READPREV request against a recoverable file, the READPREV request completes when the updating task completes its next syncpoint or rollback.

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 file resource definition. Otherwise, the file definition 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 RRRN 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.

For remote files, the KEYLENGTH can be specified in the FILE definition. If KEYLENGTH is not defined there, and is not specified in the application program, and the key is longer than 4 characters, the default value is 4.

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 READNEXT command, the LENGTH parameter contains the actual length of the record.

This option must be specified if SYSID is specified.

If the file is on a remote system, the LENGTH parameter need not be set here but must be set in the file resource definition

If the file is on a local system, the LENGTH parameter must be set for variable-length records, using the INTO option, but not for fixed-length records. It is, however, advisable to specify the length for fixed-length records 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 COBOL, 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.

If you specify the SET option, you need not also specify the LENGTH option.

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 before truncation.

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.

NOSUSPEND (RLS only)

The request does not wait if the record is locked by VSAM with an active lock, including records locked as the result of a DEADLOCK.

RBA

(VSAM KSDS or ESDS base data sets, or CICS-maintained data tables only, but 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
- Coupling facility data tables
- Any KSDS files opened in RLS access mode
- KSDS or ESDS files that hold more than 4GB

REPEATABLE (RLS only)

The record is read with a level of read integrity provided by a VSAM shared lock that lasts for the duration of the unit of work in which the read request is issued.

If the record is being modified by another task, which therefore holds an exclusive lock, the READPREV request waits until the update is complete (unless NOSUSPEND is also specified) as follows:

- For a recoverable file, the READPREV request completes when the updating transaction completes its next syncpoint or rollback.
- For a non-recoverable file, the READPREV completes as soon as the VSAM request performing the update completes.

After the READPREV request has completed, the record remains locked to the task that issued the READPREV. Other tasks can continue to read the record, but no other task is allowed to update the record until the task that issued the READPREV performs its next syncpoint or rollback.

REQID (*data-value*)

specifies as a halfword binary value a unique request identifier for a browse, used to control multiple browse operations on a file. If this option is not specified, a default value of zero is assumed.

RIDFLD (*data-area*)

specifies the record identification 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 register specified will not have been set, but can be loaded from DFHEITP1.

The pointer reference is valid until the next READPREV or READNEXT command specifying SET for the same browse (REQID) 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 returned in the SET pointer 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 user-key storage; otherwise it is in CICS-key storage.

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.

TOKEN (*data-area*) (RLS only)

Returns, as a fullword binary value, a unique identifier for this READPREV UPDATE request. This is an output value returned by file control to the requesting task, for use in associating a subsequent REWRITE or DELETE (or UNLOCK) request with the record returned on this READPREV command.

Your application program must specify the returned TOKEN on a subsequent REWRITE or DELETE command to identify the record being rewritten or deleted. Your application program can also specify the returned TOKEN on a subsequent UNLOCK command to identify the token that is being invalidated.

You must specify TOKEN if you specify UPDATE (but UPDATE is assumed if you specify TOKEN without UPDATE).

CICS supports only one active TOKEN at a time for a given REQID. Thus, a TOKEN value remains valid only until the following READNEXT, READPREV, or ENDBR command for this browse, or until a REWRITE, DELETE, or UNLOCK command (whichever is first).

TOKEN can be function shipped. However, if a request specifying TOKEN is function shipped to a member of the CICS family of products that does not recognize this keyword, the request fails.

UNCOMMITTED

The record is read without read integrity. The current data, as known to VSAM, is returned. No attempt is made to serialize this read request with any concurrent update activity for the same record. The record may be being updated by another transaction, therefore the value of the record may change later if that update is subsequently backed out.

UPDATE (RLS on1y)

Specifies that the record is to be obtained for updating or deletion. If this option and TOKEN are both omitted, read only is assumed.

If you specify UPDATE, you must also specify TOKEN.

Conditions

DUPKEY

RESP2 values:

140 A record is retrieved by way of an alternate index with the NONUNIQUEKEY attribute, and another alternate index record with the same key exists.

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

RESP2 values:

90 An end-of-file condition is detected during a browse.

Default action: terminate the task abnormally.

FILENOTFOUND

RESP2 values:

1 A file name referred to in the FILE option cannot be found in the FCT.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values (VSAM):

110 A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the EXEC interface block, "EXEC interface block" on page 745.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

20 The FILE definition does not allow updates.

- 24 A READPREV command is issued for a file for which the previous STARTBR or RESETBR command has the GENERIC option.
- 26 The KEYLENGTH option is specified and the specified length does not equal the length defined for the data set for this file refers to.
- 37 The type of record identification (for example, key or relative byte address) used to access a data set during the browse is changed.
- 39 A READPREV is issued for a BDAM file.
- 41 The REQID, if any, does not match that of any successful STARTBR command.
- 52 CONSISTENT is not allowed because the file is not a VSAM file that is accessed in RLS mode.
- 53 REPEATABLE is not allowed because the file is not a VSAM file that is accessed in RLS mode.
- 54 UPDATE is not allowed because the file is not a VSAM file that is accessed in RLS mode.
- 55 NOSUSPEND is not allowed because the file is not a VSAM file that is accessed in RLS mode.

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 120 An I/O error occurred during the browse. 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, "EXEC interface block" on page 745.)

For a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

Default action: terminate the task abnormally.

ISCVREQ

RESP2 values:

- 70 The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 10 Neither the LENGTH nor the SET option is specified for a file with variable-length records.
- 11 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.
- 13 An incorrect length is specified for a file with fixed-length records.

Default action: terminate the task abnormally.

LOCKED

RESP2 values:

- 106** The read request specifies the UPDATE keyword, or one of the read integrity keywords CONSISTENT or REPEATABLE, or the file resource definition specifies read integrity, but VSAM holds a retained lock against the record (see “Retained and active locks” on page 416).

The key of the locked record is not returned to your application program. If you handle this condition and control is returned to your program, the browse can continue and retrieve the record following the locked record by issuing another READPREV request.

The LOCKED condition can also occur for a request to a recoverable CFDT that uses the locking model, if the record being read is locked by a retained lock. See the *CICS Recovery and Restart Guide* for information about investigating retained locks on records in a coupling facility data table.

Default action: abend the task with code AEX8.

NOTAUTH

RESP2 values:

- 101** A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

- 80** An attempt to retrieve a record based on the search argument provided is unsuccessful. 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.

RECORDBUSY

RESP2 values:

- 107** NOSUSPEND is specified on the request but VSAM holds an active lock against the record, which would cause the request to wait (see “Retained and active locks” on page 416).

The key of the locked record is not returned to your application program. If you handle this condition and control is returned to your program, the browse can continue and retrieve the record following the locked record by issuing another READPREV request.

Default action: abend the task with code AEX9.

SYSIDERR

RESP2 values:

- 130** 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.

- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.

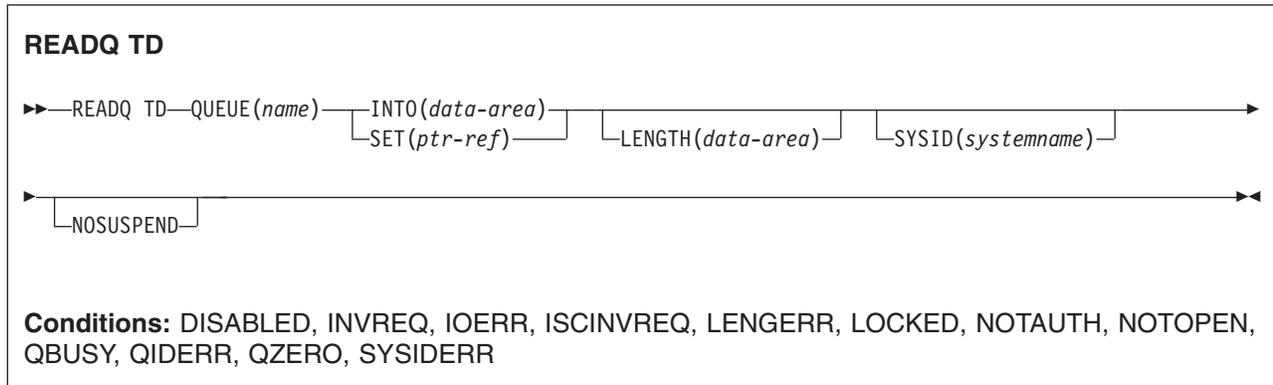
- 132** The READPREV is issued against a coupling facility data table that no

longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

READQ TD

Read data from the transient data queue.



Description

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 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.

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, or there is an active HANDLE CONDITION for QBUSY, the QBUSY condition is raised.

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 record read from the queue.

If you specify the INTO option, LENGTH need not be specified if the length can be generated by the compiler from the INTO variable. See “LENGTH options in CICS commands” on page 8 for more information about when LENGTH must be specified.

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.

Note, however, that if a HANDLE CONDITION for QBUSY is active when the command is executed, this also overrides the default action, and control is passed to the user label supplied in the HANDLE CONDITION. This takes precedence over the NOSUSPEND option but is, of course, negated by either NOHANDLE or RESP.

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 named queue must have been defined to CICS.

If SYSID is specified, the queue is assumed to be on a remote system whether or not it is defined as remote. Otherwise the transient data queue definition 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.

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.

This condition can also occur if the FREE=CLOSE operand has been used in the data set definition for an extrapartition queue, and the queue has been closed and reopened.

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.

LOCKED

occurs when the request cannot be performed because use of the queue has been restricted owing to a unit of work failing in-doubt. This can happen on any request for a logically-recoverable queue defined with WAIT(YES) and WAITACTION(REJECT) in the TDQUEUE resource definition.

Specify WAIT(YES) and WAITACTION(Queue) in the TDQUEUE resource definition if you want the transaction to wait.

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, or a HANDLE for the condition must be active, 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.

Examples

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:

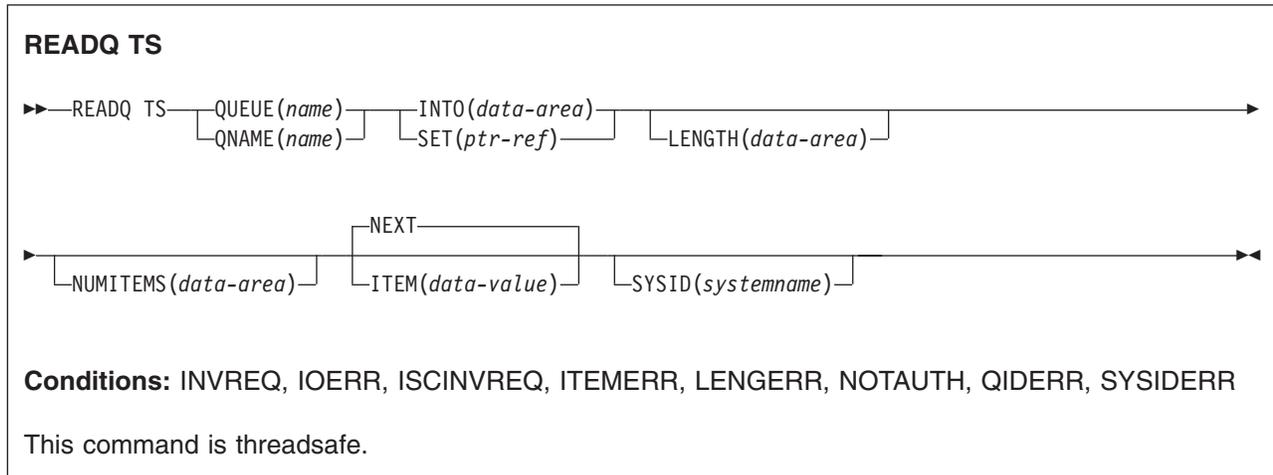
```
EXEC CICS READQ TD
      QUEUE('CSML')
      INTO(DATA)
      LENGTH(LDATA)
```

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)
```

READQ TS

Read data from temporary storage queue.



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 Application Programming Guide* for more information about transaction affinities.

Description

READQ TS retrieves data from a temporary storage queue in main or auxiliary storage.

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, LENGTH need not be specified if the length can be generated by the compiler from the INTO variable.

See “LENGTH options in CICS commands” on page 8 for more information about when LENGTH must be specified.

If you specify INTO, LENGTH defines 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, 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. If the name has less than 8 characters, you must still use an 8-character field, padded with blanks if necessary.

QNAME (*name*)

an alternative to QUEUE, QNAME specifies the symbolic name (1–16 characters) of the queue to be read from. If the name has less than 16 characters, you must still use a 16-character field, padded with blanks if necessary.

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 and shared queues only) specifies the system name (1–4 characters) identifying the remote system or shared queue pool to which the request is directed.

Conditions

INVREQ

occurs in either of the following situations:

- the queue was created by CICS internal code.
- the queue name specified consists solely of binary zeroes.

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 5** There is an irrecoverable input/output error for a shared queue.

Default action: terminate the task abnormally.

ISCVREQ

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.

This condition only applies to the INTO option and cannot occur with SET.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

101 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.

SYSIDERR

RESP2 values:

4 The CICS region in which the temporary storage command is executed fails to connect to the TS server managing the TS pool that supports the referenced temporary storage queue. (For example, this can happen if the CICS region is not authorized to access the temporary storage server).

SYSIDERR can also occur if the temporary storage server has not been started, or because the server has failed (or been stopped) while CICS continues executing. Also occurs in any of the following situations:

- 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).
- When the link to the remote system is closed.

Default action: terminate the task abnormally.

Examples

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.

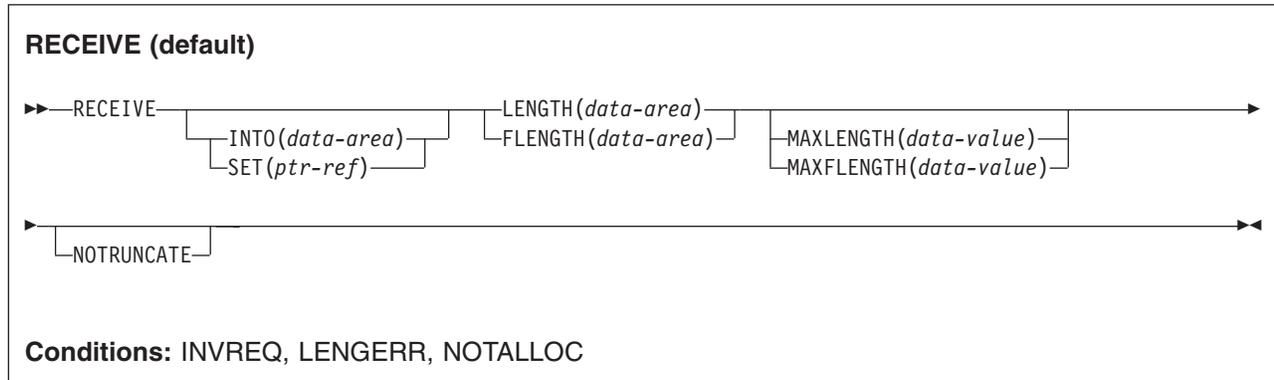
```
EXEC CICS READQ TS
      ITEM(1)
      QUEUE(UNIQNAME)
      INTO(DATA)
      LENGTH(LDATA)
```

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
```

RECEIVE (VTAM default)

Receive data from standard CICS terminal support or from a task that is not attached to a terminal.



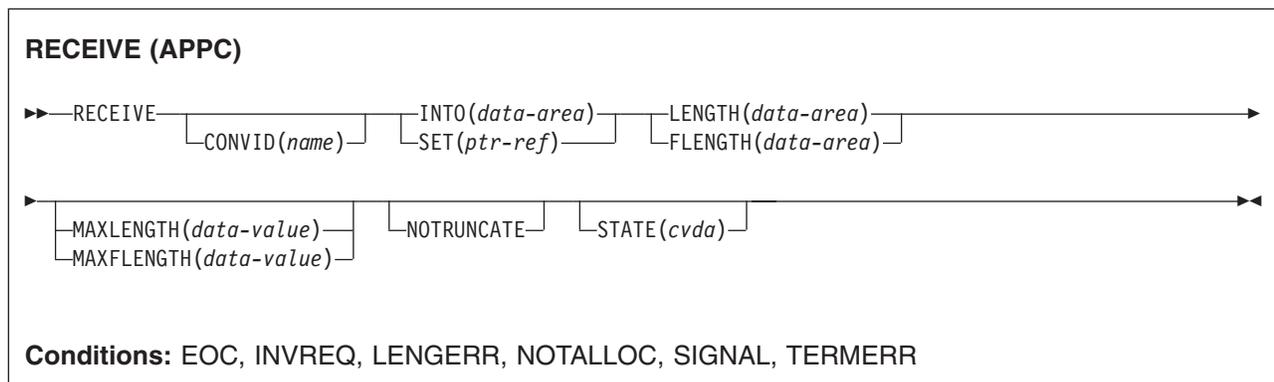
Description

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)

Receive data on an APPC mapped conversation.

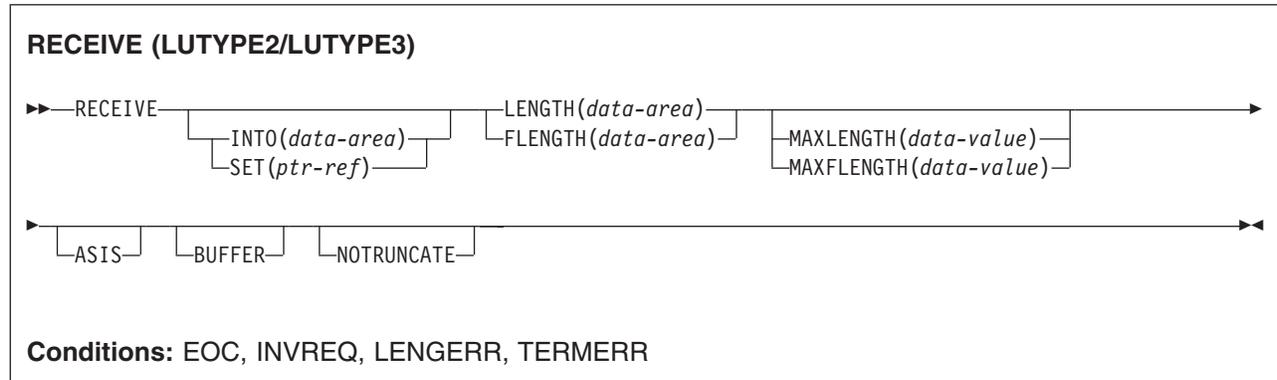


Description

RECEIVE receives data from the conversation partner in an APPC mapped conversation.

RECEIVE (LUTYPE2/LUTYPE3)

Receive data from a 3270-display logical unit (LUTYPE2) or a 3270-printer logical unit (LUTYPE3).



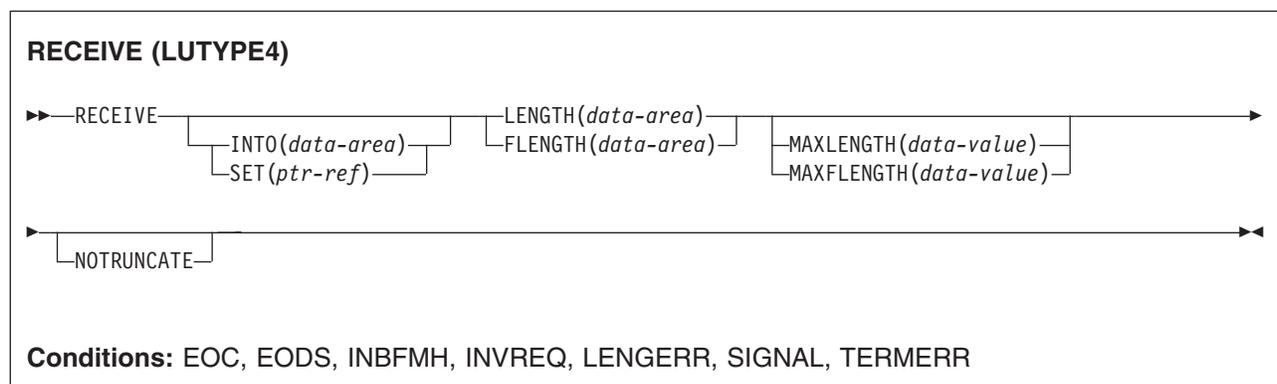
Description

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)

Receive data from an LUTYPE4 logical unit.



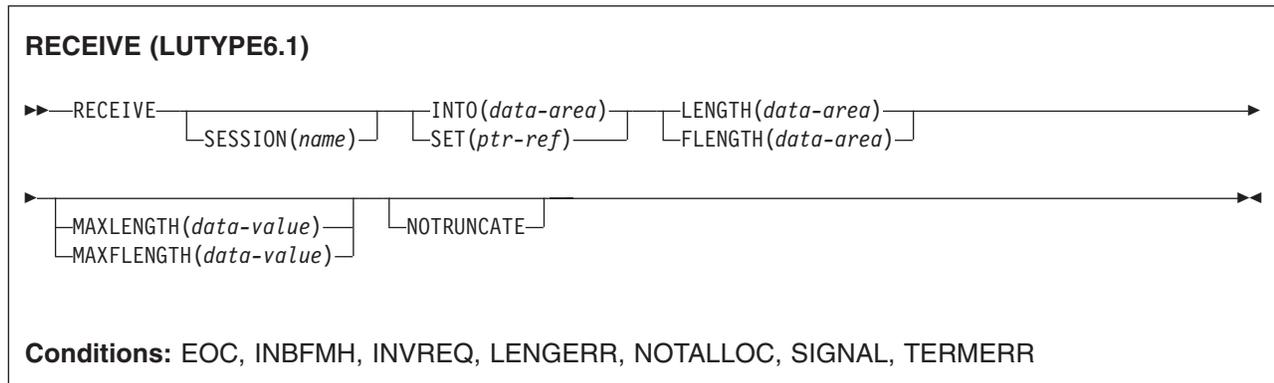
Description

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)

Receive data on an LUTYPE6.1 session.

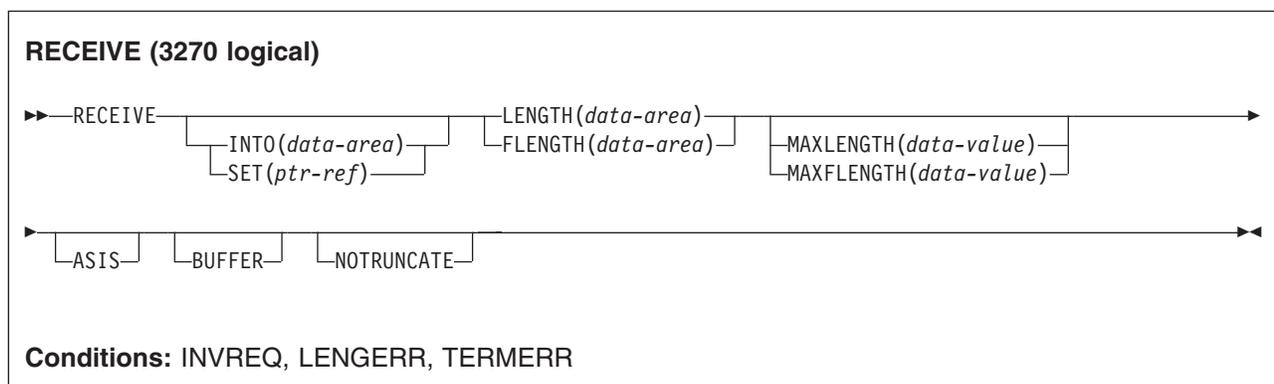


Description

RECEIVE receives data from the conversation partner in an LUTYPE6.1 conversation.

RECEIVE (3270 logical)

Receive data from a 3270 logical unit.



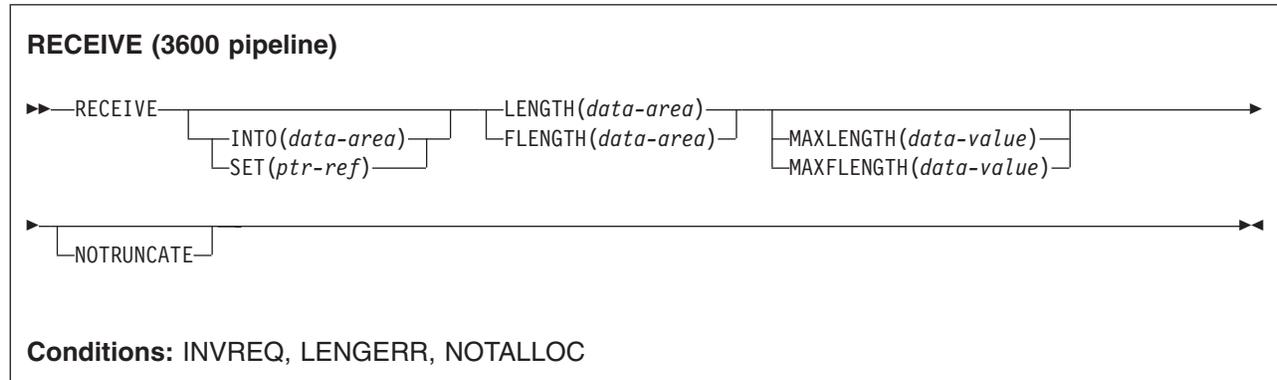
Description

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 (3600 pipeline)

Receive initial input data from a 3600 pipeline logical unit. Subsequent RECEIVES for further input data are not allowed.

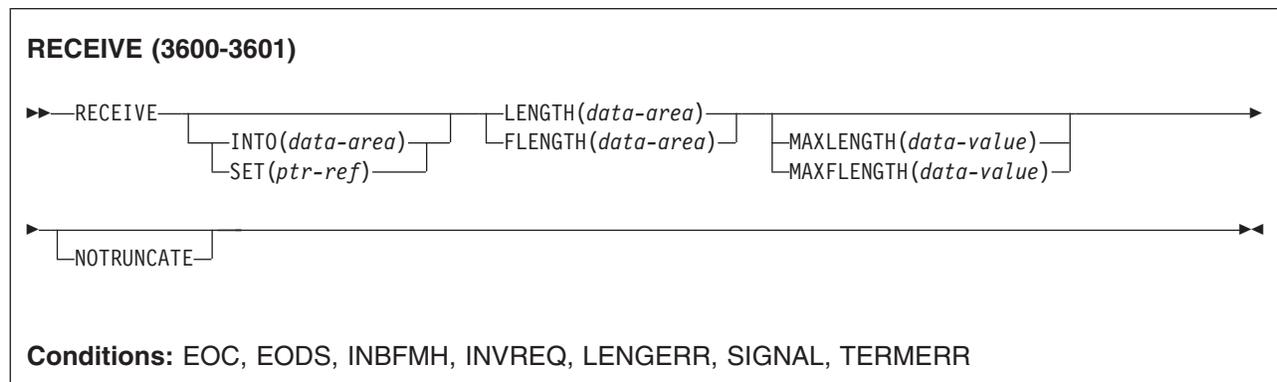


Description

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)

Receive data from a 3600 (3601) logical unit.



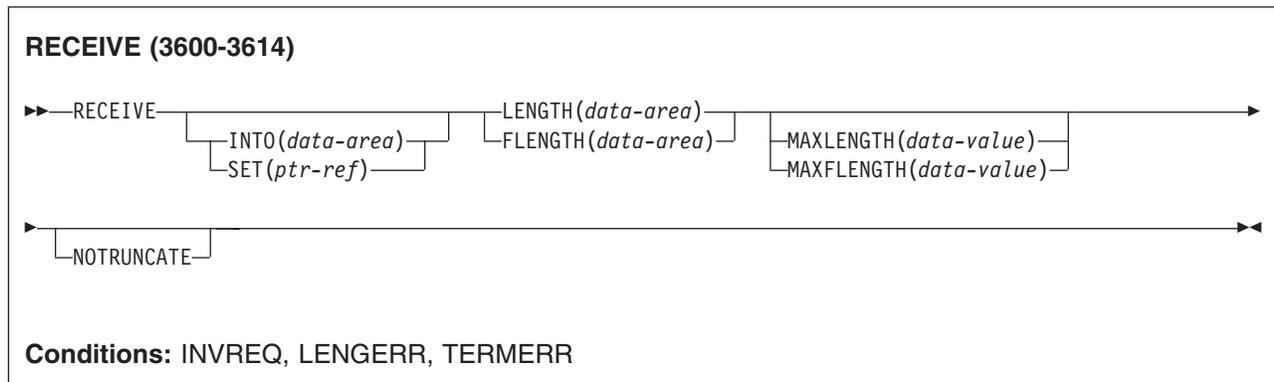
Description

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)

Receive data from a 3600 (3614) logical unit.



Description

RECEIVE receives data from the terminal.

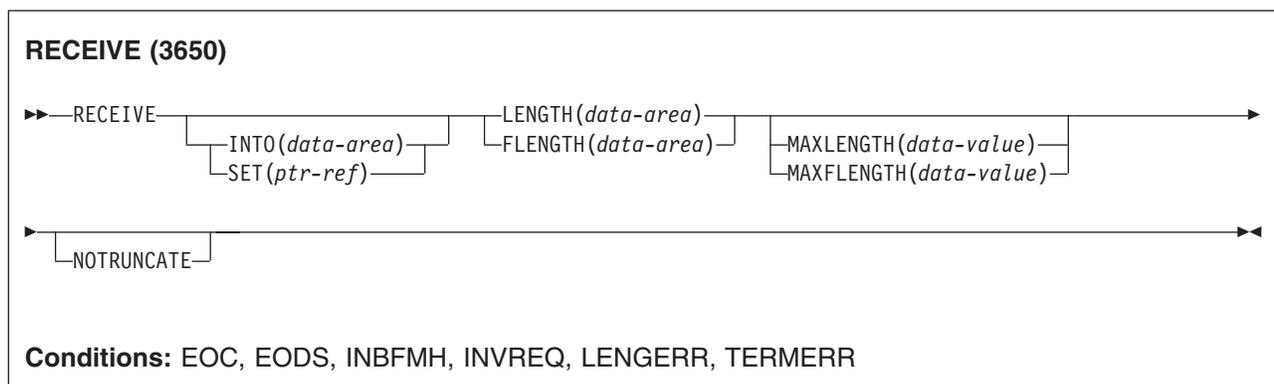
The data-stream and communication format used between a CICS application program and a 3614 is determined by the 3614. The application program is therefore device dependent when handling 3614 communication.

For further information about designing 3614 application programs for CICS, refer to the *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)

Receive data from 3650 logical units.



Description

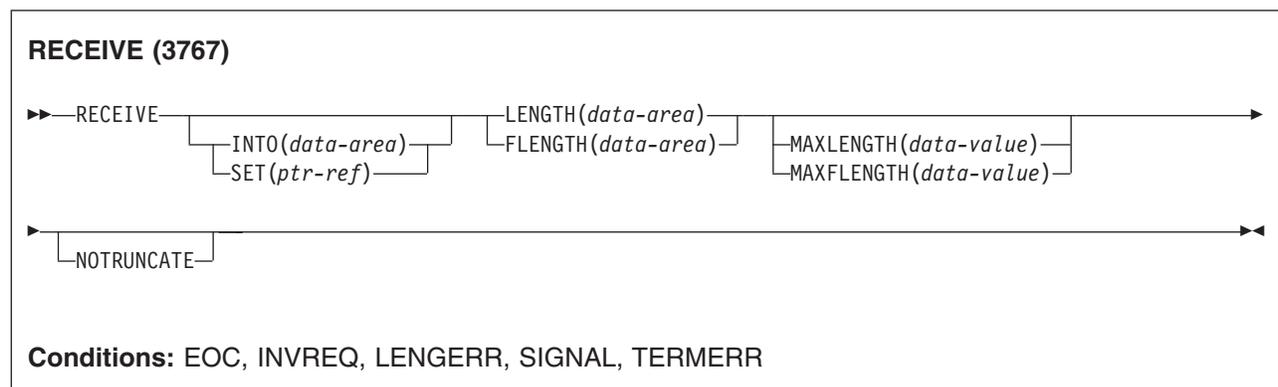
RECEIVE receives data from the terminal. This form of RECEIVE also applies to the following 3650 devices:

- 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 (3767)

Receive data from a 3767 interactive logical unit.



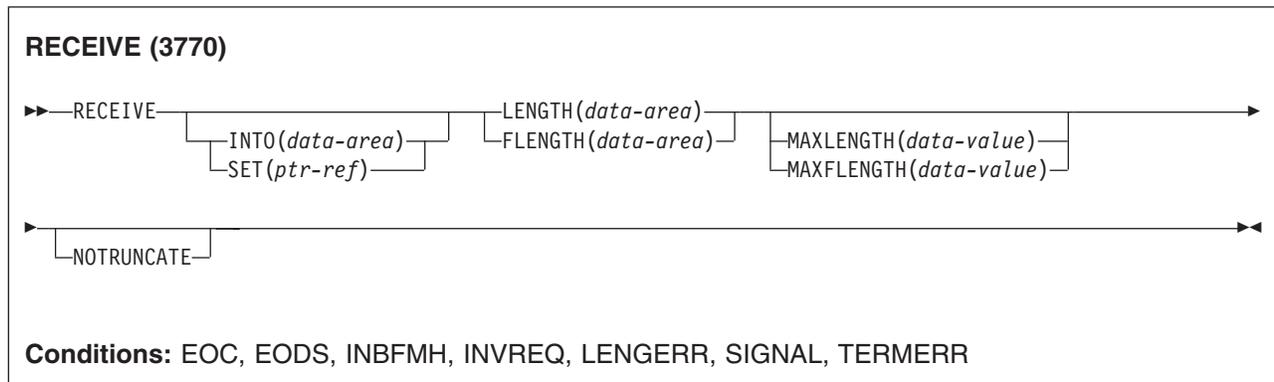
Description

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)

Receive data from a 3770 batch logical unit.

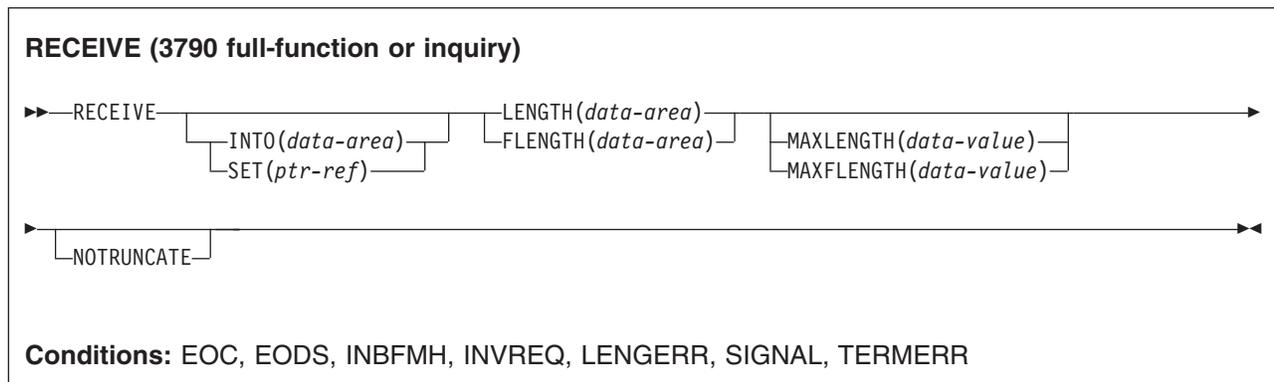


Description

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)

Receive data from a 3790 full-function or inquiry logical unit.



Description

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 options

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*)

#

An alternative to LENGTH. For architectural reasons, this option is limited to a maximum of 32K for all terminal-related RECEIVE commands.

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.

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 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

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) in any of the following situations:

- The command is used on an APPC conversation that is not using the EXEC CICS interface or that is not a mapped conversation.

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

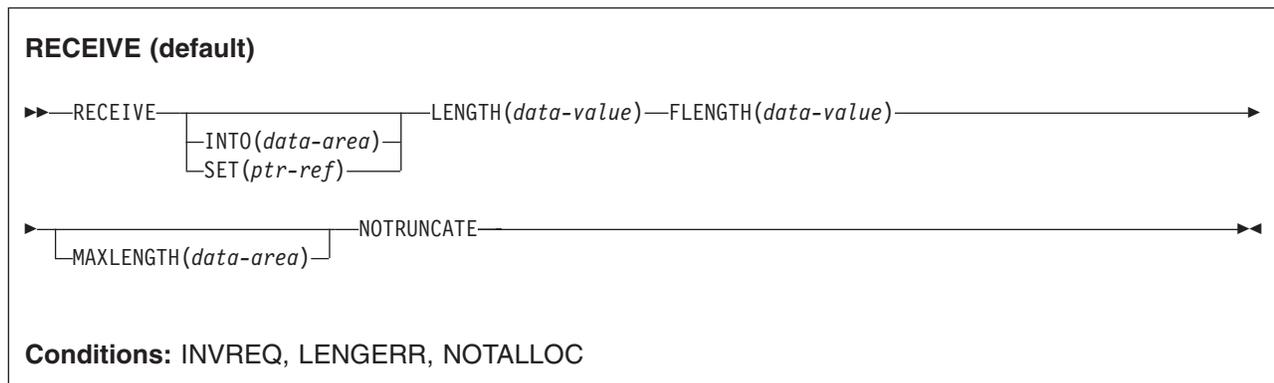
occurs for a session-related or terminal-related error. Any action on that conversation other than a FREE causes an ATCV abend.

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 default)

Receive data from standard CICS terminal support (TCAM) or from a task that is not attached to a terminal.



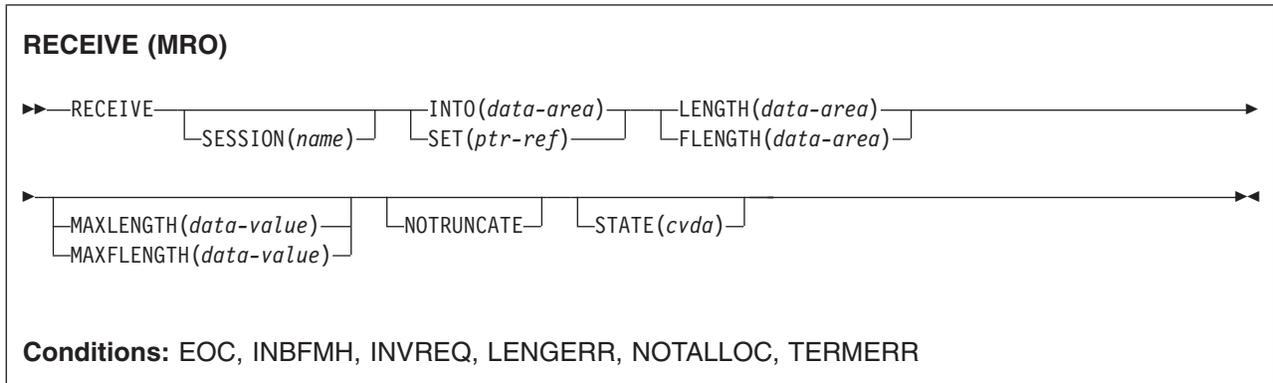
Description

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)

Receive data on an MRO conversation.

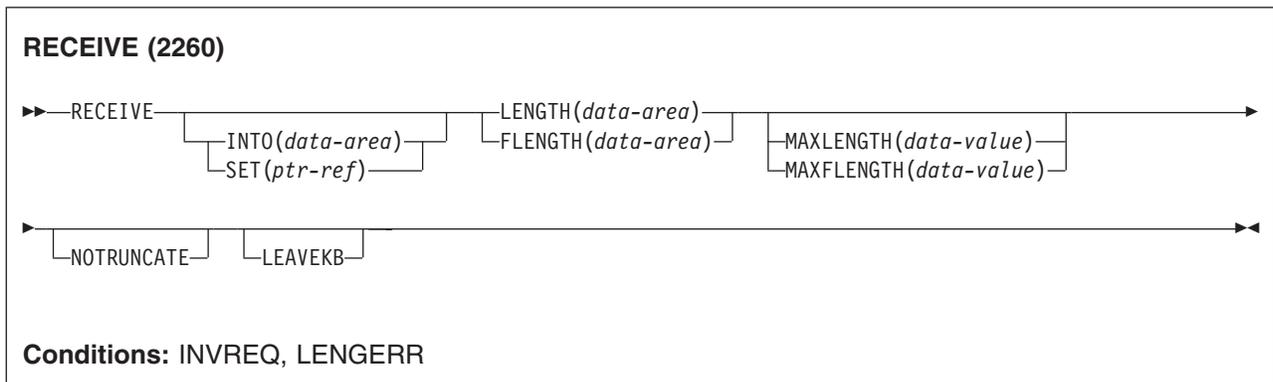


Description

RECEIVE receives data from the conversation partner in an MRO conversation.

RECEIVE (2260)

Receive data from a 2260 or 2265 display station.



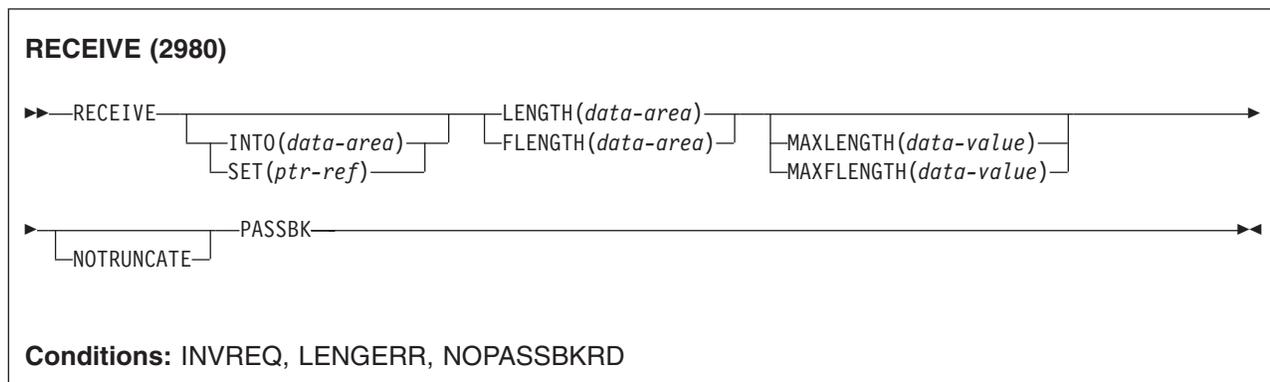
Description

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 (2980)

Receive data from a 2980 general banking terminal system.



Description

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.

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-n-A or STATION-n-N-

STATION_n_A or STATION_n_N

where **n** is an integer (0 through 9) and A and N signify alternate and 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.

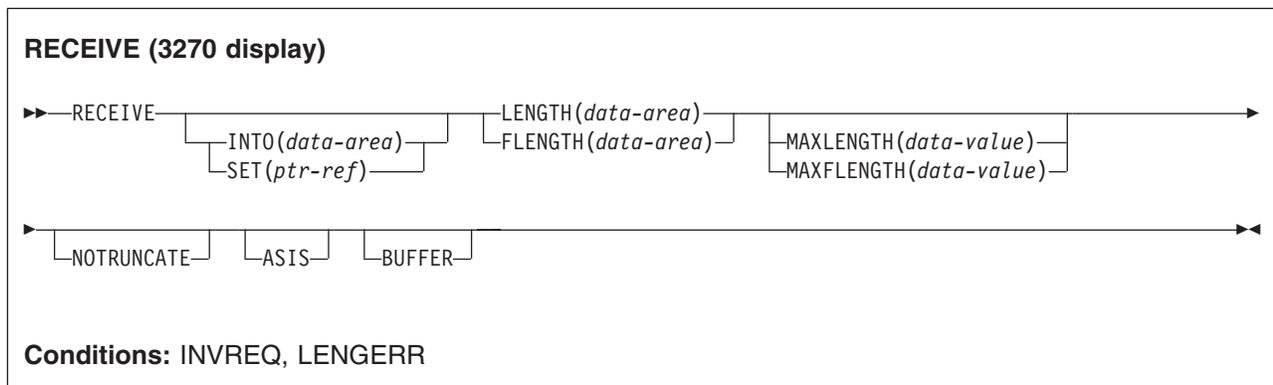
Any of the DFH2980 values TAB-ZERO through TAB-NINE for COBOL and PL/I, may be compared with NUMTAB to find out the number of tab characters that need to be inserted in an output message to get correct positioning of the print element. The tab character is included in DFH2980 as TABCHAR.

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)

Receive data from a 3270 information display system (TCAM).



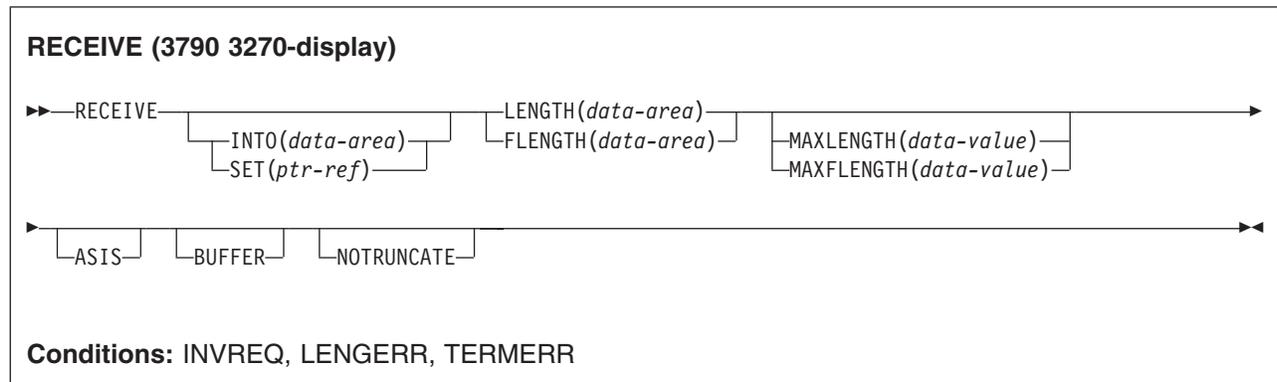
Description

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 (3790 3270-display)

Receive data from a 3790 (3270-display) logical unit.



Description

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 options

Options

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.

If you specify the INTO option, but omit the MAXLENGTH option, the argument for the LENGTH option 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 for the LENGTH option is set to the original length of the data.

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 in CICS commands” on page 8.

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 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.

If you specify the SET option, the argument for the LENGTH option must be a data area. When the data has been received, the data area is set to the length of the data.

STATE(*cvda*)

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

- ALLOCATED
- FREE
- PENDFREE
- RECEIVE
- ROLLBACK
- SEND
- SYNCFREE
- SYNCRECEIVE
- SYNCSEND

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

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

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

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

occurs if a RECEIVE command is terminated by the attention (ATTN) key rather than the return key.

Default action: ignore the condition.

TERMERR

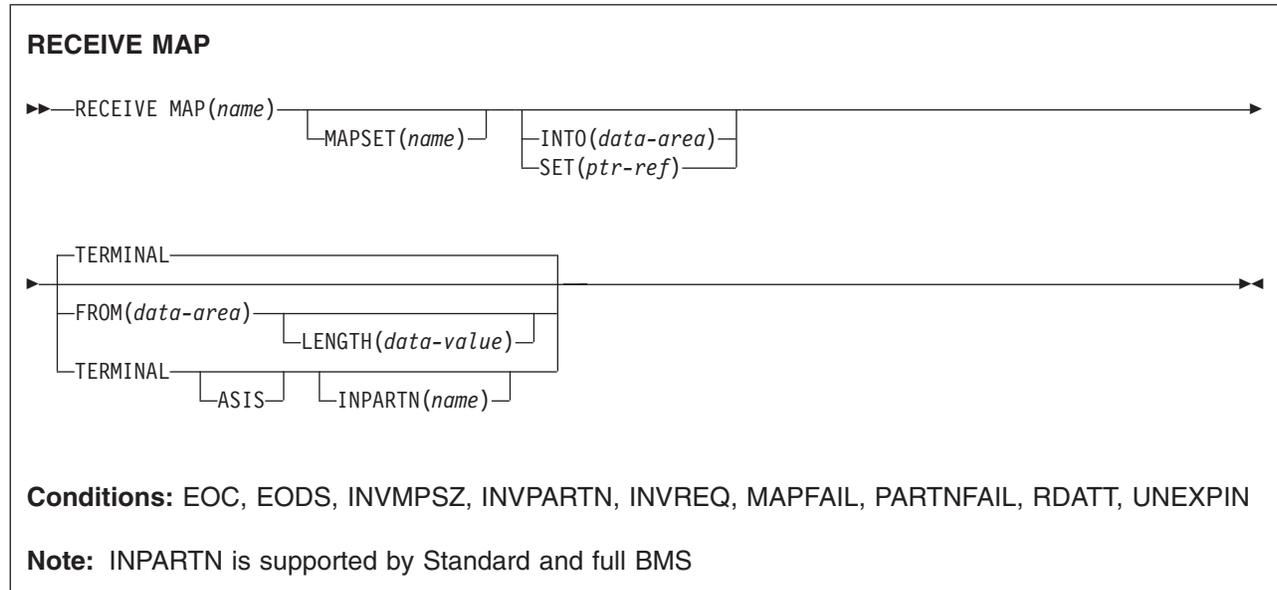
occurs for a terminal-related error, such as a session failure. This condition applies to VTAM-connected terminals only.

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

Receive screen input into an application data area. For further information about BMS, see the *CICS Application Programming Guide*.



Description

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 “BMS macros” on page 784 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.

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 DFHMDI operands, TERM and specifying NODDS in the BMS operand).

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 DFHMDI operands, TERM and specifying NODDS in the BMS operand).

For a description of a safe upper limit, see “LENGTH options in CICS commands” on page 8.

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.

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 INPARTN 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.

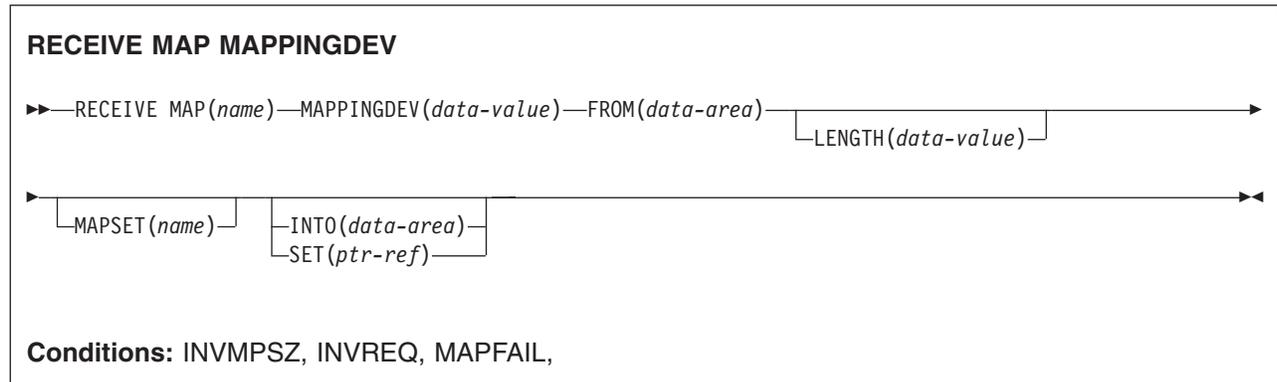
UNEXPIN

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

Receive screen input into an application data area, without reference to the principal facility, if any. Terminal characteristics are obtained from the MAPPINGDEV parameter. For further information about BMS, see the *CICS Application Programming Guide*.



Description

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.

Options

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 DFHMDI operands, TERM and specifying NODDS in the BMS operand). For a description of a safe upper limit, see "LENGTH options in CICS commands" on page 8.

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.

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.

Default action: terminate the task abnormally.

RECEIVE PARTN

Receive data from an 8775 terminal partition. This command is only available on standard and full BMS. For further information about BMS, see the *CICS Application Programming Guide*.

RECEIVE PARTN

►►—RECEIVE PARTN(*data-area*)—►►

Standard and full BMS:

RECEIVE PARTN

►►— $\left. \begin{array}{l} \text{INTO}(\textit{data-area}) \\ \text{SET}(\textit{ptr-ref}) \end{array} \right\}$ —LENGTH(*data-value*)— $\left. \text{ASIS} \right\}$ —►►

Conditions: EOC, EODS, INVPARTN, INVREQ, LENGERR

Description

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 “BMS macros” on page 784 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.

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 in CICS commands” on page 8.

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 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.

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

Release a loaded program, table, or mapset.

RELEASE

►►—RELEASE—PROGRAM(*name*)—►►

Conditions: INVREQ, NOTAUTH, PGMIDERR

This command is threadsafe.

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 Application Programming Guide* for more information about transaction affinities.

Description

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.

Note: RELEASE does not remove a program from storage. It reduces the RESCOUNT by 1 and when the count reaches zero, the storage occupied by the program can be released by CICS storage manager.

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.

Options

PROGRAM(*name*)

specifies the identifier (1–8 characters) of a program, table, or mapset to be released.

Conditions

INVREQ

RESP2 values:

- 5 An invalid attempt is made by the program to release itself. 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.
- 6 The command is issued for a program that is not loaded.

- 7 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 .
- 17 The program is defined with RELOAD=YES. It must be released by a FREEMAIN rather than a RELEASE command.
- 30 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.

Default action: terminate the task abnormally.

NOTAUTH

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

Default action: terminate the task abnormally.

PGMIDERR

RESP2 values:

- 1 A program, table, or mapset has no entry in the PPT.
- 2 A program, table, or mapset is disabled.
- 9 The installed program definition is for a remote program.
- 42 An attempt has been made to RELEASE a JVM program. This is invalid because Java byte codes programs are not managed by CICS Loader.

Default action: terminate the task abnormally.

Examples

The following example shows how to release an application program, called PROG4, loaded in response to a LOAD command:

```
EXEC CICS RELEASE PROGRAM('PROG4')
```

REMOVE SUBEVENT

Remove a sub-event from a BTS composite event.

REMOVE SUBEVENT

▶—REMOVE—SUBEVENT(*data-value*)—EVENT(*data-value*)—▶

Conditions: EVENTERR, INVREQ

Description

REMOVE SUBEVENT removes a sub-event from a named BTS composite event.

This call does not delete the removed event. Nor does it reset the event's fire status. Note that, after this call, the removed event—because it is no longer a sub-event—will cause the current activity to be reattached if it fires.

Removing a sub-event causes the composite's predicate to be re-evaluated.

Options

EVENT(*data-value*)

specifies the name (1–16 characters) of the composite event.

SUBEVENT(*data-value*)

specifies the name (1–16 characters) of the event which is to be removed from the named composite event.

Conditions

EVENTERR

RESP2 values:

- 4 The event specified on the EVENT option is not recognized by BTS.
- 5 The sub-event specified on the SUBEVENT option is not recognized by BTS.

INVREQ

RESP2 values:

- 1 The command was issued outside the scope of an activity.
- 2 The event specified on the EVENT option is not a composite event.
- 3 The event specified on the SUBEVENT option is not a sub-event of the composite event specified on the EVENT option.

RESET ACQPROCESS

Reset a BTS process to its initial state.

RESET ACQPROCESS

▶—RESET—ACQPROCESS—▶

Conditions: INVREQ, IOERR, LOCKED, NOTAUTH, PROCESSBUSY, PROCESSERR

Description

RESET ACQPROCESS resets the currently-acquired BTS process to its initial state. Any descendant activities of the root activity are deleted.

Note: RESET has no effect on the process containers, nor on the root activity's containers, the contents of which are unchanged.

Issue this command, before a second RUN command, when a process needs to be retried. When the process is re-run, the root activity is sent a DFHINITIAL event.

To be eligible to be reset, a process must:

1. Have been acquired in the current unit of work—that is, it must be the currently-acquired process.
2. Be in one of the following modes:
 - COMPLETE. This is the usual case. Perhaps the process has completed abnormally, and needs to be reset before being retried.
 - INITIAL. The process has not yet been run.

Options

ACQPROCESS

specifies that the process that is currently acquired by the requestor is to be reset.

Conditions

INVREQ

RESP2 values:

- 15** The unit of work that issued the request has not acquired a process.

IOERR

RESP2 values:

- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

NOTAUTH

RESP2 values:

101 The user associated with the issuing task is not authorized to reset the process.

PROCESSBUSY

RESP2 values:

13 The request timed out. It may be that another task using this process-record has been prevented from ending.

PROCESSERR

RESP2 values:

14 The process to be reset is not in COMPLETE or INITIAL mode.

RESET ACTIVITY

Reset a BTS activity to its initial state.

RESET ACTIVITY

▶▶—RESET—ACTIVITY(*data-value*)—————▶▶

Conditions: ACTIVITYBUSY, ACTIVITYERR, INVREQ, IOERR, LOCKED, NOTAUTH

Description

RESET ACTIVITY resets a BTS child activity to its initial state. Its completion event is added to the parent's event pool, with the fired status set to NOTFIRED. If the activity has children of its own, they are deleted.

Note: RESET has no effect on the contents of the activity's data containers, which are unchanged.

Issue this command, before a second RUN command, when an activity needs to be retried. When the activity is re-run, it is sent a DFHINITIAL event.

To be eligible to be reset, an activity must:

1. Be a child of the activity that issues the RESET command.
2. Be in one of the following modes:
 - COMPLETE. This is the usual case. Perhaps the activity has completed abnormally, and needs to be reset before being retried.
 - INITIAL. The activity has not yet been run.

Options

ACTIVITY(*data-value*)

specifies the name (1–16 characters) of the activity to be reset. This must be a child of the current activity.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19** The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8** The activity named in the ACTIVITY option is not a child of the current activity.

- 14** The activity to be reset is not in COMPLETE or INITIAL mode.

INVREQ

RESP2 values:

- 4 The RESET ACTIVITY command was issued outside the scope of a currently-active activity.

IOERR

RESP2 values:

- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

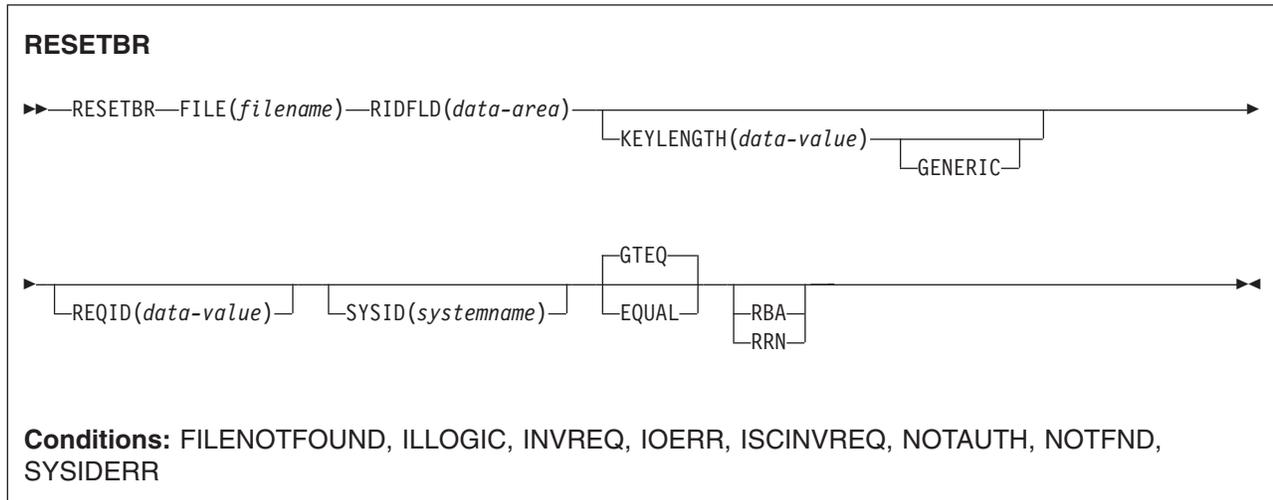
NOTAUTH

RESP2 values:

- 101 The user associated with the issuing task is not authorized to reset the activity.

RESETBR

Reset start of browse.



Description

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.

If a RESETBR request specifies the precise key at which the browse is to start (that is, it specifies a full key and the EQUAL keyword) the record returned on the following READNEXT (or READPREV) may not be the same as the record specified by the RESETBR for a file opened in VSAM NSR or RLS mode. This can occur because the initial record specified on the RESETBR command can be deleted by another transaction in between the RESETBR completing and a READNEXT or READPREV being issued. In VSAM LSR mode, the initial record cannot be deleted between the RESETBR and the READNEXT.

Note: RESETBR invalidates a TOKEN set by a previous READ or READNEXT command.

Options

EQUAL

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*)

(VSAM and data table) 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

(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.

For remote files, the KEYLENGTH can be specified in the FILE definition. If KEYLENGTH is not defined there, and is not specified in the application program, and the key is longer than 4 characters, the default value is 4.

RBA

(VSAM KSDS or ESDS base data sets, or CICS-maintained data tables 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
- Coupling facility data tables
- KSDS or ESDS files that hold more than 4GB
- Any KSDS files opened in RLS access mode

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. The contents can be a key, a relative byte address, or a 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.

Conditions**FILENOTFOUND**

RESP2 values:

- 1** A file name referred to in the FILE option cannot be found in the FCT.
Default action: terminate the task abnormally.

ILLOGIC

RESP2 values VSAM):

- 110** A VSAM error occurs that does not fall within one of the other CICS response categories.
(See EIBRCODE in the EXEC interface block, at "EXEC interface block" on page 745.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

- 25** 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.
- 26** 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.
- 36** The REQID, if any, does not match that of any successful STARTBR command.
- 42** The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than zero.

- 44** The command does not conform to the format of RESETBR for a user-maintained or coupling facility data table; for example, RBA is specified.
- 51** A RESETBR command to a KSDS file that is being accessed in RLS mode specifies the RBA keyword. RLS mode does not support RBA access to KSDS data sets.

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 120** There is an I/O error during the file control operation. 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, see "EXEC interface block" on page 745.)
For a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

Default action: terminate the task abnormally.

ISCVREQ

RESP2 values:

- 70** The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

- 101** A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

- 80** An attempt to retrieve a record based on the search argument provided is unsuccessful.
NOTFND can also occur if a generic RESETBR with KEYLENGTH(0) specifies the EQUAL option.

Default action: terminate the task abnormally.

SYSIDERR

RESP2 values:

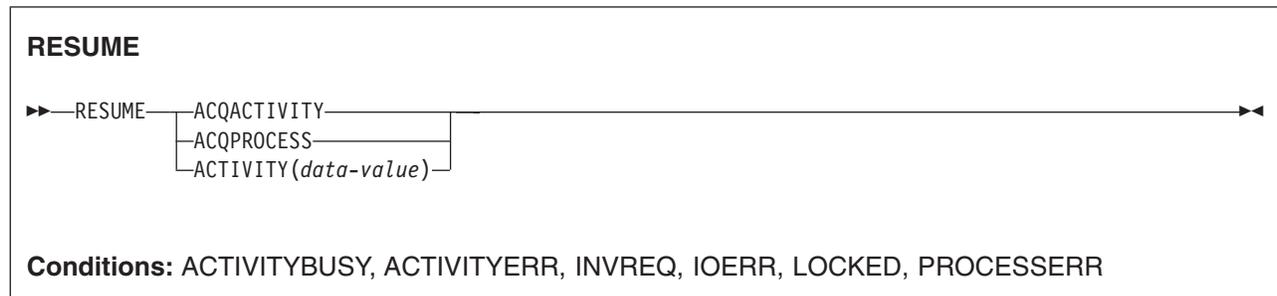
- 130** The SYSID option specifies a name that is neither the local CICS region nor a remote system (as defined by a CONNECTION definition). SYSIDERR also occurs when the link to the remote system is closed
- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The RESETBR is issued against a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS*

System Definition Guide for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

RESUME

Resume a suspended BTS process or activity.



Description

RESUME resumes a BTS process or activity that has previously been suspended (by means of a SUSPEND command). That is, it allows the process or activity to be reattached when events in its event pool are fired. If events that would normally have caused reattachment have occurred during the time the process or activity was suspended, the latter is reattached for all these events.

The only process a program can resume is the one it has acquired in the current unit of work.

The only activities a program can resume are as follows:

- If it is running as the activation of an activity, its own child activities. It can resume several of its child activities within the same unit of work.
- The activity it has acquired, by means of an ACQUIRE ACTIVITYID command, in the current unit of work.

Options

ACQACTIVITY

specifies that the activity to be resumed is the one that the current unit of work has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the process that is currently acquired by the requestor is to be resumed.

ACTIVITY(*data-value*)

specifies the name (1–16 characters) of the child activity to be resumed.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19** The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8** The activity named on the ACTIVITY option could not be found.

- 14 The activity is in COMPLETE or CANCELLING mode, and therefore cannot be resumed.

INVREQ

RESP2 values:

- 4 The ACTIVITY option was used to name a child activity, but the command was issued outside the scope of a currently-active activity.
- 15 The ACQPROCESS option was used, but the unit of work that issued the request has not acquired a process.
- 24 The ACQACTIVITY option was used, but the unit of work that issued the request has not acquired an activity.

IOERR

RESP2 values:

- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

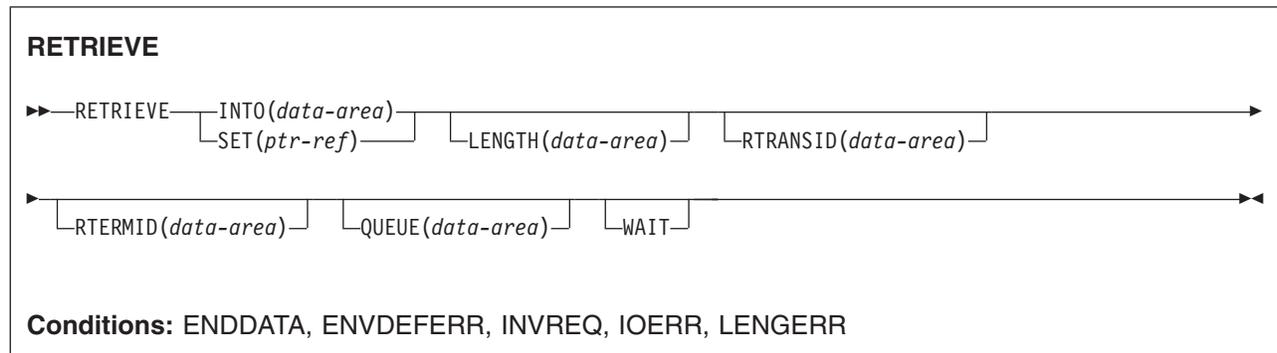
PROCESSERR

RESP2 values:

- 14 The process is in COMPLETE or CANCELLING mode, and therefore cannot be resumed.

RETRIEVE

Retrieve data stored for a task.



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 Application Programming Guide* for more information about transaction affinities.

Description

The RETRIEVE command 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 normally released on execution of the RETRIEVE command, or on termination of the task if no RETRIEVE command is executed prior to termination.

If the START command specified ATTACH, the storage is not released. (ASSIGN STARTCODE in such a task 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. 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'.

Options

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 in CICS commands” on page 8.

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.

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 (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.

Examples

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 REATTACH EVENT

Retrieve the name of an event that caused the current BTS activity to be reattached.

RETRIEVE REATTACH EVENT

▶—RETRIEVE—REATTACH—EVENT(*data-area*)—┐
└EVENTTYPE(*cvda*)—▶

Conditions: END, INVREQ

Description

RETRIEVE REATTACH EVENT:

- Returns the name of the next event in the current BTS activity's reattachment queue.
- If the retrieved event is atomic, resets its fire status to NOTFIRED. (Composite events are not reset by this command, but only when their predicates become false.)

Use this command to find the name of the event that caused the activity to be reattached. In some cases, reattachment could result from the firing of more than one event—if, for example, the activity has previously been suspended, and reattachment events occurred while it was suspended; or if two or more timer events fire simultaneously. The event name or names are placed on the reattachment queue, from where they can be retrieved by issuing one or more RETRIEVE REATTACH EVENT commands.

Each time it is activated, an activity must deal with at least one reattachment event. That is, it must issue at least one RETRIEVE REATTACH EVENT command, and (if this is not done automatically by CICS) reset the fire status of the retrieved event to NOTFIRED—see the *CICS Business Transaction Services* manual. Failure to do so results in the activity completing abnormally, because it has made no progress—it has not reset any reattachment events and is therefore in danger of getting into an unintentional loop.

If there are multiple events on its reattachment queue, an activity can, by issuing multiple RETRIEVE REATTACH EVENT commands, deal with several or all of them in a single activation. Alternatively, it can deal with them singly, by issuing only one RETRIEVE command per activation and returning; it is then reactivated to deal with the next event on its reattachment queue. Which approach you choose is a matter of program design. Bear in mind, if you deal with several reattachment events in the same activation, that a syncpoint does not occur until the activation returns.

Note: The retrieval of a composite event from the reattachment queue does not reset the state of the composite event to NOTFIRED. Thus, if it retrieves a composite reattachment event, the activity program may need to issue one or more RETRIEVE SUBEVENT commands, to retrieve (and reset) the sub-event or sub-events that have fired. This in turn causes the fire status of the composite event to be re-evaluated.

Options

EVENT(data-area)

returns the 16-character name of the event which caused this activity to be reattached.

EVENTTYPE(cvda)

returns the type of the reattachment event. CVDA values are:

ACTIVITY

Activity completion.

COMPOSITE

Composite.

INPUT

Input

SYSTEM

The BTS system event, DFHINITIAL.

TIMER

Timer.

Conditions

END

RESP2 values:

8 There are no more events to retrieve.

INVREQ

RESP2 values:

1 The command was issued outside the scope of an activity.

RETRIEVE SUBEVENT

Retrieve the name of the next sub-event in a BTS composite event's sub-event queue.

RETRIEVE SUBEVENT

▶▶—RETRIEVE—SUBEVENT (*data-area*)—EVENT (*data-value*)—┐
└EVENTTYPE (*cvda*)—▶▶

Conditions: END, EVENTERR, INVREQ

Description

RETRIEVE SUBEVENT:

- Retrieves the name of the next sub-event in a BTS composite event's sub-event queue.
- Resets the retrieved sub-event's fire status to NOTFIRED.
- Causes the composite event's fire status to be re-evaluated.

The firing of a composite event results from the firing of a set of one or more sub-events. The names of sub-events that have fired are placed on the composite event's sub-event queue, from which they can be retrieved, in sequence, by issuing successive RETRIEVE SUBEVENT commands.

You can use this command to discover which sub-event or sub-events caused a composite event to fire.

Note:

1. The presence of events on the sub-event queue does not imply that the composite event has fired. (Some sub-events in the set required to fire the composite event may still be in NOTFIRED state, and not yet on the sub-event queue.) To discover whether a composite event has fired, use the TEST EVENT command.
2. Retrieval is destructive; when the name of a fired sub-event is retrieved, that sub-event cannot be retrieved again.
3. Because it resets the fire status of the sub-event, RETRIEVE SUBEVENT causes the fire status of the composite event to be re-evaluated.

Options

EVENT(*data-value*)

specifies the name (1–16 characters) of the composite event.

EVENTTYPE(*cvda*)

returns the type of the sub-event. CVDA values are:

ACTIVITY

Activity completion.

INPUT Input

TIMER

Timer.

SUBEVENT (data-area)

returns the 16-character name of the sub-event at the head of the sub-event queue.

Conditions**END**

RESP2 values:

- 9** There are no more sub-events to retrieve.
- 10** The composite event contains no sub-events (it is empty).

EVENTERR

RESP2 values:

- 4** The event specified on the EVENT option is not recognized by BTS.

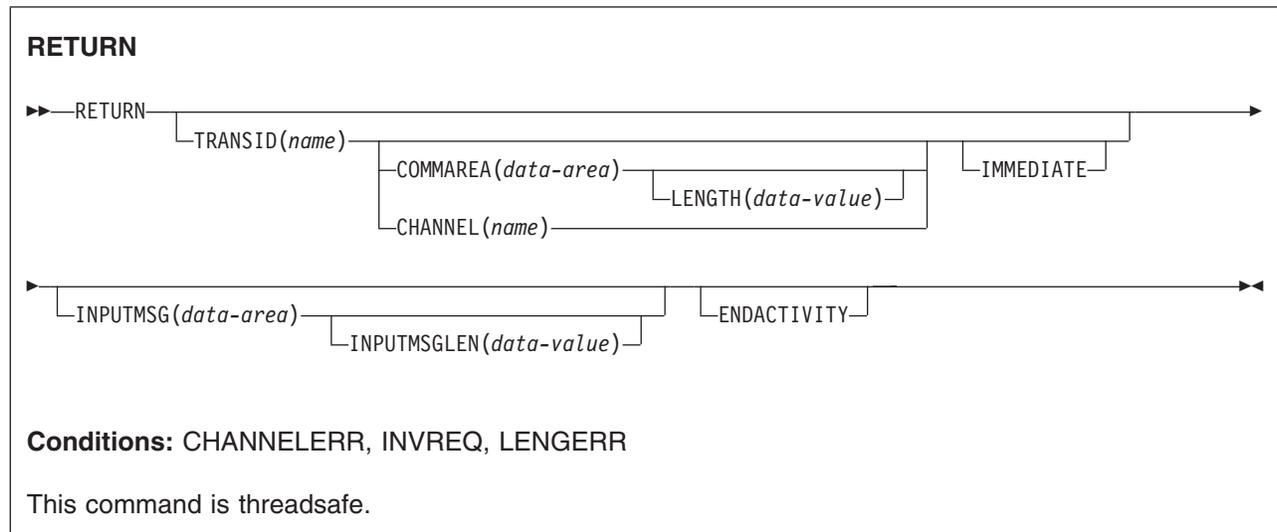
INVREQ

RESP2 values:

- 1** The command was issued outside the scope of an activity.
- 2** The event specified on the EVENT option is invalid. It is not a composite event.

RETURN

Return program control.



Description

RETURN returns control from an application program either to an application program at the next higher logical level, or to CICS.

When returning a communications area (COMMAREA), 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 Application Programming Guide*.

The valid range for the COMMAREA length is 0 through 32 763 bytes. If the length provided is outside this range, the LENGERR condition occurs.

The COMMAREA, IMMEDIATE, and CHANNEL 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.

For information about the use of this command in the CICS BTS environment, see the *CICS Business Transaction Services* manual.

Options

CHANNEL(*name*)

specifies the name (1–16 characters) of a channel that is to be made available to the next program that receives control. The acceptable characters are A-Z

a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

Channel names are always in EBCDIC. The allowable set of characters for channel names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if a channel is to be shipped between regions (that is, if the transaction named on the TRANSID option is remote), the characters used in naming it should be restricted to A-Z a-z 0-9 & : = , ; < > . - and _ .

#

The program that issues the RETURN command may:

- Have created the channel by means of one or more PUT CONTAINER CHANNEL commands
- Specify its current channel, by name
- Name a non-existent channel, in which case a new, empty, channel is created

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.

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 Application Programming Guide* for more information about the CICS COMMAREA.) 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.

ENDACTIVITY

This option is for use by programs that implement CICS business transaction services (BTS) activities. It specifies that the current activity is completing, and is not to be reactivated.

If there are no user events in the activity's event pool, the activity completes normally.

If there are user events (fired or unfired) in the activity's event pool:

- If one or more of the events are activity completion events, the activity abends. Trying to force an activity to complete before it has dealt with one or more of its child activities is a program logic error.
- If none of the events are activity completion events, the events are deleted and the activity completes normally.

For information about BTS in general and the ENDACTIVITY option in particular, see the *CICS Business Transaction Services* manual.

This option is ignored outside the CICS BTS environment.

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 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 in CICS commands” on page 8.

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.

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, INPUTMSG, or CHANNEL 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.

Conditions

CHANNELERR

RESP2 values:

- 1 The name specified on the CHANNEL option contains an illegal character or combination of characters.

INVREQ

RESP2 values:

- 1 A RETURN command with the TRANSID option is issued in a program that is not associated with a terminal.
- 2 A RETURN command with the CHANNEL, COMMAREA, or IMMEDIATE option is issued by a program that is not at the highest logical level.
- 4 A RETURN command with the TRANSID option is issued in a program that is associated with an APPC logical unit.
- 8 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.
- 30 PG domain not initialized. Parameters are not allowed on the EXEC RETURN statement in first stage PLT programs.
- 200 A RETURN command is issued with an INPUTMSG option by a program invoked by DPL.
- 203 The CHANNEL option was specified but the remote region to which control is returned does not support channels.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 11 The COMMAREA length is less than 0 or greater than 32763.
- 26 The COMMAREA ADDRESS passed was zero, but the commarea length was non-zero.
- 27 The INPUTMSG LENGTH was less than 0 or greater than 32767.

Default action: terminate the task abnormally.

REWIND COUNTER and REWIND DCOUNTER

Rewind a named counter that has reached its limit (that is, the maximum number has been assigned).

REWIND COUNTER

►► REWIND COUNTER(*name*) [POOL(*name*)] [INCREMENT(*data-value*)] ►►

Conditions: INVREQ, SUPPRESSED

REWIND DCOUNTER

►► REWIND DCOUNTER(*name*) [POOL(*name*)] [INCREMENT(*data-area*)] ►►

Conditions: INVREQ, SUPPRESSED

Description

These counter commands reset the current value of the named counter to its defined minimum number. COUNTER operates on fullword counters and DCOUNTER operates on doubleword counters.

For information about specifying fullword and doubleword variables on these named counter commands, see “CICS command argument values” on page 3.

Options

COUNTER(*name*)

specifies the name of the named counter that is to be reset to its minimum value. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

DCOUNTER(*name*)

specifies the name of the named counter that is to be reset to its minimum value. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

INCREMENT(*data-value*)

specifies, as a fullword signed binary value (doubleword unsigned binary value for DCOUNTER), an increment to be used in determining whether the named counter is in a valid state to be reset. If a previous GET command (which did not specify the REDUCE option) specified an increment that caused the GET

command to fail, specify the same increment on the REWIND. The named counter server applies the increment before testing whether the counter is in a counter-at-limit condition.

See the the INCREMENT option on the GET command for more details.

POOL(*poolname*)

specifies an 8-character string to be used as a pool selection parameter to select the pool in which the named counter resides. The string can be a logical pool name, or the actual pool name.

Valid characters for the pool selector string are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 8 characters, the name must be padded with trailing blanks.

This parameter is optional. If you omit the name of the pool, a pool selector value of 8 blanks is assumed.

If there is no matching entry in the DFHNCOPT options table, CICS uses the name specified on the NCPLDFT system initialization parameter, which specifies the default named counter pool.

For information about generating a named counter options table using the DFHNCO macro, see the *CICS Application Programming Guide*.

Conditions

INVREQ

RESP2 values:

- 201** Named counter not found.
- 301** The server has reported an error code that is not understood by the named counter interface. Generally, this is not possible unless the interface load module, DFHNCIF, is at a lower maintenance or release level than the server itself.
- 303** An unexpected error, such as structure failure or loss of connectivity, has occurred on a macro used to access the coupling facility. Further information can be found in message DFHNC0441 in the CICS job log.
- 304** The pool selection parameter specified in the program cannot be resolved to a valid server name using the current options table.
- 305** The interface is unable to establish a connection to the server for the selected named counter pool. Further information can be found in an AXM services message (AXMSC $nnnn$) in the CICS job log.
- 306** An abend occurred during server processing of a request. Further information can be found in a message in the CICS job log and the server job log.
- 308** The DFHNCOPT options table module, required for resolving a pool name, could not be loaded.
- 309** During processing of the options table, the named counter interface encountered an unknown entry format. Either the options table is not correctly generated, or the DFHNCIF interface load module is not at the same release level as the options table.
- 310** An options table entry matching the given pool name specified a user exit program, but the user exit program is not link-edited with the options table and cannot be loaded.

- 311** A response from the named counter server to the client region interface module, DFHNCIF, indicates that a system-managed rebuild is in progress but the EXEC CICS interface does not recognize the condition. This means that the CICS region is at CICS TS 2.1 or earlier.
- 403** The POOL parameter contains invalid characters or embedded spaces.
- 404** The COUNTER parameter contains invalid characters or embedded spaces.
- 406** The INCREMENT value is invalid. The value specified cannot be greater than the total range of the counter ((maximum value - minimum value) + 1).

Default action: terminate the task abnormally.

SUPPRESSED

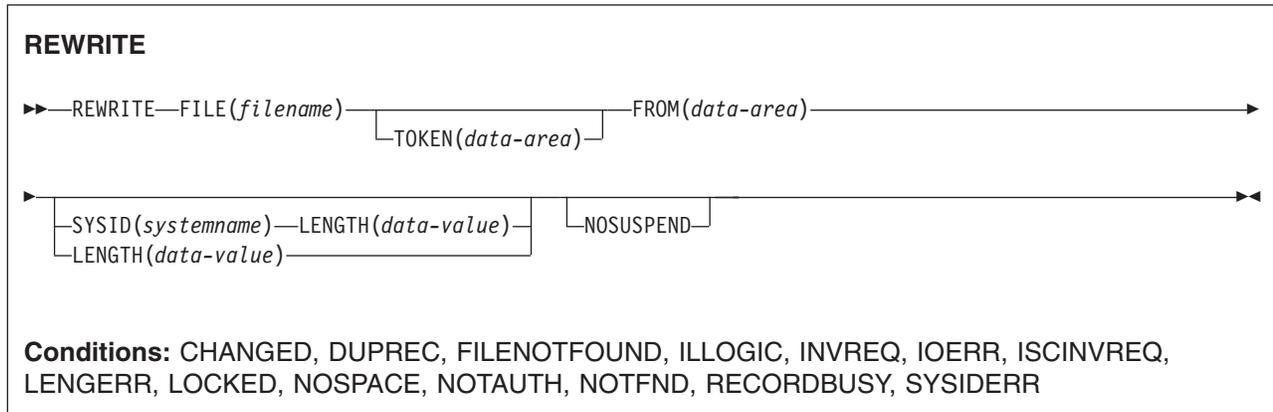
RESP2 values:

- 102** The named counter has not yet reached its limit (that is, the current value is not equal to the maximum value plus 1, giving the counter-at-limit condition). This error condition is returned if the named counter is not at its limit even after applying any specified increment.

Default action: terminate the task abnormally.

REWRITE

Update a record in a file.



Description

REWRITE updates a record in a file on a local or a remote system. You must always precede this command with a read with the UPDATE option.

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.

When this command is used to update records in a coupling facility data table, the update is made only to the data table in the coupling facility.

Note: The only VSAM data sets greater than 4GB supported by CICS are KSDS, and then only if they are accessed by key. CICS does not support ESDS or RRDS data sets defined with the extended attribute.

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 length, as a halfword binary value, of the data area where the record is written from.

This option must be specified if SYSID is specified.

If the file is on a remote system, the LENGTH parameter need not be set here but must be set in the file resource definition.

If the file is on a local system, the LENGTH parameter must be set for variable-length records, using the INTO option, but not for fixed-length records. It is, however, advisable to specify the length for fixed-length records because it causes a check to be made that the record being written is not longer than that defined for the data set.

NOSUSPEND (RLS on1y)

The request does not wait if VSAM is holding an active lock against the record, including records locked as the result of a DEADLOCK.

Generally, you should not need this option because, for example, when re-writing a record to a base-only data set, the active lock was acquired when the task issued the READ UPDATE.

However, lock contention can occur if the update involves changes made in RLS mode to records in a VSAM data set that has one or more alternate indexes, and an alternate index is defined with unique keys.

SYSID(*systemname*)

specifies the name of the system to which the request is directed.

TOKEN(*data-area*)

specifies as a fullword binary value a unique request identifier for a REWRITE, used to associate it with a previous READ, READNEXT, or READPREV command that specified UPDATE.

TOKEN can be function shipped. However, if a request specifying TOKEN is function shipped to a member of the CICS family of products that does not support the TOKEN option, the request fails:

Conditions**CHANGED**

RESP2 values:

- 109** A REWRITE command is issued for a file that is defined as a coupling facility data table using the contention update model and the record has been changed since the application program read it for update. To successfully update the record, repeat the read for update to get the latest version of the record, re-apply the change, and try the rewrite again.

Default action: terminate the task abnormally.

DUPREC

RESP2 values:

- 150** 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.

Default action: terminate the task abnormally.

FILENOTFOUND

RESP2 values:

1 A file name referred to in the FILE option cannot be found in the FCT.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values: (VSAM)

110 A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the EXEC interface block; refer to “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

30 A REWRITE command is issued without a token and no previous READ for UPDATE (also without a token) can be found.

A possible reason for the previous READ for UPDATE not being found is that it failed for some reason, and the failure has not been correctly handled or has been ignored.

46 A REWRITE command has attempted to change the length of a BDAM variable length record or block.

47 A REWRITE instruction includes a token whose value cannot be matched against any token in use for an existing read for UPDATE request.

55 NOSUSPEND is not allowed because the file is not a VSAM file that is accessed in RLS mode.

56 An attempt to update a recoverable coupling facility data table has failed because the current unit of work has already updated 1024 recoverable coupling facility data tables. You cannot update more than 1024 recoverable coupling facility data tables within a unit of work

Default action: terminate the task abnormally.

IOERR

RESP2 values:

120 An I/O error occurred during the file control operation. 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 “EXEC interface block” on page 745 for details.)

For a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

Default action: terminate the task abnormally.

ISCVREQ

RESP2 values:

70 The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 10 The LENGTH option is not specified for a file with variable-length records, or for a BDAM file with undefined format records.
- 12 The length specified exceeds the maximum record size (of the source data set for a data-table); the record is truncated.
- 14 An incorrect length is specified for a file with fixed-length records.

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

- 80 For user-maintained data tables, this condition occurs when an attempt to REWRITE a record 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. This may be caused by a logic error in the application program.

This condition can also occur when a REWRITE command is issued to a coupling facility data table using the contention model and the record has been deleted since it was read for update.

Default action: terminate the task abnormally.

LOCKED

RESP2 values:

- 106 Attempt has been made to update a record, but a retained lock exists against a unique alternate key that is involved in the request.

Default action: abend the task with code AEX8.

NOSPACE

RESP2 values:

- 100 No space is available on the direct access device for adding the updated record to the data set.
- 102 The maximum number of records specified for a recoverable coupling facility data table has been exceeded. This can occur on a rewrite operation because an extra record is required in the coupling facility data table for recovery purposes until the update has been committed.
- 103 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.
- 108 For coupling facility data tables, this condition occurs if there is insufficient space in the coupling facility data table pool to store the updated record.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

- 101 A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

RECORDBUSY

RESP2 values:

- 107 NOSUSPEND is specified but VSAM holds an active lock against a

unique alternate index key that is involved in the request, which would cause the request to wait. (See the note about active and retained locks below.)

Default action: abend the task with code AEX9.

SYSIDERR

RESP2 values:

- 130** 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.
- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The REWRITE is issued against a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

Note: RECORDBUSY refers to active locks and LOCKED refers to retained locks:

- REWRITE requests for records that have *retained* locks are always rejected with a LOCKED response.
- REWRITE requests for records that have *active* locks wait for the lock to be released, except when the NOSUSPEND keyword is specified, in which case CICS returns the RECORDBUSY response.

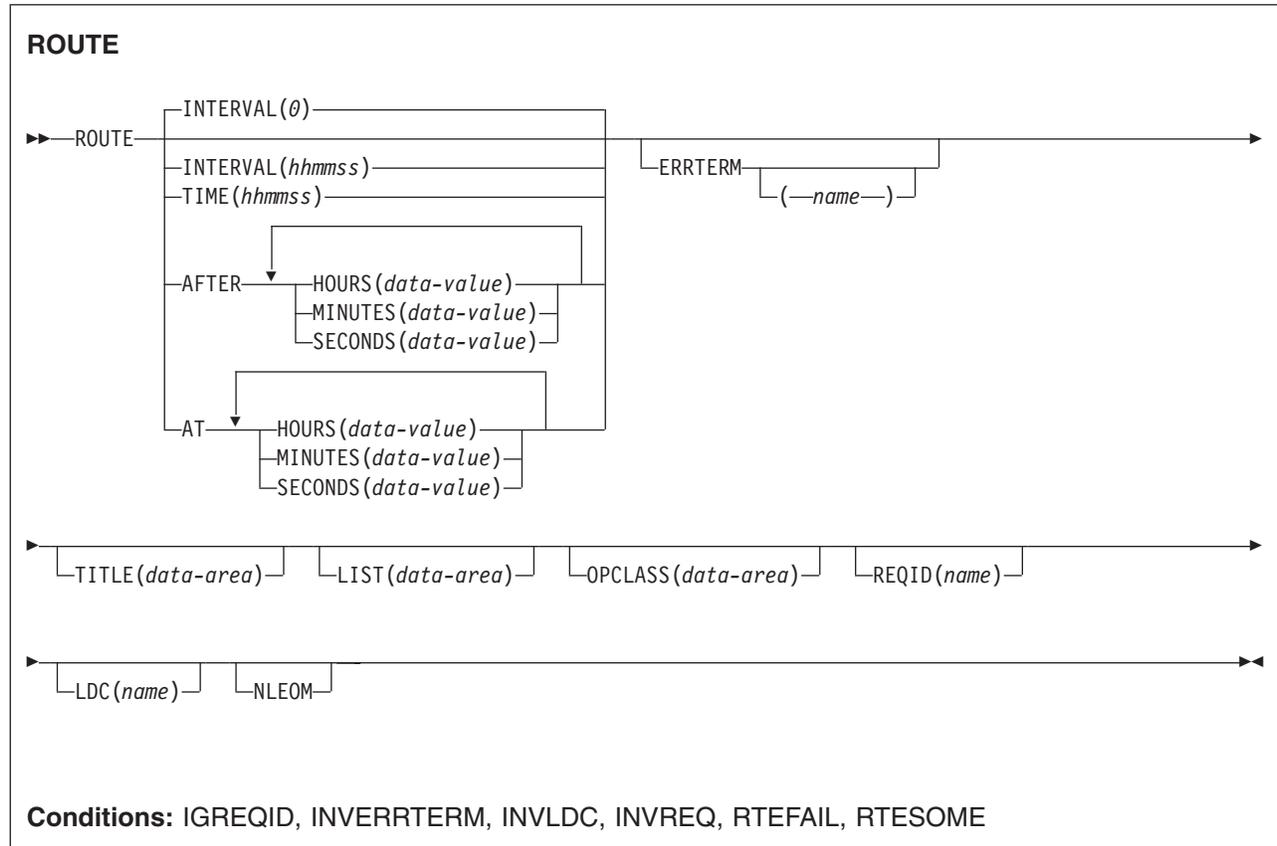
Examples

For example:

```
EXEC CICS REWRITE  
      FROM(RECORD)  
      FILE('MASTER')  
      TOKEN(APTOK)
```

ROUTE

Route a BMS message. (This command is only available on full BMS. The *CICS Application Programming Guide* has further information about BMS.)



Description

ROUTE routes a BMS logical message to one or more terminals or terminal operators.

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

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).

- As one of HOURS(0–99), MINUTES(0–5999), or SECONDS(0–359 999). 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.

This option is effective only if PRGDLAY has been specified in the system initialization parameters.

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.

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 defined by the DFHTCT TYPE=LDC macro.

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.

If the LDC option is omitted, the LDC mnemonic specified in DFHMSD is used; see “DFHMSD” on page 807 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 by the RDO TYPETERM options PAGESIZE or ALTPAGE, 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 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 for that terminal, are unavailable for data. This may cause data to wrap around in the buffer; if this occurs, the PAGESIZE or ALTPAGE 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 Application Programming Guide* for the format of the title option.

Conditions

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 (**).

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

RESP2 values:

- 4** Hours out of range
- 5** Minutes out of range
- 6** Seconds out of range
- 200** BMS commands are not supported for distributed program link.

also occurs (RESP2 not set) in the following situations:

- Bytes 10 through 15 of a route list entry do not contain blanks.

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.

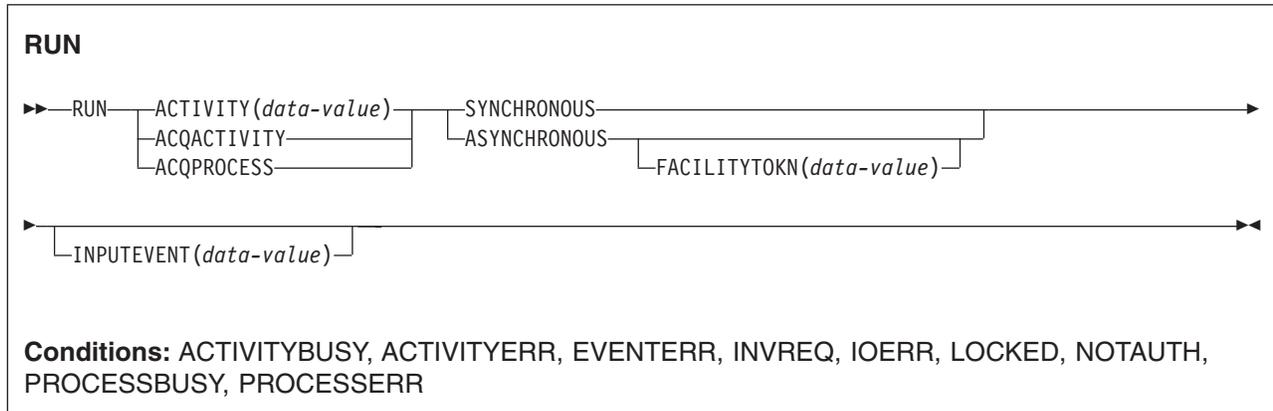
RTESOME

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.

RUN

Execute a CICS business transaction services process or activity synchronously or asynchronously, with context-switching.



Description

RUN executes a CICS business transaction services process or activity synchronously or asynchronously with the requestor, with context-switching. The process or activity must previously have been defined to BTS.

RUN causes BTS to attach the process or activity, by sending it an input event. If the process or activity is in its initial state—that is, if this is the first time it is to be run, or if the activity has been reset by a RESET ACTIVITY command—CICS sends it the DFHINITIAL system event. If the process or activity is dormant—that is, waiting for a reattachment event to occur—the input event must be specified on the INPUTEVENT option.

If the process or activity is in any mode other than INITIAL or DORMANT, it cannot be run.

The SYNCHRONOUS and ASYNCHRONOUS options allow you to specify whether the process or activity should be executed synchronously or asynchronously with the requestor.

Context-switching

When a process or activity is activated by a RUN command, it is run:

- In a separate unit of work from the requestor.
- With the transaction attributes (TRANSID and USERID) specified on the DEFINE PROCESS or DEFINE ACTIVITY command.

In other words, a **context-switch** takes place. The relationship of the process or activity to the requestor is as between separate transactions, except that:

- Data can be passed between the two units of work
- The start and finish of the activity is related to the requestor's syncpoints.

To run a process or activity *without* context-switching—that is, in the same UOW and with the same TRANSID and USERID attributes as the requesting

transaction—use the LINK ACQPROCESS, LINK ACQACTIVITY, or LINK ACTIVITY command. This is possible only if the process or activity is run synchronously.

If the ability to isolate a failure is more important than performance, use RUN SYNCHRONOUS rather than LINK.

Activities

The only activities a program can run are as follows:

- If it is running as the activation of an activity, its own child activities. It can run several of its child activities within the same unit of work.
- The activity it has acquired, by means of an ACQUIRE ACTIVITYID command, in the current unit of work.

To check the response from the activity, the CHECK ACTIVITY command must be used. This is because the response to the request to run the activity does not contain any information about the success or failure of the activity itself—only about the success or failure of the request to run it.

Typically, if the activity is run synchronously, the CHECK command is issued immediately after the RUN command. If it is run asynchronously, the CHECK command could be issued:

- When the activity's parent is reattached due to the firing of the activity's completion event
- When the requestor is reattached due to the expiry of a timer.

The activity's completion event is one of the following:

1. The event named on the EVENT option of the DEFINE command for the activity.
2. If the DEFINE command did not specify a completion event, an event of the same name as the activity.

To retry an activity:

1. Issue a RESET ACTIVITY command to reset the activity to its initial state.
2. Issue a RUN command.

Processes

The only process that a program can run is the one that it has acquired in the current unit of work—see the *CICS Business Transaction Services* manual.

To check the response from the process, the CHECK ACQPROCESS command must be used. This is because the response to the request to run the process does not contain any information about the success or failure of the process itself—only about the success or failure of the request to run it.

Typically, if the process is run synchronously, the CHECK command is issued immediately after the RUN command. If the process is run asynchronously, the CHECK command could be issued when the requestor is reattached due to the expiry of a timer.

Options

ACQACTIVITY

specifies that the activity to be run is the one that the current unit of work has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the process currently acquired by the requestor is to be run.

ACTIVITY(data-value)

specifies the name (1–16 characters) of the activity to be run. The name must be that of a child of the current activity.

ASYNCHRONOUS

specifies that the process or activity is to be executed asynchronously with the requestor.

FACILITYTKN(data-value)

specifies an 8-byte bridge facility token.

This option applies when a BTS client activity runs a 3270-based pseudoconversational transaction. To ensure that the existing bridge facility is reused for the next transaction in the pseudoconversation, the client passes its token to the next child activity. This is explained in more detail in the *CICS Business Transaction Services* manual.

INPUTEVENT(data-value)

specifies the name (1–16 characters) of the event that causes the process or activity to be attached.

You *must not* specify this option if the process or activity is in its initial state; that is, if this is the first time it is to be run, or if the activity has been reset by a RESET ACTIVITY command. In this case, CICS sends the process or activity the DFHINITIAL system event.

You *must* specify this option if the process or activity is not in its initial state; that is, if it has been activated before, and has not been reset by a RESET ACTIVITY command.

If you specify INPUTEVENT, for the RUN command to be successful the process or activity to be attached must have defined the named event as an input event.

If you issue multiple asynchronous RUN commands against the same activity within the same unit of work:

- If you specify *the same input event*, each RUN command after the first fails.
- If you specify *different input events*, the activity may or may not be invoked as many times as the number of RUN requests—the only guarantee is that it will be invoked at least once. For example, if , within the same unit of work, you issue five asynchronous RUN requests for the same activity, specifying different input events, the activity might be invoked twice. At the first invocation, three input events might be presented, and at the second two.

SYNCHRONOUS

specifies that the process or activity is to be executed synchronously with the requestor.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19 The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8 The activity named on the ACTIVITY option could not be found.
- 14 The activity to be run is not in INITIAL or DORMANT mode.
- 27 The activity named on the RUN SYNCHRONOUS command has abended.

EVENTERR

RESP2 values:

- 7 The event named on the INPUTEVENT option has not been defined by the activity or process to be run as an input event; or its fire status is FIRED.

INVREQ

RESP2 values:

- 4 The ACTIVITY option was used to name a child activity, but the command was issued outside the scope of a currently-active activity.
- 15 The task that issued the RUN ACQPROCESS command has not defined or acquired a process.
- 20 The SYNCHRONOUS option was used, but the activity to be run is suspended.
- 24 The ACQACTIVITY option was used, but the unit of work that issued the request has not acquired an activity.
- 28 CICS could not attach the transaction associated with the process or activity to be run. (This response occurs only on RUN SYNCHRONOUS commands.)
- 32 The SYNCHRONOUS option was used, but the transaction associated with the process or activity to be run is defined as remote. You cannot run a process or activity synchronously if its transaction is remote.
- 40 The program that implements the process or activity to be run is remote.

IOERR

RESP2 values:

- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

NOTAUTH

RESP2 values:

- 101 The user associated with the issuing task is not authorized to run the process or activity.

PROCESSBUSY

RESP2 values:

- 13** The request timed out. It may be that another task using this process-record has been prevented from ending.

PROCESSERR

RESP2 values:

- 6** You cannot run the current process.
- 9** The process-type could not be found.
- 14** The process to be run is not in INITIAL or DORMANT mode.
- 27** The process named on the RUN SYNCHRONOUS command has abended.

SEND (VTAM default)

Write data to a standard CICS supported terminal.

SEND (VTAM default)

▶▶ SEND FROM(*data-area*) — LENGTH(*data-value*) —
└─ FLENGTH(*data-value*) ┘ └─ WAIT ┘

Conditions: INVREQ, LENGERR, NOTALLOC

Description

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)

Send data on an APPC mapped conversation.

SEND (APPC)

▶▶ SEND — CONVID(*name*) — FROM(*data-area*) — LENGTH(*data-value*) —
└─ FLENGTH(*data-value*) ┘ └─ INVITE ┘ └─ CONFIRM ┘
└─ LAST ┘ └─ WAIT ┘

▶ └─ STATE(*cvda*) ┘

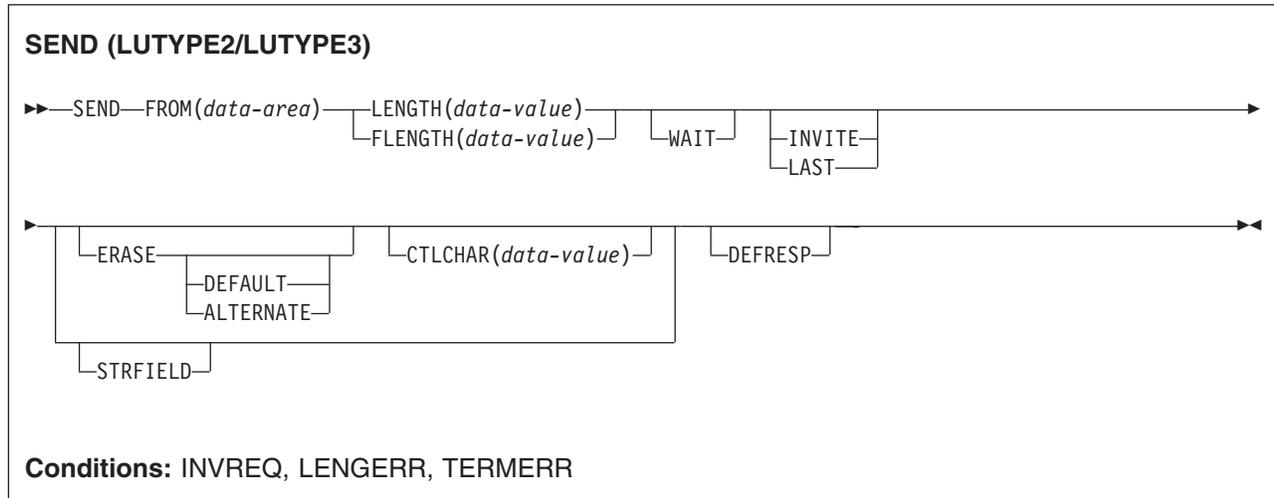
Conditions: INVREQ, LENGERR, NOTALLOC, SIGNAL, TERMERR

Description

SEND sends data to a conversation partner on an APPC mapped conversation.

SEND (LUTYPE2/LUTYPE3)

Write data to a 3270-display logical unit (LUTYPE2) or a 3270-printer logical unit (LUTYPE3).

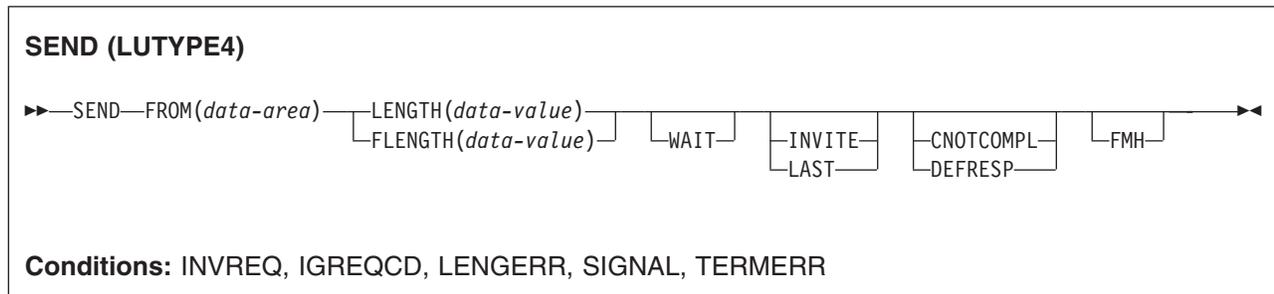


Description

SEND writes data to a terminal.

SEND (LUTYPE4)

Write data to a LUTYPE4 logical unit.

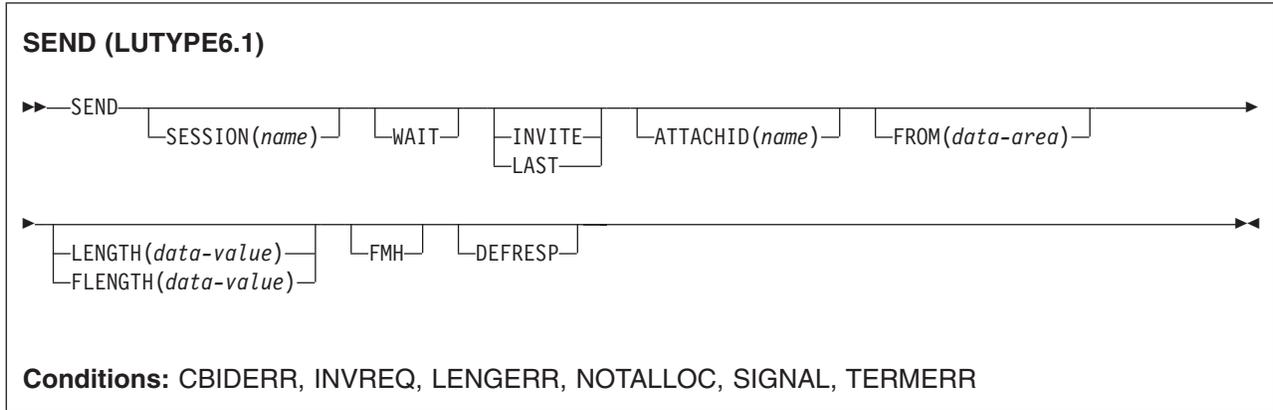


Description

SEND writes data to a terminal.

SEND (LUTYPE6.1)

Send data on an LUTYPE6.1 conversation.

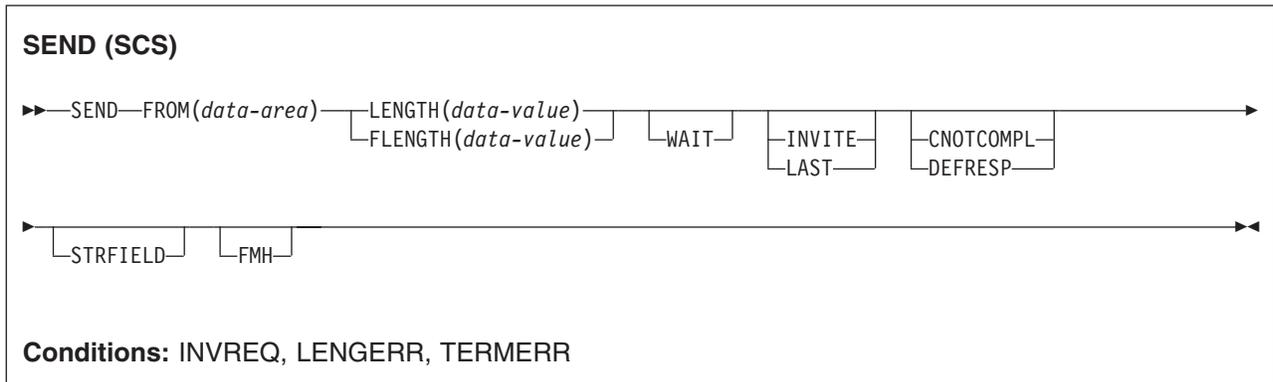


Description

SEND sends data to a conversation partner on an LUTYPE6.1 conversation.

SEND (SCS)

Write data to a 3270 SCS printer logical unit.

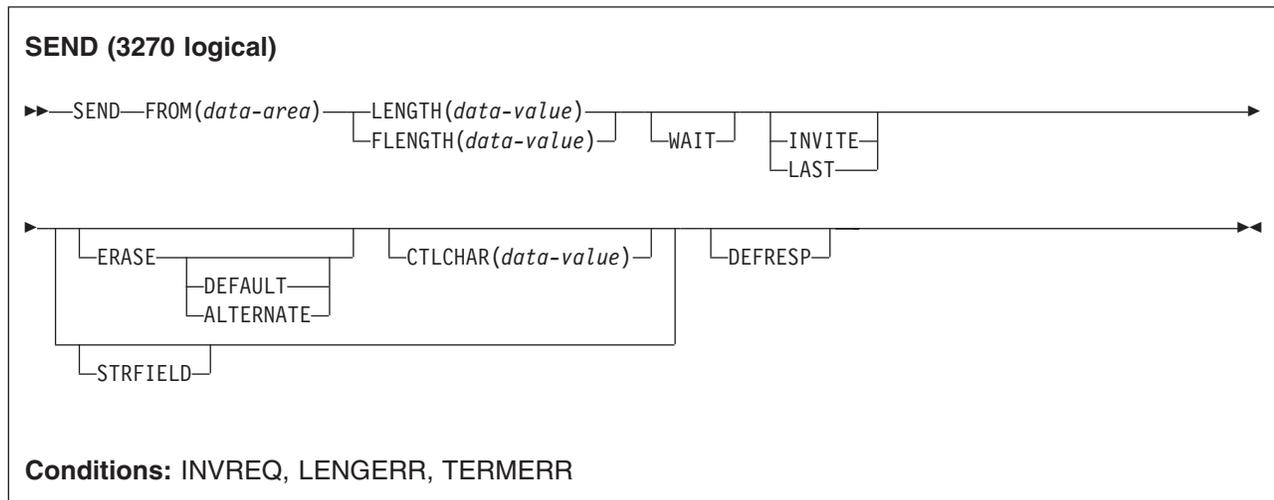


Description

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)

Write data to a 3270 logical unit.

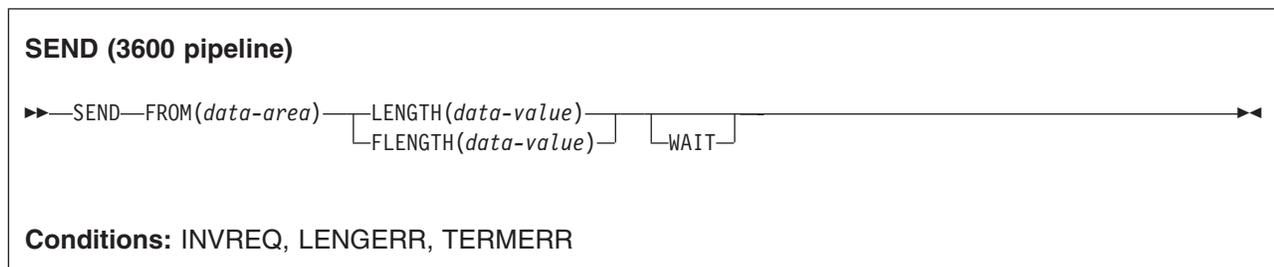


Description

SEND writes data to a terminal.

SEND (3600 pipeline)

Write data to a 3600 pipeline logical unit.

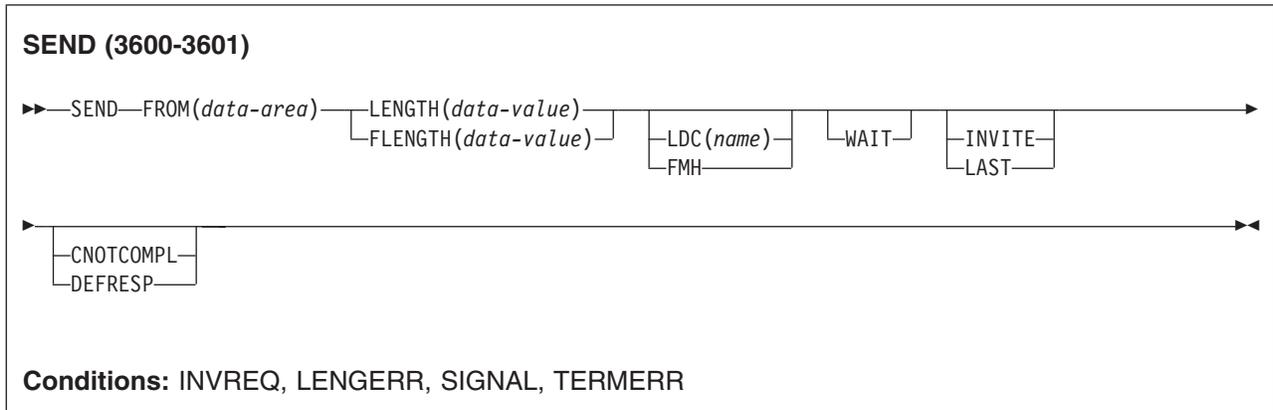


Description

SEND writes data to a terminal.

SEND (3600-3601)

Write data to a 3600 (3601) logical unit.



Description

SEND writes data to a terminal. This form of SEND also applies to the 4770 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.

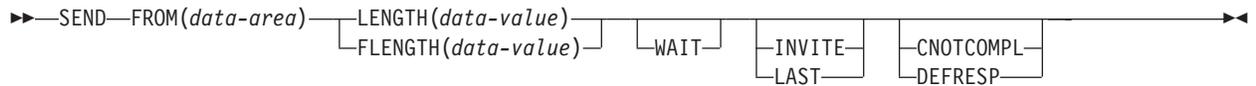
When the LDC option is specified, the numeric value associated with the mnemonic for the particular TCTTE, is inserted in the FMH. The numeric value associated with the LDC mnemonic is chosen by the installation, and is interpreted by the 3601 application program.

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)

Write data to a 3600 (3614) logical unit.

SEND (3600-3614)



Conditions: INVREQ, LENGERR, TERMERR

Description

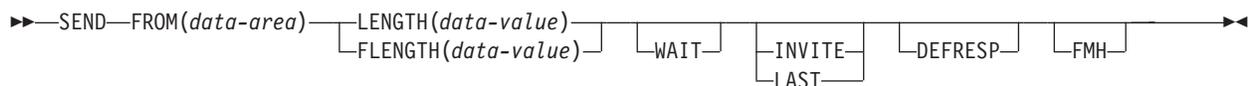
SEND writes data to a terminal. The data stream and communication format used between a CICS application program and a 3614 is determined by the 3614. The application program is therefore device dependent when handling 3614 communication.

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

SEND (3650 interpreter)

Write data to a 3650 interpreter logical unit.

SEND (3650 interpreter)



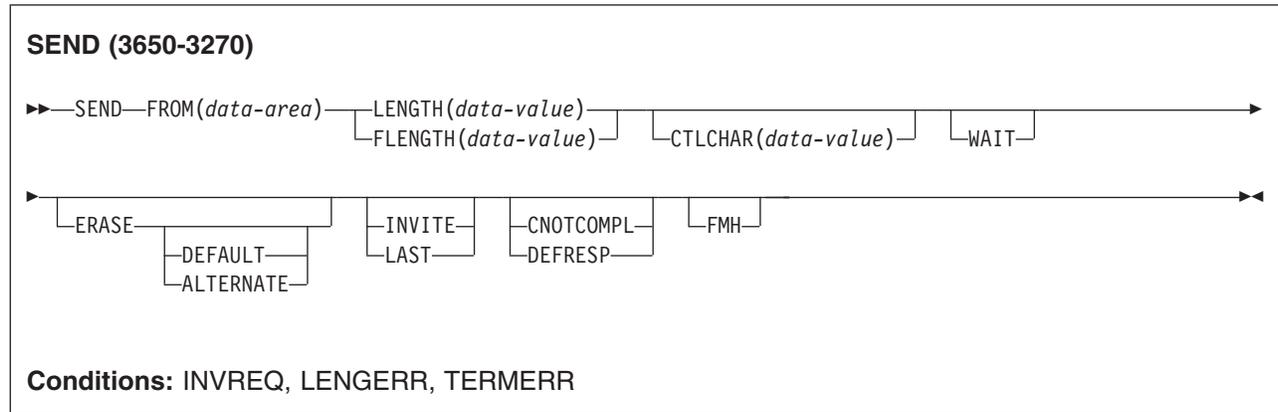
Conditions: INVREQ, LENGERR, TERMERR

Description

SEND writes data to a terminal.

SEND (3650-3270)

Write data to a 3650 logical unit.

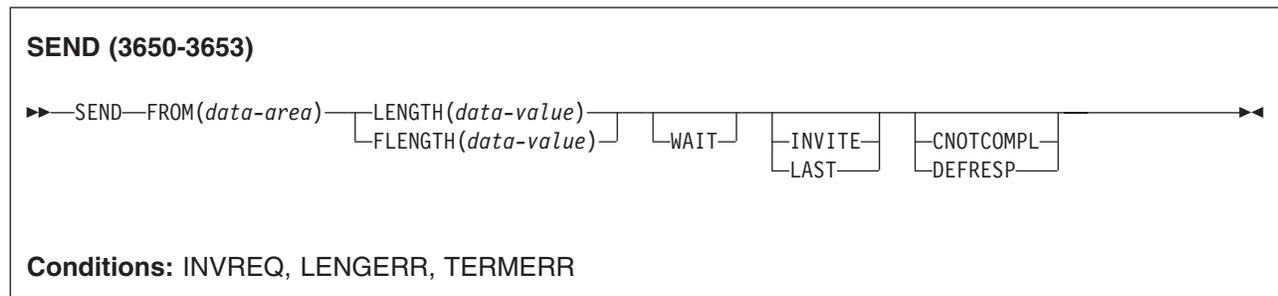


Description

SEND writes data to a terminal.

SEND (3650-3653)

Write data to a 3650 (3653) logical unit.



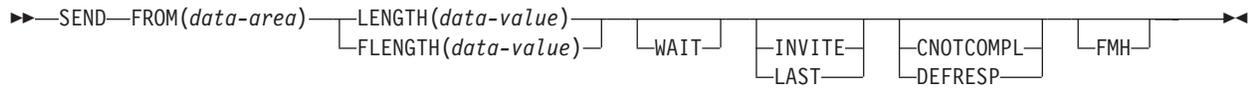
Description

SEND writes data to a terminal.

SEND (3650-3680)

Write data to a 3650 (3680) logical unit.

SEND (3650-3680)



Conditions: INVREQ, LENGERR, TERMERR

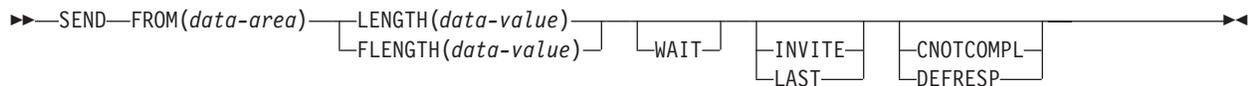
Description

SEND writes data to a terminal.

SEND (3767)

Write data to a 3767 interactive logical unit.

SEND (3767)



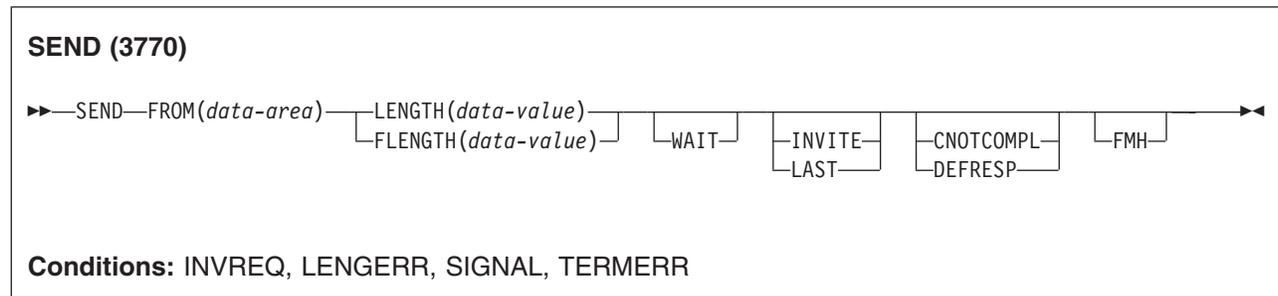
Conditions: INVREQ, LENGERR, SIGNAL, TERMERR

Description

SEND writes data to a terminal. This form of SEND also applies to the 3770 interactive logical unit.

SEND (3770)

Write data to a 3770 batch logical unit.

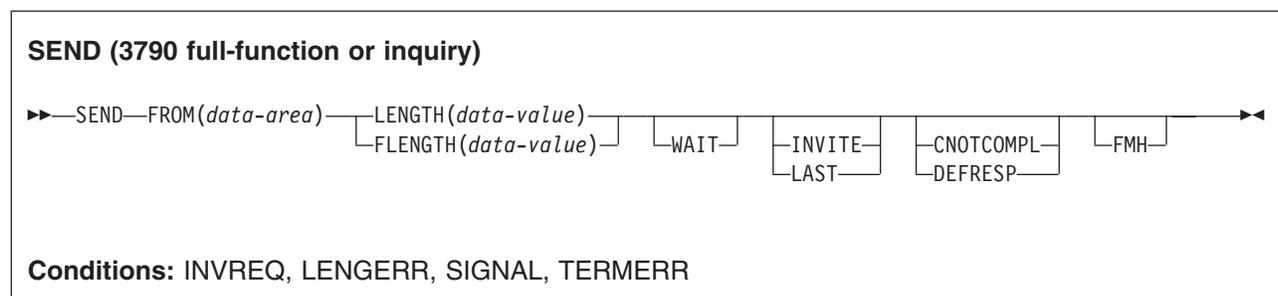


Description

SEND writes data to a terminal.

SEND (3790 full-function or inquiry)

Write data to a 3790 full-function or inquiry logical unit.



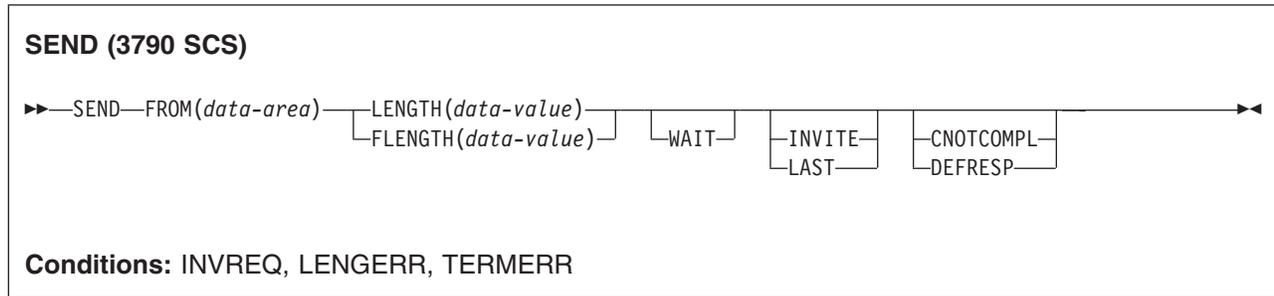
Description

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)

Write data to a 3790 SCS printer logical unit.

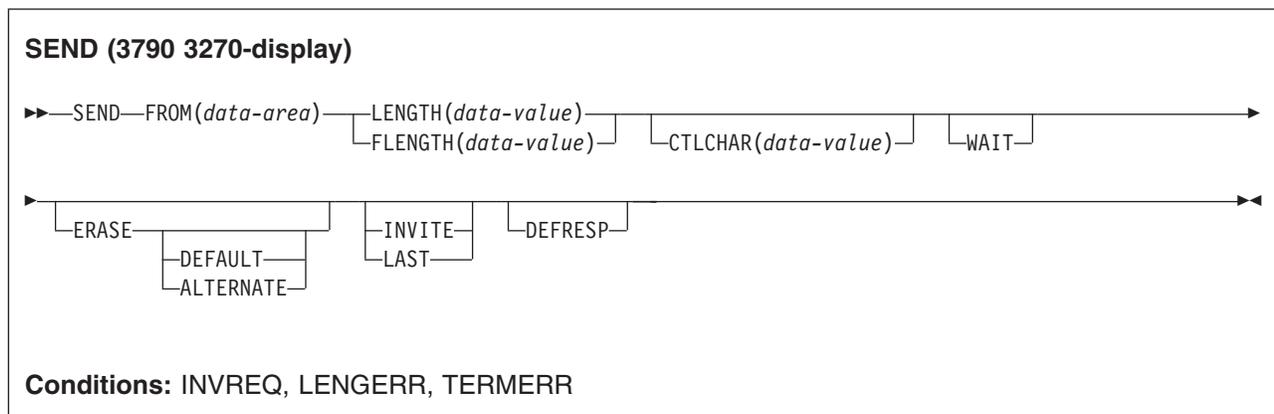


Description

SEND writes data to a terminal.

SEND (3790 3270-display)

Write data to a 3790 (3270-display) logical unit.

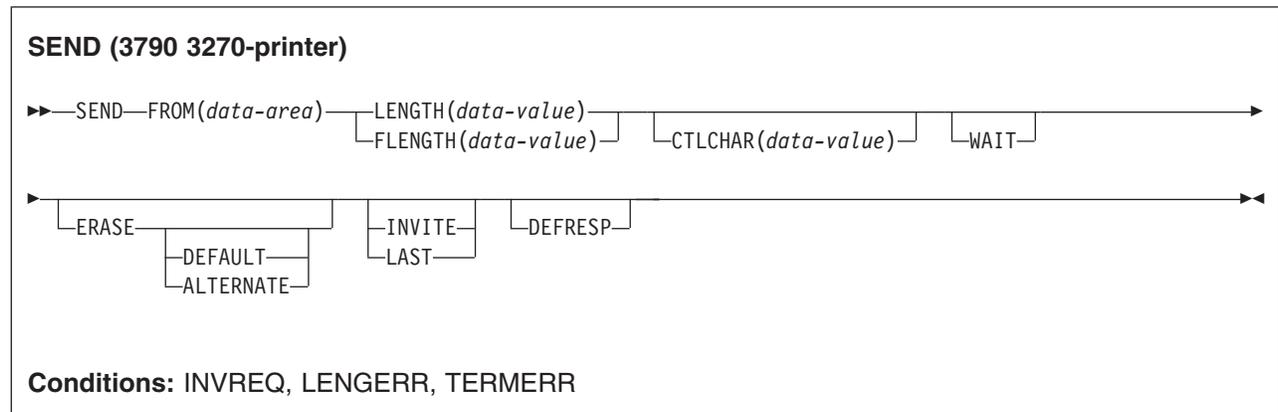


Description

SEND writes data to a terminal.

SEND (3790 3270-printer)

Write data to a 3790 (3270-printer) logical unit.



Description

SEND writes data to a terminal.

SEND: VTAM options

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. These are documented in the *IBM 3270 Data Stream Programmer's Reference*. 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.

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 RDO PROFILE resource definition.

FLENGTH(*data-value*)

An alternative to LENGTH. For architectural reasons, this option is limited to a maximum of 32K for all terminal-related SEND commands.

#

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 logical unit, or a partner transaction.

INVITE

For the SEND (APPC) command, 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.

For the other SEND commands, INVITE specifies that the next terminal control command 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.

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 in CICS commands” on page 8.

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 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.

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

RESP2 values:

200 A distributed program link server application attempted to send on its function-shipping session (its principal facility).

For SEND (APPC), a distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

also occurs (RESP2 not set) 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.

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.

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 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 default)

Write data to standard CICS terminal support.

SEND (non-VTAM default)

►► SEND FROM(*data-area*) — LENGTH(*data-value*) — FLENGTH(*data-value*) — DEST(*name*) — WAIT —

Conditions: INVREQ, LENGERR, NOTALLOC

Description

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 (MRO)

Send data on an MRO conversation.

SEND (MRO)

►► SEND — SESSION(*name*) — WAIT — INVITE — ATTACHID(*name*) — FROM(*data-area*) — LAST —

► — LENGTH(*data-value*) — FLENGTH(*data-value*) — FMH — DEFRESP — STATE(*cvda*) —

Conditions: CBIDERR, INVREQ, LENGERR, NOTALLOC, TERMERR

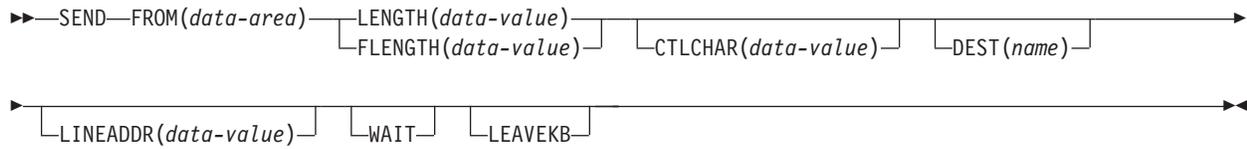
Description

SEND sends data to a conversation partner on an MRO conversation.

SEND (2260)

Write data to a 2260 or 2265 display station.

SEND (2260)



Conditions: INVREQ, LENGERR

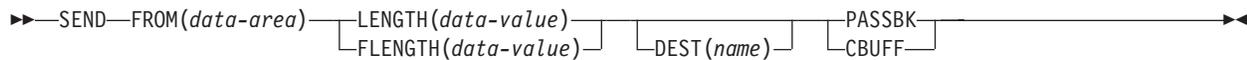
Description

SEND writes data to a terminal.

SEND (2980)

Write data to a 2980 general banking terminal system.

SEND (2980)



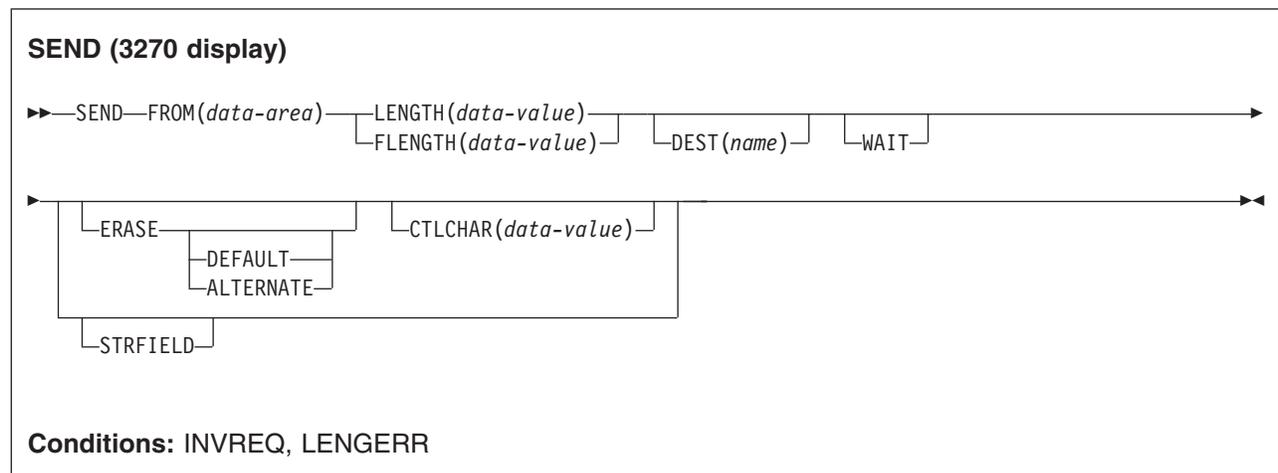
Conditions: INVREQ, LENGERR, NOPASSBKWR

Description

SEND writes data to a terminal. For more information about the 2980 general banking system, see "RECEIVE (2980)" on page 444.

SEND (3270 display)

Write data to a 3270 information display system (TCAM).



Description

SEND writes data to a terminal.

SEND: non-VTAM options

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. These are documented in the *IBM 3270 Data Stream Programmer's Reference*. 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.

Note: In CICS TS 3.1, local TCAM terminals are not supported. The only TCAM terminals supported are remote terminals connected to a pre-CICS TS 3.1 terminal-owning region by the DCB (not ACB) interface of 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.

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 RDO PROFILE resource 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 in CICS commands” on page 8.

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.

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 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.

Conditions

CBIDERR

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

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

200 occurs if a distributed program link server application attempted to send on its function-shipping session (its principal facility)

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.

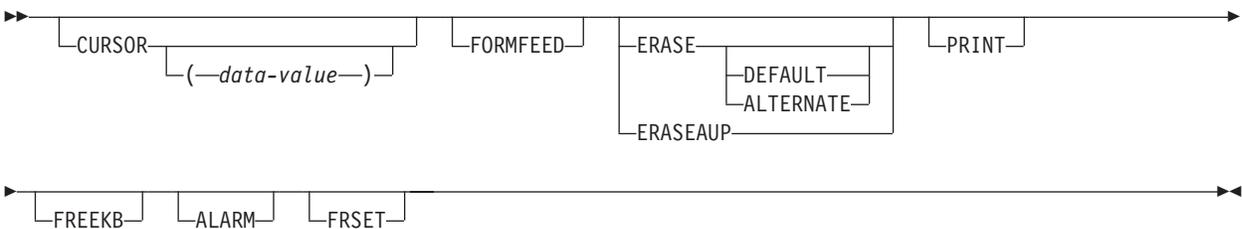
SEND CONTROL

Send 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 Application Programming Guide*.

SEND CONTROL

►► SEND CONTROL ◀◀

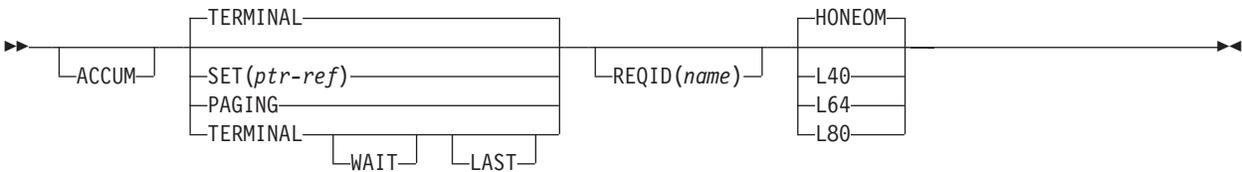
SEND CONTROL Minimum BMS



SEND CONTROL Standard BMS



SEND CONTROL Full BMS



Conditions: IGREQCD, IGREQID, INVLDC, INVPARTN, INVREQ, RETPAGE, TSOERR, WRBRK

Description

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 DFHMDI options, CTRL for a description of the option priority.

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 the 3270 or 3604 cursor is returned to 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 RDO 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 TYPETERM option FORMFEED was not used).

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 TYPETERM options PAGESIZE or ALTPAGE.

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 defined by a DFHTCT TYPE=LDC macro. 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 TYPETERM options PAGESIZE or ALTPAGE.

When using the options, refer to DFHMDDI options, CTRL 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 DFHMMSRCA to assist in setting this 4-byte area. See "Magnetic slot reader (MSR) control value constants, DFHMMSRCA" on page 782 for a complete list. This option is ignored if the RDO TYPETERM option MSRCONTROL was not used.

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 807) or the DFHMDI (see “DFHMDI” on page 798) 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 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.

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

RESP2 values:

200 A distributed program link server application attempted to send on its function-shipping session (its principal facility).

also occurs (RESP2 not set) in the following situation:

- 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.

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 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 Application Programming Guide*.

SEND MAP

►► SEND MAP (*name*) ◀◀

SEND MAP Minimum BMS

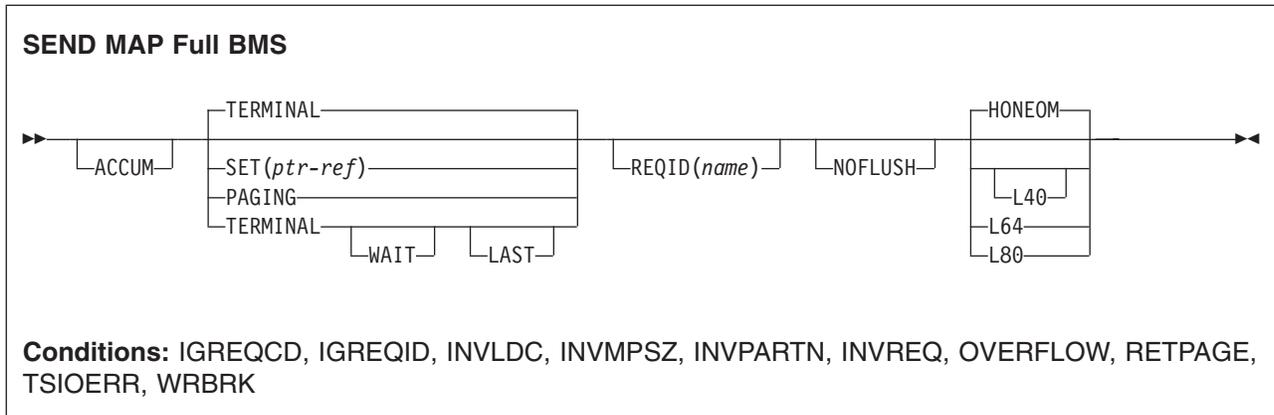
►► MAPSET (*name*) FROM (*data-area*) DATAONLY LENGTH (*data-value*)
MAPONLY ◀◀

► CURSOR (*—data-value—*) FORMFEED ERASE ERASEAUP
DEFAULT ALTERNATE PRINT ◀◀

► FREEKB ALARM FRSET ◀◀

SEND MAP Standard BMS

►► NLEOM MSR (*data-value*) FMHPARM (*name*)
OUTPARTN (*name*) LDC (*name*) ACTPARTN (*name*) ◀◀



Description

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 DFHMDI options, CTRL for a description of the option priority.

See “BMS macros” on page 784 for map definition.

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 DFHMDI options, CTRL 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 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 RDO 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 TYPETERM option FORMFEED was not used).

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 DFHMDF options, CTRL 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 DFHMDI operands, TERM and Specifying NODDS in the BMS operand).

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 DFHMDI options, CTRL 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 TYPETERM options PAGESIZE or ALTPAGE, and the same as the printer platen width; otherwise the data may not format correctly.

When using the HONEOM option, refer to DFHMDI options, CTRL 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 defined by a DFHTCT TYPE=LDC macro.

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 807. 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 RDO TYPETERM options PAGESIZE or ALTPAGE, 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 DFHMDS BMS map definitions (see DFHMDI operands, TERM and Specifying NODDS in the BMS operand). For a description of a safe upper limit, see “LENGTH options in CICS commands” on page 8.

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 TYPETERM options PAGESIZE or ALTPAGE.

When using the options, refer to DFHMDI options, CTRL 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 9 998.

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 782 for a complete list. This option is ignored if the RDO TYPETERM option MSRCONTROL was not used.

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 TYPETERM options PAGESIZE or ALTPAGE, for that terminal, is unavailable for data. This may cause data to wrap around in the buffer; if this occurs, the PAGESIZE or ALTPAGE 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 DFHMDS or DFHMDSI 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 DFHMDSI options, CTRL 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 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.

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 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.

INVMPsz

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

RESP2 values:

200 Command not allowed for a distributed program link server program.

also occurs (RESP2 not set) 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.
- 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.
- 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

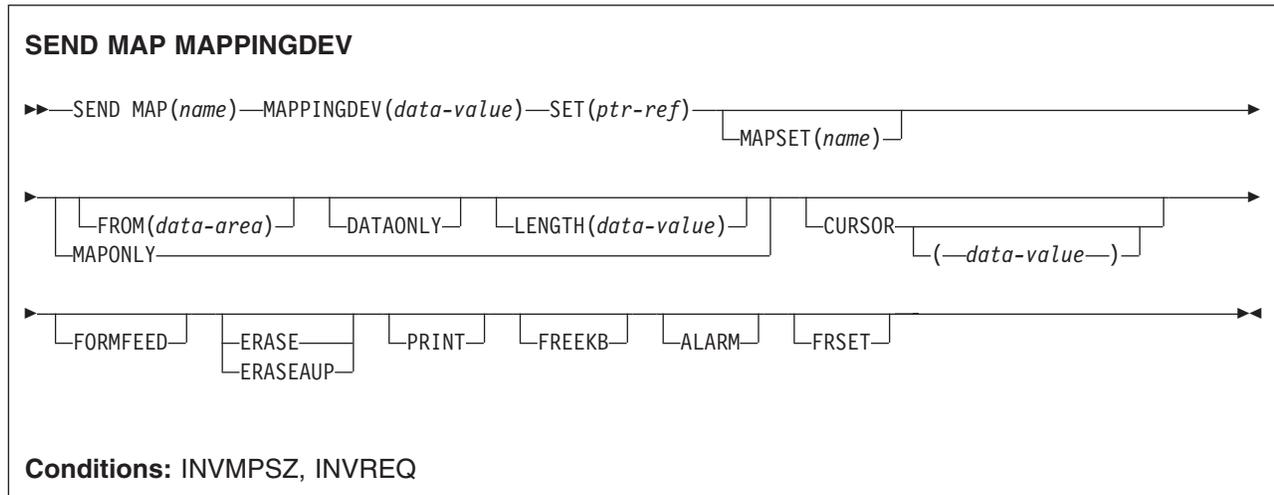
occurs if a SEND MAP command is interrupted by the terminal operator pressing the ATTN key. It applies only to the 2741 Communication Terminal, and only if write break is supported for CICS.

Default action: ignore the condition.

SEND MAP MAPPINGDEV

Create mapped output data to be sent to a terminal described by MAPPINGDEV at some later time. For further information about BMS, see the *CICS Application Programming Guide*.

Minimum BMS:



Description

SEND MAP MAPPINGDEV creates mapped output data to be sent to a terminal that is not 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.

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.

When using the ALARM option, refer to DFHMDI options, CTRL 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 *CICS Application Programming Guide* for more 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 DFHMDI options, CTRL 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 DFHMDI operands, TERM and Specifying NODDS in the BMS operand).

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 DFHMDI options, CTRL 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 DFHMDS BMS map definitions (see DFHMDI operands, TERM and Specifying NODDS in the BMS operand). For a description of a safe upper limit, see “LENGTH options in CICS commands” on page 8.

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 9 998.

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 DFHMDI options, CTRL 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 *CICS Application Programming Guide* for more guidance about using the SET option.

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.

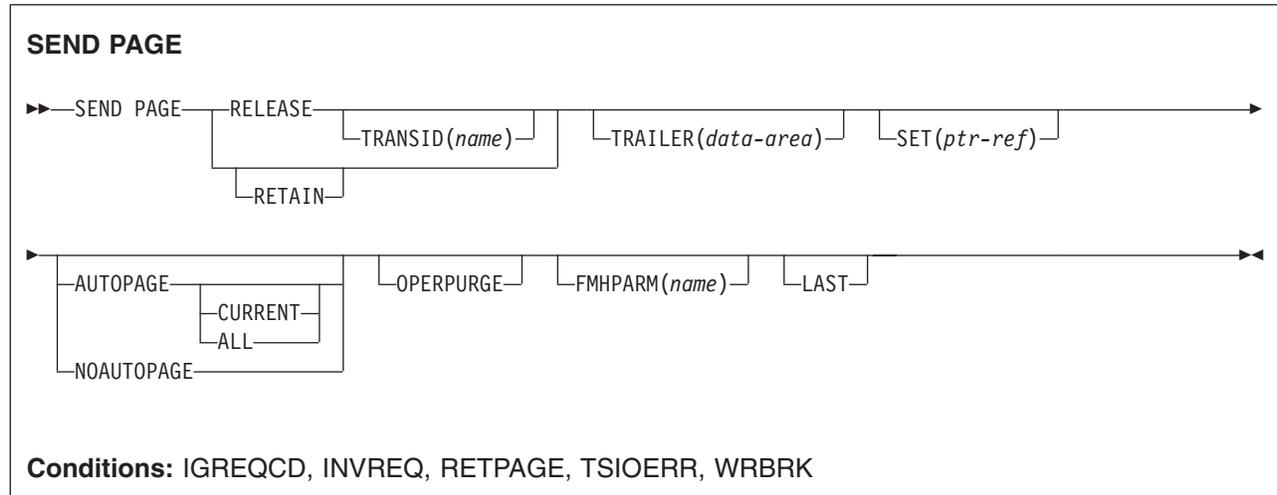
INVREQ

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 last page of data. Only supplied by full BMS. For further information about BMS, see the *CICS Application Programming Guide*.



Description

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.

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 RDO TYPETERM 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 RDO TYPETERM 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 Supplied Transactions* for more information about terminal operator paging commands.)

If automatic paging is specified for the terminal by the RDO TYPETERM option AUTOPAGE(YES), 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 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 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 to CICS via a RDO TRANSACTION resource 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.

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

RESP2 values:

200 Command not allowed for a distributed program link server program.

also occurs (RESP2 not set) in any of the following situations:

- 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.

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 unrecoverable 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.

Default action: ignore the condition.

SEND PARTNSET

This command is available on standard and full BMS only. For further information about BMS, see the *CICS Application Programming Guide*.

SEND PARTNSET

►—SEND PARTNSET—┐
└(—name—)┘◄

Conditions: INVPARTNSET, INVREQ

Description

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.

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

RESP2 values:

200 Command not allowed for a distributed program link server program.

also occurs (RESP2 not set) in the following situation:

- A SEND PARTNSET command is issued while a logical message is active.

Default action: terminate the task abnormally.

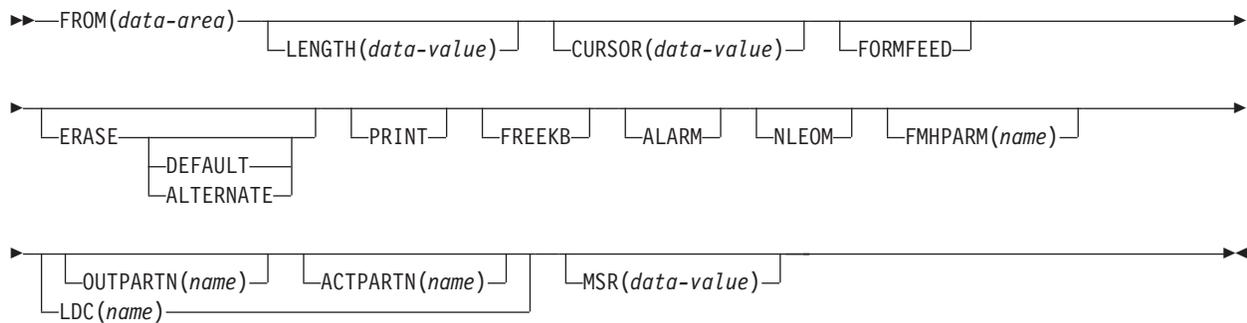
SEND TEXT

Send data without mapping. The keywords are separated into those supported by standard and full BMS. For further information about BMS, see the *CICS Application Programming Guide*.

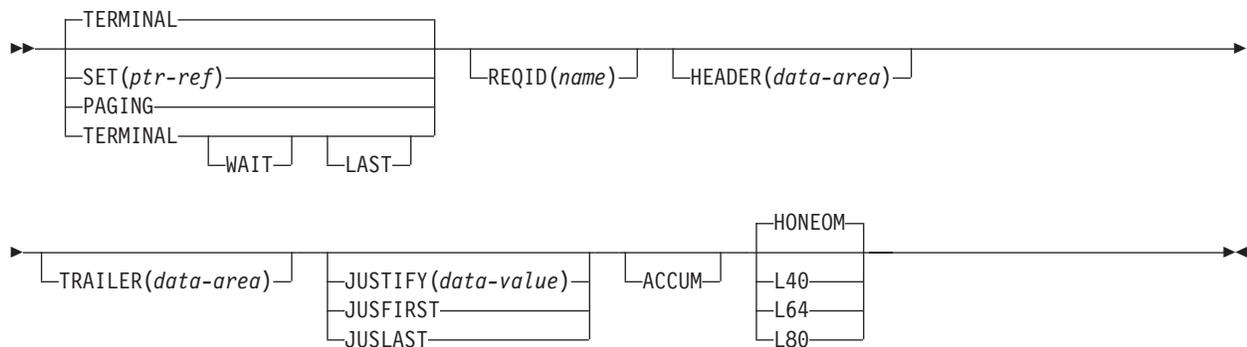
SEND TEXT

►► SEND TEXT ◀◀

SEND TEXT Standard BMS



SEND TEXT Full BMS



Conditions: IGRREQCD, IGRREQID, INVLDC, INVPARTN, INVREQ, LENGERR, RETPAGE, TSIOERR, WRBRK

Description

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 DFHMDI options, CTRL for a description of the option priority.

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.

HEADER, JUSFIRST, JUSLAST, JUSTIFY and TRAILER all imply ACCUM.

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 RDO 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 TYPETERM option FORMFEED was not used).

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 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, see DFHMDI options, CTRL 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 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 TYPETERM options PAGESIZE or ALTPAGE, and the same as the printer platen width; otherwise the data may not format correctly.

When using the HONEOM option, see DFHMDI options, CTRL 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.

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 defined by a DFHTCT TYPE=LDC macro.

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 in CICS commands" on page 8.

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 TYPETERM options PAGESIZE or ALTPAGE.

When using the options, see DFHMDI options, CTRL 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 782 for a complete list. This option is ignored if the RDO TYPETERM option MSRCONTROL was not used.

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 TYPETERM options PAGESIZE or ALTPAGE for that terminal, is unavailable for data. This may cause data to wrap around in the buffer; if this occurs, the PAGESIZE 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, see DFHMDI options, CTRL 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 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 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.

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

RESP2 values:

200 Command not allowed for a distributed program link server program.

also occurs (RESP2 not set) 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.

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.

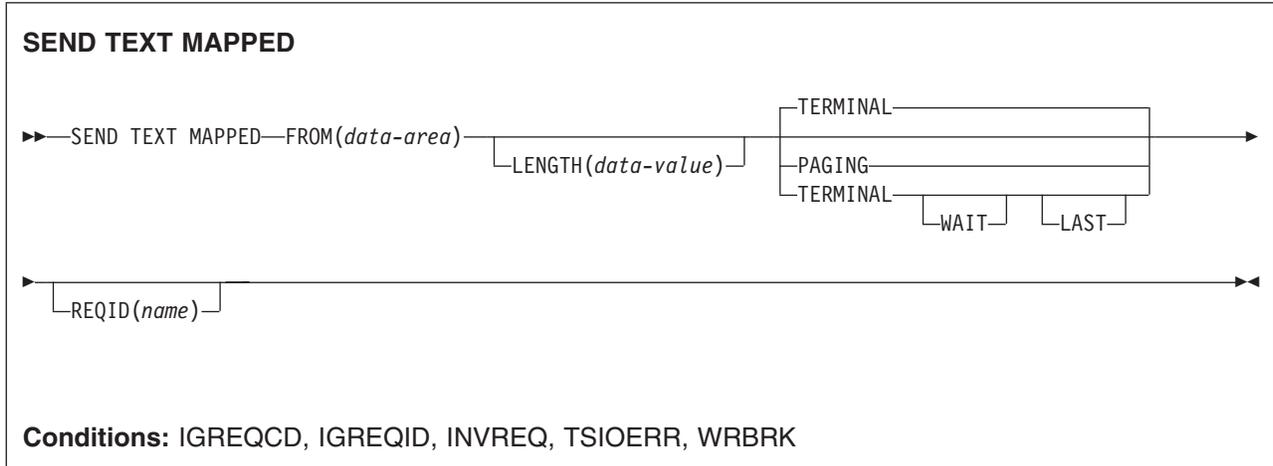
WRBRK

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

Send data with mapping. Only supplied by full BMS. For further information about BMS, see the *CICS Application Programming Guide*.



Description

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.

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 in CICS commands” on page 8.

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.

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

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID option.

Default action: terminate the task abnormally.

TSIOERR

occurs if there is an irrecoverable temporary storage input/output error.

Default action: terminate the task abnormally.

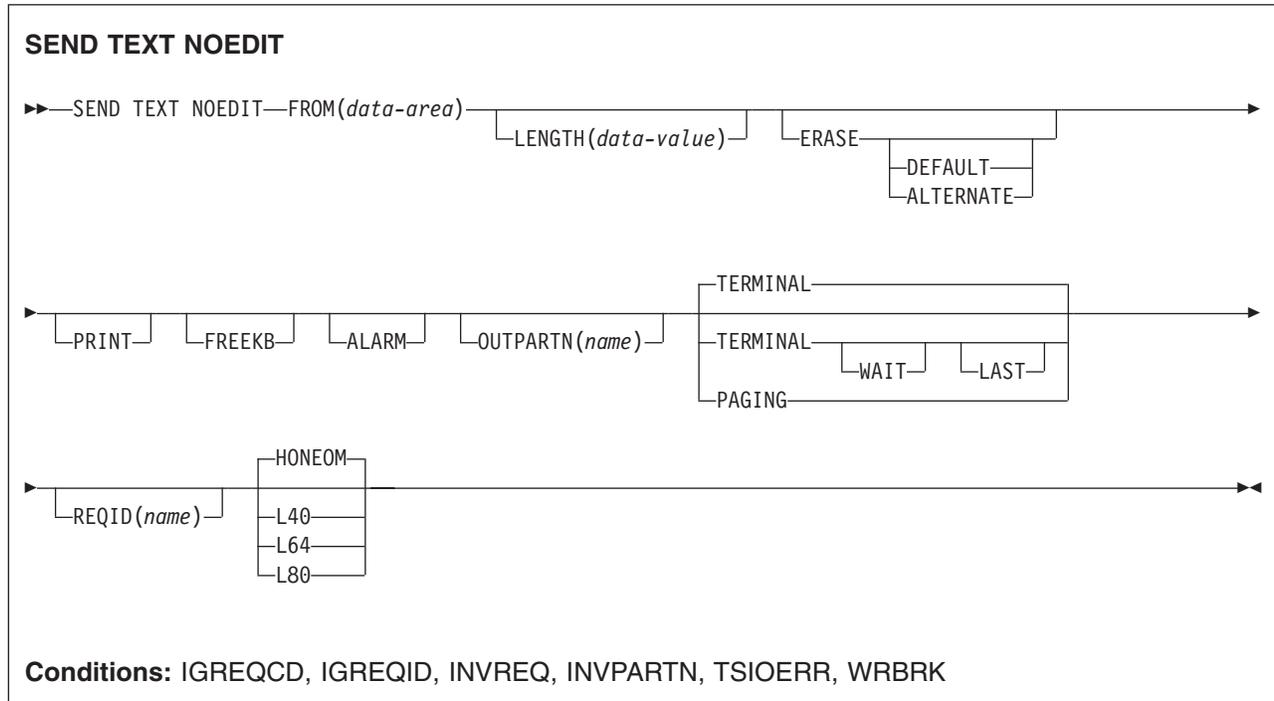
WRBRK

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

Send a page. Only supplied by full BMS. For further information about BMS, see the *CICS Application Programming Guide*.



Description

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 DFHMDI options, CTRL for a description of the option priority.

This command is supported on full BMS only.

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 RDO 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 DFHMDI options, CTRL 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 TYPETERM options PAGESIZE or ALTPAGE, and the same as the printer platen width; otherwise the data may not format correctly.

When using the HONEOM option, refer to DFHMDI options, CTRL 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 in CICS commands" on page 8.

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 TYPETERM options PAGESIZE or ALTPAGE.

When using the options, refer to CTRL DFHMDI options, CTRL 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 DFHMDI options, CTRL 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.

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

RESP2 values:

200 Command not allowed for a distributed program link server program.

also occurs (RESP2 not set) 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.

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

Sign off from a terminal.

SIGNOFF

▶▶—SIGNOFF—◀◀

Condition: INVREQ

Description

SIGNOFF enables you to sign off from the terminal or principal facility that you previously signed on to. When sign-off 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.

When this command is executed, CICS immediately recognizes the sign-off 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 sign-off before the transaction terminates.**

The default attributes apply for all RMs invoked by subsequent transactions at the terminal.

Conditions

INVREQ

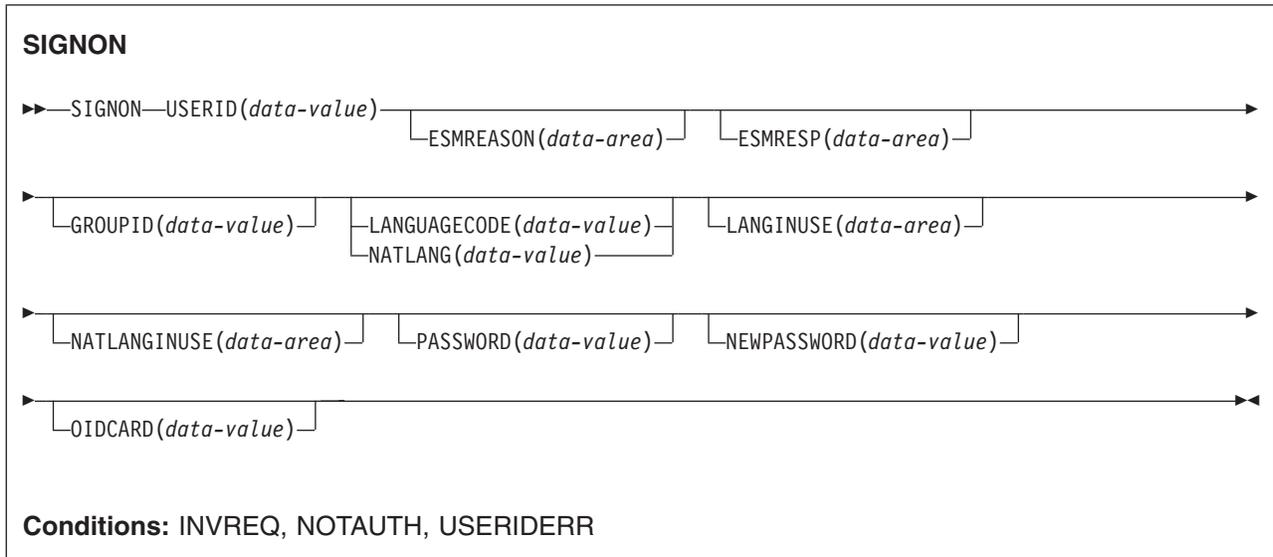
RESP2 values:

- 1** No user is currently signed on. This could be because the CICS ESM is not initialized.
- 2** There is no terminal with this task.
- 3** This task's terminal has preset security.
- 4** Sign-off is attempted using transaction routing without using the CRTE transaction.
- 18** The CICS ESM interface is not initialized.
- 200** Command not allowed for a distributed program link server program.

Default action: terminate the task abnormally.

SIGNON

Sign on to a terminal.



Description

The SIGNON command enables your application to associate a new user ID with the current terminal. When you use the SIGNON command, the following rules apply:

- The signon operation is terminal related only. Signon has no meaning if the transaction does not have a terminal as its principal facility.
- When you issue an EXEC CICS SIGNON command, CICS modifies the state of the terminal that is the principal facility of the transaction that issues the command.
- Signon does not affect the user ID and security capabilities currently in effect for the transaction issuing the command. This is because:
 - A transaction's user ID and security capabilities are established at transaction-attach time. It is not possible to modify these subsequently during the life of the transaction.
 - All actions performed by a transaction (whether to a local or remote resource, or to a connected system) take place in the security context established at the time the transaction was attached.

There is no implied sign-off with the SIGNON command. If your application program attempts to associate a new user with a terminal that already has a signed-on user ID, CICS returns an INVREQ (Resp2=9) error response. Note that there is no default value for the USERID option.

PASSWORD is used as a parameter which means that if CICS takes a dump, the password may be visible. You should therefore clear the PASSWORD field as soon as possible after using it in a SIGNON command.

For more information on how CICS uses the USERID and GROUPLD, see *CICS RACF Security Guide*.

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.

GROUPLD(*data-value*)

assigns, to a RACF user group, the user that is being signed on. This overrides, for this session only, the default group name specified for the user in the RACF database.

LANGUAGECODE(*data-value*)

specifies the national language that the user being signed on wants CICS to use. You specify the language as a standard 3-character IBM code. This is an alternative to the 1-character code that you specify on the NATLANG option.

See “National language codes” on page 766 for possible values of the code.

LANGINUSE(*data-area*)

the LANGINUSE option allows an application program to receive the national language chosen by the sign-on 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 “National language codes” on page 766 for possible values of the code.

NATLANG(*data-value*)

specifies a 1-character field identifying the national language the user wants to use during the signed-on session.

See “National language codes” on page 766 for possible values of the code.

NATLANGINUSE(*data-area*)

specifies a one character the national language used during the signed-on session. The current implementation always returns the character “E” (U.S. English), which corresponds to the language supplied in the NATLANG option. 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 “National language codes” on page 766 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 sign-on USERID.

Conditions**INVREQ**

RESP2 values:

- 9** The terminal is already signed on.
- 10** No terminal is associated with this task.
- 11** This task's terminal has preset security.
- 12** The response from CICS security modules is unrecognized.
- 13** 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.
- 14** The required national language is not available.
- 15** Signon was attempted using transaction routing without using the CRTE transaction.
- 18** The CICS ESM interface is not initialized (SEC=NO specified as a System initialization parameter).
- 25** The terminal is of an invalid type.
- 26** An error occurred during SNSCOPE checking. The limit of MVS ENQ requests was reached.
- 27** The external security manager (ESM) is not active.
- 28** The required national language is invalid.
- 29** The user is already signed on. This relates to the sign-on scope checking.
- 200** Command not allowed for a distributed program link server program.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

- 1** A password is required.
- 2** The supplied password is wrong.
- 3** A new password is required.
- 4** The new password is not acceptable.
- 5** An OIDCARD is required.
- 6** The supplied OIDCARD is wrong.
- 16** The USERID is not authorized to use this terminal.
- 17** The USERID is not authorized to use the application.

- 19** The USERID is revoked.
- 20** The USERID's access to the specified group has been revoked.
- 21** The sign-on failed during SECLABEL checking.
- 22** The sign-on failed because the ESM is not currently accepting sign-on.
- 23** The GROUPID is not known to the ESM.
- 24** The USERID is not contained in the GROUPID.

Default action: terminate the task abnormally.

USERIDERR

RESP2 values:

- 8** The USERID is not known to the external security manager.
- 30** The USERID is all blanks or nulls.

Default action: terminate the task abnormally.

Options

SUBCODESTR(*data-value*)

Specifies the contents of a <Subcode> element that is to be added to the SOAPFAULT object. The subcode can be up to 64 characters in length, and must be an XML qualified name (QName). An XML qualified name has the form *prefix:name*.

- For SOAP 1.1, this option is ignored.
- For SOAP 1.2, this option supplies the contents of the <Subcode> element.

SUBCODELEN(*data-value*)

specifies the length, as a fullword binary value, of the <Subcode> element specified in the SUBCODESTR option.

FAULTSTRING(*data-value*)

specifies a human-readable explanation of the fault. The FAULTSTRING can be up to 2056 characters in length.

- For SOAP 1.1, this option supplies the contents of the <faultstring> element.
- For SOAP 1.2, this option supplies the contents of the <Reason> element.

FAULTSTRLEN(*data-value*)

Specifies the length, as a fullword binary value, of the FAULTSTRING option.

FROMCCSID(*data-value*)

Specifies, as a fullword decimal number, the current Coded Character Set Identifier (CCSID) of the character data to be put into the SOAP fault. If this option is not specified, CICS uses the value which is specified in the LOCALCCSID system initialization parameter. For more information about CCSIDs, and a list of the CCSIDs supported by CICS, see *CICS Family: Communicating from CICS on System/390*.

NATLANG(*data-value*)

Specifies an eight character field containing the national language used for the FAULTSTRING. The language is specified using the XML 1.0 language identification. The default value is 'en' (English).

When the language identifier is shorter than eight characters, you must pad it on the right with space characters in the character set specified in the FROMCCSID option (or the CICS LOCALCCSID). For example, if you specify the UTF-8 character set with FROMCCSID(1208), you must pad the NATLANG value with X'20' characters.

Conditions

INVREQ

RESP2 values are:

- 3** The command was issued outside the environment of a CICS supplied SOAP handler.
- 12** The contents of the SUBCODESTR is invalid.

LENGERR

RESP2 values are:

- 6** The FAULTSTRLEN value is invalid.
- 10** The SUBCODELEN value is invalid.

CCSIDERR

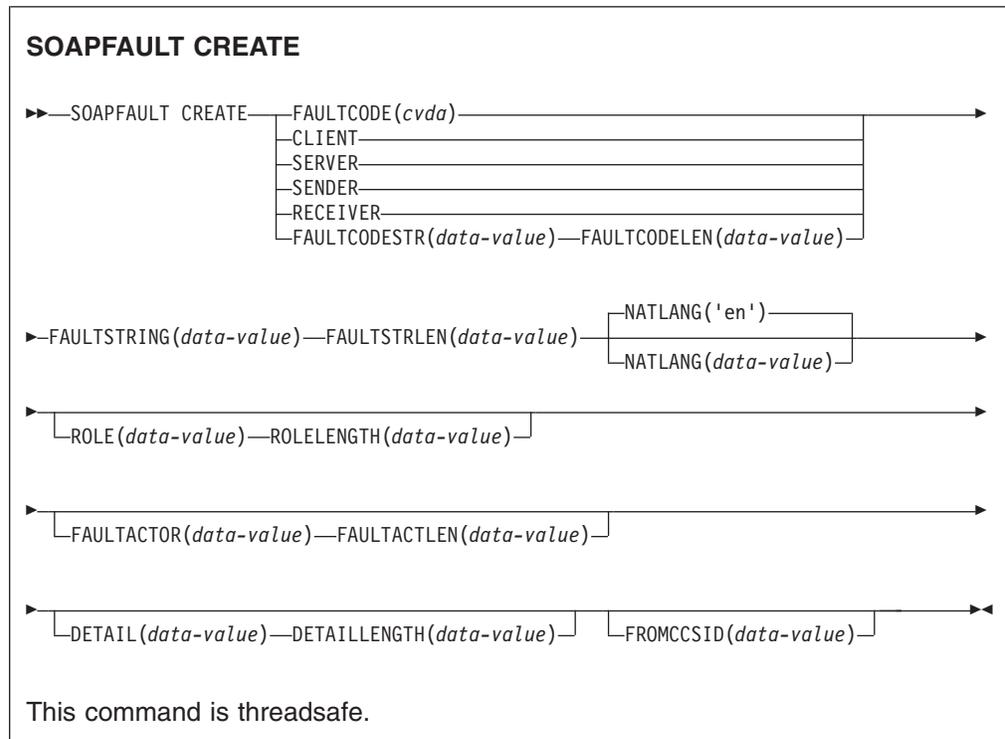
RESP2 values are:

#

#

SOAPFAULT CREATE

This command creates a SOAP fault. You can use this command only in a program that is invoked from a CICS-supplied SOAP message handler.



Description

SOAPFAULT CREATE creates a SOAP fault. If a SOAP fault already exists in the context of the SOAP message that is being processed by the message handler, the existing fault is overwritten.

This command requires information that is held in containers on the channel of the CICS-supplied SOAP message handler. To use this command, you must have access to the channel. Only the following types of programs have this access:

- Programs that are invoked as SOAP header handlers
- Programs that are invoked directly from a CICS-supplied SOAP message handler
- Programs deployed with the CICS Web services assistant that have a channel interface. Programs with a COMMAREA interface do **not** have access to the channel.

Many of the options on this command apply to SOAP 1.1 and SOAP 1.2 faults, although their behavior is slightly different for each level of SOAP. Other options apply to one SOAP level or the other, but not to both, and if you specify any of them when the message uses a different level of SOAP, the command will raise an INVREQ condition. To help you determine which SOAP level applies to the message, container DFHWS-SOAPLEVEL contains a binary fullword with one of the following values:

- 1 The request or response is a SOAP 1.1 message.

2 The request or response is a SOAP 1.2 message.

10 The request or response is not a SOAP message.

Options

DETAIL(*data-value*)

- For SOAP 1.1, this option supplies the contents of the <detail> element of the SOAP fault.
- For SOAP 1.2, this option supplies the contents of the <Detail> element of the SOAP fault..

It should contain either one or more valid namespace-qualified XML elements, or whitespace. Refer to the appropriate SOAP specifications for a full description of the valid content of the element.

The element carries application-specific error information related to the <Body> element, and is used when the contents of the <Body> element could not be successfully processed. For SOAP 1.1, the <detail> element must be present if the contents of the <Body> element could not be successfully processed; for SOAP 1.2, the <Detail> element is optional.

If the SOAPFAULT CREATE command is issued in a header handler program the <detail> or <Detail> element is carried in a header block.

DETAILLENGTH(*data-value*)

specifies the length, as a fullword binary value, of the DETAIL option.

FAULTACTLEN(*data-value*)

Specifies the length, as a fullword binary value, of the FAULTACTOR option.

FAULTACTOR(*data-value*)

- For SOAP 1.1, this option supplies the contents of the <faultactor> element.
- For SOAP 1.2, this option supplies the contents of the <Node> element.

The FAULTACTOR option can be up to 2056 characters in length, and must be a valid URI (anyURI).

FAULTCODE(*cvda*)

CLIENT

SENDER

For SOAP 1.1 specifies a SOAP Fault code of Client

For SOAP 1.2 specifies a SOAP Fault code of Sender

SERVER

RECEIVER

For SOAP 1.1 specifies a SOAP Fault code of Server

For SOAP 1.2 specifies a SOAP Fault code of Receiver

FAULTCODELEN(*data-value*)

Specifies the length, as a fullword binary value, of the FAULTCODESTR option.

FAULTCODESTR(*data-value*)

Specifies a user-defined SOAP Fault code. The Fault code can be up to 64 characters in length, and must be an XML qualified name (QName). The use of the "." (dot) character to separate Fault code values is not supported.

- For SOAP 1.1, this option supplies the contents of the <faultcode> element.
- For SOAP 1.2, this option supplies the contents of the <Code> element.

FAULTSTRING(*data-value*)

Specifies a human-readable explanation of the fault. The FAULTSTRING can be up to 2056 characters in length.

- For SOAP 1.1, this option supplies the contents of the <faultstring> element.
- For SOAP 1.2, this option supplies the contents of the <Reason> element.

FAULTSTRLEN(*data-value*)

Specifies the length, as a fullword binary value, of the FAULTSTRING option.

FROMCCSID(*data-value*)

Specifies, as a fullword decimal number, the current Coded Character Set Identifier (CCSID) of the character data to be put into the SOAP fault. If this option is not specified, CICS uses the value which is specified in the LOCALCCSID system initialization parameter. For more information about CCSIDs, and a list of the CCSIDs supported by CICS, see *CICS Family: Communicating from CICS on System/390*.

NATLANG(*data-value*)

Specifies an eight character field containing the national language used for the FAULTSTRING. The language is specified using the XML 1.0 language identification. The default value is 'en' (English).

When the language identifier is shorter than eight characters, you must pad it on the right with space characters in the character set specified in the FROMCCSID option (or the CICS LOCALCCSID). For example, if you specify the UTF-8 character set with FROMCCSID(1208), you must pad the NATLANG value with X'20' characters.

ROLE(*data-value*)

Specifies the URI that describes the role of the SOAP node that generated the fault. The ROLE option can be up to 2056 characters in length, and must be a valid URI (XML type anyURI).

- For SOAP 1.1, this option is ignored.
- For SOAP 1.2, this option supplies the contents of the <Role> element.

ROLELENGTH(*data-value*)

Specifies the length, as a fullword binary value, of the ROLE option.

Conditions**INVREQ**

RESP2 values are:

- 3** The command was issued outside the environment of a CICS supplied SOAP handler.
- 11** The FAULTCODE specified is invalid.
- 13** The DETAIL option does not contain valid namespace-qualified XML or whitespace.

LENGERR

RESP2 values are:

- 5** The FAULTCODELEN value is invalid
- 6** The FAULTSTRLEN value is invalid
- 7** The ROLELENGTH value is invalid
- 8** The FAULTACTLEN value is invalid
- 9** The DETAILLENGTH value is invalid

SOAPFAULT DELETE

This command deletes an existing SOAPFAULT object. You can use it only in a program that is invoked from a CICS-supplied SOAP message handler.

SOAPFAULT DELETE

▶—SOAPFAULT DELETE—▶

This command is threadsafe.

Description

This command deletes a SOAPFAULT object that was created with an earlier **SOAPFAULT CREATE** command.

This command requires information that is held in containers on the channel of the CICS-supplied SOAP message handler. To use this command, you must have access to the channel. Only the following types of programs have this access:

- Programs that are invoked as SOAP header handlers
- Programs that are invoked directly from a CICS-supplied SOAP message handler
- Programs deployed with the CICS Web services assistant that have a channel interface. Programs with a COMMAREA interface do **not** have access to the channel.

Conditions

INVREQ

RESP2 values are:

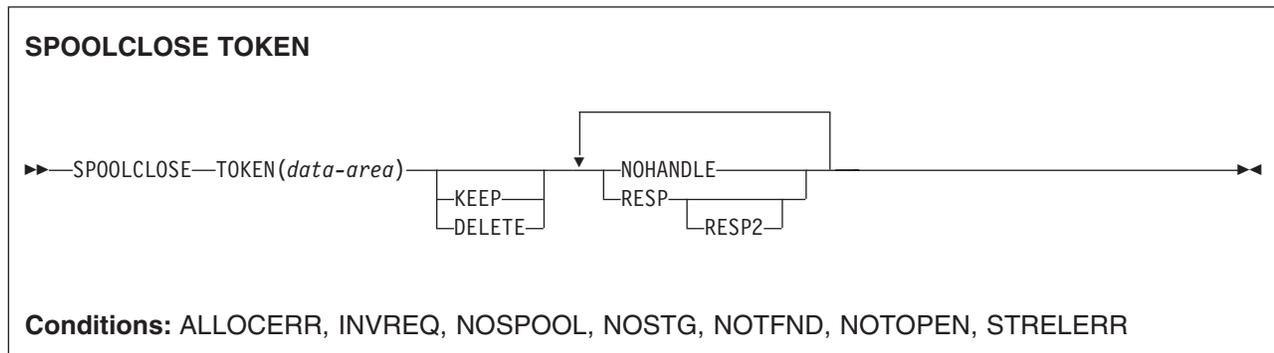
- 3** The function was called when a CICS supplied SOAP node was not in use

NOTFND

- 2** There is no SOAPFAULT to delete

SPOOLCLOSE

Close a spool report.



Description

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.

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.

Conditions

Note: There are no default actions.

ALLOCERR

occurs in any of the following situations:

- 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 *z/OS: MVS Programming: Authorized Assembler Services Guide, SA22-7608*.

INVREQ

RESP2 values:

- 4** Unsupported language.
- 8** Unsupported function.
- 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.

NOSP00L

RESP2 values:

- 4** No subsystem present.
- 8** Interface being disabled; CICS is quiescing.
- 12** Interface has been stopped.

NOSTG

occurs in any of the following situations:

- A GETMAIN has failed within the JES interface subtask (DFHPSPSS).

RESP2 gives the GETMAIN register 15 return code.

NOTFND

RESP2 values:

- 1024** Input or output function has been corrupted, and SPOOLCLOSE could not complete.

NOTOPEN

RESP2 values:

- 8** Data set has not been opened.

STRELERR

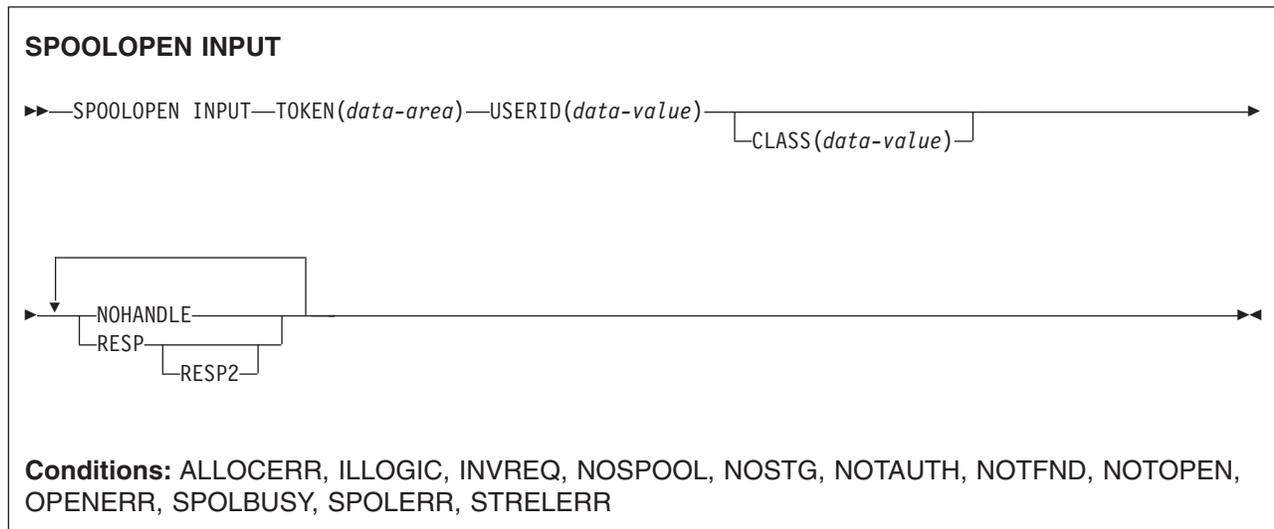
occurs in any of the following situations:

- A FREEMAIN has failed within the JES interface subtask (DFHPSPSS).

RESP2 gives the FREEMAIN register 15 return code.

SPOOLOPEN INPUT

Open a spool report.



Description

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.

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.

Conditions

Note: There are no default actions.

ALLOCERR

occurs in any of the following situations:

- 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 *z/OS: MVS Programming: Authorized Assembler Services Guide, SA22-7608*.

ILLOGIC

RESP2 values:

- 3** Invalid CLASS value specified.

INVREQ

RESP2 values:

- 4** Unsupported language.
8 Unsupported function.
16 USERID missing.
36 INPUTOUTPUT missing.
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

RESP2 values:

- 4** No subsystem present.
8 Interface being disabled; CICS is quiescing.
12 Interface has been stopped.

NOSTG

occurs in any of the following situations:

- A GETMAIN has failed within the JES interface subtask (DFHPSPSS).

RESP2 gives the GETMAIN register 15 return code.

NOTAUTH

occurs in any of the following situations:

- 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.

NOTFND

RESP2 values:

- 4** No data sets could be located for retrieval for the specified external writer name; or the data set exists, but it is in HELD status.

Can also be returned if the CICS region USERID does not have ALTER access to the appropriate PROFILE in the JESSPOOL class. See the *CICS RACF Security Guide* for more information about RACF authorization of JES.

1024 Input or output function has been corrupted, and SPOOLCLOSE could not complete.

NOTOPEN

RESP2 values:

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.

1024 Subtask OPEN macro failure.

OPENERR

RESP2 values:

4 A VSAM SHOWCB macro failed to return the lengths of the VSAM control blocks used to access the JES spool file.

Also occurs (RESP2 not set) in any of the following situations:

- An internal error occurred during SPOOLOPEN processing that has forced the request to fail.

SPOLBUSY

RESP2 values:

4 Interface already in use by another task.

8 Interface already in use by current task.

Also occurs (RESP2 not set) in any of the following situations:

- The JES/input single thread within the JES interface was not available.

SPOLERR

occurs in any of the following situations:

- The subsystem interface macro (IEFSSREQ) has failed. No input data set name was selected.

RESP2 gives the 'IEFSSREQ' response code.

STRELERR

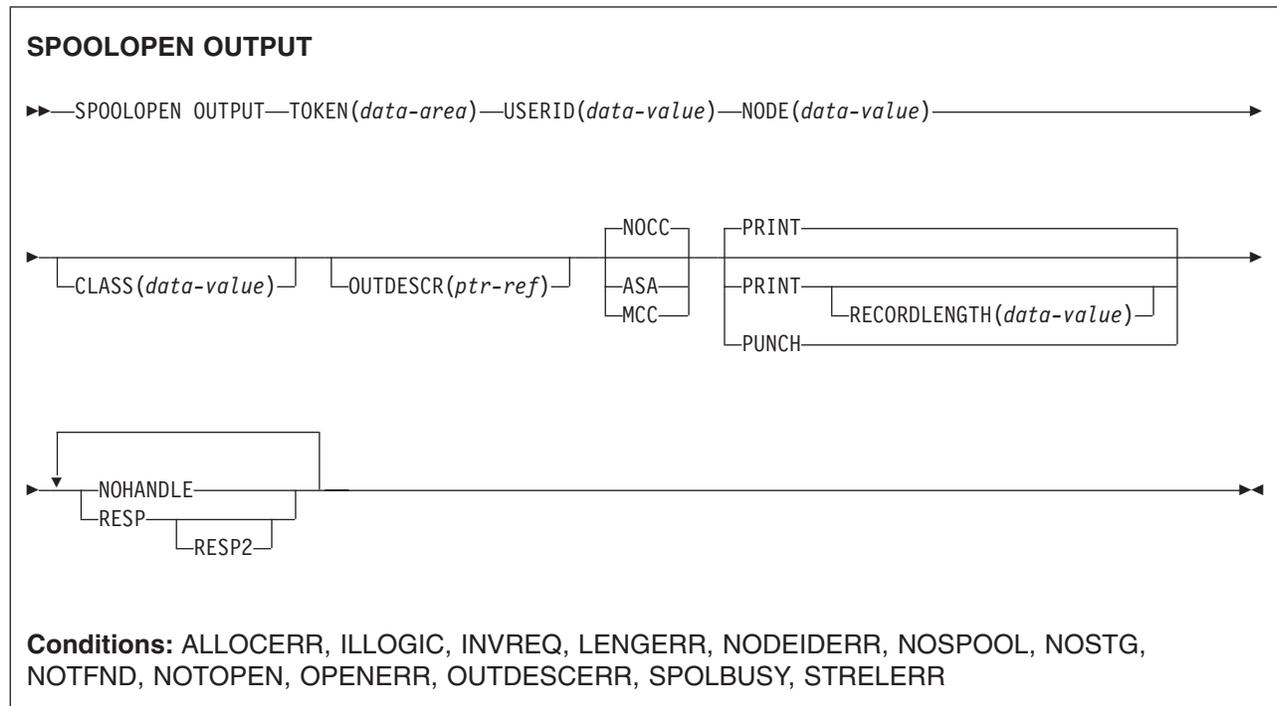
occurs in any of the following situations:

- A FREEMAIN has failed within the JES interface subtask (DFHPSPSS).

RESP2 gives the FREEMAIN register 15 return code.

SPOOLOPEN OUTPUT

Open a spool report for output.



Description

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 OUTPUT command 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 NOCCIASAIMCC options to control output formatting. If you do not specify a format, the default value of NOCC is used. NODE and USERID can be used to write the data set directly to the local spool file only if specified with a value of '*1'.

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.

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. If you want to specify the local spool file and to enable the OUTDESCR operand to override the NODE and USERID operands, code NODE(*) and also USERID(*). (Do not use NODE(*) with any other userid.) Otherwise, code the actual NODE, which is the name of the operating system (for example, MVS, VM) as that system is known to VTAM in the MVS system in which your CICS is executing. NODE(LOCAL) is also a valid specification.

Validity checking is performed for NODE. Checks are made for blanks (X'40'), and nulls (X'00').

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 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 Transaction Server for z/OS.

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 32 760.

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.

If you want to specify the local spool file and to enable the OUTDESCR operand to override the NODE and USERID operands, code USERID('*') and also NODE('*'). Otherwise, code the actual USERID. The meaning of USERID varies with the operating system. In VM, it is a particular user; in MVS, it might be a JES external writer or another JES destination, a TSO user, or another job executing on that system. One such destination is the JES internal reader, which normally has the reserved name INTRDR. If you code an actual USERID, do not use NODE('*'); code the actual NODE instead.

#

The USERID parameter is equivalent to the WRITER parameter in JES.

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 *z/OS JCL User's Guide*.

Conditions

Note: There are no default actions.

ALLOCERR

occurs in any of the following situations:

- 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 *z/OS: MVS Programming: Authorized Assembler Services Guide, SA22-7608*.

ILLOGIC

occurs in any of the following situations:

- Invalid CLASS value specified.

INVREQ

RESP2 values:

- 4** Unsupported language.
- 8** Unsupported function.
- 16** USERID missing.
- 20** NODE missing.
- 36** INPUTOUTPUT missing.
- 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.

- 44** Error in the OUTDESCR string.
- 48** OUTDESCR specified but function not available (wrong level of CICS or JES).
- 52** OUTDESCR specified but bad pointer found on keyword or in OUTDESCR condition.

LENGERR

occurs in any of the following situations:

- RECORDLENGTH not in the range 0 through 32760. RESP2 shows the incorrect value.

NODEIDERR

occurs in any of the following situations:

- JES cannot identify the NODE/USERID combination specified on SPOOL OPEN 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 *z/OS: MVS Programming: Authorized Assembler Services Guide, SA22-7608*.

NOSPOOL

RESP2 values:

- 4** No subsystem present.
- 8** Interface being disabled; CICS is quiescing.
- 12** Interface has been stopped.

NOSTG

occurs in any of the following situations:

- A GETMAIN has failed within the JES interface subtask(DFHPSPSS).
RESP2 gives the GETMAIN register 15 return code.

NOTFND

RESP2 values:

- 4** No data sets could be located for retrieval for the specified external writer name.

NOTOPEN

RESP2 values:

- 8** Data set has not been opened.
- 1024** Subtask OPEN macro failure.

OPENERR

RESP2 values:

- 4** A VSAM SHOWCB macro failed to return the lengths of the VSAM control blocks used to access the JES spool file.

Also occurs (RESP2 not set) in any of the following situations:

- An internal error occurred during SPOOLOPEN processing that has forced the request to fail.

OUTDESCRERR

occurs in any of the following situations:

- The 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 *z/OS: MVS Programming: Authorized Assembler Services Guide, SA22-7608*, for descriptions of codes.

SPOLBUSY

RESP2 values:

- 4** Interface already in use by another task.
- 8** Interface already in use by current task.

Also occurs in the following situation:

- The JES/input single thread within the JES interface was not available.

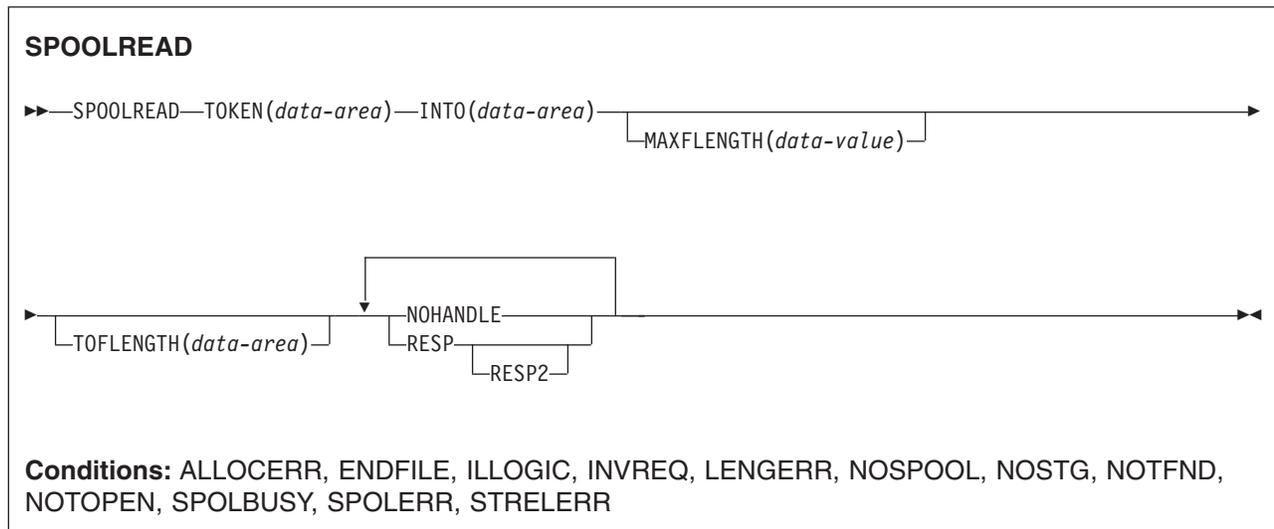
STRELERR

occurs in the following situation:

- A FREEMAIN has failed within the JES interface subtask (DFHPSPSS). RESP2 gives the FREEMAIN register 15 return code.

SPOOLREAD

The SPOOLREAD command obtains the next record from the system spooler.



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 32 760 bytes. This is the maximum size of the CICS buffer used to read a record.

TOFLENGTH(*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.

Conditions

Note: There are no default actions.

ALLOCERR

occurs in any of the following situations:

- 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 *z/OS: MVS Programming: Authorized Assembler Services Guide, SA22-7608*.

ENDFILE

occurs in any of the following situations:

- 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

RESP2 values:

- 3 Invalid CLASS value specified.

INVREQ

RESP2 values:

- 4 Unsupported language.
- 8 Unsupported function.
- 12 Read attempt after end of file.
- 24 INTO missing.
- 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.

LENGERR

occurs in any of the following situations:

- You provided insufficient buffer space to read your record, or you requested more than the maximum allowable buffer size of 32 760 bytes (the size of a CICS buffer used to read a record). If the buffer space is too small, it receives as much data as possible. The amount of data truncated is then placed in the RESP2 field. If the TOFLLENGTH operand is specified, the actual length of the record is placed here.

Note: In the event of a length error due to insufficient buffer space, the next record is not read until the error has been corrected and the current record reread.

RESP2 indicates the amount of data truncated, or shows zero if the MAXFLLENGTH field is greater than the maximum allowable buffer size 32 760 bytes.

NOSPOOL

RESP2 values:

- 4 No subsystem present.
- 8 Interface being disabled; CICS is quiescing.
- 12 Interface has been stopped.

NOSTG

occurs in any of the following situations:

- A GETMAIN has failed within the JES interface subtask (DFHSPSS).
RESP2 gives the GETMAIN register 15 return code.

NOTFND

RESP2 values:

- 4 No data sets could be located for retrieval for the specified external writer name.

NOTOPEN

RESP2 values:

- 8** Data set has not been opened.
- 12** Attempt to read an output file.
- 1024** Subtask OPEN macro failure.

SPOLBUSY

RESP2 values:

- 4** Interface already in use by another task.
- 8** Interface already in use by current task.

Also occurs (RESP2 not set) in any of the following situations:

- The JES/input single thread within the JES interface was not available.

SPOLERR

occurs in any of the following situations:

- The subsystem interface macro (IEFSSREQ) has failed. No input data set name was selected.

RESP2 gives the 'IEFSSREQ' response code.

STRELERR

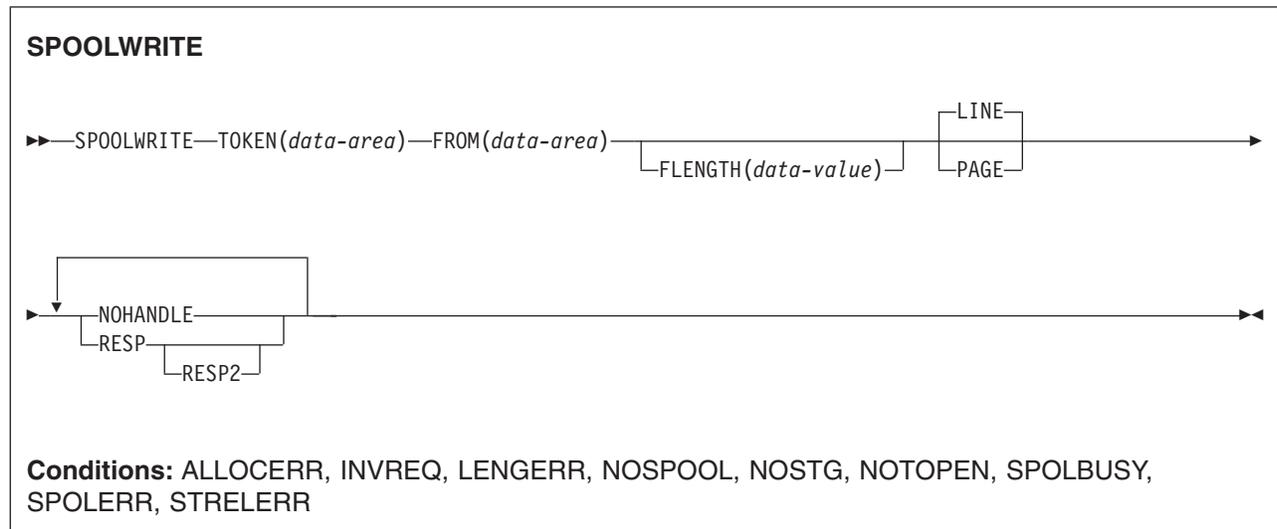
occurs in any of the following situations:

- A FREEMAIN has failed within the JES interface subtask (DFHPSPSS).

RESP2 gives the FREEMAIN register 15 return code.

SPOOLWRITE

The SPOOLWRITE command writes data to a spool report.



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.

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.

Conditions

Note: There are no default actions.

ALLOCERR

occurs in any of the following situations:

- 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 *z/OS: MVS Programming: Authorized Assembler Services Guide, SA22-7608*.

INVREQ

RESP2 values:

- 4** Unsupported language.
- 8** Unsupported function.
- 28** FROM missing.
- 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.

LENGERR

occurs in any of the following situations:

- 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

RESP2 values:

- 4** No subsystem present.
- 8** Interface being disabled; CICS is quiescing.
- 12** Interface has been stopped.

NOSTG

occurs in any of the following situations:

- A GETMAIN has failed within the JES interface subtask (DFHPSPSS).

RESP2 gives the GETMAIN register 15 return code.

NOTOPEN

RESP2 values:

- 8** Spool report has not been opened.
- 16** Attempt to write an input file.
- 1024** Subtask OPEN macro failure.

SPOLBUSY

RESP2 values:

- 4** Interface already in use by another task.
- 8** Interface already in use by current task.

Also occurs (RESP2 not set) in the following situation:

- The JES/input single thread within the JES interface was not available.

SPOLERR

occurs in the following situation:

- The subsystem interface macro (IEFSSREQ) has failed. No input data set name was selected.

RESP2 gives the 'IEFSSREQ' response code.

STRELERR

occurs in the following situation:

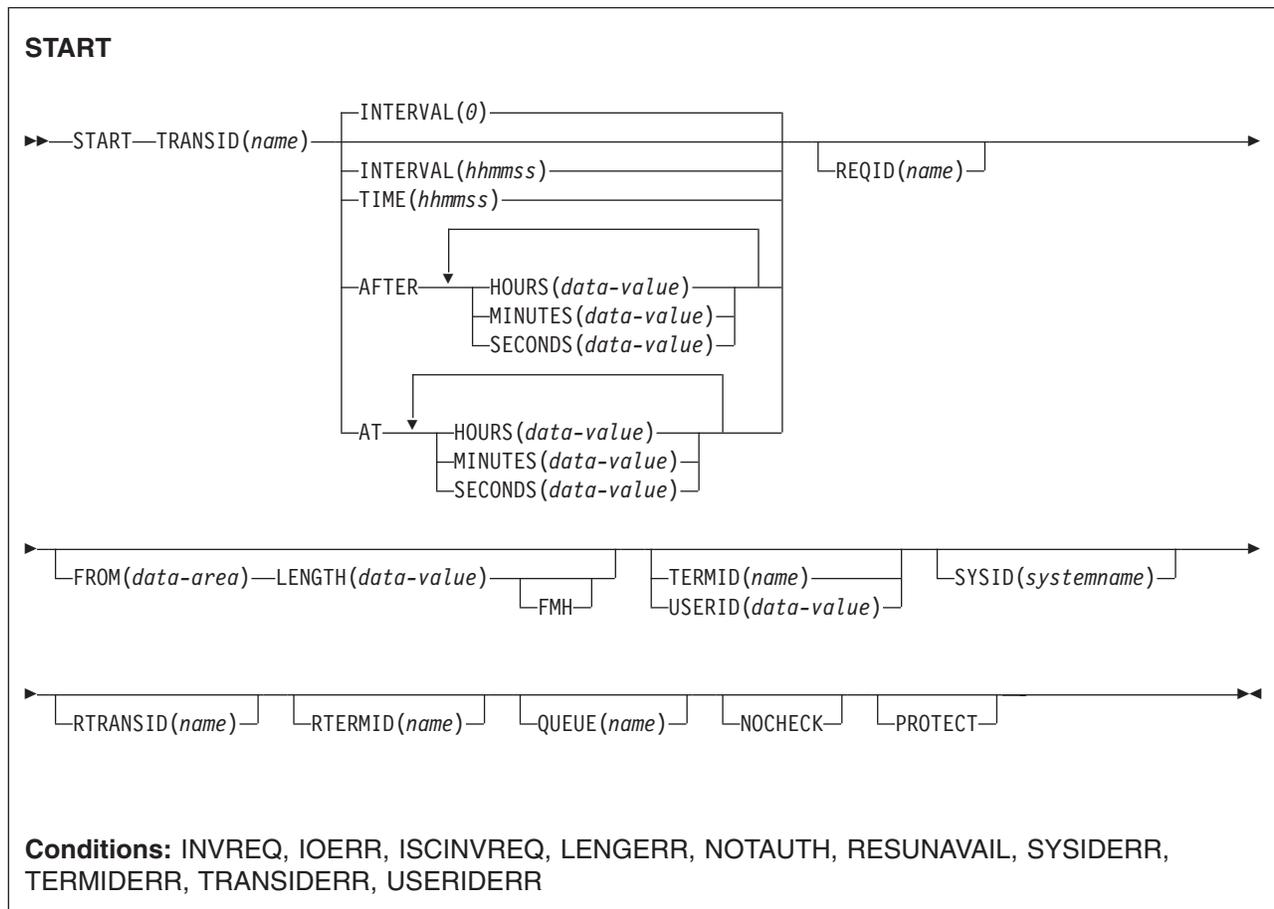
- A FREEMAIN has failed within the JES interface subtask (DFHPSPSS).
RESP2 gives the FREEMAIN register 15 return code.

START

Start task at a specified time.

See also:

- “START ATTACH” on page 598
- “START BREXIT” on page 600
- “START CHANNEL” on page 603



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 Application Programming Guide* for more information about transaction affinities.

Description

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 Application Programming Guide*.

The starting task may pass data to the started task. The starting task may also specify a terminal to be used by the started task as its principal facility.

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

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, RTERMID, 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 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 TERMID 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.
- 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. 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 can be queued by specifying the LOCALQ option on the RDO TRANSACTION resource definition, as described in the *CICS 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 *CICS 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 (see the *CICS Recovery and Restart Guide*).

If you specify the FROM and not the REQID option, a default 'DF' prefix temporary storage queue is used. The same rules apply as above; only specify the PROTECT option if you define the 'DF' prefix temporary storage queues as recoverable.

Note: A START command with REQID, issued from within a task that was itself initiated by a START with the same REQID, returns an AEIQ abend (IOERR condition) if the FROM data for the task has not yet been read by a RETRIEVE.

This is also true if more than one START command with the same REQID is issued by a task or tasks in the same CICS system. This is due to a tightening of the rules governing the use of the REQID option for start requests that have associated data. In CICS/ESA 4.1, and earlier releases, if you specify the same REQID on more than one START command, in some circumstances CICS accepts the start request. However, this can cause the behavior of subsequent RETRIEVE or CANCEL requests to be unpredictable. In particular, the association between each START and its data is lost. CICS TS regions always reject with an IOERR any START commands that specify a duplicate REQID.

Started tasks without data run without a facility address. Started tasks with data run with a facility address of an ICE until the data is retrieved.

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 these examples, CICS assigns one and returns it to the application program in the EIBREQID field of 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 transaction abends and 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)
      :
```

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 or 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.

Dynamically routed transactions started by START commands

Some transactions started by a subset of START commands can be dynamically routed to a remote region. For general information about dynamic transaction routing, and specific information about which transactions started by START commands are eligible for dynamic routing, see the *CICS Intercommunication Guide*.

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 subsequently, as occurs if the user logs off, your START request is discarded with the terminal definition.

Options

AFTER

specifies the interval of time that is to elapse before the new task is started.

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(0–359 999). 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 at which the new task is to be started. For the ways to enter the time, see the AFTER option.

FMH

specifies that the user data to be passed to the started task contains function management headers. This is not valid for LUTYPE2 or LUTYPE3 terminals.

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. This is a suboption of the AFTER and AT options. For its use and meaning, see the AFTER option.

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.

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 as a fullword binary value the number of minutes for use in conjunction with AFTER or AT. The value must be in the range 0 through 59 if HOURS or SECONDS is also specified, or in the range 0 through 5999 otherwise. This is a suboption of the AFTER and AT options. For its use and meaning, see the AFTER option.

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 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 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 must 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 a TS queue using the REQID name specified (or CICS generated) as the identifier. The queue 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.

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. This is a suboption of the AFTER and AT options. For its use and meaning, see the AFTER option.

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.

Conditions

INVREQ

RESP2 values:

- 4** The value specified in HOURS, for AFTER or AT options, or the *hh* value specified for INTERVAL, is out of range.
- 5** The value specified in MINUTES, for AFTER or AT options, or the *mm* value specified for INTERVAL, is out of range.
- 6** The value specified in SECONDS, for AFTER or AT options, or the *ss* value specified for INTERVAL, is out of range.
- 17** The STARTed transaction is not shutdown-enabled, and the CICS region is in the process of shutting down..
- 18** A USERID is specified and the CICS external security manager interface is not initialized.

Also occurs (RESP2 not set) in any of the following situations:

- The START command is not valid for processing by CICS.
- Values specified in the INTERVAL option are out of range.

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 a temporary storage queue when the DFHTEMP dataset is full. is full.
- A START operation uses a REQID name that already exists. 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

RESP2 values:

7 A resource security check fails on TRANSID (name).

9 A surrogate user security check fails on USERID (name).

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.

RESUNAVAIL

RESP2 values:

121 A resource required by the transaction to be started is unavailable on the target region. The RESUNAVAIL condition applies only to *dynamically-routed, non-terminal-related* EXEC CICS START requests.

RESUNAVAIL is returned on the EXEC CICS START command *executed by the mirror in the target region*, if an XICERES global user exit program indicates that a required resource is unavailable on the target region. It is not returned to the application.

Default action: reinvoke the distributed routing program for route selection failure.

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 Customization Guide* for programming information).

The following error is indicated by a RESP2 value:

1 The dynamic routing program rejected the START request.

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

RESP2 values:

- 8** The specified USERID is not known to the external security manager.
- 10** The external security manager is in a state such that CICS cannot determine whether a specified USERID is valid.
- 19** The specified USERID is revoked.

Default action: terminate the task abnormally.

Options

FROM(*data-area*)

specifies the data to be passed to a started task.

LENGTH(*data-value*)

specifies a halfword binary data value that is the length of the data to be passed to a started task.

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 ATTACH command.

Conditions

INVREQ

RESP2 values:

- 11** An attempt was made to route a START ATTACH request.
- 12** A START ATTACH request has failed.

Default action: terminate the task abnormally.

LENGERR

occurs if LENGTH is not greater than zero.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

- 7** A resource security check fails on TRANSID (name).

Default action: terminate the task abnormally.

TRANSIDERR

occurs if the transaction identifier specified in a START command has not been defined to CICS.

RESP2 values:

- 11** The specified transaction is defined as remote.

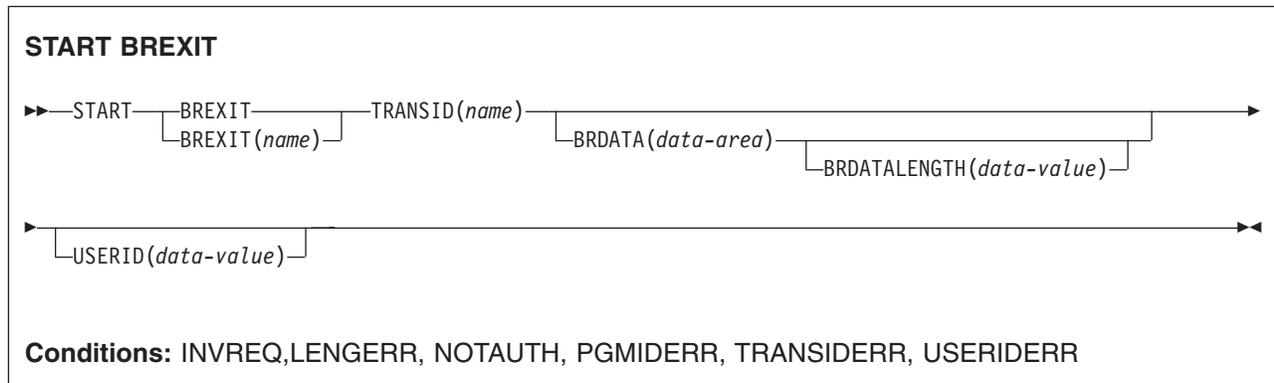
Default action: terminate the task abnormally.

START BREXIT

Start task in the 3270 bridge environment and associate it with the named bridge exit.

See also:

- “START” on page 588
- “START ATTACH” on page 598
- “START CHANNEL” on page 603



Description

START BREXIT starts a task immediately in the local CICS region, and initializes the specified transaction (TRANSID) and bridge exit (BREXIT).

In the 3270 bridge environment, all 3270 terminal requests issued by the transaction specified by TRANSID, are intercepted and passed to the user-replaceable program (the bridge exit) specified by BREXIT.

The bridge exit (BREXIT) emulates the 3270 interface by passing the terminal requests to a client application that may be executing inside or outside of CICS.

See *CICS External Interfaces Guide* for more information about the 3270 bridge and its interfaces.

The attached task cannot be CANCELLED; its STARTCODE is defined by the bridge exit.

Options

BREXIT(*name*)

specifies the name (1-8 characters) of the bridge exit to be associated with the started task. If no name is specified, the value of BREXIT on the TRANSACTION resource definition for TRANSID is used.

BRDATA(*data-area*)

specifies the data to be passed to the bridge exit specified by BREXIT when the task is started.

BRDATALENGTH(*data-value*)

specifies a fullword binary data value that is the length of the BRDATA to be passed to the bridge exit specified by BREXIT when the task is started.

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 BREXIT command. The transaction will be started in the 3270 bridge environment, and will execute in association with the bridge exit specified in BREXIT.

USERID(*data-value*)

Specifies the userid under whose authority the started transaction is to run.

Conditions**INVREQ**

RESP2 values:

- 11** An attempt was made to route a START BREXIT request.
- 12** A START BREXIT request has failed..
- 18** A USERID is specified and the CICS external security manager interface is not initialized.

Default action: terminate the task abnormally.

LENGERR

occurs if BRDATALENGTH is not greater than zero.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

- 7** A resource security check fails on TRANSID (name).
- 9** A surrogate user security check fails on USERID (name). 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.

Default action: terminate the task abnormally.

PGMIDERR

occurs if no name is supplied by the BREXIT option and the transaction definition for TRANSID does not provide a default BREXIT name.

Default action: terminate the task abnormally.

TRANSIDERR

occurs if the TRANSID specified in a START BREXIT command has not been defined to CICS.

RESP2 values:

- 11** The specified transaction is defined as remote.

Default action: terminate the task abnormally.

USERIDERR

RESP2 values:

- 8** The specified USERID is not known to the external security manager.
- 10** The external security manager is in a state such that CICS cannot determine whether a specified USERID is valid.

Default action: terminate the task abnormally.

Passing data to the bridge exit

Data can be passed to the bridge exit using the BRDATA and BRDATALENGTH options.

The following example shows how to start a specified task, in the 3270 bridge environment and pass data to its bridge exit:

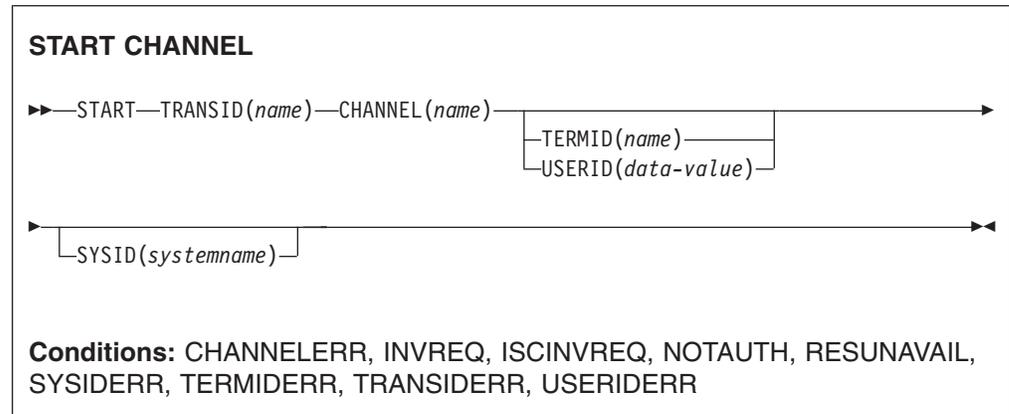
```
EXEC CICS START BREXIT('DFH0CBRE')  
          TRANSID('TRNL')  
          BRDATA(BRSD)  
          BRDATALENGTH(72)  
          :
```

START CHANNEL

Start a task, passing it a channel.

See also:

- “START” on page 588
- “START ATTACH” on page 598
- “START BREXIT” on page 600



Description

START CHANNEL starts a task, on a local or remote system, passing it a channel.

Typically, the starting task uses the channel to pass data to the started task (although in some circumstances the channel may be empty—see the description of the CHANNEL option). The starting task may also specify a terminal to be used by the started task as its principal facility.

The started task can, for example:

1. Use an ASSIGN CHANNEL command to discover the name of the channel it's been passed
2. Use STARTBROWSE CONTAINER CHANNEL and GETNEXT CONTAINER commands to browse the containers in the channel
3. Use GET CONTAINER CHANNEL commands to access the data in the containers

Some constraints have to be satisfied before the transaction to be executed can be started, as follows:

- If the TERMID option is specified, the named terminal must exist and be available. If the named terminal does not exist, the START is discarded.
- START CHANNEL does not support IMS—that is, you cannot use START CHANNEL to start a transaction on a remote IMS system.

Each START CHANNEL command results in a separate task being started.

Dynamically routed transactions started by START commands

Some transactions started by a subset of START commands can be dynamically routed to a remote region. For general information about dynamic transaction

routing, and specific information about which transactions started by START commands are eligible for dynamic routing, see the *CICS Intercommunication Guide*.

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 that is not defined (and cannot be located by the XICTENF or XALTENF exits) at the time CICS attempts to create the task.
- You get a TERMIDERR condition if the requested terminal does not exist at the time of the START. However, if the terminal becomes unavailable subsequently, as occurs if the user logs off, your START request is discarded and no TERMIDERR occurs.

These exposures result from the delay between the execution of the START and the time of task creation. Even on a START CHANNEL request, when the START is always 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.

Options

CHANNEL(*name*)

specifies the name (1–16 characters) of a channel that is to be made available to the started task. The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

Channel names are always in EBCDIC. The allowable set of characters for channel names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if channels are to be shipped between regions, the characters used in naming them should be restricted to A-Z a-z 0-9 & : = , ; < > . - and _ .

The program that issues the START command may:

- Have created the channel by means of one or more PUT CONTAINER CHANNEL commands
- Specify its current channel, by name
- Name a non-existent channel, in which case a new, empty, channel is created

The started task is given a *copy* of the channel's containers (and the data they contain). The copy is made when the START command is issued.

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.

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.

Conditions

CHANNELERR

RESP2 values:

- 1** The channel specified on the CHANNEL option contains an illegal character or combination of characters.

INVREQ

RESP2 values:

- 9** The options specified on the command are incompatible.
- 17** The STARTed transaction is not shutdown-enabled, and the CICS region is in the process of shutting down.
- 18** A USERID is specified and the CICS external security manager interface is not initialized.

Also occurs (RESP2 not set) if the START command is not valid for processing by CICS.

Default action: terminate the task abnormally.

ISCVREQ

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

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

- 7 A resource security check fails on TRANSID (name).
- 9 A surrogate user security check fails on USERID (name).

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.

RESUNAVAIL

RESP2 values:

- 121 A resource required by the transaction to be started is unavailable on the target region. The RESUNAVAIL condition applies only to *dynamically-routed, non-terminal-related* EXEC CICS START requests.

RESUNAVAIL is returned on the EXEC CICS START command *executed by the mirror in the target region*, if an XICERES global user exit program indicates that a required resource is unavailable on the target region. It is not returned to the application.

Default action: reinvoke the distributed routing program for route selection failure.

SYSIDERR

occurs in all of the following cases:

- The SYSID option specifies a name that is neither the local system nor a remote system (made known to CICS by defining a CONNECTION).
- The link to the remote system is known but unavailable.

In all the above cases, the nature of the error is indicated by the second byte of the EIBRCODE.

The following errors are indicated by RESP2 values:

- 1 The dynamic routing program rejected the START request.
- 2 The CHANNEL option was used and the START request was shipped or routed to a remote system which doesn't support it. (MRO connections only)

Default action: terminate the task abnormally.

- 20 The CHANNEL option is specified and the START request is to be shipped over an LUTYPE61 connection. START CHANNEL requests cannot be shipped over LUTYPE61 connections.

The SYSIDERR condition may not be raised if the user exit XISLCLQ is enabled (see the *CICS Customization Guide* for programming information).

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

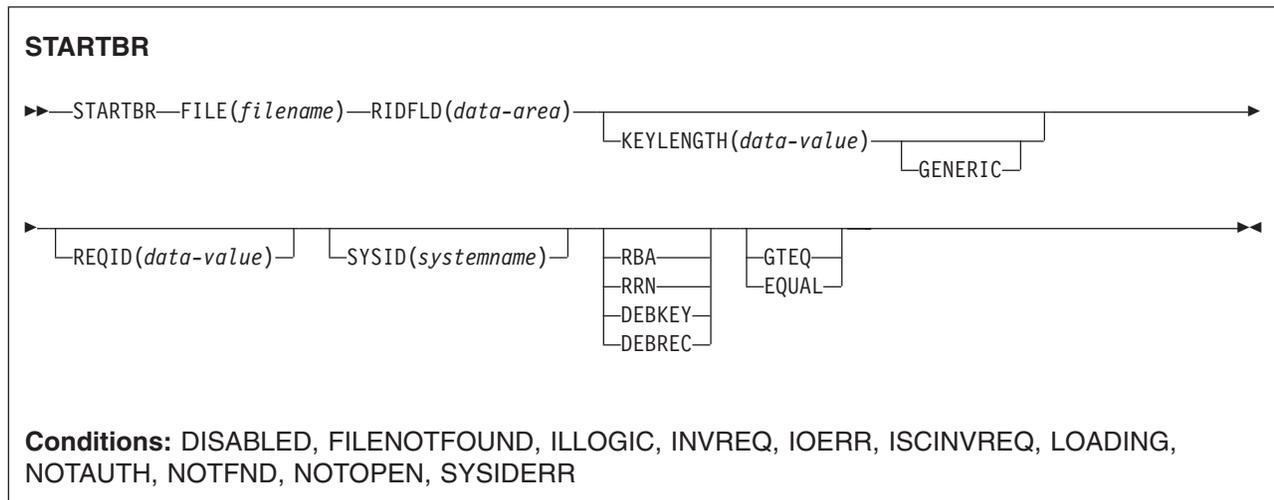
RESP2 values:

- 8** The specified USERID is not known to the external security manager.
- 10** The external security manager is in a state such that CICS cannot determine whether a specified USERID is valid.

Default action: terminate the task abnormally.

STARTBR

Start browse of a file.



Description

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 tables, a READPREV command) is executed.

A browse operation, where direct means browse of the base data set (using the primary key), may be:

- A direct browse of a key sequenced data set (KSDS or data-table) 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 ESDS 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.
- A browse of a KSDS by RBA.

Note: The only VSAM data sets greater than 4GB supported by CICS are KSDS, and then only if they are accessed by key. CICS does not support ESDS or RRDS data sets defined with the extended attribute.

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.

If a STARTBR request specifies the precise key at which the browse is to start (that is, it specifies a full key and the EQUAL keyword) the record returned on the following READNEXT (or READPREV) may not be the same as the record specified by the STARTBR for a file opened in VSAM NSR or RLS mode. This can occur because the initial record specified on the STARTBR command can be deleted by another transaction in between the STARTBR completing and a READNEXT or READPREV being issued. In VSAM LSR mode, the initial record cannot be deleted between the STARTBR and the READNEXT.

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 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

(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.

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 RRRN 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.

For remote files, the **KEYLENGTH** can be specified in the **FILE** definition. If **KEYLENGTH** is not defined there, and is not specified in the application program, and the key is longer than 4 characters, the default value is 4.

RBA

(**VSAM KSDS** or **ESDS** base data sets, or **CICS**-maintained data tables 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
- Coupling facility data tables
- Any **KSDS** files opened in **RLS** access mode
- **KSDS** or **ESDS** files that hold more than 4GB

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. 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.

See 'Identifying **BDAM** records' and 'VSAM record identification' in the *CICS Application Programming Guide* for more information about defining the record identification field.

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.

Conditions

DISABLED

RESP2 values:

- 50** A file is disabled. 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

RESP2 values:

- 1** A file name referred to in the FILE option cannot be found in the FCT and SYSID has not been specified.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values: (VSAM)

- 110** A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the EXEC interface block; refer to “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

- 20** Browse operations are not allowed according to the file entry specification in the FCT.
- 25** 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.
- 26** 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.
- 33** An attempt is made to start a browse with a REQID already in use for another browse.
- 42** The KEYLENGTH and GENERIC options are specified, and the length specified in the KEYLENGTH option is less than zero.
- 44** The specified file is a user-maintained or coupling facility data table and the command does not conform to the format of STARTBR for such a data table (for example, RBA is specified).
- 51** A STARTBR command to a KSDS file that is being accessed in RLS mode specifies the RBA keyword. RLS mode does not support RBA access to KSDS files.

Default action: terminate the task abnormally.

IOERR

RESP2 values:

120 There is an I/O error during the file control operation. 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 “EXEC interface block” on page 745 for details.

For a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

Default action: terminate the task abnormally.

ISCVNREQ

RESP2 values:

70 The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

LOADING

RESP2 values:

104 The request cannot be satisfied because it is issued against a data table that is still being loaded. The condition can be raised for one of the following reasons:

- The STARTBR specifies a record that has not yet been loaded into a coupling facility data table. Records can be read while a CFDT is loading only if the requested key is within the range of those records already loaded.

The LOADING response can also be returned for a coupling facility data table that has failed during loading. For more information about what happens if the load for a coupling facility data table fails, see the description of the XDTLC global user exit in the *CICS Customization Guide*.

- The READ specifies the GENERIC or GTEQ options for a user-maintained data table. While a UMT is being loaded, you can issue start browse requests with precise keys only.

If your application programs encounter the LOADING condition persistently or too frequently, check that this is not caused by conflicting file definitions that reference the same data set.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

101 A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTFND

RESP2 values:

80 An attempt to position on a record based on the search argument provided is unsuccessful.

NOTFND can also occur if a generic STARTBR with KEYLENGTH(0) specifies the EQUAL option.

Default action: terminate the task abnormally.

NOTOPEN

RESP2 values:

- 60** NOTOPEN (RESP2 60) is returned for one of the following reasons:
- 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. You can also make CLOSED, UNENABLED the initial state, by specifying STATUS(UNENABLED) and OPENTIME(FIRSTREF) on the FILE resource definition. (For BDAM files, you use the FILSTAT parameter of the DFHFCT TYPE=FILE macro.)
 - The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.
 - A STARTBR command is issued against a data set that is quiesced, or is being quiesced, as a result of a SET DSNAME QUIESCED or IMMQUIESCED command.
 - The requested file is CLOSED and ENABLED, so CICS has tried to open the file as part of executing the request. This file open has failed for some reason. You should examine the console for messages that explain why the file open has been unsuccessful.

This condition does not occur if the request is made to a CLOSED, DISABLED file. In this case, the DISABLED condition occurs.

Default action: terminate the task abnormally.

SYSIDERR

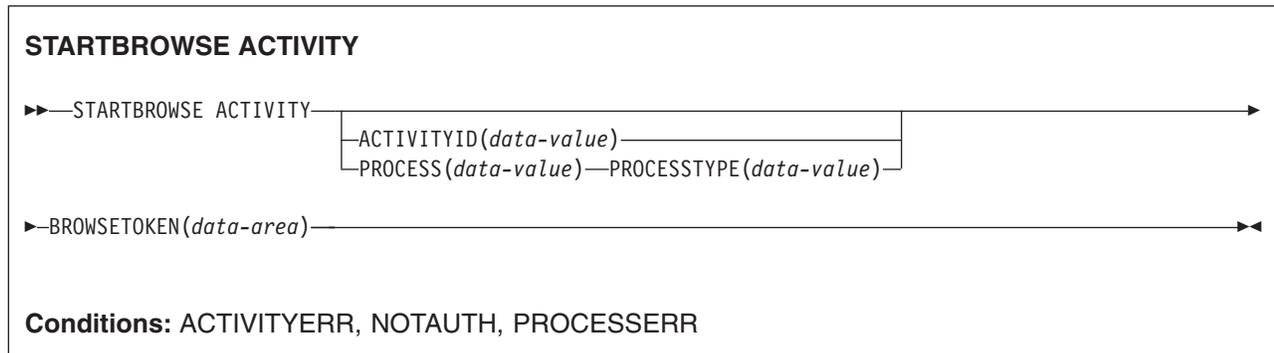
RESP2 values:

- 130** The SYSID option specifies a name that is neither the local CICS region nor a remote system defined by a CONNECTION definition. SYSIDERR also occurs when the link to the remote system is closed.
- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The start browse is operating on a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

STARTBROWSE ACTIVITY

Start a browse of the child activities of a BTS activity, or of the descendant activities
of a process. Although this command appears in the Application Programming set
of topics, to use it you must specify the system programming (SP) parameter in the
EXEC statement of the translate step of your compile job. See Technote 1265081
for further information.



Description

STARTBROWSE ACTIVITY initializes a browse token which can be used to identify either:

- Each child activity of a specified BTS parent activity
- Each descendant activity of a specified BTS process.

If you specify the ACTIVITYID option, the children (but not the grandchildren nor other descendants) of the specified activity can be browsed. This option takes as its argument an activity identifier. This identifier may, for example, have been returned on a previous GETNEXT ACTIVITY command. If it was, the command starts a browse of child activities one level down the activity tree.

If you specify the PROCESS and PROCESSTYPE options, all the descendant activities of the specified process can be browsed. This type of browse is known as a **flat browse**. A flat browse is one which can return every descendant activity exactly once. A parent activity is always returned before its children. The value returned in the LEVEL option of a GETNEXT ACTIVITY command indicates the depth at which the activity lies in the process's activity-tree, with the root activity having a level of zero.

If you specify neither the ACTIVITYID nor the PROCESS and PROCESSTYPE options, the children of the current activity can be browsed.

Options

ACTIVITYID(*data-value*)

specifies the identifier (1–52 characters) of the activity whose child activities are to be browsed.

Typically, the activity identifier specified on this option has been returned on a previous GETNEXT ACTIVITY command (or, in the case of a root activity, on a GETNEXT PROCESS command). ACTIVITYID allows you to start a browse of child activities one level down the activity tree.

If you omit both this and the PROCESS option, the children of the current activity are browsed.

BROWSETOKEN(data-area)

specifies a fullword binary data area, into which CICS will place the browse token.

PROCESS(data-value)

specifies the name (1–36 characters) of the process whose descendant activities are to be browsed.

PROCESSTYPE(data-value)

specifies the process-type (1–8 characters) of the process named on the PROCESS option.

Conditions

ACTIVITYERR

RESP2 values:

- 1 The activity indicated by the ACTIVITYID option could not be found.
- 2 Because neither the ACTIVITYID nor the PROCESS options were specified, a browse of the children of the current activity was implied—but there is no current activity associated with the request.
- 19 The request timed out. It may be that another task using this activity-record has been prevented from ending.
- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

- 101 The user associated with the issuing task is not authorized to access the file whose data set contains the records to be browsed.

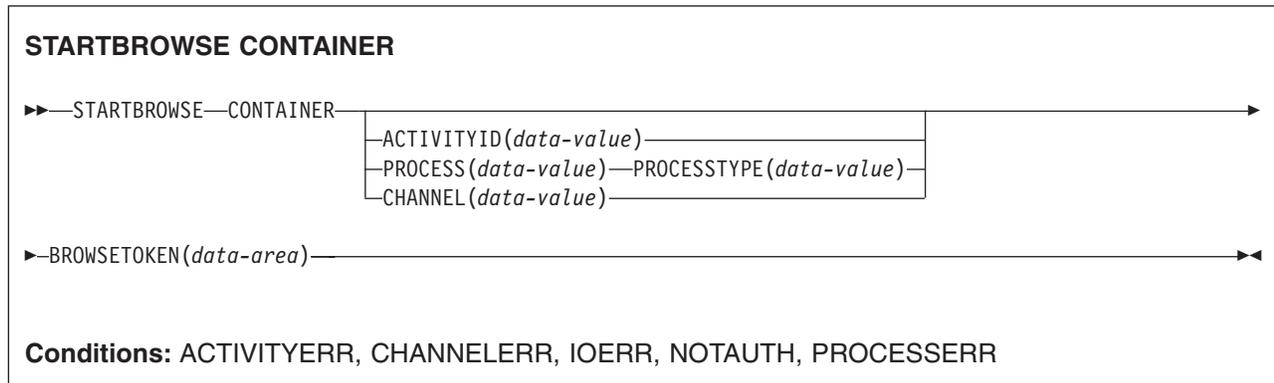
PROCESSERR

RESP2 values:

- 3 The process specified on the PROCESS option could not be found.
- 4 The process-type specified on the PROCESSTYPE option could not be found.

STARTBROWSE CONTAINER

Start a browse of the containers associated with a channel, or with a BTS activity or
process. Although this command appears in the Application Programming set of
topics, to use it you must specify the system programming (SP) parameter in the
EXEC statement of the translate step of your compile job. See Technote 1265081
for further information.



Description

STARTBROWSE CONTAINER initializes a browse token which can be used to identify the name of each data-container associated with a specified channel, or with a BTS activity or process.

Note: The browse token should be used only by the program that issues the STARTBROWSE command.

If you specify none of the ACTIVITYID, PROCESS, or CHANNEL options, CICS examines the context (channel or BTS) of the request. If a current channel exists, its containers are browsed. If a current activity exists, its containers are browsed. If neither exists, an ACTIVITYERR 2 is raised: see the description of the ACTIVITYERR condition, below.

Options

ACTIVITYID(data-value)

specifies the identifier (1–52 characters) of the activity whose containers are to be browsed.

Typically, the identifier specified on this option has been returned on a previous GETNEXT ACTIVITY command.

BROWSETOKEN(data-area)

specifies a fullword binary data area, into which CICS will place the browse token.

CHANNEL(data-value)

specifies the name (1–16 characters) of the channel whose containers are to be browsed. This must be the name of either the current channel or of a channel created by the program that issues the STARTBROWSE CONTAINER command.

If this option is not specified, and the current context is channel, the current channel's containers are browsed.

#

The order in which containers are returned is undefined.

PROCESS(data-value)

specifies the name (1–36 characters) of the process whose containers are to be browsed.

Note: The containers associated with a process (*process containers*) are globally available throughout the process. They are not the same as the root activity's containers.

PROCESSTYPE(data-value)

specifies the process-type (1–8 characters) of the process named on the PROCESS option.

Conditions

ACTIVITYERR

RESP2 values:

- 1 The activity indicated by the ACTIVITYID option could not be found.
- 2 None of the ACTIVITYID, PROCESS, or CHANNEL options were specified and there is no current channel and no current activity associated with the request.
- 29 The repository file is unavailable.
- 30 An input/output error has occurred on the repository file.

CHANNELERR

RESP2 values:

- 2 The channel specified on the CHANNEL option could not be found.

IOERR

RESP2 values:

- 30 An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

- 101 The user associated with the issuing task is not authorized to access this resource in the way requested.

PROCESSERR

RESP2 values:

- 3 The process specified on the PROCESS option could not be found.
- 4 The process-type specified on the PROCESSTYPE option could not be found.
- 13 The request timed out. It may be that another task using this process-record has been prevented from ending.

STARTBROWSE EVENT

Start a browse of events known to a BTS activity. Although this command appears
in the Application Programming set of topics, to use it you must specify the system
programming (SP) parameter in the EXEC statement of the translate step of your
compile job. See Technote 1265081 for further information.

STARTBROWSE EVENT

▶▶—STARTBROWSE—EVENT—BROWSETOKEN(*data-area*)—┬—ACTIVITYID(*data-value*)—┘▶▶

Conditions: ACTIVITYERR, INVREQ, IOERR, NOTAUTH

Description

STARTBROWSE EVENT initializes a browse token which can be used to identify each event (including each sub-event and system event) that is within the scope of a specified BTS activity. If you do not specify an activity, events within the scope of the current activity are browsed.

A browse started by STARTBROWSE EVENT returns:

- Atomic events. An atomic event returned on this command may or may not be included in the predicate of a composite event—that is, it may or may not be a sub-event.
- Composite events.
- System events.

Options

ACTIVITYID(*data-value*)

specifies the identifier (1–52 characters) of the activity whose events are to be browsed.

If you omit this option, events known to the current activity are browsed.

BROWSETOKEN(*data-area*)

specifies a fullword binary data area, into which CICS will place the browse token.

Conditions

ACTIVITYERR

RESP2 values:

- 1** The activity identifier specified on the ACTIVITYID option does not relate to any activity that is within the scope of this task.
- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

INVREQ

RESP2 values:

- 1** There is no current activity within the scope of this task.

IOERR

RESP2 values:

30 An input/output error has occurred on the repository file.**NOTAUTH**

RESP2 values:

101 The user associated with the issuing task is not authorized to access this resource in the way requested.

STARTBROWSE PROCESS

Start a browse of all processes of a specified type within the CICS business
transaction services system. Although this command appears in the Application
Programming set of topics, to use it you must specify the system programming (SP)
parameter in the EXEC statement of the translate step of your compile job. See
Technote 1265081 for further information.

STARTBROWSE PROCESS

▶▶—STARTBROWSE—PROCESS—PROCESSTYPE(*data-value*)—BROWSETOKEN(*data-area*)—▶▶

Conditions: IOERR, NOTAUTH, PROCESSERR

Description

STARTBROWSE PROCESS initializes a browse token which can be used to identify each process of a specified type within the CICS business transaction services system.

When you add a process to the BTS system, you use the PROCESSTYPE option of the DEFINE PROCESS command to categorize it. You specify the name of a PROCESSTYPE resource definition, which in turn names a CICS file definition that maps to a physical VSAM data set (the repository) on which details of the process and its constituent activities will be stored. (Records for multiple process-types can be stored on the same repository data set.)

The STARTBROWSE PROCESS command enables you to start a browse of processes of a specified type.

Options

BROWSETOKEN(*data-area*)

specifies a fullword binary data area, into which CICS will place the browse token.

PROCESSTYPE(*data-value*)

specifies the process-type (1–8 characters) of the processes to be browsed.

Conditions

IOERR

RESP2 values:

29 The repository file is unavailable.

30 An input/output error has occurred on the repository file.

NOTAUTH

RESP2 values:

101 The user associated with the issuing task is not authorized to access this resource in the way requested.

PROCESSERR

RESP2 values:

- 1** No processes of this process-type could be found.
- 4** The process-type specified on the PROCESSTYPE option could not be found.
- 13** The request timed out. It may be that another task using this process-record has been prevented from ending.

SUSPEND

Suspend a task.

SUSPEND

▶▶—SUSPEND—◀◀

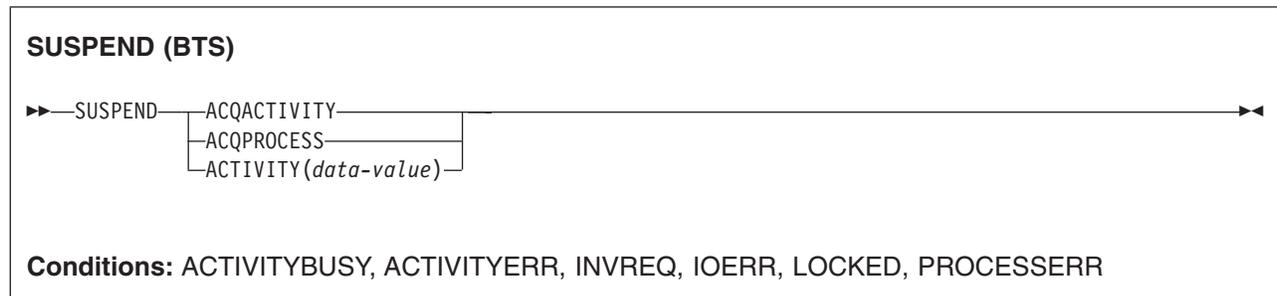
This command is threadsafe.

Description

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.

SUSPEND (BTS)

Suspend a BTS process or activity.



Description

SUSPEND (BTS) prevents a BTS process or activity being reattached when events in its event pool are fired.

The only process a program can suspend is the one that it has acquired in the current unit of work.

The only activities a program can suspend are as follows:

- If it is running as the activation of an activity, its own child activities. It can suspend several of its child activities within the same unit of work.
- The activity it has acquired, by means of an ACQUIRE ACTIVITYID command, in the current unit of work.

To resume a suspended process or activity, a RESUME command must be issued.

Options

ACQACTIVITY

specifies that the activity to be suspended is the one that the current unit of work has acquired by means of an ACQUIRE ACTIVITYID command.

ACQPROCESS

specifies that the process that is currently acquired by the requestor is to be suspended.

ACTIVITY(*data-value*)

specifies the name (1–16 characters) of the child activity to be suspended.

Conditions

ACTIVITYBUSY

RESP2 values:

- 19** The request timed out. It may be that another task using this activity-record has been prevented from ending.

ACTIVITYERR

RESP2 values:

- 8** The activity named on the ACTIVITY option could not be found.

INVREQ

RESP2 values:

- 4** The **ACTIVITY** option was used to name a child activity, but the command was issued outside the scope of a currently-active activity.
- 14** The activity is in **COMPLETE** or **CANCELLING** mode, and therefore cannot be suspended.
- 15** The **ACQPROCESS** option was used, but the unit of work that issued the request has not acquired a process.
- 24** The **ACQACTIVITY** option was used, but the unit of work that issued the request has not acquired an activity.

IOERR

RESP2 values:

- 29** The repository file is unavailable.
- 30** An input/output error has occurred on the repository file.

LOCKED

The request cannot be performed because a retained lock exists against the relevant record on the repository file.

PROCESSERR

RESP2 values:

- 5** The process could not be found.

SYNCPOINT

Establish a syncpoint.

SYNCPOINT



Conditions: INVREQ, ROLLEDBACK

Description

SYNCPOINT divides a task (usually a long-running one) into smaller units of work. It specifies that all changes to recoverable resources made by the task since its last syncpoint are to be committed.

Note: A failure occurring during the commit phase (phase 2) of syncpoint processing does not return an error condition and the transaction is not abnormally terminated. Subsequent units of work in the transaction are allowed to continue normally. See the *CICS Recovery and Restart Guide* for further information.

Conditions

INVREQ

RESP2 values:

200 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.

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 unit of work are backed out.

Default action: terminate the task abnormally.

SYNCPOINT ROLLBACK

Back out to last syncpoint.

SYNCPOINT ROLLBACK

▶—SYNCPOINT—ROLLBACK—▶

Condition: INVREQ

Options

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 unit of work 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 unit of work 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.

Note: A failure occurring during the backout phase (phase 2) of syncpoint processing does not return an error condition and the transaction is not abnormally terminated. Subsequent units of work in the transaction are allowed to continue normally. See *CICS Recovery and Restart Guide* for further information.

Conditions

INVREQ

RESP2 values:

- 200** 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

Default action: terminate the task abnormally.

TEST EVENT

Test whether a BTS event has fired.

TEST EVENT

▶▶—TEST—EVENT(*data-value*)—FIRESTATUS(*cvda*)————▶▶

Conditions: EVENTERR, INVREQ

Description

TEST EVENT tests whether a named BTS event has occurred (fired).

Options

EVENT(*data-value*)

specifies the name (1–16 characters) of the event to test for completion.

FIRESTATUS(*cvda*)

FIRESTATUS returns the fire status of the event. CVDA values are:

FIRED The event has fired.

NOTFIRED

The event has not fired.

Conditions

EVENTERR

RESP2 values:

4 The event specified on the EVENT option is not recognized by BTS.

INVREQ

RESP2 values:

1 The command was issued outside the scope of an activity.

UNLOCK

Release exclusive control.

UNLOCK

▶—UNLOCK—FILE(*filename*)—
 └─TOKEN(*data-area*)—┘ └─SYSID(*systemname*)—┘

Conditions: DISABLED, FILENOTFOUND, ILLOGIC, INVREQ, IOERR, ISCINVREQ, NOTAUTH, NOTOPEN, SYSIDERR

Description

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 recoverable file (other than one that refers to a coupling facility data table), the resource remains locked until either a syncpoint 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.

If the UNLOCK command refers to a record in a recoverable coupling facility data table the record lock is released immediately provided that the task has not made any previous change to the same record (or added it as a new record) within the current unit of work. If changes have been made to the record, or it is a new record added to the table, it remains locked until either a syncpoint command is executed or the task is terminated.

If an UNLOCK command does not have a token, an attempt is made to match it to either a read with the UPDATE option that also does not have a token, or to a WRITE MASSINSERT. If neither of these is found, no action is taken and a NORMAL response is returned.

Use this command to terminate a VSAM WRITE MASSINSERT operation against a VSAM file.

Releasing locks when updating in browse

The UNLOCK command does not release locks held against records locked in response to READNEXT or READPREV commands that specify the update option. It only invalidates the TOKEN value so that it cannot be used to complete an update.

Options

FILE(*filename*)

specifies the name 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-area*)

specifies as a fullword binary value a unique request identifier for an UNLOCK, used to associate it with a previous READ, READNEXT, or READPREV command that specified the UPDATE option.

If you specify UNLOCK with the TOKEN returned on a READNEXT UPDATE or READPREV UPDATE command for a file accessed in RLS mode, the unlock command simply invalidates the TOKEN value so that it cannot be used to complete an update. It does not release the record lock.

TOKEN can be function shipped. However, if a request specifying TOKEN is function shipped to CICS region that does not support this keyword, the request fails.

Conditions

DISABLED

RESP2 values:

50 A file is disabled because it was initially defined as disabled and has not since been enabled.

A file is disabled by an EXEC CICS 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

RESP2 values:

1 A file name referred to in the FILE option cannot be found in the FCT and SYSID has not been specified.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values: (VSAM and CICS-maintained data tables)

110 A VSAM error occurs that does not fall within one of the other CICS response categories.

(See EIBRCODE in the EXEC interface block; refer to “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

47 An unlock includes a token whose value cannot be matched against any token in use for an existing READ with the UPDATE option.

48 An attempt is made to function-ship a request which includes a TOKEN keyword.

Default action: terminate the task abnormally.

IOERR

RESP2 values:

120 There is an I/O error during the file control operation. 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 “EXEC interface block” on page 745 for details.)

For a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

Default action: terminate the task abnormally.

ISCVREQ

RESP2 values:

70 The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

101 A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTOPEN

RESP2 values:

60 NOTOPEN (RESP2 60) is returned for one of the following reasons:

- 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. You can also make CLOSED, UNENABLED the initial state, by specifying STATUS(UNENABLED) and OPENTIME(FIRSTREF) on the FILE resource definition. (For BDAM files, you use the FILSTAT parameter of the DFHFCT TYPE=FILE macro.)
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.
- An UNLOCK command is issued against a data set that is quiesced, or is being quiesced, as a result of a SET DSNAME QUIESCED or IMMQUIESCED command.
- The requested file is CLOSED and ENABLED, so CICS has tried to open the file as part of executing the request. This file open has failed for some reason. You should examine the console for messages that explain why the file open has been unsuccessful.

This condition does not occur if the request is made to a CLOSED, DISABLED file. In this 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

RESP2 values:

- 130** The SYSID option specifies a name that is neither the local CICS region nor a remote system defined to CICS by a CONNECTION definition. SYSIDERR also occurs when the link to the remote system is closed.
- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The UNLOCK is issued against a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

UPDATE COUNTER and UPDATE DCOUNTER

Update the current value.

UPDATE COUNTER

```
▶▶ UPDATE COUNTER(name) [ POOL(name) ] VALUE(data-value) [ COMPAREMIN(data-value) ]
▶ [ COMPAREMAX(data-value) ] ▶▶
```

Conditions: INVREQ, SUPPRESSED

UPDATE DCOUNTER

```
▶▶ UPDATE DCOUNTER(name) [ POOL(name) ] VALUE(data-area) [ COMPAREMIN(data-area) ]
▶ [ COMPAREMAX(data-area) ] ▶▶
```

Conditions: INVREQ, SUPPRESSED

Description

These counter commands set a new current value for the named counter. COUNTER operates on fullword signed counters and DCOUNTER on doubleword unsigned counters.

You can use the COMPAREMAX and COMPAREMIN options to set a new current value only if it falls within a specified range, or is above or below a specified value.

For information about specifying fullword and doubleword variables on these named counter commands, see “CICS command argument values” on page 3.

Options

COMPAREMAX(*data-value*)

specifies, as a fullword signed binary value (doubleword unsigned binary value for DCOUNTER), a value to be compared with the named counter's current value, and makes the result of the UPDATE command conditional on the comparison:

- If the current value to be assigned is less than, or equal to, the value specified on the COMPAREMAX parameter, the current value is reset, with response normal

- If the current value is greater than the specified value, CICS returns an exception condition.

The value you specify on the COMPAREMAX parameter can be less than the value on the COMPAREMIN parameter, in which case the current value is considered to be in range if it satisfies the COMPAREMIN *or* the COMPAREMAX comparison. In the normal case where the COMPAREMIN value is less than the COMPAREMAX value, the current value must satisfy both comparisons (that is, it must lie between the two values).

COMPAREMIN (*data-value*)

specifies, as a fullword signed binary value (doubleword unsigned binary value for DCOUNTER), a value to be compared with the named counter's current value, and makes the result of the UPDATE command conditional on the comparison:

- If the current value to be assigned is equal to, or greater than, the value specified on the COMPAREMIN parameter, the CICS resets the current value, with response normal
- If the current value is less than the specified value, CICS returns an exception condition.

Note: The value you specify on the COMPAREMIN parameter can be greater than the value on the COMPAREMAX parameter. See the COMPAREMAX parameter for the effect of this.

COUNTER (*name*)

specifies the name of the named counter for which the current number is to be reset to the value specified on the value parameter. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

DCOUNTER (*name*)

specifies the name of the named counter for which the current number is to be reset to the value specified on the value parameter. The name can be up to 16 alphanumeric characters. If *name* is a variable that contains a name that is less than 16 characters, the name must be padded with trailing blanks.

POOL (*poolname*)

specifies the name of the pool in which the named counter resides.

Valid characters for the pool selector string are A through Z, 0 through 9, \$ @ # and _ (underscore). If *name* is a variable that contains a name that is less than 8 characters, the name must be padded with trailing blanks.

This parameter is optional. If you omit the name of the pool, a pool selector value of 8 blanks is assumed.

If there is no matching entry in the DFHNCOPT options table, CICS uses the name specified on the NCPLDFT system initialization parameter, which specifies the default named counter pool.

For information about generating a named counter options table using the DFHNCO macro, see the *CICS Application Programming Guide*.

VALUE (*data-value*)

specifies the new number to be set as the current value for the named counter, using a fullword signed binary value for COUNTER and a doubleword unsigned value for DCOUNTER.

Conditions

INVREQ

RESP2 values:

- 201** Named counter not found.
- 301** The server has reported an error code that is not understood by the named counter interface. Generally, this is not possible unless the interface load module, DFHNCIF, is at a lower maintenance or release level than the server itself.
- 303** An unexpected error, such as structure failure or loss of connectivity, has occurred on a macro used to access the coupling facility. Further information can be found in message DFHNC0441 in the CICS job log.
- 304** The pool selection parameter specified in the program cannot be resolved to a valid server name using the current options table.
- 305** The interface is unable to establish a connection to the server for the selected named counter pool. Further information can be found in an AXM services message (AXMSC $nnnn$) in the CICS job log.
- 306** An abend occurred during server processing of a request. Further information can be found in a message in the CICS job log and the server job log.
- 308** The DFHNCOPT options table module, required for resolving a pool name, could not be loaded.
- 309** During processing of the options table, the named counter interface encountered an unknown entry format. Either the options table is not correctly generated, or the DFHNCIF interface load module is not at the same release level as the options table.
- 310** An options table entry matching the given pool name specified a user exit program, but the user exit program is not link-edited with the options table and cannot be loaded.
- 311** A response from the named counter server to the client region interface module, DFHNCIF, indicates that a system-managed rebuild is in progress but the EXEC CICS interface does not recognize the condition. This means that the CICS region is at CICS TS 2.1 or earlier.
- 403** The POOL parameter contains invalid characters or embedded spaces.
- 404** The COUNTER parameter contains invalid characters or embedded spaces.
- 406** The VALUE parameter is invalid. You cannot set the current value to less than the minimum value, or greater than the maximum value plus 1.

Default action: terminate the task abnormally.

SUPPRESSED

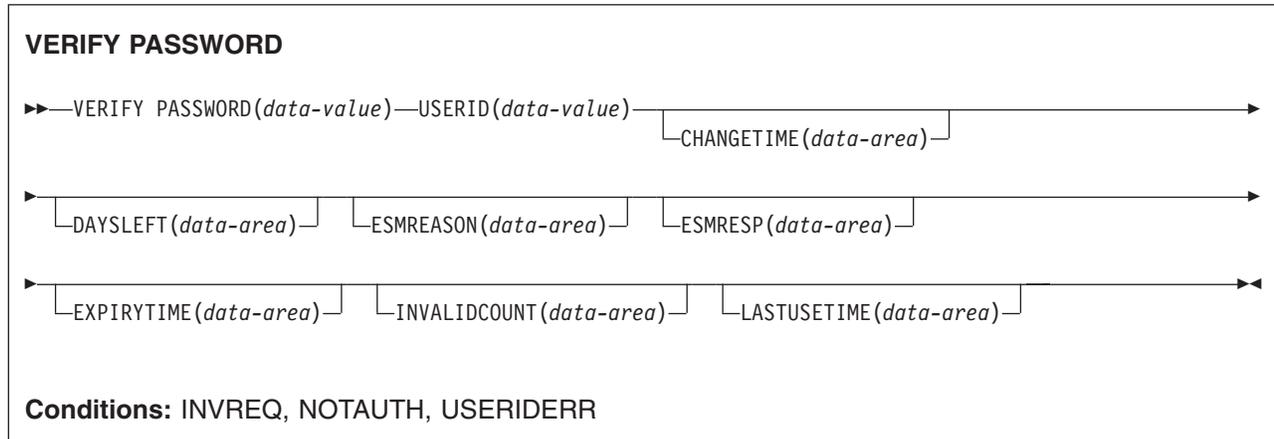
RESP2 values:

- 103** The current value of the named counter is:
 - not within the range specified by the COMPAREMAX and COMPAREMIN parameters, when both are specified
 - greater than the COMPAREMAX parameter or less than the COMPAREMIN parameter, when only one option is specified.

Default action: terminate the task abnormally.

VERIFY PASSWORD

Allow an application to check that a password matches the password recorded by an external security manager.



Description

The VERIFY PASSWORD command 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.

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. The userid might not be able to sign on in the CICS region, for example, because:

- The userid might not be authorized to access the CICS address space (identified by the APPLID).
- The userid might not be authorized to use the terminal at which the user is signing on (identified by the TERMINAL class).

Attention: 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.

Note: In the CHANGETIME, LASTUSETIME, and EXPIRYTIME options, the time value returned is in the same format as the ASKTIME command, that is, in ABSTIME units. ABSTIME is the time, in packed decimal, since 00:00 on 1 January 1900 (in milliseconds rounded to the nearest hundredth of a second). The data can be reformatted as a date and time, in a format specified by the caller, by using the FORMATTIME command.

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.

The external security manager does not always return response and reason
codes to CICS. Make sure that you check the EIBRESP and EIBRESP2 values
returned by this command in addition to checking the ESMRESP and
ESMREASON values.

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.

The external security manager does not always return response and reason
codes to CICS. Make sure that you check the EIBRESP and EIBRESP2 values
returned by this command in addition to checking the ESMRESP and
ESMREASON values.

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, in a halfword binary field, that 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.

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.

Conditions

INVREQ

RESP2 values:

13 There is an unknown return code in ESMRESP from the external security manager.

18 The CICS external security manager interface is not initialized.

29 The external security manager is not responding.

32 The userid field contains a blank character in an invalid position.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

2 The supplied password is wrong. If the external security manager is RACF, the revoke count maintained by RACF is incremented.

3 A new password is required.

19 The user has been revoked.

20 The user's connection to their default group has been revoked.

Default action: terminate the task abnormally.

USERIDERR

RESP2 values:

8 The USERID is not known to the external security manager.

Default action: terminate the task abnormally.

WAIT CONVID (APPC)

Ensure accumulated data is transmitted on an APPC mapped conversation.

WAIT CONVID (APPC)

▶—WAIT CONVID(*name*)—
└─STATE(*cvda*)┘

Conditions: INVREQ, NOTALLOC

Description

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.

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

Conditions

INVREQ

RESP2 values:

- 200** Command not supported for distributed program link when it refers to the principal facility.

also occurs (RESP2 not set) if the command is used on a conversation that is not using the EXEC CICS interface or that is not a mapped conversation.

Default action: terminate the task abnormally.

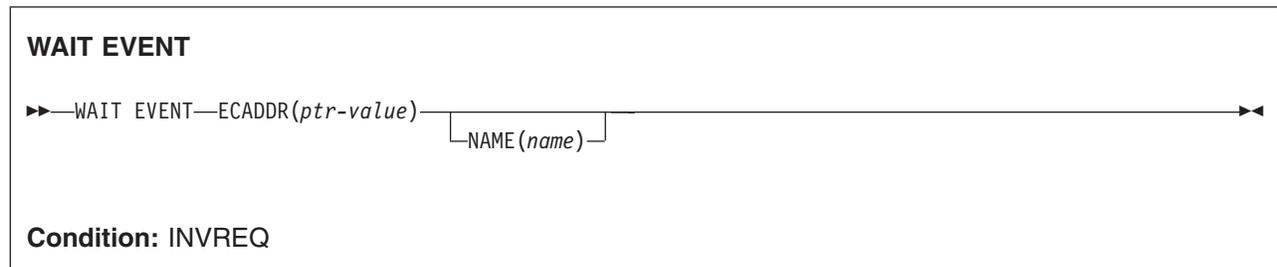
NOTALLOC

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

Default action: terminate the task abnormally.

WAIT EVENT

Wait for an event to occur.



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 Application Programming Guide* for more information about transaction affinities.

Description

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 371. 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 WAIT facility when it runs out of work.

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.

#

Make sure that asynchronous cross memory post (posting completion of an event in an address space other than the user's own) is not used more frequently than necessary. Large numbers of cross memory posts can consume excessive amounts of system resources.

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 INQUIRE TASK command or a CEMT INQ TASK.

Conditions

INVREQ

RESP2 values:

- 2 The ECB address is a null pointer, (X'00000000') or (X'FF000000').
- 3 The specified event control area address is above the 16MB line for programs executing in 24-bit mode.
- 4 The event control area address is not aligned on a fullword boundary.
- 6 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.

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.

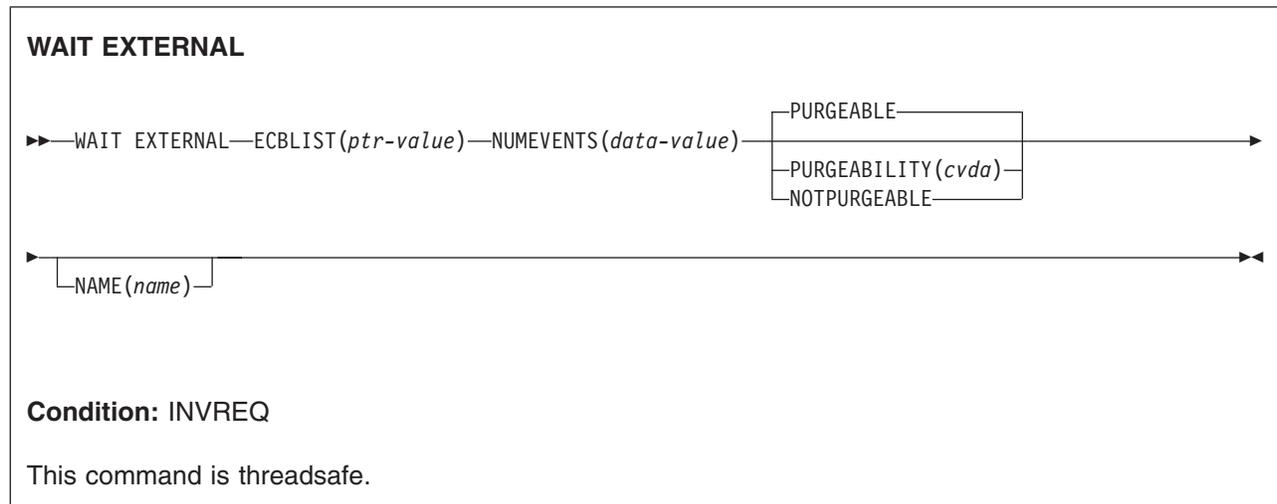
Examples

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 EXTERNAL

Synchronize events.



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 Application Programming Guide* for more information about transaction affinities.

Description

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.

Make sure that asynchronous cross memory post (posting completion of an event in
 # an address space other than the user's own) is not used more frequently than
 # necessary. Large numbers of cross memory posts can consume excessive amounts
 # of system resources.

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, 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 PURGEIFORCEPURGE command
- A CEMT SET TASK PURGEIFORCEPURGE

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 Recovery and Restart Guide* for information about DTIMOUT and SET TASK PURGEIFORCEPURGE.

Conditions

INVREQ

RESP2 values: CVDA values are:

- 1 An ECB is not valid, for example the ECB is not fullword aligned.
- 2 The ECB address is a null pointer, (X'00000000') or (X'FF000000').

- 3 NUMEVENTS is not a positive number.
- 4 PURGEABILITY is specified with an incorrect CVDA.
- 5 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.

The ECBs specified are in read-only storage.

Default action: terminate the task abnormally.

Examples

The following figure 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.

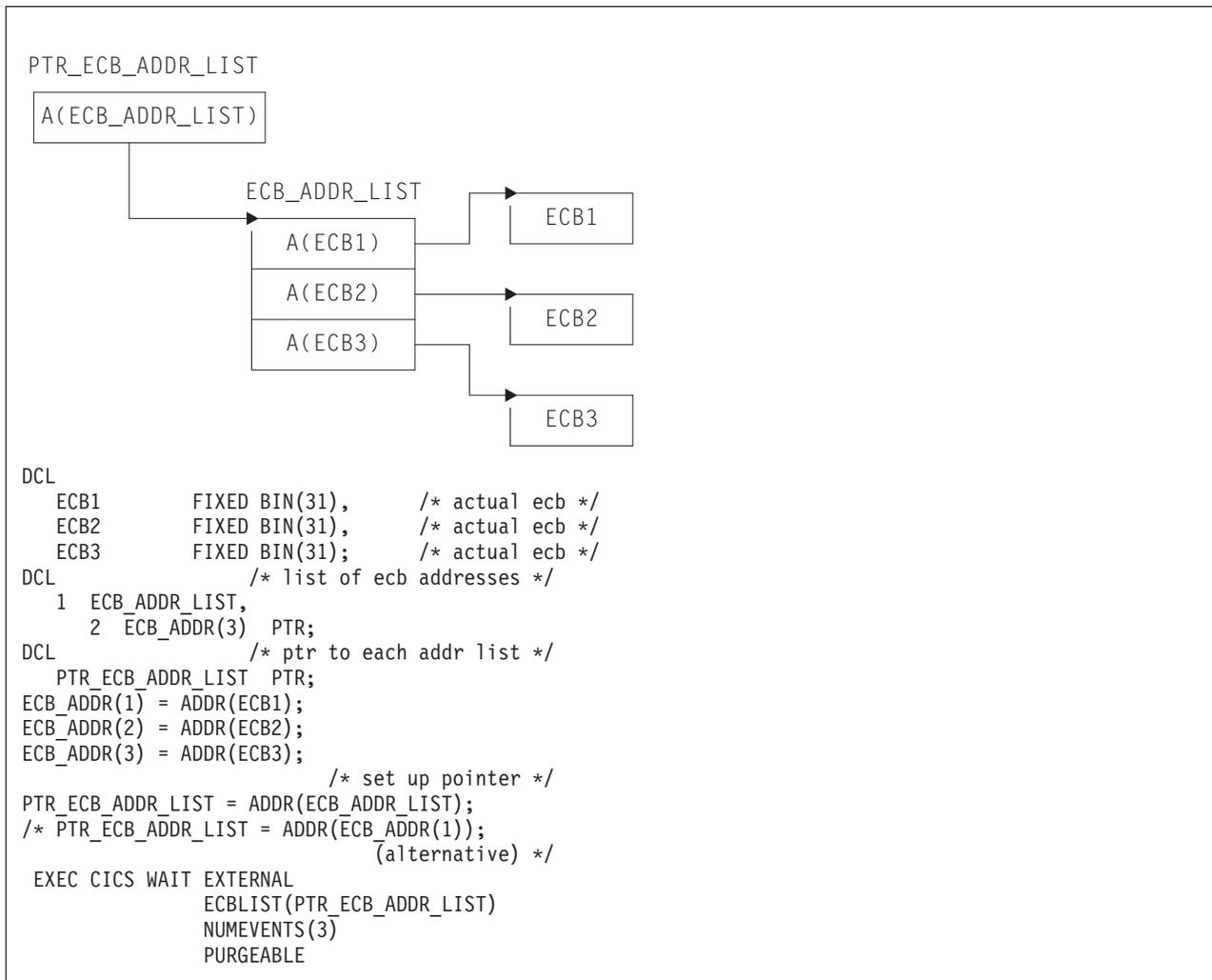


Figure 4. ECBLIST option, EXEC CICS WAIT EXTERNAL

WAIT JOURNALNAME

Synchronize the task with journal output being written to a system logger log stream.

WAIT JOURNALNAME

▶—WAIT JOURNALNAME(*data-value*)—▶
└─REQID(*data-value*)─┘

Conditions: IOERR, JIDERR, NOTOPEN

Description

WAIT JOURNALNAME 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 may already be written out from the journal buffer area to the corresponding system logger log stream, or the system logger output operation may be in progress. If the log stream 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 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 been output to the system logger log stream. This may be done either by issuing a stand-alone WAIT JOURNALNAME command, or by making the last output command itself synchronous (by specifying the WAIT option in the WRITE JOURNALNAME command).

Options

JOURNALNAME(*data-value*)

specifies a 1– to 8– character journal name identifying the journal on which the task is to wait for synchronization. The name must be a journal name known to CICS.

To issue the wait against the CICS system log, specify DFHLOG as the journal name.

To issue the wait against journals defined using the journal numbering convention, as in file resource definitions, specify the name as DFHJnn, where nn is the journal number in the range 1 to 99.

Note: Specifying DFHJ01 on this command refers to a user journal, *not* the system log.

REQID(*data-value*)

specifies, in a fullword binary variable, the token that refers to a journal record

that has been created but possibly not yet written out. The token is returned by CICS from a previous WRITE JOURNALNAME command issued by this task.

If you do not specify REQID, the task is synchronized with the output of the current buffer for the journal specified by JOURNALNAME.

Conditions

IOERR

a journal record was not output because of an irrecoverable error condition returned by the system logger or by SMF.

Default action: If the log is the system log, CICS either quiesces or abends. If it is a general log, the task is terminated abnormally.

JIDERR

occurs in either of the following situations:

- The specified journal name is not known in the CICS region.
- The specified journal name refers to a DASD-only log stream to which a CICS region in another MVS image is currently connected.

Default action: terminate the task abnormally.

NOTOPEN

occurs in any of the following situations:

- The command cannot be satisfied because the user explicitly disabled the specified journal.
- A wait request is issued against a journal that has not previously been written to.
- This journal was defined using a model that maps it into the logstream used by the system log for this system. This error is not detected when trying to connect to the logstream. A definition for the journal will be installed and set to failed.

Default action: terminate the task abnormally.

Examples

The following example shows how to request synchronization with the output of journal records written to a user journal named 'ACCOUNTS'

```
EXEC CICS WAIT JOURNALNAME('ACCOUNTS')  
      REQID(RECTOKEN)
```

WAIT JOURNALNUM

Synchronize with journal output.

This command is supported for compatibility with earlier releases of CICS. It is superseded by the WAIT JOURNALNAME command, which you are recommended to use instead.

The syntax is the same as for WAIT JOURNALNAME except that JOURNALNUM specifies a numeric instead of a character value. The numeric value, nn, is in the range 01-99 and corresponds to journal name DFHJnn.

WAIT SIGNAL

Suspend task on a logical unit.

WAIT SIGNAL

▶—WAIT SIGNAL—▶

Conditions: NOTALLOC, SIGNAL, TERMERR

Description

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

Conditions

NOTALLOC

occurs if the principal facility of the task is not a terminal.

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

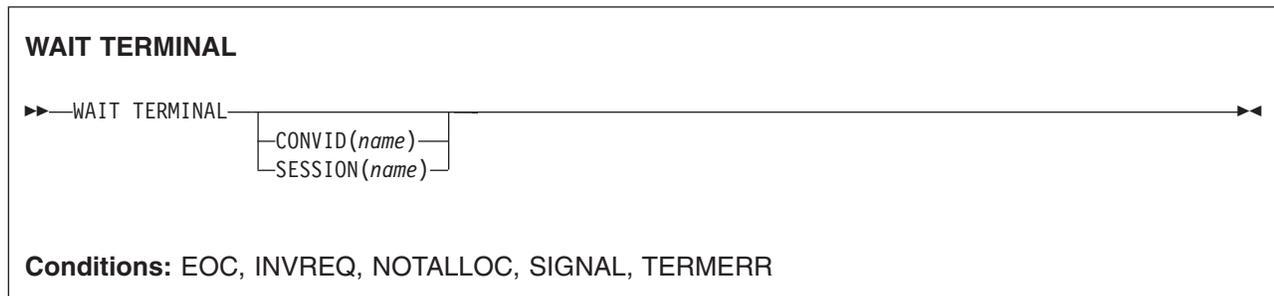
occurs for a terminal-related error.

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

Ensure terminal operation has completed.



Description

WAIT TERMINAL ensures that terminal operation has completed.

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.

Conditions

#

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.

INVREQ

RESP2 values:

200 A distributed program link server application specified the function-shipping session (its principal facility) on the CONVID 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.

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.

#

TERMERR

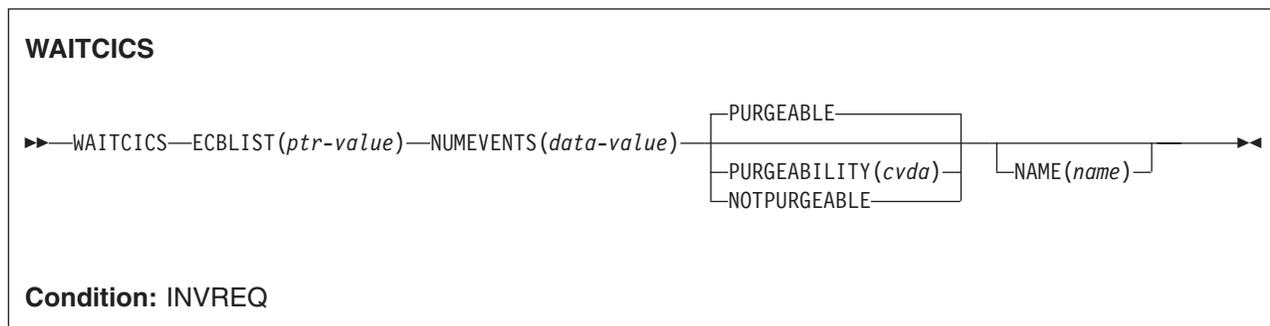
occurs for a terminal-related error.

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.

WAITCICS

Synchronize events.



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 Application Programming Guide* for more information about transaction affinities.

Description

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.

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 EXEC CICS 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 Recovery and Restart Guide* for information about DTIMOUT and *CICS System Programming Reference* for information about SET TASK PURGEIFORCEPURGE.

Conditions**INVREQ**

RESP2 values:

- 1 An ECB is not valid, for example the ECB is not fullword aligned.
- 3 NUMEVENTS is not a positive number.
- 4 PURGEABILITY is specified with an incorrect CVDA.
- 5 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.

The ECBs specified are in read-only storage.

Default action: terminate the task abnormally.

Examples

The following figure 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.

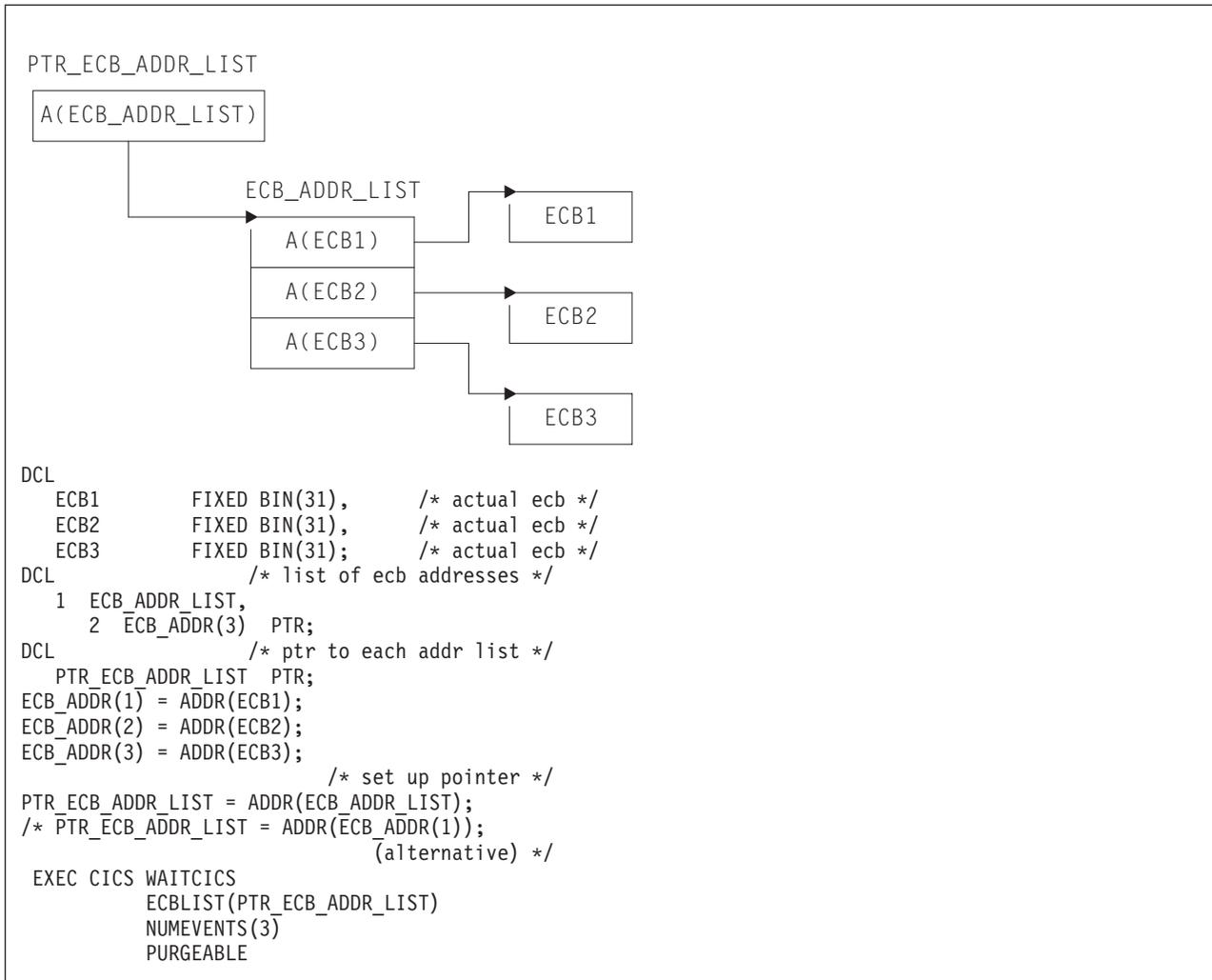


Figure 5. ECBLIST option, EXEC CICS WAITCICS

WEB CLOSE

Closes a connection to a server for CICS as an HTTP client.

WEB CLOSE

►► WEB CLOSE SESSTOKEN(*data-value*) ◄◄

Conditions: NOTOPEN

This command is threadsafe.

Description

WEB CLOSE enables an application program to close a connection with a server. The session token identifies the connection that is to be closed. When the connection is closed, the session token that applies to it is no longer valid for use.

The session token is required to receive a response from the server and to read the HTTP headers for the response, so the WEB CLOSE command should not be issued until all interaction with the server and with the response that it sends is complete. The command releases CICS resources involved with the connection.

The WEB CLOSE command does not cause CICS to notify the server that the connection should be terminated. It only makes CICS close the connection on the client side. On the final request that you make using the connection, you should specify the CLOSESTATUS(CLOSE) option on the WEB SEND or WEB CONVERSE command. When this option is specified, CICS writes a Connection: close header on the request, or, for a server at HTTP/1.0 level, omits the Connection: Keep-Alive header. The information in the headers means that the server can close its connection with you immediately after sending the final response, rather than waiting to see if you send further requests before timing out.

The connection might also be closed at the request of the server before the WEB CLOSE command is issued. If you need to test whether the server has requested termination of the connection, use the WEB READ HTTPHEADER command to look for the Connection: close header in the last message from the server.

If the server does request termination of the connection, the data relating to that connection is still kept available within CICS until the WEB CLOSE command is issued. The available data includes the most recent message received from the server, and the parameters used to open the connection (such as the scheme and the host name of the server). When a server has terminated the connection, the application program cannot:

- Send further requests on that connection, using the WEB SEND or WEB CONVERSE commands.
- Write HTTP headers, using the WEB WRITE HTTPHEADER command.

However, the application program can still:

- Receive a response, using the WEB RECEIVE command.
- Examine HTTP headers, using the WEB READ HTTPHEADER and HTTP header browsing commands.
- Extract connection information, using the WEB EXTRACT command.

When the WEB CLOSE command is issued, the data relating to the connection is cleared.

If the WEB CLOSE command is not issued by the application program, then at end of task CICS clears the data relating to the connection and closes the connection, if it has not already been closed.

Options

SESSTOKEN(*data-value*)

specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. When you issue the WEB CLOSE command for the connection identified by the session token, CICS ends that connection and clears the data associated with it, and makes the session token invalid for further use by the application program. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

Conditions

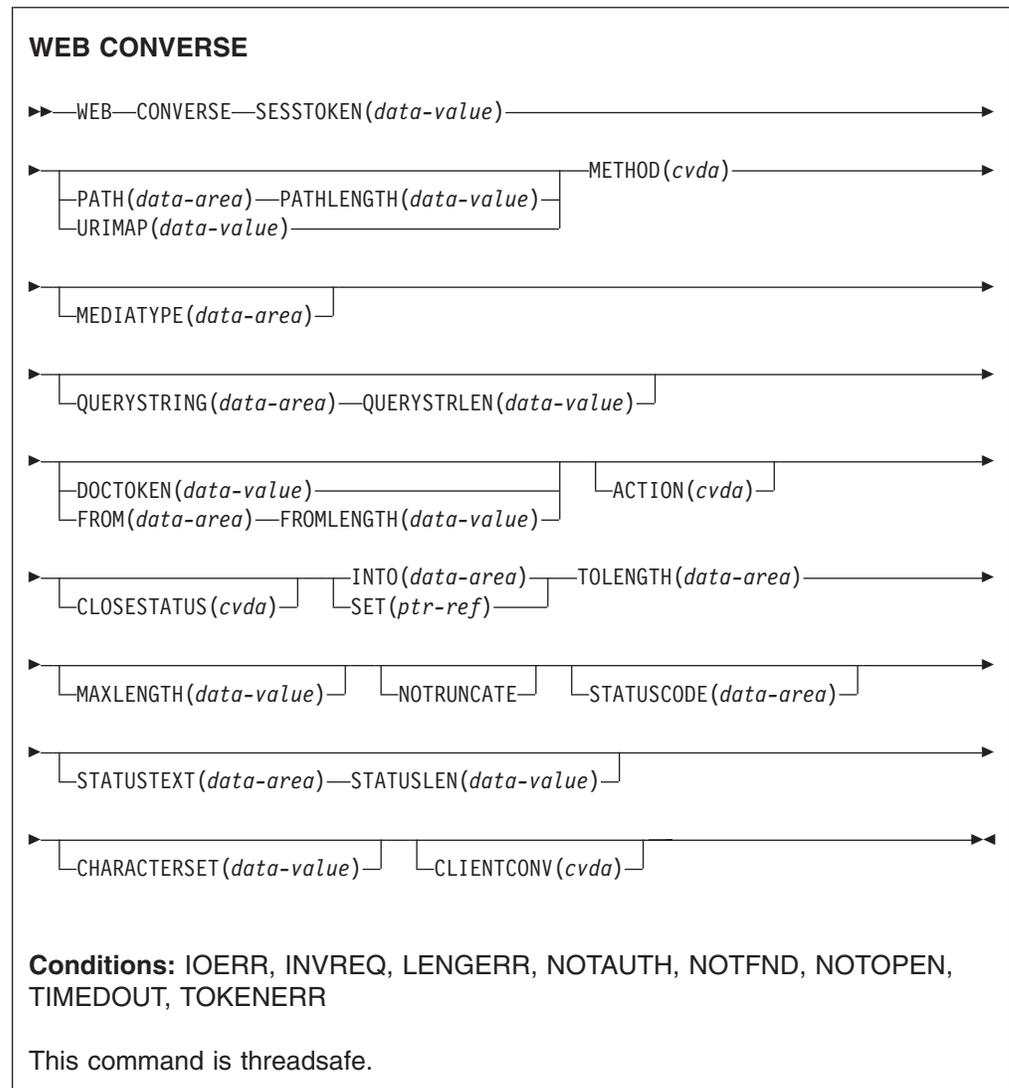
NOTOPEN

RESP2 values are:

27 Invalid session token.

WEB CONVERSE

Send an HTTP request by CICS as an HTTP client, and receive a response from the server, using a single command. An alternative to the WEB SEND and WEB RECEIVE commands for CICS as an HTTP client.



Description

WEB CONVERSE enables an application program to compose and send an HTTP client request, and receive a response from the server. A session token must be included on this command. For guidance on the correct use of the WEB CONVERSE command, see the *CICS Internet Guide*.

- **The HTTP client request** is made using a connection that has been opened using the WEB OPEN command. The WEB CONVERSE command can be used in place of the WEB SEND command to compose and send the request.
- **The response from the server** is received by CICS Web support and passed to the application. The WEB CONVERSE command can be used in place of the WEB RECEIVE command to make the application program wait for and receive

the HTTP response. The headers for the HTTP response can be examined separately using the WEB READ HTTPHEADER command or the HTTP header browsing commands.

Note: The RTIMOUT value specified for the transaction that starts the user application indicates the time that the application is prepared to wait to receive the incoming message. (RTIMOUT is specified on the transaction profile definition). When the period specified by RTIMOUT expires, CICS returns a TIMEDOUT response to the application. An RTIMOUT value of zero means that the application is prepared to wait indefinitely. The default setting for RTIMOUT on transaction profile definitions is zero, so it is important to check and change that setting for applications that are making HTTP client requests.

The WEB CONVERSE command does not support chunked transfer-coding for the request, because this requires a sequence of send actions, and the WEB CONVERSE command provides a single send action. If you want to send a chunked message, use the WEB SEND command to send it, and the WEB RECEIVE command to receive it. If the server sends a chunked response, this can be received using the WEB CONVERSE command.

The WEB CONVERSE command cannot be used after the connection to the server has been closed. If you need to test whether the server has requested termination of the connection, use the WEB READ HTTPHEADER command to look for the Connection: close header in the last message from the server.

The WEB CONVERSE command performs a single send action and a single receive action, and it is designed to be used in place of a WEB SEND command and a WEB RECEIVE command. You may use WEB SEND and WEB RECEIVE commands, and WEB CONVERSE commands, in relation to the same connection (that is, with the same SESSTOKEN). However, if you are pipelining requests (that is, sending a sequence of requests without waiting for a response), you must not follow a WEB SEND command with a WEB CONVERSE command. CICS checks at program run time that each WEB SEND command has a subsequent WEB RECEIVE command before any WEB CONVERSE command is issued. For example, if you use the WEB SEND command three times to issue a pipelined sequence of requests, you must use the WEB RECEIVE command three times to receive the responses for those requests, before you may use the WEB CONVERSE command.

Options for sending the HTTP client request

ACTION(*cvda*)

This option is used to specify how the message should be sent out. The CVDA value that applies for CICS as an HTTP client is:

EXPECT

makes CICS send an Expect header along with the request line and headers for the request, and await a 100-Continue response before sending the message body to the server. If a response other than 100-Continue is received, CICS informs the application program and cancels the send. If no response is received after a period of waiting, CICS sends the message body anyway.

This option must only be used if your request has a message body.

CLOSESTATUS(*cvda*)

specifies whether or not a Connection header with the "close" connection option

(Connection: close) should be included on the request. The default is that the header is not included. The CVDA values are:

CLOSE

makes CICS write a Connection: close header for this request. The header notifies the server that the connection should be closed after the server has sent its response to the request. (For a server at HTTP/1.0 level, CICS achieves the same effect by omitting the Connection: Keep-Alive header.)

NOCLOSE

means that the Connection: close header is not used for this request. If the server is identified as HTTP/1.0, CICS sends a Connection header with the "Keep-Alive" connection option (Connection: Keep-Alive), to notify that a persistent connection is desired.

DOCTOKEN(*data-value*)

specifies the 16-byte binary token of a document to be sent as the message body. The document must be created using the CICS Document interface (EXEC CICS DOCUMENT CREATE, INSERT, and SET commands), as described in the *CICS Application Programming Guide*. The FROM option provides an alternative way to create a message body.

CICS documents cannot be converted to the UTF-8 and UTF-16 character encodings.

FROM(*data-area*)

specifies a buffer of data which holds the message body. The message body is built by the application program. When you specify the FROM option, use the FROMLENGTH option to specify the length of the buffer of data. The DOCTOKEN option provides an alternative way to create the message body.

There is no set maximum limit for the size of the data-area, but its size is limited in practice by storage considerations. The *CICS Internet Guide* has more information about these.

FROMLENGTH(*data-area*)

specifies the length, as a fullword binary value, of the buffer of data supplied on the FROM option (the message body). It is important to state this value correctly, because an incorrect data length can cause problems for the recipient of the message.

MEDIATYPE(*data-area*)

specifies the data content of any request body, for example `text/xml`. The media type is up to 56 alphanumeric characters, including appropriate punctuation, but not spaces. See the *CICS Internet Guide* for more information about media types. CICS checks that the format of the media type is correct, but does not check the validity of the media type against the data content. CICS uses this information to produce the Content-Type header for the message.

#

For requests which require a body, you must specify the MEDIATYPE option. There is no default. However, if the required Content-Type header needs to contain spaces or more than 56 characters, the application can provide it using the WEB WRITE HTTPHEADER command. In this case, do not specify the MEDIATYPE option.

For code page conversion to take place, the MEDIATYPE option must specify a type of data content that can be identified as text according to the IANA definitions. For non-text media types, CICS does not convert the message body.

The **MEDIATYPE** option is used for both the sending and receiving functions of the **WEB CONVERSE** command. If it is specified with a value, the value is used to construct the Content-Type header in the request, and the same field is used to receive the media type of the response that is returned by the server. If it is specified without a value, it is used only to receive the media type of the response.

METHOD(*cvda*)

specifies the HTTP method for the request.

The **GET**, **HEAD**, **POST**, **PUT**, **TRACE**, **OPTIONS**, and **DELETE** methods are supported by this command. However, some HTTP servers, particularly HTTP/1.0 servers, might not implement all of these methods.

The *CICS Internet Guide* has more information about the correct use of methods, including the HTTP versions that apply to each.

CICS bars the sending of a message body for methods where it is inappropriate, and requires it for methods where it is appropriate. The CVDA values are:

GET Obtain a resource from the server. A request body is not allowed.

HEAD Obtain the HTTP headers, but not the response body, for a resource. A request body is not allowed.

POST Send data to a server. A request body is required.

PUT Create or modify a resource on the server. A request body is required.

TRACE

Trace the route of your request to the server. A request body is not allowed.

OPTIONS

Obtain information about the server. A request body is allowed, but there is no defined purpose for the body. If you do use a request body, then you must specify a media type.

DELETE

Delete a resource on the server. A request body is not allowed.

PATH(*data-area*)

specifies the path information for the specific resource within the server that the application needs to access.

If the **URIMAP** option was used to specify an existing **URIMAP** definition on the **WEB OPEN** command for this connection, the path specified in that **URIMAP** definition is the default path for the **WEB SEND** command. In these circumstances, if you do not specify path information on the **WEB SEND** command, the path from the **URIMAP** definition is used. If you specify a different path from that given in the **URIMAP** definition, this overrides the path from the **URIMAP** definition.

If the **URIMAP** option was not used on the **WEB OPEN** command, there is no default path, and you must provide path information. Path information can be extracted from a known URL using the **WEB PARSE URL** command.

As an alternative to using the **PATH** option to provide the path information, you can use the **URIMAP** option on the **WEB CONVERSE** command to specify a **URIMAP** definition from which the path information is taken directly.

PATHLENGTH(*data-value*)

specifies the length of the path, as a fullword binary value. If you are providing

path information using the PATH option, you need to specify the PATHLENGTH option. Path length information is returned if you use the WEB PARSE URL command to parse a URL.

QUERYSTRING(*data-area*)

specifies a query string that is to be supplied to the server as part of the request. You do not need to include a question mark (?) at the beginning of the query string; if you do not include it, CICS supplies it for you automatically when constructing the request. If you include escaped characters in the query string, CICS passes them to the server in their escaped format.

QUERYSTRLEN(*data-value*)

specifies the length of the query string supplied on the QUERYSTRING option, as a fullword binary value.

SESSTOKEN(*data-value*)

specifies the session token, an 8-byte binary value that uniquely identifies this connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

URIMAP(*data-value*)

specifies the name (up to 8 characters, in mixed case) of a URIMAP definition that provides the path information for the specific resource within the server that the application needs to access. The URIMAP definition must be for CICS as an HTTP client (with USAGE(CLIENT) specified). Its HOST attribute must be the same as the HOST attribute of the URIMAP definition that was specified on the WEB OPEN command for this connection, or the same as the host name specified in the HOST option on the WEB OPEN command for this connection. A URIMAP definition specified on the WEB CONVERSE command applies only to this request.

If the URIMAP option is specified, do not specify the PATH or PATHLENGTH options.

Options for receiving the server's response

INTO(*data-area*)

specifies the buffer that is to contain the data being received.

MAXLENGTH(*data-value*)

specifies the maximum amount, as a fullword binary value, of data that CICS is to pass to the application. The MAXLENGTH option applies even when the INTO option is specified for receiving data. If the data has been sent using chunked transfer-coding, CICS assembles the chunks into a single message before passing it to the application, so the MAXLENGTH option applies to the total length of the chunked message, rather than to each individual chunk. The data is measured after any code page conversion has taken place.

#

Do not use the MAXLENGTH and SET options together, because MAXLENGTH will be ignored.

If the length of data exceeds the value specified and the NOTTRUNCATE option **is not** specified, the data is truncated to that value, and the remainder of the data is discarded.

If the length of data exceeds the value specified and the NOTTRUNCATE option **is** specified, CICS retains the remaining data and can use it to satisfy subsequent RECEIVE commands.

MEDIATYPE (*data-area*)

specifies a 56-character data-area to receive the media type (that is, the type of data content) for the body, for example `text/xml`. See the *CICS Internet Guide* for more information about media types.

The MEDIATYPE option is used for both the sending and receiving functions of the WEB CONVERSE command. If it is specified with a value, the value is used to construct the Content-Type header in the request, and the same field is used to receive the media type of the response that is returned by the server. If it is specified without a value, it is used only to receive the media type of the response.

NOTRUNCATE

specifies that when the data available exceeds the length requested on the MAXLENGTH option, the remaining data is not to be discarded immediately but is to be retained for retrieval by subsequent RECEIVE commands. (If no further RECEIVE commands are issued, the data is discarded during transaction termination.)

A single RECEIVE command using the SET option and without the MAXLENGTH option receives all the remaining data, whatever its length. Alternatively, you can use a series of RECEIVE commands with the NOTRUNCATE option to receive the remaining data in appropriate chunks. Keep issuing the RECEIVE command until you are no longer getting a LENGERR response. Bear in mind that if you receive less than the length requested on the MAXLENGTH option, this does not necessarily indicate the end of the data; this could happen if CICS needs to avoid returning a partial character at the end of the data.

SET (*ptr-ref*)

specifies a pointer reference that is to be set to the address of data received. The pointer reference is valid until the next receive command or the end of task.

STATUSCODE (*data-area*)

specifies a data-area to receive the HTTP status code sent by the server. The code is a binary halfword value. Examples are 200 (normal) or 404 (not found). Receiving the status code is optional, but you should always receive and check the status code in the following circumstances:

- If you intend to make an identical request to the server, now or during a future connection.
- If you intend to make further requests to the server using this connection.
- If your application is carrying out any further processing that depends on the information you receive in the response.

The *CICS Internet Guide* has basic guidance on appropriate actions for an application to take in response to the status codes for HTTP/1.1.

STATUSTEXT (*data-area*)

specifies a data-area to receive any text returned by the server to describe the status code. The text is known as a reason phrase. Examples are "OK" (accompanying a 200 status code), or "Bad Request" (accompanying a 400 status code). The STATUSLEN option gives the length allowed for the text.

STATUSLEN (*data-value*)

specifies, as a fullword binary value, the length of the data-area to receive any text returned by the server to describe the status code (the STATUSTEXT option). The text is known as a reason phrase. Most reason phrases

recommended for HTTP are short, but a data-area length of 256 characters is suggested here, in case the server replaces the recommended reason phrase with more detailed information.

TOLENGTH(*data-area*)

specifies a fullword binary variable which is set to the amount of data that CICS has returned to the application. Note that this might be slightly less than the limit that you set using the MAXLENGTH option, especially if a double-byte or multi-byte character set is involved, because CICS does not return a partial character at the end of the data.

- If the NOTRUNCATE option **is not** specified, any further data present in the message has now been discarded. A LENGERR response with a RESP2 value of 57 is returned if further data was present.
- If the NOTRUNCATE option **is** specified, any additional data is retained. A LENGERR response with a RESP2 value of 36 is returned if additional data is available. The description for the NOTRUNCATE option tells you what to do in this case.

This option is the equivalent of the LENGTH option on the WEB RECEIVE command.

Options for converting items sent and received

CHARACTERSET(*data-value*)

specifies the character set into which CICS translates the entity body of the HTTP request before sending. The name of the character set can consist of up to 40 alphanumeric characters, including appropriate punctuation. CICS does not support all the character sets named by IANA. The *CICS Internet Guide* lists the IANA character sets that are supported by CICS for code page conversion.

For conversion of the request body to take place, the CLIENTCONV option must be allowed to default to CLICONVERT, or specified as NOINCONVERT. Specifying NOCLICONVERT or NOOUTCONVERT suppresses conversion of the request body. If conversion is requested, ISO-8859-1 is used as the default if the CHARACTERSET attribute is not specified.

CLIENTCONV(*cvda*)

specifies whether or not CICS translates the entity body of the HTTP request before sending, and translates the entity body of the server's response. The default is that the entity body **is** converted both when the request is sent out, and when the response is received (CLICONVERT).

- For the request body, you can use the CHARACTERSET option on this command to specify a character set that is suitable for the server. If conversion is requested (or happens as the default) but you do not specify a character set, the default is that CICS converts the entity body to the ISO-8859-1 character set.
- For the response body, you do not need to specify the character set used by the server. CICS identifies this by examining the Content-Type header of the message. If the header does not provide this information, or if the named character set is not supported by CICS for code page conversion, the ISO-8859-1 character set is used.
- For the application's code page, the default code page for the local CICS region (as specified in the LOCALCCSID system initialization parameter) is used, or an alternative EBCDIC code page that you specified on the WEB OPEN COMMAND.

Note: For message bodies with non-text media types, CICS does not convert the message body, even if an appropriate conversion option is specified. CVDA values are:

CLICONVERT

CICS converts the entity body of the request into the character set that you identify for the server, and converts the entity body of the response into a code page suitable for the application.

NOINCONVERT

CICS converts the entity body of the request into the character set that you identify for the server. However, CICS does **not** convert the entity body of the response, and it is passed to the application in the character set used by the server.

NOOUTCONVERT

CICS does **not** convert the entity body of the request, and it is sent to the server in the code page used by the application. However, CICS does convert the entity body of the response into a code page suitable for the application.

NOCLICONVERT

CICS does not convert the entity body of the request, and it is sent to the server in the code page used by the application. CICS does not convert the entity body of the response, and it is passed to the application in the character set used by the server.

Conditions

NOTOPEN

RESP2 values are:

27 Invalid session token.

INVREQ

RESP2 values are:

10 Invalid response header.

11 Action code invalid.

13 Close status invalid.

15 Code page conversion failure.

17 Expect-100 request was rejected by the server.

22 Invalid chunk size.

32 Media type invalid.

33 Method does not support a body.

34 Method requires a body.

41 The connection has been closed.

45 The character set specified is invalid.

46 The CLIENTCONV option is invalid.

49 The format of the path option is invalid.

54 The HTTP method is not valid.

63 URIMAP object disabled.

- 64 Host in URIMAP definition does not match host specified when this session was opened.
- 67 HTTP error in response.
- 74 The connection has been closed (CICS sent a Connection: close header to the server).
- 76 MEDIATYPE option required.
- 79 Pipelining is in progress. WEB CONVERSE command cannot be used.

LENGERR

RESP2 values are:

- 5 The PATHLENGTH option value was not greater than zero.
- 8 The QUERYSTRLEN option value was not greater than zero.
- 16 Invalid MAXLENGTH.
- 36 Partial response body returned. Use additional RECEIVES to obtain remainder.
- 50 The FROMLENGTH option value was not greater than zero.
- 57 The response body exceeds the length specified, and the remainder of the body has been discarded.
- 58 The status text exceeds the length specified.
- 59 The STATUSLEN option value was not greater than zero.

NOTFND

RESP2 values are:

- 61 The URIMAP object specified was not found.

TOKENERR

RESP2 values are:

- 47 The document token specified is invalid.

IOERR

RESP2 values are:

- 42 Socket error.

TIMEDOUT

RESP2 values are:

- 62 Timeout on socket receive.

NOTAUTH

RESP2 values are:

- 100 Path barred by security exit.

WEB ENDBROWSE FORMFIELD

Signal the end of a formfield browse in an HTML form.

WEB ENDBROWSE FORMFIELD

▶—WEB—ENDBROWSE—FORMFIELD—▶

Conditions: INVREQ

This command is threadsafe.

Description

WEB ENDBROWSE FORMFIELD terminates the browse of a set of name-value pairs in an HTML form. The form is part of the body of an HTTP request being processed by the current CICS task. No information is returned on the ENDBROWSE.

Conditions

INVREQ

RESP2 values are:

- 1 The command is being issued in a non-CICS Web support application.
- 3 The command is being issued for a non-HTTP request.
- 4 The command is being issued before a WEB STARTBROWSE command is issued.

WEB ENDBROWSE HTTPHEADER

Signal the end of an HTTP header browse.

WEB ENDBROWSE HTTPHEADER



Conditions: INVREQ, NOTOPEN

This command is threadsafe.

Description

WEB ENDBROWSE HTTPHEADER terminates the browse. No information is returned on the ENDBROWSE. The SESSTOKEN option is required if the HTTP header information is part of a response sent to CICS as an HTTP client.

Options

SESSTOKEN(*data-value*)

For CICS as an HTTP client, this option is required. It specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

Conditions

INVREQ

RESP2 values are:

- 1 The command is being issued in a non-CICS Web support application.
- 3 The command is being issued for a non-HTTP request.
- 4 The command is being issued before a WEB STARTBROWSE command is issued.

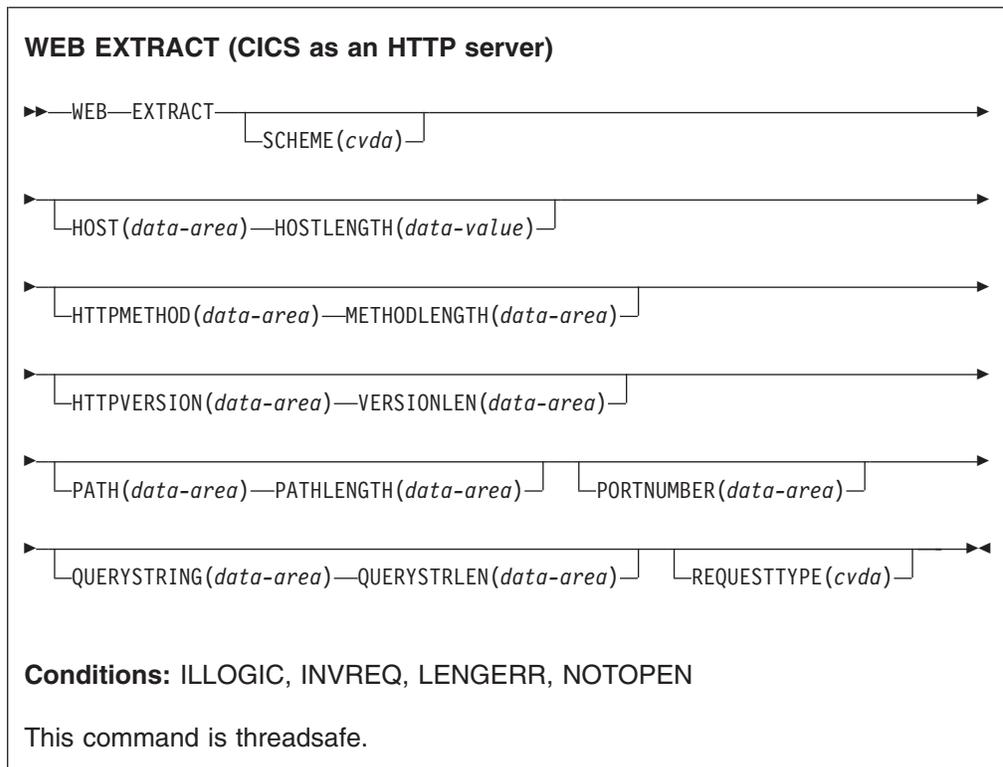
NOTOPEN

RESP2 value is:

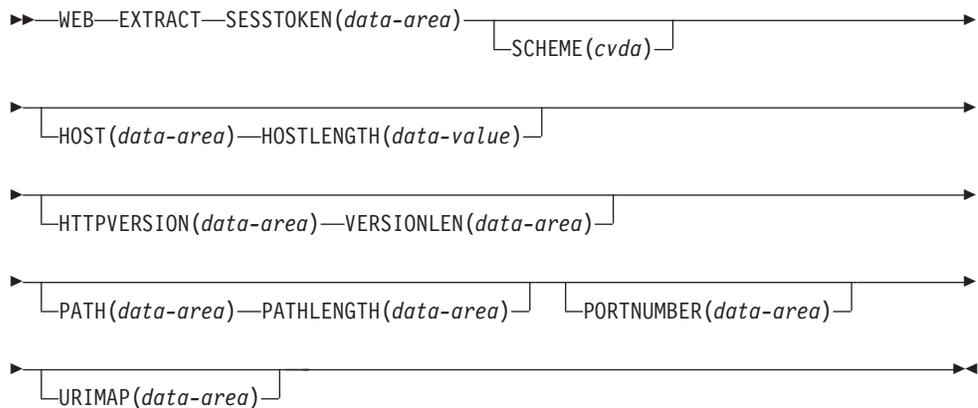
- 27 Invalid session token.

WEB EXTRACT

Obtain information about an HTTP request that has been made to CICS as an HTTP server, or about a connection between an Internet server and CICS as an HTTP client.



WEB EXTRACT (CICS as an HTTP client)



Conditions: ILLOGIC, INVREQ, LENGERR, NOTOPEN

This command is threadsafe.

Description

For CICS as an HTTP server, WEB EXTRACT enables an application to obtain information about the most recent HTTP request that has been made to CICS by a Web client and assigned to the application for handling.

For CICS as an HTTP client, when the SESSTOKEN option is specified, the command enables an application to obtain information about a connection that it has opened with a server on the Internet. The information returned to the application comprises global information about the connection, such as the host name of the server and its HTTP version. Information about specific requests made by the application, and responses made by the server, is not available using this command. The WEB RECEIVE command is used to receive information from a server's response.

Options

HOST(*data-area*)

For CICS as an HTTP server, this option specifies a buffer to contain the host component of the URL, as specified either in the Host header field for the request, or in the request line (if an absolute URI was used for the request). The port number is presented separately using the PORTNUMBER option.

For CICS as an HTTP client (with the SESSTOKEN option), this option specifies a buffer to contain the host name of the server in the connection identified by the SESSTOKEN option. The port number is presented separately using the PORTNUMBER option.

HOSTLENGTH(*data-area*)

specifies the length of the buffer supplied on the HOST option, as a fullword binary variable, and is set to the actual length of the data returned to the

application. 116 characters is an appropriate size to specify for this data-area. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

HTTPMETHOD (*data-area*)

For CICS as an HTTP server, this option specifies a buffer to contain the HTTP method string on the request line of the message.

This option is not relevant for CICS as an HTTP client.

HTTPVERSION (*data-area*)

For CICS as an HTTP server, this option specifies a buffer to contain the HTTP version for the Web client, as stated on its request.

For CICS as an HTTP client (with the SESSTOKEN option), this option specifies a buffer to contain the HTTP version of the server in the connection identified by the SESSTOKEN option.

"1.1" indicates HTTP/1.1, and "1.0" indicates HTTP/1.0.

Note: CICS does not make any special provision for a server or Web client that is below HTTP/1.0 level. CICS behaves as though they were at HTTP/1.0 level, and returns "1.0" as the HTTP version.

If your application program writes HTTP headers that might be unsuitable for a Web client or server at HTTP/1.0 level, or if you intend to send chunked information to the Web client or server (which cannot be received by a client or server at HTTP/1.0 level), your application program should check the HTTP version information.

METHODLENGTH (*data-area*)

specifies the length of the buffer supplied on the HTTPMETHOD option, as a fullword binary variable, and is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

PATH (*data-area*)

For CICS as an HTTP server, this option specifies a buffer to contain the path specified in the request line of the message.

For CICS as an HTTP client (with the SESSTOKEN option), this option specifies a buffer to contain the default path that applies to requests made using the connection. If a URIMAP definition was specified on the WEB OPEN command for the connection, the default path is the path specified in the URIMAP definition. Otherwise, the default path is a single forward slash.

PATHLENGTH (*data-area*)

specifies the length of the buffer supplied on the PATH option, as a fullword binary variable, and is set to the actual length of the data returned to the application. 256 characters is an appropriate size to specify for this data-area. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

PORTNUMBER (*data-area*)

For CICS as an HTTP server, this option returns a data area containing the port number specified in the request line of the message.

For CICS as an HTTP client (with the SESSTOKEN option), this option returns a data containing the port number used to access the server in the connection specified by the SESSTOKEN option.

The value returned in the data area is a fullword binary value.

Well-known port numbers for a service are normally omitted from the URL. If the port number is not present in the URL, the WEB EXTRACT command identifies and returns it based on the scheme. For HTTP, the well-known port number is 80, and for HTTPS, the well-known port number is 443. If a port number is returned which is not the default for the scheme, you need to specify the port number explicitly to gain access to the URL (for example, if you are using this information in a WEB OPEN command).

QUERYSTRING(*data-area*)

For CICS as an HTTP server, this option specifies a buffer to contain the query string on the request line of the message. The query string is the value or values encoded after the question mark (?) delimiting the end of the path. The query string is returned in its escaped form.

This option is not relevant for CICS as an HTTP client.

QUERYSTRLEN(*data-area*)

specifies the length of the buffer supplied on the QUERY option, as a fullword binary variable, and is set to the actual length of the data returned to the application (the query string). 256 characters is an appropriate size to specify for this data-area. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

REQUESTTYPE(*cvda*)

For CICS as an HTTP server, this option specifies the type of request received. This option is not relevant for CICS as an HTTP client. CVDA values are:

HTTPYES

indicates an HTTP request.

HTTPNO

indicates a non-HTTP request.

SCHEME(*cvda*)

For both CICS as an HTTP server, and CICS as an HTTP client (with the SESSTOKEN option), this option returns the scheme used for the connection between CICS and the Web client or server. CVDA values are:

HTTP is the HTTP protocol, without SSL.

HTTPS

is the HTTPS protocol, which is HTTP with SSL.

SESSTOKEN(*data-value*)

For CICS as an HTTP client, this option is required. It specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token. For the WEB EXTRACT command, information is returned about the specified connection.

This option is not relevant for CICS as an HTTP server.

URIMAP(*data-value*)

For CICS as an HTTP client (with the SESSTOKEN option), this option returns the 8-character name (in mixed case) of any URIMAP definition that was specified on the WEB OPEN command to open the connection specified by the SESSTOKEN option. The INQUIRE URIMAP command can be used to find information about the attributes of this URIMAP definition.

This option is not relevant for CICS as an HTTP server.

VERSIONLEN (*data-area*)

specifies the length of the buffer supplied on the HTTPVERSION option, as a fullword binary variable, and is set to the actual length of the data returned to the application.

Conditions**INVREQ**

RESP2 values are:

- 1** The command is being issued in a non-CICS Web support application.
- 3** The command is being issued for a non-HTTP request (this is set only if one or more of HTTPMETHOD, HTTPVERSION, or PATH is specified and the request is a non-HTTP request).

LENGERR

RESP2 values are:

- 4** The method exceeds the length specified (METHODLENGTH option).
- 5** The PATHLENGTH option value was not greater than zero.
- 6** The HTTP version exceeds the length specified (VERSIONLEN option).
- 7** The VERSIONLEN option value was not greater than zero.
- 8** The query string exceeds the length specified (QUERYSTRLEN option).
- 21** The HOSTLENGTH option value was not greater than zero.
- 29** The host name exceeds the length specified (HOSTLENGTH option).
- 30** The path exceeds the length specified (PATHLENGTH option).

NOTOPEN

RESP2 values are:

- 27** Invalid session token.

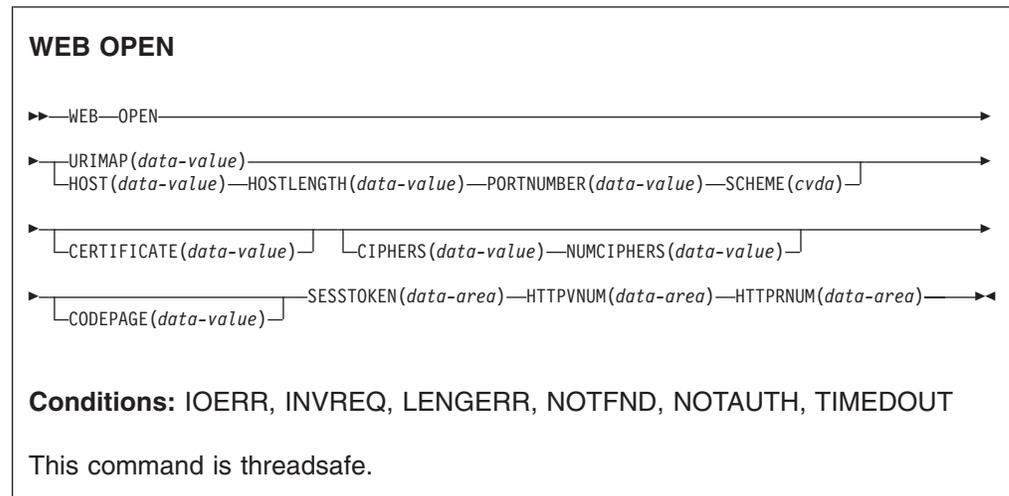
ILLOGIC

RESP2 values are:

- 9** CICS logic error.

WEB OPEN

Open a connection to a server for CICS as an HTTP client.



Description

WEB OPEN enables an application program, through CICS Web support, to open a connection with a specified host on an HTTP server on the Internet. The host name and scheme can be used from a preset URIMAP definition, which also supplies a default path for requests.

When the connection is open, the application program can make HTTP client requests to the server and receive responses from it. CICS queries the HTTP version of the server (using an OPTIONS request) when the connection is opened, and uses this information for subsequent communications. CICS also returns the HTTP version information to the application program, to be checked if you plan to write HTTP headers or send chunked information.

The WEB OPEN command drives the XWBOPEN user exit, which can make the connection to the server go through a proxy server, if required.

Options

CERTIFICATE(data-value)

specifies the label of the X.509 certificate that is to be used as the SSL client certificate during the SSL handshake. Certificate labels can consist of up to 32 alphanumeric characters. This option is only relevant when SCHEME(HTTPS) is specified. If SCHEME(HTTPS) is specified, but the CERTIFICATE option is omitted, the default certificate defined in the key ring for the CICS region user ID is used. The certificate must be stored in a key ring in the external security manager's database. "Building a key ring" in the *CICS RACF Security Guide* tells you how to do this.

CIPHERS(data-value)

specifies a string of up to 56 hexadecimal digits that is interpreted as a list of up to 28 2-digit cipher suite codes. The cipher suite codes are used when SSL is active for the connection, so this option is only relevant when SCHEME(HTTPS) is specified. They indicate the method of encryption to be used for this connection.

Use the NUMCIPHERS option to specify the number of cipher suite codes in your list. The codes that are available depend on what level of encryption has been specified by the ENCRYPTION system initialization parameter. If you specify any cipher codes that are not in the default list for the active encryption level, they are ignored. "Cipher suites" in the *CICS RACF Security Guide* has more information about using cipher suite codes.

You can specify the URIMAP option to use this information directly from an existing URIMAP definition, in which case the CIPHERS option is not required. You may still specify the CIPHERS option, and the setting in the URIMAP definition is overridden by any codes that you specify for this option.

If you omit the CIPHERS option and the URIMAP option, but SSL is active for the connection, the default cipher list for the encryption level for the running system is used.

CODEPAGE(*data-value*)

specifies a code page, usually EBCDIC, that is suitable for the application program. The code page name can be up to 8 alphanumeric characters. The default is the default code page for the local CICS region, as specified in the LOCALCCSID system initialization parameter. The code page applies for the duration of this connection. When the server returns a response to an HTTP request, if conversion is requested (which is the default), CICS converts the request body into this code page before passing it to the application.

HOST(*data-value*)

specifies the host name on the server to which you want to connect. This information can be extracted from a known URL using the WEB PARSE URL command, or from an existing URIMAP definition using the WEB EXTRACT URIMAP command. You can specify the URIMAP option to use this information directly from an existing URIMAP definition, in which case the HOST option is not required.

An IPv4 address can be used as a host name, but IPv6 addresses are **not** supported.

If a port number is required, do not include this with the host name, but use the PORTNUMBER option to specify it.

HOSTLENGTH(*data-value*)

specifies, as a fullword binary value, the length of the host name. This information is returned if you use the WEB PARSE URL command to parse a URL, or it can be extracted from an existing URIMAP definition using the WEB EXTRACT URIMAP command. You can specify the URIMAP option to use this information directly from an existing URIMAP definition, in which case the HOSTLENGTH option is not required.

HTTPRNUM(*data-area*)

returns the release number for the HTTP version of the server, as a halfword binary value. (HTTPVNUM returns the version number.) For example, if the server is at HTTP/1.0 level, HTTPRNUM returns 0.

HTTPVNUM(*data-area*)

returns the version number for the HTTP version of the server, as a halfword binary value. (HTTPRNUM returns the release number.) For example, if the server is at HTTP/1.0 level, HTTPVNUM returns 1.

CICS obtains the HTTP version information when it opens the connection to the server. If the server does not provide HTTP version information, CICS assumes that it is at HTTP/1.0 level.

If your application program writes HTTP headers that might be unsuitable for a server at HTTP/1.0 level, or if you intend to send a chunked message to the server (which cannot be received by a server at HTTP/1.0 level), your application program should also consult the HTTP version information.

Note: CICS does not make any special provision for a server that is below HTTP/1.0 level. CICS behaves as though these servers were at HTTP/1.0 level, and returns HTTP/1.0 as the HTTP version.

NUMCIPHERS(*data-value*)

specifies, as a halfword binary value, the number of cipher suite codes that you specified for the CIPHERS option.

PORTNUMBER(*data-value*)

specifies the port number, as a fullword binary value. You only need to specify the port number if it is **not** the default for the specified scheme. For HTTP, the default port number is 80, and for HTTPS, the default port number is 443. Port number information can be extracted from a known URL using the WEB PARSE URL command, or from an existing URIMAP definition using the WEB EXTRACT URIMAP command. You can specify the URIMAP option to use this information directly from an existing URIMAP definition, in which case the PORTNUMBER option is not required.

SCHEME(*cvda*)

specifies the scheme that is to be used for the connection to the server, which can be with or without SSL. CVDA values are:

HTTP is the HTTP protocol, without SSL.

HTTPS

is the HTTPS protocol, which is HTTP with SSL. If HTTPS is used, the CICS address space must be enabled for SSL.

This information can be extracted from a known URL using the WEB PARSE URL command, or from an existing URIMAP definition using the WEB EXTRACT URIMAP command. You can specify the URIMAP option to use this information directly from an existing URIMAP definition, in which case the SCHEME option is not required.

SESSTOKEN(*data-area*)

returns the session token, an 8-byte binary value that uniquely identifies this connection between CICS and a server. It is returned when the connection is opened successfully. The session token must be used on all CICS WEB commands that relate to this connection. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

URIMAP(*data-value*)

specifies the name (up to 8 characters, in mixed case) of a URIMAP definition that provides the following information:

- The scheme that is to be used for the connection to the server.
- The host name on the server to which you want to connect.
- A port number, if required.
- A path component for the URI, representing the resource on the server that you want to access. This path becomes the default path for WEB SEND or WEB CONVERSE commands relating to this connection, but it can be overridden by specifying another path on the WEB SEND or WEB CONVERSE command.

- The label of the X.509 certificate that is to be used as the SSL client certificate, if required.
- The cipher suite codes that can be used for the connection.

If the URIMAP option is specified, do not specify the CERTIFICATE, HOST, HOSTLENGTH, PORTNUMBER, PORTLENGTH, or SCHEME options. The CIPHERS and NUMCIPHERS options can be omitted or specified in the command; if specified, they override these settings in the URIMAP definition. The URIMAP definition must be for CICS as an HTTP client, with USAGE(CLIENT) specified.

Conditions

IOERR

RESP2 values are:

- 38** Proxy error.
- 42** Socket error.

INVREQ

RESP2 values are:

- 14** Code page invalid.
- 22** Invalid chunk received during the initial OPTIONS request.
- 23** Invalid client certificate.
- 40** Scheme invalid.
- 41** Server closed the connection during the initial OPTIONS request.
- 48** The format of the host option is invalid.
- 63** URIMAP object disabled.
- 66** An error occurred in processing for the XWBOPEN exit.
- 67** HTTP error in response.
- 96** SSL not supported.
- 137** All requested cipher codes have been rejected.

#

#

LENGERR

RESP2 values are:

- 21** Invalid host length.

NOTFND

RESP2 values are:

- 20** Host name not resolved by name server.
- 39** Unknown proxy.
- 61** The URIMAP object specified was not found.

NOTAUTH

RESP2 values are:

- 100** Host name barred by security exit.

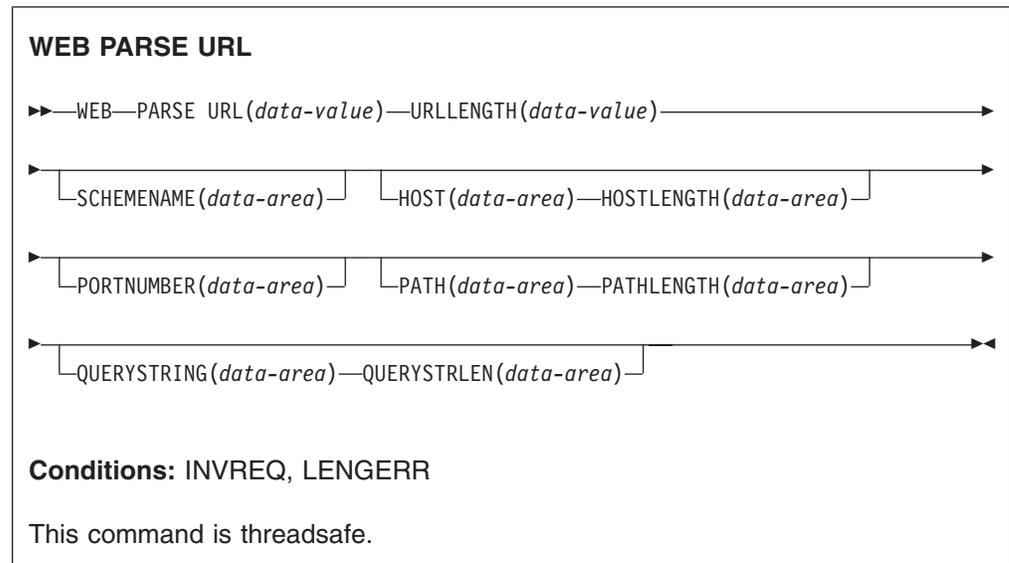
TIMEDOUT

RESP2 values are:

- 62** Timeout on socket receive.

WEB PARSE URL

Breaks down a URL string into its component parts.



Description

WEB PARSE URL enables you to break down a URL string into its component parts: scheme, host, port, path and query string. The *CICS Internet Guide* explains these components. You can use this process to examine the construction of the URL, and to separate out the components. The returned information can be used in the WEB OPEN command to open a client connection to the host named in the URL.

Any escape sequences found in the URL are checked for validity. An escape sequence consists of the percent character (%) followed by two hexadecimal characters. Valid hexadecimal characters are the digits 0 to 9 and the letters A to F.

Note that where the string input to the WEB PARSE URL command has been delimited in the correct way for a URL, the command does not detect invalid content, such as a host name that does not represent an existing host on the Internet, or a character that is not permitted in a URL.

Options

HOST (*data-area*)

returns the host component of the URL. This can be either an alphanumeric host name or a numeric IP address. If a port number was specified explicitly in the URL, this is returned separately as the PORTNUMBER option.

An IPv4 address can be used as a host name in the WEB OPEN command, but IPv6 addresses are **not** supported. IPv6 addresses are rejected as invalid by the WEB PARSE URL command because they do not conform to the expected structure.

HOSTLENGTH (*data-area*)

specifies the length of the buffer supplied on the HOST option, as a fullword binary variable, and is set to the actual length of the data returned to the

application (the host name). 116 characters is suggested as an appropriate size to specify for this data-area. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

PATH(*data-area*)

returns the path component of the URL.

PATHLENGTH(*data-area*)

specifies the length of the buffer supplied on the PATH option, as a fullword binary variable, and is set to the actual length of the data returned to the application (the path component of the URL). 256 characters is suggested as an appropriate size to specify for this data-area. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

PORTNUMBER(*data-area*)

returns (as a fullword binary data area) the port number that is specified in, or appropriate for, the URL. Port numbers are sometimes specified explicitly in a URL, following the host name. However, well-known port numbers for a service are normally omitted from a URL. If the port number is not present in the URL, the WEB PARSE URL command identifies and returns it based on the scheme. For HTTP, the well-known port number is 80, and for HTTPS, the well-known port number is 443. If a port number is returned which is not the default for the scheme, you need to specify the port number explicitly to gain access to the URL (for example, if you are using this information in a WEB OPEN command).

QUERYSTRING(*data-area*)

returns the query string from the URL. The query string is the value or values encoded after the question mark (?) delimiting the end of the path. The query string is returned in its escaped form.

QUERYSTRLEN(*data-area*)

specifies the length of the buffer supplied on the QUERYSTRING option, as a fullword binary variable, and is set to the actual length of the data returned to the application (the query string). 256 characters is suggested as an appropriate size to specify for this data-area. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

SCHEMENAME(*data-area*)

returns the scheme component of the URL, as a 16-character data area. Only the HTTP and HTTPS schemes (the HTTP protocol with and without SSL) are supported by CICS and can be used in a WEB OPEN command.

#

The scheme name is always returned in upper case.

URL(*data-value*)

specifies the complete URL string.

URLLENGTH(*data-value*)

specifies the length of the buffer containing the URL string, as a fullword binary value.

Conditions

INVREQ

RESP2 values are:

28 Invalid URL.

65 Bad escape sequence.

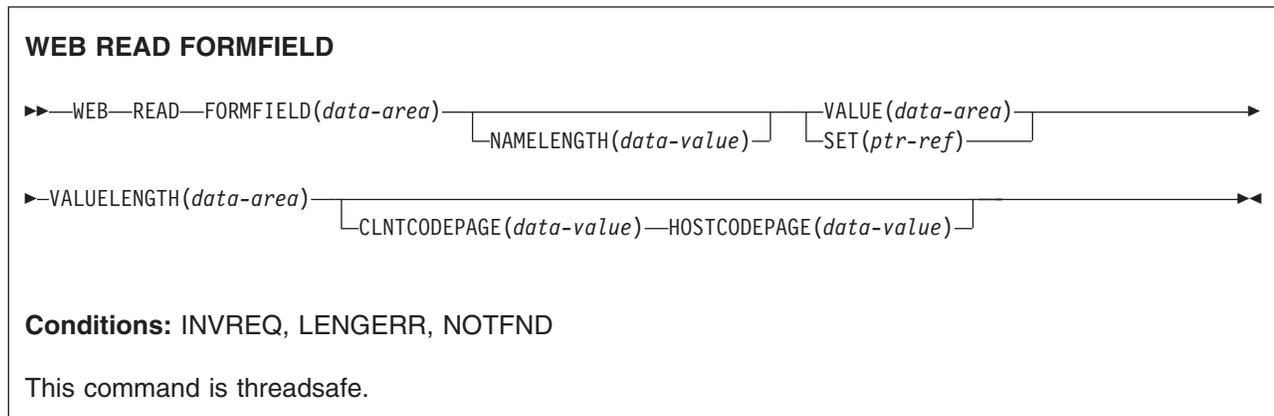
LENGERR

RESP2 values are:

- 8** Length of query string returned is greater than QUERYSTRLEN.
- 29** Length of host name returned is greater than HOSTLENGTH.
- 30** Length of path returned is greater than PATHLENGTH.

WEB READ FORMFIELD

Retrieve the value of a field from an HTML form.



Description

WEB READ FORMFIELD retrieves the value of a specific field from an HTML form. The name of the form field is given in the FORMFIELD parameter. The form data is sent as part of an HTTP request being processed by the current CICS task.

The Web client sends form data in a query string when the GET method is used, and in the entity body when the POST method is used. CICS can extract the data from either of these locations.

The form data is returned in its unescaped form (see the *CICS Internet Guide* for an explanation of this).

If the data that is received represents a file, the uploaded file does *not* undergo code page conversion.

CICS only reads form data when CICS is the HTTP server. The facility is not available when CICS is an HTTP client.

Options

CLNTCODEPAGE(*data-value*)

specifies the name of the character set that was used by the Web client for the HTTP request. The name of the character set can consist of up to 40 alphanumeric characters, including appropriate punctuation. If this is not specified, CICS obtains it from the Content-Type header of the HTTP request. If the Content-Type header is not present, CICS assumes the ISO-8859-1 character set. If you specify CLNTCODEPAGE you must also specify HOSTCODEPAGE. The *CICS Internet Guide* lists the IANA character sets that are supported by CICS for code page conversion.

FORMFIELD(*data-area*)

specifies the name of the form field to extract. It is a string of text containing the name of the requested field. The string of text supplied is not case sensitive.

HOSTCODEPAGE(*data-value*)

specifies the 8-character name of the host code page used by the application program, into which the form data is to be converted. If you specify

#

HOSTCODEPAGE you must also specify CLNTCODEPAGE. This code page is
normally an EBCDIC code page. If this is not specified, the default is the
EBCDIC code page 037.

If the form data is in a query string (GET method), this option is ignored, and
the data is returned in the EBCDIC code page specified by the LOCALCCSID
system initialization parameter (which applies to the whole of the local CICS
region, and has a default of 037). If the ASCII Latin-1 character set ISO-8859-1
(code page 819) cannot be converted into that code page, CICS uses the
default EBCDIC code page 037 instead.

NAMELENGTH(*data-value*)

specifies the length, as a fullword binary value, of the form field name.

SET(*ptr-ref*)

specifies a pointer reference that is to be set to the address of data received.
The pointer reference is valid until the end of the task.

VALUE(*data-area*)

specifies the buffer to contain the value of the named form field. CICS
unescapes any escaped characters before placing them in the buffer.

VALUELENGTH(*data-area*)

specifies the length, as a fullword binary value, of the buffer that is to contain
the form field value. If you specify the VALUE option, VALUELENGTH specifies
the maximum length of the data that the program accepts. If the value exceeds
the length of the buffer, it is truncated. If the length of the form field value is
less than the size of the buffer, the form field value is placed in the leftmost
byte positions.

Conditions

INVREQ

RESP2 values are:

- 1 The command is being issued in a non-CICS Web support application.
- 3 The command is being issued for a non-HTTP request.
- 11 The client code page cannot be found.
- 12 The host code page cannot be found.
- 13 No form data has been supplied in the body of the HTTP request.
- 14 The code page combination for client and server is invalid.
- 17 Invalid forms data was found in the input message.

LENGERR

RESP2 values are:

- 1 The length in VALUELENGTH is less than or equal to zero.
- 5 The form field value has been truncated during a read operation
because the receiving buffer is too small.

NOTFND

RESP2 values are:

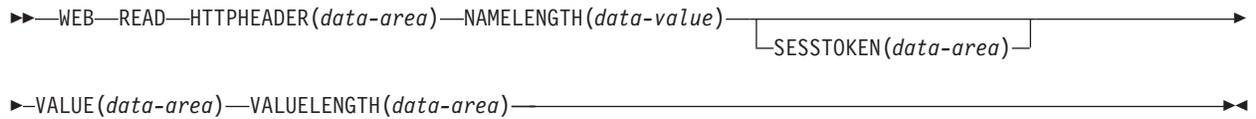
- 1 The form field with the given name cannot be found.

#

WEB READ HTTPHEADER

Extract HTTP header information.

WEB READ HTTPHEADER



Conditions: INVREQ, LENGERR, NOTFND

This command is threadsafe.

Description

WEB READ HTTPHEADER enables an application to extract HTTP header information from a message. When CICS is an HTTP server, the message is a request from a Web client. When CICS is an HTTP client, the message is a response from a server, and the SESSTOKEN option is specified.

The *CICS Internet Guide* lists HTTP/1.1 headers that you are likely to receive, and gives guidance for the actions that you might take in response to them.

The HTTP header browsing commands (WEB STARTBROWSE HTTPHEADER, WEB READNEXT HTTPHEADER, WEB ENDBROWSE HTTPHEADER) can be used to browse through all the HTTP header information for a message.

Options

HTTPHEADER(*data-area*)

specifies the name of the HTTP header to be extracted.

NAMELENGTH(*data-value*)

specifies the length, as a fullword binary value, of the HTTP header name.

SESSTOKEN(*data-value*)

For CICS as an HTTP client, this option is required. It specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

VALUE(*data-area*)

specifies the buffer to contain the value of the HTTP header being extracted.

VALUELENGTH(*data-area*)

specifies the length of the buffer supplied on the VALUE option, as a fullword binary variable, and is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

Conditions

INVREQ

RESP2 values are:

- 1 The command is being issued in a non-CICS Web support application.
- 3 The command is being issued for a non-HTTP request.
- 43 No HTTP headers found.

LENGERR

RESP2 values are:

- 1 The length in VALUELENGTH is not greater than zero (CICS as an HTTP server).
- 2 The header value has been truncated because the receiving buffer is too small (CICS as an HTTP server).
- 35 The length in NAMELENGTH is not greater than zero (CICS as an HTTP client).
- 52 The header value has been truncated because the receiving buffer is too small (CICS as an HTTP client).
- 55 The length in VALUELENGTH is not greater than zero (CICS as an HTTP client).

NOTFND

RESP2 value is:

- 1 The header with the given name could not be found.

NOTOPEN

RESP2 value is:

- 27 Invalid session token.

WEB READNEXT FORMFIELD

Retrieve next name-value pair in an HTML form.

WEB READNEXT FORMFIELD

►►—WEB—READNEXT—FORMFIELD(*data-area*)—NAMELENGTH(*data-area*)—VALUE(*data-area*)—————►
►—VALUELENGTH(*data-area*)—————►►

Conditions: ENDFILE, INVREQ, LENGERR

This command is threadsafe.

Description

WEB READNEXT FORMFIELD retrieves the next name-value pair in an HTML form.

The data is returned in its unescaped form (see the *CICS Internet Guide* for an explanation of this).

Options

FORMFIELD(*data-area*)

specifies the buffer to contain the name of the form field being retrieved. The case of the name is as it is stored in the form.

NAMELENGTH(*data-area*)

specifies the length, as a fullword binary value, of the form field name. If the length of the form field value is less than the size of the buffer, the form field value is placed in the leftmost byte positions.

VALUE(*data-area*)

specifies the buffer to contain the value corresponding to the name returned in the FORMFIELD data area. CICS unescapes any escaped characters before placing them in the buffer.

VALUELENGTH(*data-area*)

specifies the length, as a fullword binary value, of the buffer that is to contain the form field value. If the value exceeds the buffer length, it is truncated. If the length of the form field value is less than the size of the buffer, the form field value is placed in the leftmost byte positions.

Conditions

ENDFILE

The end of the list of name-value pairs has been reached.

INVREQ

RESP2 values are:

- 1 The command is being issued in a non-CICS Web support application.

- 3 The command is being issued for a non-HTTP request.
- 4 The command is being issued before a WEB STARTBROWSE FORMFIELD has been issued.
- 6 A form field has been found that is not in the format NAME:VALUE.

LENGERR

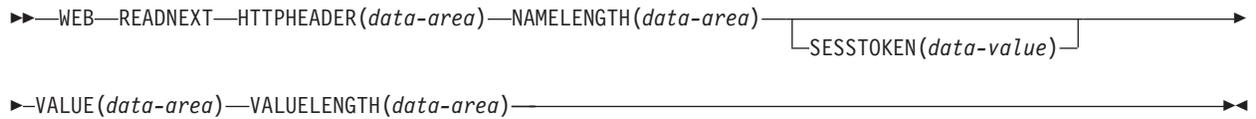
RESP2 values are:

- 1 NAMELENGTH or VALUELENGTH is less than or equal to zero.
- 4 The form field name has been truncated during a browse operation because the receiving buffer is too small.
- 5 The form field value has been truncated because the receiving buffer is too small.

WEB READNEXT HTTPHEADER

Retrieve next HTTP header.

WEB READNEXT HTTPHEADER



Conditions: ENDFILE, INVREQ, LENGERR, NOTOPEN

This command is threadsafe.

Description

WEB READNEXT HTTPHEADER retrieves the next HTTP header in the list of headers. The SESSTOKEN option is required if the HTTP header information is part of a response sent to CICS as an HTTP client.

Options

HTTPHEADER(*data-area*)

specifies the buffer to contain the name of the HTTP header being extracted.

NAMELENGTH(*data-area*)

specifies the length of the buffer supplied on the HTTPHEADER option, as a fullword binary data area, and is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

SESSTOKEN(*data-value*)

For CICS as an HTTP client, this option is required. It specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

VALUE(*data-area*)

specifies the buffer to contain the value of the HTTP header being extracted.

VALUELENGTH(*data-area*)

specifies the length of the buffer supplied on the VALUE option, as a fullword binary data area, and is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

Conditions

ENDFILE

The end of the list of HTTP headers has been reached.

INVREQ

RESP2 values are:

- 1 The command is being issued in a non-CICS Web support application.
- 3 The command is being issued for a non-HTTP request.
- 4 The command is being issued before a WEB STARTBROWSE has been issued.
- 6 A header has been found which is not in the format NAME:VALUE.

LENGERR

RESP2 values are:

- 1 NAMELENGTH or VALUELENGTH is less than or equal to zero.
- 4 The header name has been truncated during a browse operation because the receiving buffer is too small.
- 5 The header value has been truncated because the receiving buffer is too small.

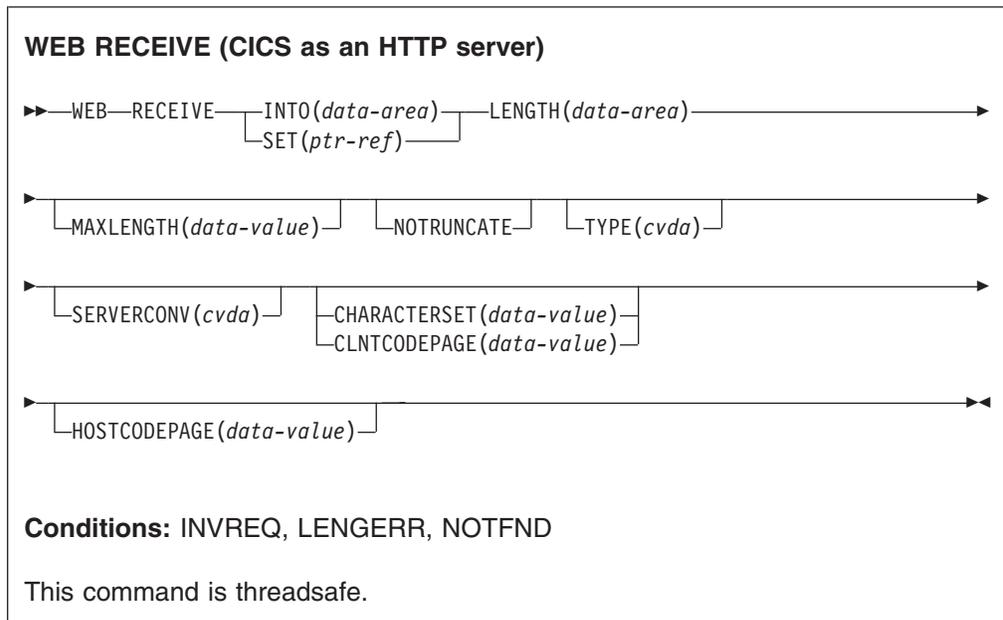
NOTOPEN

RESP2 value is:

- 27 Invalid session token.

WEB RECEIVE (Server)

Receive an HTTP request, or a non-HTTP message.



Description

WEB RECEIVE receives the body of an HTTP request, or all the data for a non-HTTP message, into an application-supplied buffer. The headers for an HTTP request can be examined separately using the WEB HTTPHEADER commands. The item received by the WEB RECEIVE command can be:

- The body of an HTTP request that a Web client has made to CICS as an HTTP server. For guidance on the correct use of the WEB RECEIVE command for this purpose, see "Writing Web-aware application programs for CICS as an HTTP server" in the *CICS Internet Guide*.
- A non-HTTP message handled by CICS Web support facilities, with the user-defined (USER) protocol on the TCPIP SERVICE definition. For guidance on non-HTTP messages, see "CICS Web support and non-HTTP requests" in the *CICS Internet Guide*.
- A request from another application that has used the CICS business logic interface to contact the application program directly, rather than going through the CICS HTTP listener. For guidance on the CICS business logic interface, see "The CICS business logic interface" in the *CICS Internet Guide*.

The data is returned in its escaped form. The type of code page conversion used for incoming data received by the CICS application program can be specified on this command. If you omit all of the code page conversion options (SERVERCONV, CLNTCODEPAGE, CHARACTERSET, HOSTCODEPAGE), no code page conversion takes place.

Options

CHARACTERSET (*data-value*)

specifies the character set that was used by the Web client for the entity body

of the received item. The name of the character set can consist of up to 40 alphanumeric characters, including appropriate punctuation. CICS does not support all the character sets named by IANA. The *CICS Internet Guide* lists the IANA character sets that are supported by CICS for code page conversion.

When the CHARACTERSET option is specified, SERVERCONV(SRVCONVERT) is assumed, so code page conversion of the entity body takes place. As an alternative to identifying the character set yourself, specifying either SERVERCONV(SRVCONVERT), or HOSTCODEPAGE, or both, and omitting CHARACTERSET, lets CICS identify the character set for the message body. The description for the SERVERCONV option tells you what happens in this case.

CLNTCODEPAGE (*data-value*)

This option is supported for migration purposes only. CHARACTERSET replaces it. The action taken by CICS is the same for both keywords. This means that code page conversion does take place when CLNTCODEPAGE or HOSTCODEPAGE is specified, even if the SERVERCONV option is not specified.

HOSTCODEPAGE (*data-value*)

specifies the 8-character name of the CICS (host) code page used by the application program, into which the entity body of the received item should be converted from the character set in which it was received from the Web client. When the HOSTCODEPAGE option is specified, SERVERCONV(SRVCONVERT) is assumed, so code page conversion of the entity body takes place. Specifying either SERVERCONV(SRVCONVERT), or CHARACTERSET, or both, and omitting HOSTCODEPAGE, lets CICS determine the host code page.

The default if this option is not specified is the default code page for the local CICS region, as specified in the LOCALCCSID system initialization parameter.

INTO (*data-area*)

specifies the buffer that is to contain the data being received.

LENGTH (*data-area*)

specifies a fullword binary variable which is set to the amount of data that CICS has returned to the application. Note that this might be slightly less than the limit that you set using the MAXLENGTH option, especially if a double-byte or multi-byte character set is involved, because CICS does not return a partial character at the end of the data.

- If the NOTTRUNCATE option **is not** specified, any further data present in the message has now been discarded. A LENGERR response with a RESP2 value of 57 is returned if further data was present.
- If the NOTTRUNCATE option **is** specified, any additional data is retained. A LENGERR response with a RESP2 value of 36 is returned if additional data is available. The description for the NOTTRUNCATE option tells you what to do in this case.

MAXLENGTH (*data-value*)

specifies the maximum amount, as a fullword binary value, of data that CICS is to pass to the application. The MAXLENGTH option applies whether the INTO or the SET option is specified for receiving data. If the data has been sent using chunked transfer-coding, CICS assembles the chunks into a single message before passing it to the application, so the MAXLENGTH option applies to the total length of the chunked message, rather than to each individual chunk. The data is measured after any code page conversion has taken place.

If the length of data exceeds the value specified and the NOTRUNCATE option **is not** specified, the data is truncated to that value, and the remainder of the data is discarded.

If the length of data exceeds the value specified and the NOTRUNCATE option **is** specified, CICS retains the remaining data and can use it to satisfy subsequent RECEIVE commands.

NOTRUNCATE

specifies that when the data available exceeds the length requested on the MAXLENGTH option, the remaining data is not to be discarded immediately but is to be retained for retrieval by subsequent RECEIVE commands. (If no further RECEIVE commands are issued, the data is discarded during transaction termination.)

A single RECEIVE command using the SET option and without the MAXLENGTH option receives all the remaining data, whatever its length. Alternatively, you can use a series of RECEIVE commands with the NOTRUNCATE option to receive the remaining data in appropriate chunks. Keep issuing the RECEIVE command until you are no longer getting a LENGERR response. Bear in mind that if you receive less than the length requested on the MAXLENGTH option, this does not necessarily indicate the end of the data; this could happen if CICS needs to avoid returning a partial character at the end of the data.

SERVERCONV (*cvda*)

specifies whether or not CICS translates the entity body of the item received, from the character set used by the Web client, to a code page suitable for the application. You can use the CHARACTERSET and HOSTCODEPAGE options on this command to specify the character set and code page that are used. If you specify either of these options, code page conversion (SRVCONVERT) is assumed. Alternatively, you can omit either or both of these options, specify SERVERCONV(SRVCONVERT) and let CICS determine a suitable character set and code page.

SRVCONVERT

CICS converts the entity body of the message.

When you specify SRVCONVERT without CHARACTERSET, CICS identifies the character set as follows:

1. If the Web client's request has a Content-Type header naming a character set supported by CICS, that character set is used.
2. If the Web client's request has no Content-Type header or the named character set is unsupported, the ISO-8859-1 character set is used.
3. For non-HTTP messages (sent using the USER protocol), the ISO-8859-1 character set is used.

When you specify SRVCONVERT without HOSTCODEPAGE, CICS determines the host code page as the default code page for the local CICS region, as specified in the LOCALCCSID system initialization parameter.

If you specify SRVCONVERT alone, note that for code page conversion to take place, the media type for the message must specify a type of data content that can be identified as text according to the IANA definitions. For messages where no media type is given but SRVCONVERT is specified, code page conversion also takes place. If a non-text media type is present, CICS does not convert the message

body. However, for compatibility with Web-aware applications coded in earlier releases, if you specify either of the CHARACTERSET or HOSTCODEPAGE options or omit the SERVERCONV option, the media type for the message does not influence code page conversion.

NOSRVCONVERT

CICS does not convert the entity body of the item, and it is passed to the application in the character set used by the Web client. If you specify NOSRVCONVERT, you cannot specify the CHARACTERSET or HOSTCODEPAGE options.

SET(*ptr-ref*)

specifies a pointer reference that is to be set to the address of data received. The pointer reference is valid until the next receive command or the end of task.

TYPE(*cvda*)

returns the type of request received. CVDA values are:

HTTPYES

indicates an HTTP request.

HTTPNO

indicates a non-HTTP request.

#

In CICS Transaction Server for z/OS, Version 3, HTTP requests and non-HTTP requests use different protocols, which are specified on TCIPSERVICE definitions, and must therefore use different ports. Non-HTTP requests use the user-defined (USER) protocol. You might use the TYPE option to distinguish between the request types if you specify the same user-written application program for responding to both HTTP and non-HTTP requests.

Conditions

INVREQ

RESP2 values are:

- 1** The command is being issued in a non-CICS Web support application.
- 14** Invalid code page combination.
- 46** The SERVERCONV option is invalid.
- 80** CHARACTERSET cannot be specified with SERVERCONV(NOSRVCONVERT).
- 81** HOSTCODEPAGE cannot be specified with SERVERCONV(NOSRVCONVERT).
- 84** Body incomplete.

LENGERR

RESP2 values are:

- 1** The MAXLENGTH option value was not greater than zero.
- 36** Partial response body returned. Use additional RECEIVES to obtain remainder.
- 57** The response body exceeds the length specified, and the remainder of the body has been discarded.

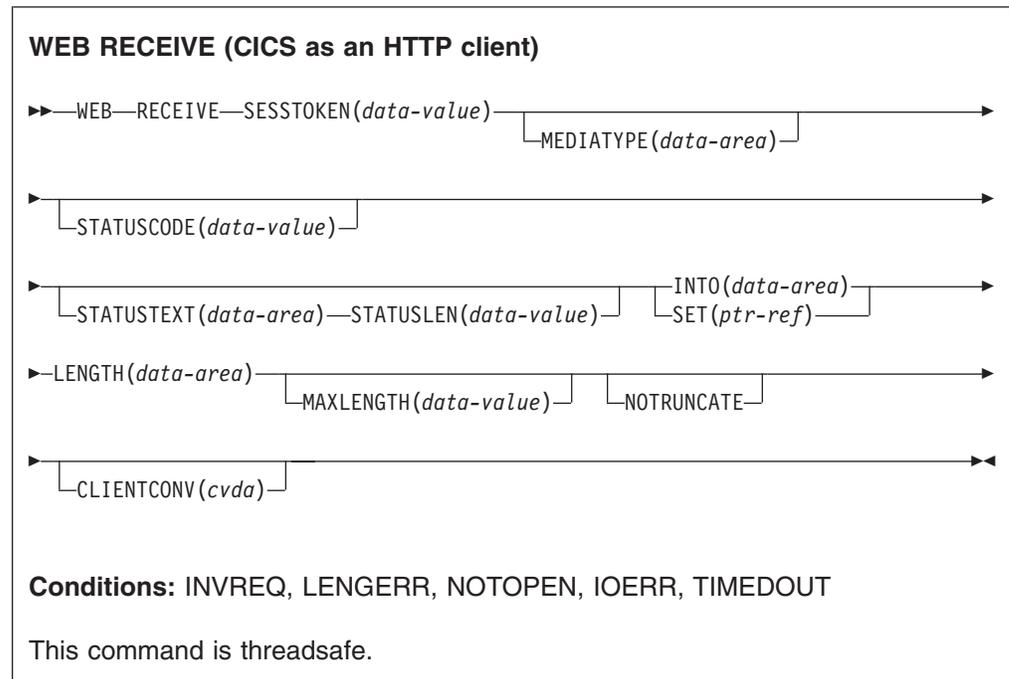
NOTFND

RESP2 values are:

- 7 Code page not found.
- 82 Client code page (character set) not found.
- 83 Host code page (for server) not found.

WEB RECEIVE (Client)

Receive an HTTP response for CICS as an HTTP client.



Description

WEB RECEIVE for CICS as an HTTP client receives the body of an HTTP response that a server has made. The headers for the HTTP response can be examined separately using the WEB READ HTTPHEADER command or the HTTP header browsing commands. A session token must be included on this command. For guidance on the correct use of the WEB RECEIVE command for CICS as an HTTP client, see "HTTP client requests from a CICS application" in the *CICS Internet Guide*.

Code page conversion for the incoming message can be specified on this command.

Note: The RTIMOUT value specified for the transaction that starts the user application indicates the time that the application is prepared to wait to receive the incoming message. (RTIMOUT is specified on the transaction profile definition). When the period specified by RTIMOUT expires, CICS returns a TIMEDOUT response to the application. An RTIMOUT value of zero means that the application is prepared to wait indefinitely. The default setting for RTIMOUT on transaction profile definitions is zero, so it is important to check and change that setting for applications that are making HTTP client requests.

Tip: For CICS as an HTTP client, the CONVERSE command can be used as an alternative to issuing a WEB SEND command followed by a WEB RECEIVE command.

Options

CLIENTCONV (*cvda*)

specifies whether or not CICS translates the entity body of the response from the character set used by the server, to a code page suitable for the application. The default is that the entity body is converted.

CLICONVERT

CICS converts the entity body of the response from the character set used by the server, into the code page that you identify for the application.

NOCLICONVERT

CICS does not convert the entity body of the response, and it is passed to the application in the character set used by the server.

You do not need to specify a character set or application code page on the WEB RECEIVE command for CICS as an HTTP client. If code page conversion is required, CICS identifies the character set used by the server by examining the Content-Type header of the message. If the header does not provide this information, or if the named character set is not supported by CICS for code page conversion, the ISO-8859-1 character set is used. For the application's code page, the default code page for the local CICS region (as specified in the LOCALCCSID system initialization parameter) is used, or an alternative EBCDIC code page that you specified on the WEB OPEN COMMAND.

Note that for code page conversion to take place, the media type for the message must specify a type of data content that can be identified as text according to the IANA definitions. For messages where no media type is given but CLICONVERT is specified, code page conversion also takes place. If a non-text media type is present, CICS does not convert the message body.

INTO (*data-area*)

specifies the buffer that is to contain the data being received.

LENGTH (*data-area*)

specifies a fullword binary variable which is set to the amount of data that CICS has returned to the application. Note that this might be slightly less than the limit that you set using the MAXLENGTH option, especially if a double-byte or multi-byte character set is involved, because CICS does not return a partial character at the end of the data.

- If the NOTRUNCATE option **is not** specified, any further data present in the message has now been discarded. A LENGERR response with a RESP2 value of 57 is returned if further data was present.
- If the NOTRUNCATE option **is** specified, any additional data is retained. A LENGERR response with a RESP2 value of 36 is returned if additional data is available. The description for the NOTRUNCATE option tells you what to do in this case.

MAXLENGTH (*data-value*)

specifies the maximum amount, as a fullword binary value, of data that CICS is to pass to the application. The MAXLENGTH option applies whether the INTO or the SET option is specified for receiving data. If the data has been sent using chunked transfer-coding, CICS assembles the chunks into a single message before passing it to the application, so the MAXLENGTH option applies to the total length of the chunked message, rather than to each individual chunk. The data is measured after any code page conversion has taken place.

If the length of data exceeds the value specified and the NOTRUNCATE option **is not** specified, the data is truncated to that value, and the remainder of the data is discarded.

If the length of data exceeds the value specified and the NOTRUNCATE option **is** specified, CICS retains the remaining data and can use it to satisfy subsequent RECEIVE commands.

MEDIATYPE (*data-area*)

specifies a 56-character data-area to receive the media type (that is, the type of data content) for the body, for example `text/xml`. See the *CICS Internet Guide* for more information about media types.

NOTRUNCATE

specifies that when the data available exceeds the length requested on the MAXLENGTH option, the remaining data is not to be discarded immediately but is to be retained for retrieval by subsequent RECEIVE commands. (If no further RECEIVE commands are issued, the data is discarded during transaction termination.)

A single RECEIVE command using the SET option and without the MAXLENGTH option receives all the remaining data, whatever its length. Alternatively, you can use a series of RECEIVE commands with the NOTRUNCATE option to receive the remaining data in appropriate chunks. Keep issuing the RECEIVE command until you are no longer getting a LENGERR response. Bear in mind that if you receive less than the length requested on the MAXLENGTH option, this does not necessarily indicate the end of the data; this could happen if CICS needs to avoid returning a partial character at the end of the data.

SET (*ptr-ref*)

specifies a pointer reference that is to be set to the address of data received. The pointer reference is valid until the next receive command or the end of task.

SESSTOKEN (*data-value*)

specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

STATUSCODE (*data-value*)

specifies a data-area to receive the HTTP status code sent by the server. The code is a binary halfword value. Examples are 200 (normal) or 404 (not found). Receiving the status code is optional, but you should always receive and check the status code in the following circumstances:

- If you intend to make an identical request to the server, now or during a future connection.
- If you intend to make further requests to the server using this connection.
- If your application is carrying out any further processing that depends on the information you receive in the response.

The *CICS Internet Guide* has basic guidance on appropriate actions for an application to take in response to the status codes for HTTP/1.1.

STATUSTEXT (*data-area*)

specifies a data-area to receive any text returned by the server to describe the status code. The text is known as a reason phrase. Examples are "OK" (accompanying a 200 status code), or "Bad Request" (accompanying a 400 status code). The STATUSLEN option gives the length allowed for the text.

STATUSLEN(*data-value*)

specifies, as a fullword binary value, the length of the data-area to receive any text returned by the server to describe the status code (the STATUSTEXT option). The text is known as a reason phrase. Most reason phrases recommended for HTTP are short, but a data-area length of 256 characters is suggested here, in case the server replaces the recommended reason phrase with more detailed information.

Conditions**NOTOPEN**

RESP2 values are:

27 Invalid session token.

INVREQ

RESP2 values are:

10 Invalid response header.

15 Code page conversion failure.

22 Invalid chunk received.

41 The connection has been closed.

46 The CLIENTCONV option is invalid.

67 HTTP error in response.

68 Message send with chunked transfer-coding is in progress.

71 Chunked transfer-coding error.

LENGERR

RESP2 values are:

16 Invalid MAXLENGTH.

36 Partial response body returned. Use additional RECEIVES to obtain remainder.

57 The response body exceeds the length specified, and the remainder of the body has been discarded.

58 The status text exceeds the length specified and has been truncated.

59 The STATUSLEN option value was not greater than zero.

IOERR

RESP2 values are:

42 Socket error.

TIMEDOUT

RESP2 values are:

62 Timeout on socket receive.

WEB RETRIEVE

Retrieve the DOCTOKEN for a CICS document that was sent using a WEB SEND command.

WEB RETRIEVE

►►—WEB—RETRIEVE—DOCTOKEN(*data-area*)—————►◄

Conditions:INVREQ

This command is threadsafe.

Description

#

The WEB RETRIEVE command retrieves the DOCTOKEN of the document which was sent using an earlier WEB SEND command. The command can only retrieve the DOCTOKEN of a document which was sent with the ACTION(EVENTUAL) option specified on the WEB SEND command. If the ACTION(IMMEDIATE) option was used, the WEB RETRIEVE command cannot retrieve the DOCTOKEN, and an INVREQ response with a RESP2 value of 2 is returned.

Options

DOCTOKEN(*data-area*)

specifies a 16-byte buffer to contain the symbolic name of the document to be retrieved.

Conditions

INVREQ

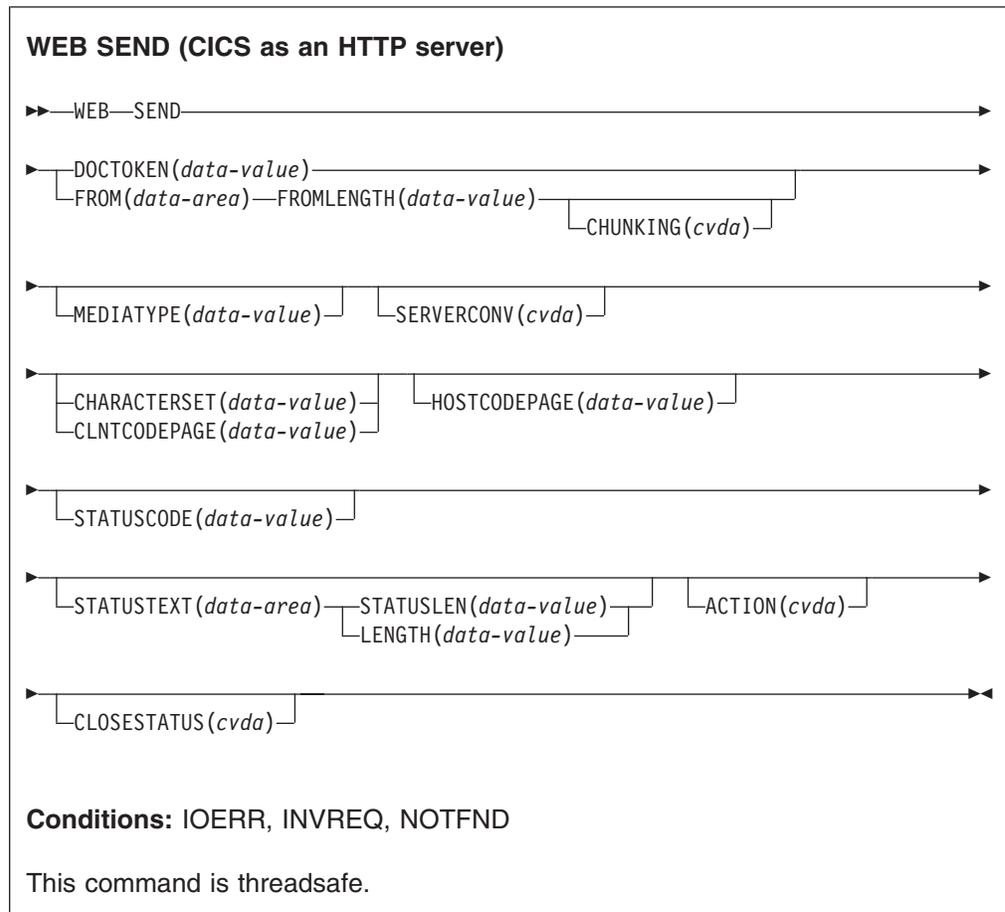
RESP2 values:

- 1 The command is issued in a non-CICS Web support application.
- 2 A WEB SEND command has not been issued, or a WEB SEND command has been issued but with the ACTION(IMMEDIATE) option.

#

WEB SEND (Server)

Send an HTTP response, or a non-HTTP message.



Description

WEB SEND for CICS as an HTTP server selects an item for delivery by CICS Web support or the CICS business logic interface, and specifies options for sending it. The item can be:

- A response to an HTTP request that was made by a Web client, to CICS as an HTTP server. For guidance on the correct use of the WEB SEND command for this purpose, see "Writing Web-aware application programs for CICS as an HTTP server" in the *CICS Internet Guide*.
- A non-HTTP message handled by CICS Web support facilities, with the user-defined (USER) protocol on the TCPIP SERVICE definition. For guidance on non-HTTP messages, see "CICS Web support and non-HTTP requests" in the *CICS Internet Guide*.
- A response to a request from another application that has used the CICS business logic interface to contact the program directly, rather than going through the CICS HTTP listener. For guidance on the CICS business logic interface, see "The CICS business logic interface" in the *CICS Internet Guide*.

Only one response can be sent during a task. This can be a standard response using one WEB SEND command, or a chunked response using a sequence of WEB SEND commands.

If you attempt to send a second response during the same task, the result depends on whether the ACTION(IMMEDIATE) option or the ACTION(EVENTUAL) option was specified on the WEB SEND command for the first response.

- If the ACTION(IMMEDIATE) option was used for the first response, an error is returned when you attempt the second response.
- If the ACTION(EVENTUAL) option was used for the first response, the second response overwrites the components of the previous response (status line, HTTP headers and message body). The first response is lost, and the second response is sent.

Each time a request from a Web client is received, CICS starts a new task to process the request.

Options

ACTION(*cvda*)

specifies how the message should be sent out. The CVDA values that apply for CICS as an HTTP server are:

IMMEDIATE

sends the response immediately to the Web client. If CHUNKING is specified, the IMMEDIATE option is assumed. For message sends that do not use chunked transfer-coding, EVENTUAL is the default, which sends the response at end of task.

EVENTUAL

sends the response to the Web client at end of task. If CHUNKING is specified, the EVENTUAL option is ignored. This option produces the same behaviour as CICS Web support had in releases before CICS Transaction Server for z/OS, Version 3 Release 1, and it is the default for CICS as an HTTP server.

CHARACTERSET(*data-value*)

specifies a character set into which CICS translates the entity body of the item sent by the command before sending. The name of the character set can consist of up to 40 alphanumeric characters, including appropriate punctuation. CICS does not support all the character sets named by IANA. The *CICS Internet Guide* lists the IANA character sets that are supported by CICS for code page conversion.

When the CHARACTERSET option is specified, SERVERCONV(SRVCONVERT) is assumed, so code page conversion of the entity body takes place. As an alternative to selecting the character set yourself, specifying either SERVERCONV(SRVCONVERT), or HOSTCODEPAGE (if allowed), or both, and omitting CHARACTERSET, lets CICS determine a suitable character set for the message body. The description for the SERVERCONV option tells you what happens in this case.

If you omit all of the code page conversion options, no code page conversion takes place.

CHUNKING(*cvda*)

is used for controlling the message send when the message is being sent in chunks (known as chunked transfer-coding). The default when the option is not

specified is that chunked transfer-coding is not in use. Chunked transfer-coding is only acceptable to HTTP/1.1 clients, and it cannot be used with HTTP/1.0 clients or non-HTTP messages.

The content of a chunked message can be divided into chunks in whatever way is most convenient for the application program. The body of a chunked message cannot be formed directly from CICS documents, so the DOCTOKEN option cannot be used.

Use a separate WEB SEND command with CHUNKING(CHUNKYES) for each chunk of the message. Use the FROM option to specify the chunk of data, and the FROMLENGTH option to specify the length of the chunk. Other options for the message, such as the CLOSESTATUS option, can be specified on the first WEB SEND command of the sequence (which sends the first chunk), but do not specify them on subsequent commands (which send the second and subsequent chunks).

When you have sent the last chunk of the data, specify a further WEB SEND command with CHUNKING(CHUNKEND) and no FROM or FROMLENGTH option. CICS then sends an empty chunk to the recipient to complete the chunked message.

If one of the WEB SEND commands fails during the sequence, an error response is returned, and subsequent sends will also fail. The application should handle this situation appropriately. If all of the chunks are sent successfully but the application does not issue the final WEB SEND command with CHUNKING(CHUNKEND), the transaction is abended with abend code AWBP. An incomplete chunked message should be ignored and discarded by the recipient.

The *CICS Internet Guide* has a full description of the procedure for chunked transfer-coding, which should be followed in order for your chunked message to be acceptable to the recipient. CVDA values are:

CHUNKNO

Chunked transfer-coding is not used for the message. This is the default if the CHUNKING option is not specified.

CHUNKYES

Chunked transfer-coding is in progress. The data specified by the FROM option represents a chunk of the message.

CHUNKEND

Chunked transfer-coding is complete. No data is specified for this send. CICS sends an empty chunk to the recipient to complete the chunked message.

CLNTCODEPAGE (*data-value*)

This option is supported for migration purposes only. CHARACTERSET replaces it. The action taken by CICS is the same for both keywords. This means that code page conversion does take place when CLNTCODEPAGE is specified, even if the SERVERCONV option is not specified. Code page conversion does not take place if all the code page conversion options are omitted.

CLOSESTATUS (*cvda*)

specifies whether or not CICS closes the connection after sending the message. The default is that the connection is not closed. The CVDA values are:

CLOSE

CICS writes a Connection header with the "close" connection option (Connection: close) for this response, and closes the connection with the Web client after sending the response. The header notifies the Web client of the closure. (For a Web client at HTTP/1.0 level, CICS achieves the same effect by omitting the Connection: Keep-Alive header.)

If chunked transfer-coding is in use, the CLOSESTATUS(CLOSE) option can be specified on the first chunk of the message, to inform the Web client that the connection is closed after the chunked message is complete.

NOCLOSE

means that the Connection: close header is not used for this response, and the connection is kept open. If the Web client is identified as HTTP/1.0 and has sent a Connection header with the "Keep-Alive" connection option (Connection: Keep-Alive), CICS sends the same header, to notify that a persistent connection will be maintained.

DOCTOKEN (*data-value*)

specifies the 16-byte binary token of a document to be sent as the message body. The document is created using the CICS Document interface (EXEC CICS DOCUMENT CREATE, INSERT, and SET commands), as described in the *CICS Application Programming Guide*. The FROM option provides an alternative way to create a message body.

The body of a chunked message cannot be formed from CICS documents, so the DOCTOKEN option cannot be used for chunked transfer-coding.

CICS documents cannot be converted to the UTF-8 and UTF-16 character encodings.

FROM (*data-area*)

specifies a buffer of data which holds the complete message body, or a chunk of the message body. The message body is built by the application program. When you specify the FROM option, use the FROMLENGTH option to specify the length of the buffer of data. The DOCTOKEN option provides an alternative way to create the message body, but that option cannot be used for the body of a chunked message.

There is no set maximum limit for the size of the data-area, but its size is limited in practice by storage considerations. The *CICS Internet Guide* has more information about these.

FROMLENGTH (*data-value*)

specifies the length, as a fullword binary value, of the buffer of data supplied on the FROM option. It is important to state this value correctly, because an incorrect data length can cause problems for the recipient of the message.

HOSTCODEPAGE (*data-value*)

specifies the 8-character name of the CICS (host) code page that was used by the application program for the entity body of the response. When the HOSTCODEPAGE option is specified, SERVERCONV(SRVCONVERT) is assumed, so code page conversion of the entity body takes place. Specifying either SERVERCONV(SRVCONVERT), or CHARACTERSET, or both, and omitting HOSTCODEPAGE, lets CICS identify the host code page.

If a CICS document is used to form the response body (DOCTOKEN option), do not specify the HOSTCODEPAGE option, because CICS identifies the host code page from the CICS document domain's record of the host code pages for the document.

If a buffer of data is used to form the response body (FROM option), you may need to specify HOSTCODEPAGE. The default if this option is not present is the default code page for the local CICS region, as set in the LOCALCCSID system initialization parameter. If you require code page conversion but your application has used a different code page, use HOSTCODEPAGE to specify it.

If you omit all of the code page conversion options, no code page conversion takes place.

LENGTH(*data-value*)

This option is supported for migration purposes only. STATUSLEN replaces it.

MEDIATYPE(*data-value*)

specifies the data content of the message body, for example `text/xml`. The media type is up to 56 alphanumeric characters, including appropriate punctuation, but not spaces.. See the *CICS Internet Guide* for more information about media types. CICS checks that the format of the media type is correct, but does not check the validity of the media type against the data content. CICS does not provide a default. In some circumstances, the media type that you specify affects whether or not code page conversion is carried out; see the description of the SERVERCONV option for more information.

SERVERCONV(*cvda*)

specifies whether or not CICS translates the entity body of the item sent by the command before sending, from the code page used by the application, to a character set suitable for the recipient. You can use the CHARACTERSET and HOSTCODEPAGE options on this command to specify the character set and code page that are used. If you specify either of these options, code page conversion (SRVCONVERT) is assumed. Alternatively, you can omit either or both of these options, specify SERVERCONV(SRVCONVERT) and let CICS determine a suitable character set and code page.

SRVCONVERT

CICS converts the entity body of the message.

When you specify SRVCONVERT without CHARACTERSET, CICS determines a suitable character set as follows:

1. If the Web client's request has a Content-Type header naming a character set supported by CICS, that character set is used.
2. If the Web client's request has no Content-Type header or the named character set is unsupported, the ISO-8859-1 character set is used.
3. For non-HTTP messages (sent using the USER protocol), the ISO-8859-1 character set is used.

When you specify SRVCONVERT without HOSTCODEPAGE, CICS identifies the host code page as follows:

- If the FROM option is used, CICS identifies the host code page as the default code page for the local CICS region, as specified in the LOCALCCSID system initialization parameter.
- If the DOCTOKEN option is used, CICS identifies the host code page from the CICS document domain's record of the host code pages for the document.

#

If you specify SRVCONVERT alone, note that for code page conversion to take place, the MEDIATYPE option must specify a type of data content that can be identified as text according to the IANA definitions. For non-text media types, CICS does not convert the message body. However, for compatibility with Web-aware applications coded in earlier releases, if you specify either of the CHARACTERSET or HOSTCODEPAGE options or omit the SERVERCONV option, the MEDIATYPE option does not influence code page conversion.

NOSRVCONVERT

CICS does not convert the entity body of the HTTP request, and it is sent to the server in the code page used by the application. If you specify NOSRVCONVERT, you cannot specify the CHARACTERSET or HOSTCODEPAGE options.

Note: If you omit all of the code page conversion options (SERVERCONV, CLNTCODEPAGE, CHARACTERSET, HOSTCODEPAGE), no code page conversion takes place.

STATUSCODE(*data-value*)

specifies a standard HTTP status code determined by the application program, which is to be inserted on the status line of the HTTP response. The code is a halfword binary value. Examples are 200 (normal response) or 404 (not found). If this option is not specified, CICS supplies a default of 200.

The *CICS Internet Guide* has information about the use of status codes for CICS Web support. For status codes 204, 205, and 304, a message body is not allowed, and CICS returns an error response to the command if you attempt to include one. Other than that, CICS does not check that your use of the status code is appropriate.

STATUSLEN(*data-value*)

specifies the length, as a fullword binary value, of the string supplied on the STATUSTEXT option.

STATUSTEXT(*data-area*)

specifies a data-area containing human-readable text to describe the reason for the status code. The text is known as a reason phrase. Examples are "OK" (accompanying a 200 status code), or "Bad Request" (accompanying a 400 status code). The HTTP/1.1 specification (RFC 2616) defines a recommended reason phrase for each status code, but you do not have to use these.

Conditions

INVREQ

RESP2 values are:

- 1** The command is being issued in a non-CICS Web support application.
- 11** Action code invalid.
- 13** Close status invalid.
- 14** Invalid code page combination.
- 32** Media type invalid.
- 41** The connection has been closed.
- 46** The SERVERCONV option is invalid.
- 72** Status code does not support a message body.

- 75 Invalid send sequence.
- 77 Chunk incomplete.
- 80 CHARACTERSET cannot be specified with SERVERCONV(NOSRVCONVERT).
- 81 HOSTCODEPAGE cannot be specified with SERVERCONV(NOSRVCONVERT).
- 85 Chunking cannot be used with non-HTTP messages.
- 86 Chunking cannot be used with HTTP/1.0 clients.
- 87 Status code not allowed.
- 88 Host code page not allowed.
- 89 Previous send over this connection failed. No further sends permitted.
- # 90 STATUSCODE and STATUSTEXT options not allowed for second or
- # subsequent chunks.
- # 91 CHARACTERSET and CLNTCODEPAGE options not allowed for
- # second or subsequent chunks.
- # 92 HOSTCODEPAGE option not allowed for second or subsequent
- # chunks.
- # 93 MEDIATYPE option not allowed for second or subsequent chunks.
- # 94 CLOSESTATUS option not allowed for second or subsequent chunks.
- # 95 SERVERCONV option not allowed for second or subsequent chunks.
- 120 The CHUNKING option is invalid.
- 121 FROMLENGTH option required.
- 122 FROM option required.
- 123 No message body specified. Use FROM, DOCTOKEN or
- CHUNKING(CHUNKEND).
- 124 CHUNKING option not specified, FROMLENGTH option required.
- 125 CHUNKNO specified, FROM option required.
- 126 CHUNKNO specified, FROMLENGTH option required.
- 127 CHUNKYES specified, FROM option required.
- 128 CHUNKYES specified, FROMLENGTH option required.
- 129 FROM option not allowed with CHUNKING(CHUNKEND).
- 130 FROMLENGTH option not allowed with CHUNKING(CHUNKEND).
- 131 FROMLENGTH option specified as zero.

NOTFND

RESP2 values are:

- 1 The document has not been created or the name is incorrectly specified.
- 7 Client code page (character set) not found.
- 83 Host code page (for server) not found.

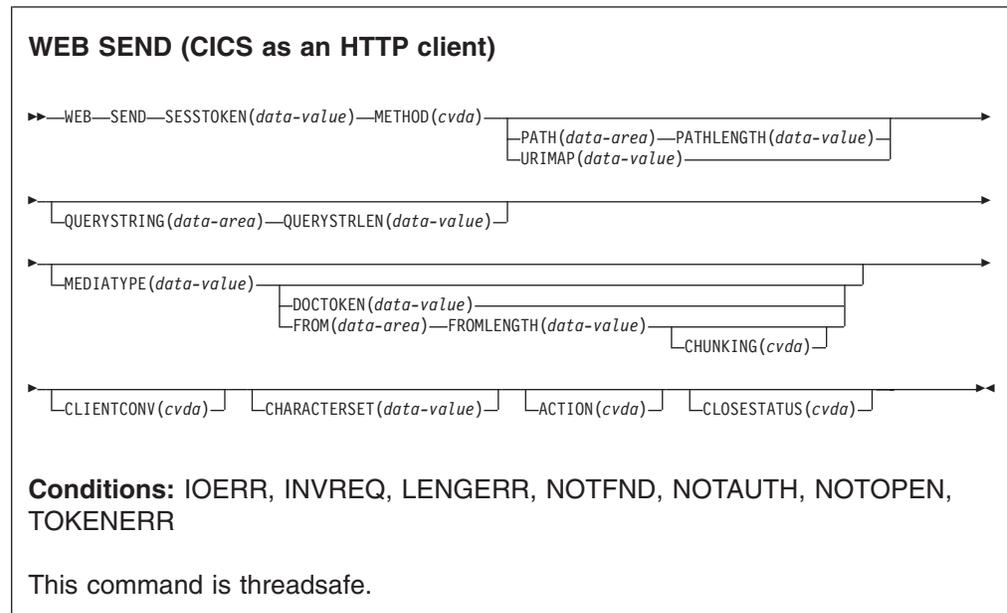
IOERR

RESP2 values are:

42 Socket error.

WEB SEND (Client)

Send an HTTP request by CICS as an HTTP client, using CICS Web support.



Description

WEB SEND for CICS as an HTTP client is used to make an HTTP request to a server. A session token must be included on this command. For guidance on the correct use of the WEB SEND command for CICS as an HTTP client, see .

Tip: For CICS as an HTTP client, the WEB SEND command cannot be used after the connection to the server has been closed. This happens if either the application program, or the server, sends a Connection: close header on a message. If you need to test whether the server has requested termination of the connection, use the WEB READ HTTPHEADER command to look for the Connection: close header in the last message from the server.

Tip: For CICS as an HTTP client, the CONVERSE command can be used as an alternative to issuing a WEB SEND command followed by a WEB RECEIVE command. However, bear in mind that the WEB CONVERSE command does not support chunked transfer-coding, because this requires a sequence of send actions, and the WEB CONVERSE command provides a single send action.

Options

ACTION(cvda)

This option is used to specify how the message should be sent out. The CVDA value that applies for CICS as an HTTP client is:

EXPECT

makes CICS send an Expect header along with the request line and headers for the request, and await a 100-Continue response before sending the message body to the server. If a response other than 100-Continue is received, CICS informs the application program and

cancels the send. If no response is received after a period of waiting, CICS sends the message body anyway.

This option must only be used if your request has a message body.

CHARACTERSET (*data-value*)

specifies the character set into which CICS translates the entity body of the request before sending. The name of the character set can consist of up to 40 alphanumeric characters, including appropriate punctuation. CICS does not support all the character sets named by IANA. The *CICS Internet Guide* lists the IANA character sets that are supported by CICS for code page conversion.

For conversion of the entity body to take place, the CLIENTCONV option must be specified as (or allowed to default to) CLICONVERT. Specifying NOCLICONVERT suppresses conversion of the entity body. If conversion is requested, ISO-8859-1 is used as the default if the CHARACTERSET attribute is not specified.

CHUNKING (*cvda*)

is used for controlling the message send when the message is being sent in chunks (known as chunked transfer-coding). The default when the option is not specified is that chunked transfer-coding is not in use.

The content of a chunked message can be divided into chunks in whatever way is most convenient for the application program. The body of a chunked message cannot be formed directly from CICS documents, so the DOCTOKEN option cannot be used.

Use a separate WEB SEND command with CHUNKING(CHUNKYES) for each chunk of the message. Use the FROM option to specify the chunk of data, and the FROMLENGTH option to specify the length of the chunk. Other options for the message, such as the CLOSESTATUS option, can be specified on the first WEB SEND command of the sequence (which sends the first chunk), but do not specify them on subsequent commands (which send the second and subsequent chunks).

When you have sent the last chunk of the data, specify a further WEB SEND command with CHUNKING(CHUNKEND) and no FROM or FROMLENGTH option. CICS then sends an empty chunk to the recipient to end the chunked message.

If your application program is informed of an error at any point in the chunking process, use the WEB CLOSE command to stop the process and close the connection. The recipient of the chunked message will not receive the final empty chunk, and so should ignore and discard the data that you have sent so far.

The *CICS Internet Guide* has a full description of the procedure for chunked transfer-coding, which should be followed in order for your chunked message to be acceptable to the recipient. CVDA values are:

CHUNKNO

Chunked transfer-coding is not used for the message. This is the default if the CHUNKING option is not specified.

CHUNKYES

Chunked transfer-coding is in progress. The data specified by the FROM option represents a chunk of the message.

CHUNKEND

Chunked transfer-coding is complete. No data is specified for this send. CICS sends an empty chunk to the recipient to complete the chunked message.

Note:

1. The method (METHOD option) must be compatible with chunked transfer-coding.
2. When you have begun sending the parts of a chunked message, the application program cannot send any different messages or receive any items until the final empty chunk is sent and the chunked message is complete.

CLOSESTATUS(*cvda*)

specifies whether or not a Connection header with the "close" connection option (Connection: close) should be included on the message. The default is that the header is not included. The CVDA values are:

CLOSE

makes CICS write a Connection: close header for this request. The header notifies the server that the connection should be closed after the server has sent its response to the request. (For a server at HTTP/1.0 level, CICS achieves the same effect by omitting the Connection: Keep-Alive header.)

If chunked transfer-coding is in use, the CLOSESTATUS(CLOSE) option can be specified on the first chunk of the message, to inform the server that the connection should be closed after the chunked message is complete and a response has been sent.

If chunked transfer-coding is not in use, and the CLOSESTATUS(CLOSE) option is specified on a WEB SEND command, no further messages can be sent to the server until a new connection is made.

NOCLOSE

means that the Connection: close header is not used for this request. If the server is identified as HTTP/1.0, CICS sends a Connection header with the "Keep-Alive" connection option (Connection: Keep-Alive), to notify that a persistent connection is desired.

CLIENTCONV(*cvda*)

specifies whether or not CICS translates the entity body of the HTTP request before sending, from the code page used by the application, to a character set suitable for the recipient. If this option is omitted, the default is that any entity body **is** converted, unless a non-text media type is specified. CVDA values are:

CLICONVERT

CICS converts the entity body of the HTTP request from the code page used by the application, into the character set that you identify for the server. You can use the CHARACTERSET option on this command to specify the character set that is used. If conversion is requested but you do not specify a character set, the default is that CICS converts the entity body to the ISO-8859-1 character set. (The code page used by the application was identified on the WEB OPEN command for the connection.)

Note: For non-text media types, CICS does not convert the message body, even if CLICONVERT is specified.

NOCLICONVERT

CICS does not convert the entity body of the HTTP request, and it is sent to the server in the code page used by the application, as identified on the WEB OPEN command for the connection.

DOCTOKEN(*data-value*)

specifies the 16-byte binary token of a document to be sent as the message body. The document must be created using the CICS Document interface (EXEC CICS DOCUMENT CREATE, INSERT, and SET commands), as described in the *CICS Application Programming Guide*. You do not need to retrieve the document before sending it. The FROM option provides an alternative way to create a message body.

The body of a chunked message cannot be formed from CICS documents, so the DOCTOKEN option cannot be used for chunked transfer-coding.

CICS documents cannot be converted to the UTF-8 and UTF-16 character encodings.

FROM(*data-area*)

specifies a buffer of data which holds the message body. The message body is built by the application program. When you specify the FROM option, use the FROMLENGTH option to specify the length of the buffer of data. The DOCTOKEN option provides an alternative way to create the message body, but that option cannot be used for the body of a chunked message.

There is no set maximum limit for the size of the data-area, but its size is limited in practice by storage considerations. The *CICS Internet Guide* has more information about these.

FROMLENGTH(*data-value*)

specifies the length, as a fullword binary value, of the buffer of data supplied on the FROM option (the message body). It is important to state this value correctly, because an incorrect data length can cause problems for the recipient of the message.

MEDIATYPE(*data-value*)

specifies the data content of any message body provided, for example text/xml. The media type is up to 56 alphanumeric characters, including appropriate punctuation, but not spaces. See the *CICS Internet Guide* for more information about media types. CICS checks that the format of the media type is correct, but does not check the validity of the media type against the data content. CICS uses this information to produce the Content-Type header for the message.

#

#

#

#

#

#

#

#

For requests which require a body, you must specify the MEDIATYPE option. There is no default. However, if the required Content-Type header needs to contain spaces or more than 56 characters, the application can provide it using the WEB WRITE HTTPHEADER command. In this case, do not specify the MEDIATYPE option.

For code page conversion to take place, the MEDIATYPE option must specify a type of data content that can be identified as text according to the IANA definitions. For non-text media types, CICS does not convert the message body.

METHOD(*cvda*)

specifies the HTTP method for the request.

The GET, HEAD, POST, PUT, TRACE, OPTIONS, and DELETE methods are supported by this command. However, some HTTP servers, particularly HTTP/1.0 servers, might not implement all of these methods.

The *CICS Internet Guide* has more information about the correct use of methods, including the HTTP versions that apply to each.

CICS bars the sending of a message body for methods where it is inappropriate, and requires it for methods where it is appropriate. Chunked transfer-coding is not relevant for methods that do not have a request body. The CVDA values are:

GET Obtain a resource from the server. A request body is not allowed.

HEAD Obtain the HTTP headers, but not the response body, for a resource. A request body is not allowed.

POST Send data to a server. A request body is required.

PUT Create or modify a resource on the server. A request body is required.

TRACE

Trace the route of your request to the server. A request body is not allowed.

OPTIONS

Obtain information about the server. A request body is allowed, but there is no defined purpose for the body. If you do use a request body, then you must specify a media type.

DELETE

Delete a resource on the server. A request body is not allowed.

PATH(*data-area*)

specifies the path information for the specific resource within the server that the application needs to access.

If the URIMAP option was used to specify an existing URIMAP definition on the WEB OPEN command for this connection, the path specified in that URIMAP definition is the default path for the WEB SEND command. In these circumstances, if you do not specify path information on the WEB SEND command, the path from the URIMAP definition is used. If you specify a different path from that given in the URIMAP definition, this overrides the path from the URIMAP definition.

If the URIMAP option was not used on the WEB OPEN command, there is no default path, and you must provide path information. Path information can be extracted from a known URL using the WEB PARSE URL command.

As an alternative to using the PATH option to provide the path information, you can use the URIMAP option on the WEB SEND command to specify a URIMAP definition from which the path information is taken directly.

PATHLENGTH(*data-value*)

specifies the length of the path, as a fullword binary value. If you are providing path information using the PATH option, you need to specify the PATHLENGTH option. Path length information is returned if you use the WEB PARSE URL command to parse a URL.

QUERYSTRING(*data-area*)

specifies a query string that is to be supplied to the server as part of the request. You do not need to include a question mark (?) at the beginning of the query string; if you do not include it, CICS supplies it for you automatically when constructing the request. If you include escaped characters in the query string, CICS passes them to the server in their escaped format.

QUERYSTRLEN(*data-value*)

specifies the length of the query string supplied on the QUERYSTRING option, as a fullword binary value.

SESSTOKEN(*data-value*)

specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

URIMAP(*data-value*)

specifies the name (up to eight characters, in mixed case) of a URIMAP definition that provides the path information for the specific resource within the server that the application needs to access. The URIMAP definition must be for CICS as an HTTP client (with USAGE(CLIENT) specified). Its HOST attribute must be the same as the HOST attribute of the URIMAP definition that was specified on the WEB OPEN command for this connection, or the same as the host name specified in the HOST option on the WEB OPEN command for this connection. A URIMAP definition specified on the WEB SEND command applies only to this request.

If the URIMAP option is specified, do not specify the PATH or PATHLENGTH options.

Conditions**NOTOPEN**

RESP2 values are:

27 Invalid session token.

INVREQ

RESP2 values are:

11 Action code invalid.

12 URIMAP and PATH specified. Only one allowed. OR: URIMAP option not allowed for second or subsequent chunks.

13 Close status invalid.

15 Code page conversion failure.

17 Expect-100 request was rejected by the server.

22 Invalid chunk size.

32 Media type invalid.

33 Method does not support a body.

34 Method requires a body.

45 The character set specified is invalid.

46 The CLIENTCONV option is invalid.

49 The format of the path option is invalid.

54 The HTTP method is not valid.

63 URIMAP object disabled.

64 Host in URIMAP definition does not match host specified when this session was opened.

69 Chunked transfer-coding not supported with this HTTP version.

#

- 71 Chunked transfer-coding error.
- 74 The connection has been closed.
- 76 MEDIATYPE option required.
- 79 Pipelining is in progress. Expect header cannot be sent.
- 120 The CHUNKING option is invalid.
- 121 FROMLENGTH option required.
- 122 FROM option required.
- 123 No message body specified. Use FROM, DOCTOKEN or CHUNKING(CHUNKEND).
- 124 CHUNKING option not specified, FROMLENGTH option required.
- 125 CHUNKNO specified, FROM option required.
- 126 CHUNKNO specified, FROMLENGTH option required.
- 127 CHUNKYES specified, FROM option required.
- 128 CHUNKYES specified, FROMLENGTH option required.
- 129 FROM option not allowed with CHUNKING(CHUNKEND).
- 130 FROMLENGTH option not allowed with CHUNKING(CHUNKEND).
- 131 FROMLENGTH option specified as zero.
- 132 METHOD option not allowed for second or subsequent chunks.
- # 133 MEDIATYPE option not allowed for second or subsequent chunks, or
when Content-Type header already provided.
- # 135 PATH option not allowed for second or subsequent chunks.
- # 136 METHOD option required.

LENGERR

RESP2 values are:

- 5 The PATHLENGTH option value was not greater than zero.
- 8 The QUERYSTRLEN option value was not greater than zero.
- 50 The FROMLENGTH option value was not greater than zero.

NOTFND

RESP2 values are:

- 61 The URIMAP object specified was not found.

TOKENERR

RESP2 values are:

- 47 The document token specified is invalid.

IOERR

RESP2 values are:

- 42 Socket error.

NOTAUTH

RESP2 values are:

- 100 Path barred by security exit.

WEB STARTBROWSE FORMFIELD

Signal start of HTML form field browse.

WEB STARTBROWSE FORMFIELD

►►—WEB—STARTBROWSE—FORMFIELD(*data-area*)—NAMELENGTH(*data-area*)—►►

►—CLNTCODEPAGE(*name*)—HOSTCODEPAGE(*name*)—►

Conditions: INVREQ, LENGERR, NOTFND

This command is threadsafe.

Description

WEB STARTBROWSE FORMFIELD signals the start of a browse of a set of name-value pairs in an HTML form that is part of the body of an HTTP request being processed by the current CICS task.

Options

CLNTCODEPAGE(*name*)

specifies the 40-character name of the character set that was used by the Web client for the HTTP request. If this is not specified, CICS obtains it from the charset parameter on the Content-Type header of the HTTP request. If the Content-Type header is not present, CICS assumes ISO-8859-1, the default character set for the Internet. If you specify CLNTCODEPAGE you must also specify HOSTCODEPAGE. CICS does not support all the character sets named by IANA. The *CICS Internet Guide* lists the IANA character sets that are supported by CICS for code page conversion.

FORMFIELD(*data-area*)

specifies the name of the form field at which browsing is to start. It is a string of text containing the name of the requested field. If a name is not specified, browsing starts at the first name/value pair in the HTML form.

HOSTCODEPAGE(*name*)

specifies the 8-character name of the CICS (host) code page required by the
application program, into which the form data is to be converted. If you specify
HOSTCODEPAGE you must also specify CLNTCODEPAGE. This code page is
normally an EBCDIC code page. If this is not specified, the default is the
EBCDIC code page 037.

If the form data is in a query string (GET method), this option is ignored, and
the data is returned in the EBCDIC code page specified by the LOCALCCSID
system initialization parameter (which applies to the whole of the local CICS
region, and has a default of 037). If the ASCII Latin-1 character set ISO-8859-1
(code page 819) cannot be converted into that code page, CICS uses the
default EBCDIC code page 037 instead.

NAMELENGTH(*data-value*)

specifies the length, as a fullword binary value, of the form field name. This field must be specified if FORMFIELD is specified.

Conditions

INVREQ

occurs for the following conditions. RESP2 values are:

- 1 The command is being issued in a non-CICS Web support application.
- 3 The command is being issued for a non-HTTP request.
- 5 There is already a WEB STARTBROWSE in progress.
- 11 The client code page cannot be found.
- 12 The host code page cannot be found.
- 13 No forms data has been supplied in the body of the HTTP request.
- 14 The code page combination for client and server is invalid.
- 17 Invalid forms data was found in the input message.

#

LENGERR

occurs for the following conditions. RESP2 values are:

- 1 NAMELENGTH or VALUELENGTH is less than or equal to zero.

NOTFND

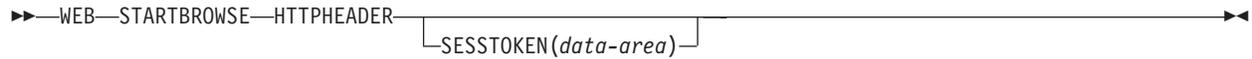
occurs for the following conditions. RESP2 values are:

- 1 The form field name given in the FORMFIELD parameter could not be found.

WEB STARTBROWSE HTTPHEADER

Signal start of HTTP header browse.

WEB STARTBROWSE HTTPHEADER



Conditions:ILLOGIC, INVREQ, NOTOPEN

This command is threadsafe.

Description

WEB STARTBROWSE HTTPHEADER signals the start of a browse of the HTTP header information. The SESSTOKEN option is required if the HTTP header information is part of a response sent to CICS as an HTTP client.

Options

SESSTOKEN(*data-value*)

For CICS as an HTTP client, this option is required. It specifies the session token, an 8-byte binary value that uniquely identifies a connection between CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

Conditions

ILLOGIC

RESP2 value is:

- 10** An HTTP header browse is already in progress.

INVREQ

RESP2 values are:

- 1** The command is being issued in a non-CICS Web support application.
- 3** The command is being issued for a non-HTTP request.
- 43** No HTTP headers found.

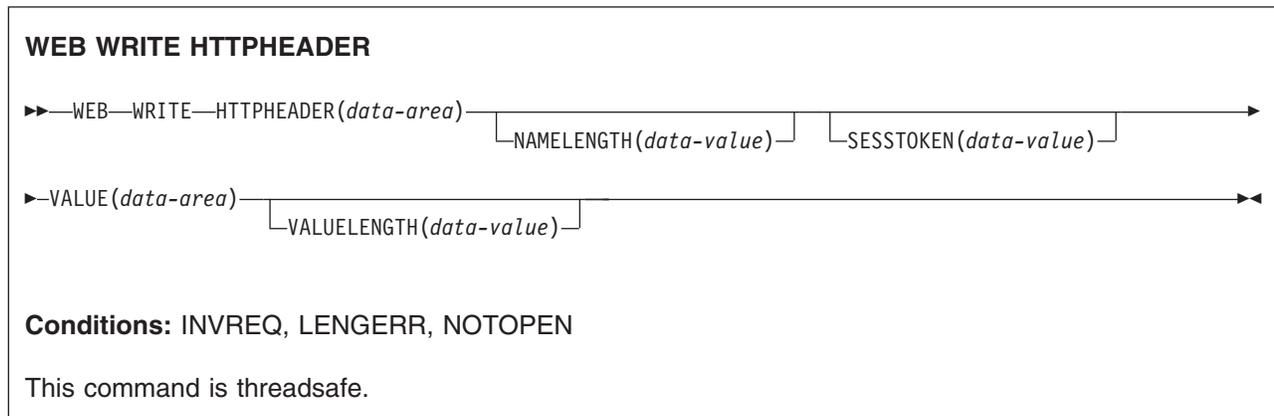
NOTOPEN

RESP2 value is:

- 27** Invalid session token.

WEB WRITE HTTPHEADER

Build HTTP header information.



Description

WEB WRITE HTTPHEADER enables an application to add HTTP header information to a message. When CICS is an HTTP server, the message is a response to a Web client. When CICS is an HTTP client, the message is a request to a server, and the SESSTOKEN option is specified.

Some HTTP headers are created automatically by CICS if the message requires them, and the application does not need to write these headers. These are:

- Connection
- Content-Length
- Content-Type (written by CICS, but can be supplied by a client application if a complex header is required)
- Date
- Expect
- Host
- Server
- TE (written by CICS, but the application can add further instances)
- Transfer-Encoding
- User-Agent
- WWW-Authenticate

"HTTP header reference for CICS Web support" in the *CICS Internet Guide* describes the circumstances in which these headers are created. If the user application program writes a header that CICS also generates, CICS handles this depending on the situation:

- For CICS as an HTTP server, if the header is appropriate for a response, CICS does not overwrite it, but allows the application's version to be used.
- For CICS as an HTTP client, if the header is appropriate for a request, CICS does not allow the application to write it, and returns an error response to the WEB WRITE HTTPHEADER command. The exceptions are the TE header and

the Content-Type header. Application programs can add further instances of the
TE header. They can also supply the Content-Type header, if the required header
needs to contain spaces or more than 56 characters, and so cannot be specified
on the MEDIATYPE option of the WEB SEND command.

- If the header is not normally appropriate for the type of message (request or response), CICS allows it, as is the case for all user-defined headers. This situation should not occur if your message is compliant with the HTTP specification to which you are working.

The WEB WRITE HTTPHEADER command adds a single header, and you can repeat the command to add further headers. If you write a header that you have already written for the request or response, CICS adds the new header to the request or response in addition to the existing header.

The name and value of the headers you write, and the circumstances in which you choose to write them, should conform to the requirements of the HTTP specification to which you are working.

If any of the HTTP headers you use might be unsuitable for a Web client or remote server below HTTP/1.1 level, check the version information that they supply before writing those headers. For CICS as an HTTP server, the WEB EXTRACT command with the HTTPVERSION option gives this information about the Web client. For CICS as an HTTP client, the options HTTPVNUM and HTTPRNUM that are returned on the WEB OPEN command for the session give this information about the server.

The WEB WRITE HTTPHEADER command cannot be used if the connection with the server or Web client has been closed by either party sending a Connection: close header on a request or response.

For guidance on the correct use of this command:

- When writing headers for an HTTP response sent by CICS as an HTTP server, see "Writing Web-aware application programs for CICS as an HTTP server" in the *CICS Internet Guide*.
- When writing headers for an HTTP request sent by CICS as an HTTP client, see "HTTP client requests from a CICS application" in the *CICS Internet Guide*.
- When using chunked transfer-coding to send an HTTP request or response, see "Using chunked transfer-coding to send an HTTP request or response" in the *CICS Internet Guide*. That topic explains the correct procedure for writing trailing headers for a chunked message.

Options

HTTPHEADER(*data-area*)

specifies the name of the HTTP header to be added to the request or response. The name, which is a string of text, should conform to the standards in the HTTP specification to which you are working.

NAMELENGTH(*data-value*)

specifies the length, as a fullword binary value, of the HTTP header name.

SESSTOKEN(*data-value*)

For CICS as an HTTP client, this option is required. It specifies the session token, an 8-byte binary value that uniquely identifies a connection between

CICS and a server. This value is returned by a WEB OPEN command for CICS as an HTTP client. "Session tokens" in the *CICS Internet Guide* explains the use of the session token.

VALUE (*data-area*)

specifies the value of the named HTTP header. The value, which is a string of text, should conform to the standards in the HTTP specification to which you are working.

VALUELENGTH (*data-value*)

specifies the length, as a fullword binary value, of the HTTP header value.

Conditions

INVREQ

RESP2 values are:

- 1** The command is being issued in a non-CICS Web support application.
- 6** Client did not send TE: trailers on request, so trailing headers cannot be used.
- 19** Header not allowed. Some request headers may only be generated by CICS.
- 44** Header not allowed as a trailing header (trailer).
- 69** Chunked transfer-coding not supported.
- 70** Trailer header has not been created, so trailing headers cannot be written.
- 71** Chunked transfer-coding error.
- 74** Previous send failed.
- 78** Too late to write trailing headers for this message.

LENGERR

RESP2 values are:

- 35** The length in NAMELENGTH is not greater than zero.
- 55** The length in VALUELENGTH is not greater than zero.

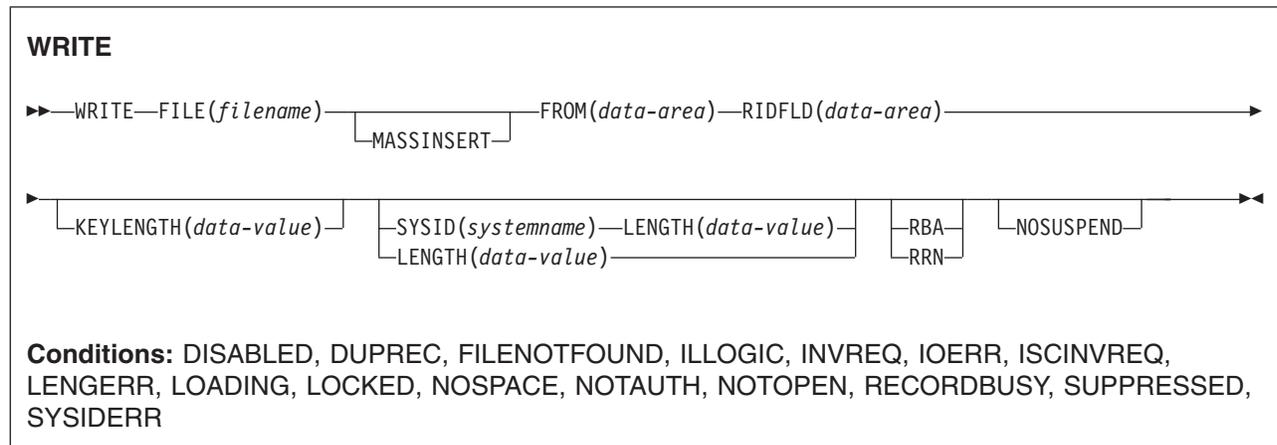
NOTOPEN

RESP2 values are:

- 27** Invalid session token.

WRITE

Write a record.



Description

WRITE writes a new record to a file on a local or a remote system.

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 (unless rejected by the XDTAD user exit).

When this command is used to write a record to a coupling facility data table, the update is made to the data table in the coupling facility (unless it is rejected by the XDTAD user exit).

For a VSAM ESDS, the record is always added at the end of the data set. CICS 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.

For a VSAM ESDS or KSDS, records can be either fixed-length or variable-length. MASSINSERT operations must proceed with ascending keys, and must be terminated by an UNLOCK before any other request to the same data set.

Note: The only VSAM data sets greater than 4GB (that is, defined with the extended format / extended addressability attribute) supported by CICS are

KSDS, and then only if they are accessed by key. CICS does not support ESDS or RRDS data sets defined with the extended attribute.

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.

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 data area where the record is written from.

This option must be specified if SYSID is specified.

If the file is on a remote system, the LENGTH parameter need not be set here but must be set in the file resource definition.

If the file is on a local system, the LENGTH parameter must be set for variable-length records, using the INTO option, but not for fixed-length records. It is, however, advisable to specify the length for fixed-length records because it causes a check to be made that the record being written is not longer than that defined for the data set.

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 or coupling facility data tables.

NOSUSPEND (RLS on1y)

The request does not wait if VSAM is holding an active lock against the record, including records locked as the result of a DEADLOCK.

A task could wait when it issues a WRITE request if the key is for a record that is being modified, created, or deleted by another task, because VSAM always acquires the lock first.

RBA

(VSAM ESDS base data sets only) 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 data sets that are greater than 4GB in size.

RIDFLD(*data-area*)

specifies the record identification 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 to be written. If RRN is specified, RIDFLD contains the relative record number (greater than or equal to 1) of the record to be written.

See 'Identifying BDAM records' and 'VSAM record identification' in the *CICS Application Programming Guide* for more information about defining the record identification field.

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 C.

Conditions

DISABLED

RESP2 values:

- 50** A file was initially defined as disabled and has not since been enabled, or was disabled by a SET FILE or a CEMT SET FILE command.

Default action: terminate the task abnormally.

DUPREC

RESP2 values:

- 150** 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.

This condition is also raised for a coupling facility data table that uses the contention model, even if another task has read the record with the same key for update. (For a coupling facility data table that uses the locking model, and for all other kinds of files, if another task has read the record for update, it is locked, and the WRITE request waits for the lock to be released, rather than returning a DUPREC response immediately.)

Default action: terminate the task abnormally.

FILENOTFOUND

RESP2 values:

- 1** A file name referred to in the FILE option cannot be found in the FCT.

Default action: terminate the task abnormally.

ILLOGIC

RESP2 values: (VSAM)

- 110** A VSAM error occurs that does not fall within one of the other CICS response categories.
(See EIBRCODE in the EXEC interface block; refer to “EXEC interface block” on page 745 for details.)

Default action: terminate the task abnormally.

INVREQ

RESP2 values:

- 20** Add operations are not allowed according to the file entry specification in the FCT.
- 23** When writing records containing embedded keys, the key in the record area (FROM option) and the key in RIDFLD do not match.
- 26** The KEYLENGTH option is specified, and the specified length does not equal the length defined for the data set that this file refers to.
- 38** A WRITE with the MASSINSERT option is issued against a BDAM file.
- 40** A BDAM key conversion error occurred.
- 44** The WRITE command does not conform to the format of WRITE for a user-maintained or coupling facility data table (for example, MASSINSERT or RBA is specified).
- 51** A WRITE command specifying the RBA keyword was issued against a KSDS file that is being accessed in RLS mode. RLS mode does not support relative byte address access to KSDS files.
- 55** NOSUSPEND is not allowed because the file is not a VSAM file that is accessed in RLS mode.
- 56** An attempt to update a recoverable coupling facility data table has failed because the current unit of work has already updated 1024 recoverable coupling facility data tables. You cannot update more than 1024 recoverable coupling facility data tables within a unit of work

Default action: terminate the task abnormally.

IOERR

RESP2 values:

- 120** There is an I/O error during the file control operation. 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 “EXEC interface block” on page 745 for details.)

For VSAM files, IOERR normally indicates a hardware error.

For BDAM files, IOERR could mean that you are trying to write to a BDAM track address that is not defined for the data set.

For a coupling facility data table, an IOERR indicates a bad response returned from a coupling facility access.

Default action: terminate the task abnormally.

ISCVREQ

RESP2 values:

- 70** The remote system indicates a failure that does not correspond to a known condition.

Default action: terminate the task abnormally.

#

LENGERR

RESP2 values:

- 12** The length specified for the write operation exceeds the maximum record size; the record is truncated.
- 10** LENGTH is omitted for a WRITE to a file with variable-length records or to a BDAM file with undefined format records.
- 14** An incorrect length is specified for a write operation involving fixed-length records.

LOADING

RESP2 values:

- 104** The request cannot be satisfied because it is issued against a data table that is still being loaded. The condition can be raised for one of the following reasons:
 - The WRITE specifies a record key that has out of range of the records so far loaded into a coupling facility data table. Records can be added while a CFDT is loading only if the specified key is within the range of those records already loaded.

The LOADING response can also be returned for a coupling facility data table that has failed during loading. For more information about what happens if the load for a coupling facility data table fails, see the description of the XDTLC global user exit in the *CICS Customization Guide*.
 - A WRITE is issued to a user-maintained data table that is currently being loaded. A user-maintained data table cannot be modified during loading.

If your application programs encounter the LOADING condition persistently or too frequently, check that this is not caused by conflicting file definitions that reference the same data set.

Default action: terminate the task abnormally.

LOCKED

RESP2 values:

- 106** An attempt has been made to write a record, but a retained lock exists against the key of this record.

Default action: abend the task with code AEX8.

NOSPACE

RESP2 values:

- 100** No space is available on the direct access device for adding records to a data set.
- 102** The maximum number of table entries specified for the user-maintained table or coupling facility data table has already been reached.

This condition can also occur for a recoverable coupling facility data table when the table apparently contains fewer than the maximum number of records allowed if there are uncommitted updates outstanding.
- 103** CICS is unable to get sufficient storage in the CICS address space to create an in-memory table entry for the record being written.

108 There is insufficient space in the coupling facility data table pool to store the record.

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

101 A resource security check has failed on FILE(filename).

Default action: terminate the task abnormally.

NOTOPEN

RESP2 values:

60 NOTOPEN (RESP2 60) is returned for one of the following reasons:

- 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. You can also make CLOSED, UNENABLED the initial state, by specifying STATUS(UNENABLED) and OPENTIME(FIRSTREF) on the FILE resource definition. (For BDAM files, you use the FILSTAT parameter of the DFHFCT TYPE=FILE macro.)
- The requested file is OPEN and in use by other transactions, but a CLOSE request against the file has been received.
- A WRITE request is issued against a data set is quiesced, or is being quiesced, as a result of a SET DSNAME QUIESCED or IMMQUIESCED command.
- The requested file is CLOSED and ENABLED, so CICS has tried to open the file as part of executing the request. This file open has failed for some reason. You should examine the console for messages that explain why the file open has been unsuccessful.

This condition does not occur if the request is made to a CLOSED, DISABLED file. In this case, the DISABLED condition occurs.

Default action: terminate the task abnormally.

RECORDBUSY

RESP2 values:

107 NOSUSPEND is specified on the request but VSAM holds an active lock against the record, which would cause the request to wait (see "Retained and active locks" below).

Default action: abend the task with code AEX9.

SUPPRESSED

RESP2 values:

105 A user exit program that is invoked at the XDTAD exit point decides not to add the record to the user-maintained or coupling facility data table.

Default action: terminate the task abnormally.

SYSIDERR

RESP2 values:

130 The SYSID option specifies a name that is neither the local CICS region nor a remote system defined to CICS by a CONNECTION definition. SYSIDERR also occurs when the link to the remote system is closed.

- 131** For a coupling facility data table, the connection to the coupling facility data table server has failed. This could be because the server itself has failed, or the server is available, but CICS has failed to connect to it.
- 132** The WRITE is issued against a coupling facility data table that no longer exists, probably because of a coupling facility failure, in which case the coupling facility data table server also fails. See the *CICS System Definition Guide* for information about restarting a coupling facility data table server and reloading a table.

Default action: terminate the task abnormally.

Retained and active locks: RECORDBUSY refers to active locks, and LOCKED refers to retained locks:

- READNEXT requests for records that have *retained* locks are always rejected with a LOCKED response.
- READNEXT requests for records that have *active* locks wait for the lock to be released, except when the NOSUSPEND keyword is specified, in which case CICS returns the RECORDBUSY response.

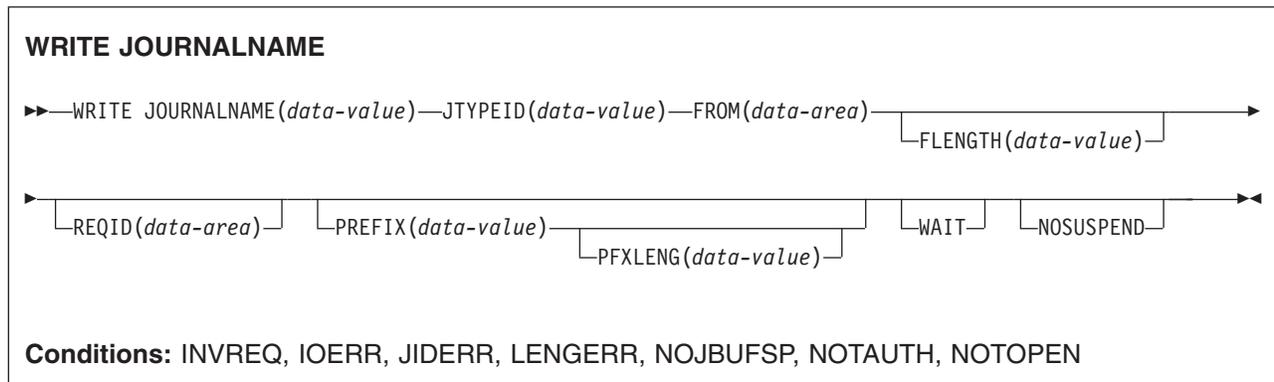
Examples

For example:

```
EXEC CICS WRITE  
      FROM(RECORD)  
      LENGTH(DATLEN)  
      FILE('MASTER')  
      RIDFLD(KEYFLD)
```

WRITE JOURNALNAME

Create a journal record



Description

WRITE JOURNALNAME writes a journal record from the specified data area to the system logger log stream that corresponds to the CICS journal name, or to SMF. The request can be for synchronous or asynchronous output; definitions of these terms, and information regarding the synchronization of journal output, are in the *CICS Application Programming Guide*.

Options

FLENGTH(*data-value*)

specifies, as a full word binary value, the length in bytes of the user data to be built into the journal record.

Note that the maximum total length of a journal record depends on a number of factors:

- There is a limit of 32KB minus 400 bytes if the journal is using SMF.
- The limit for journals that map to log streams is the value expressed in the MAXBUFSIZE attribute for the structure being used minus 400 bytes. This has to include the user data, the prefix data, and the 2-byte JTYPEID.

Note: Data longer than 32K bytes cannot be read by offline jobs using the SUBSYS=LOGR interface.

FROM(*data-area*)

specifies the user data to be built into the journal record.

JOURNALNAME(*data-value*)

specifies a 1- to 8-character journal name. The valid characters for a journal name are the upper-case letters A through Z, the numeric characters 0 through 9, and the special symbols \$ @ and #.

On first reference to this journal name, CICS must be able to map the journal name to a corresponding MVS system loggerlog stream, or MVS SMF data set. To do this, CICS searches the installed JOURNALMODEL definitions, looking

for a matching journal name in a journal model. CICS looks for either a specific match or a generic match. If a matching entry cannot be found, CICS attempts to use a default log stream name.

To write to the CICS system log, specify DFHLOG as the journal name.

Note: The CICS system log should be used only for short-lived data required for recovery purposes. You should not write user records for such things as audit trails to it.

To write to journals defined using the journal numbering convention (for example, to the auto journals defined in file resource definitions), specify the name as DFHJnn, where nn is the journal number in the range 1 to 99.

You cannot write to a forward recovery log that is known to CICS only by its 26-character log stream name (as derived directly from the VSAM ICF catalog) unless you write to a journal whose matching JOURNALMODEL is associated with the same log stream name.

Specifying DFHJ01 on this command refers to a user journal, *not* the system log.

JTYPEID(*data-value*)

specifies a 2-character identifier to be placed in the journal record to identify its origin.

NOSUSPEND

specifies that the application program is not to be suspended for the NOJBUFSP condition. The user record is ignored.

PFXLENG(*data-value*)

specifies the length (halfword binary value) in bytes of the user prefix data to be included in the journal record.

Note that the maximum total length of a journal record depends on a number of factors:

- There is a limit of 32KB minus 400 bytes if the journal is using SMF.
- The limit for journals that map to log streams is the value expressed in the MAXBUFSIZE attribute for the structure being used minus 400 bytes. This has to include the prefix data, the user data, and the 2-byte JTYPEID.

The minimum value is 0. See FLENGTH for the limits to the size of a journal record.

Note: Data longer than 32K bytes cannot be read by offline jobs using the SUBSYS=LOGR interface.

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 token that can be used for synchronization. REQID is only valid for asynchronous output (that is, the WAIT option is not specified).

WAIT

specifies that synchronous journal output is required. The requesting task waits until the record has been hardened.

Conditions

INVREQ

the command is not valid for processing by CICS.

Default action: Terminate the task abnormally.

IOERR

a journal record has not been output because an irrecoverable error condition was returned by the system logger log stream or by SMF.

Default action: If the log is the system log, CICS either quiesces or abends CICS. If the log is a general log, the task is terminated abnormally.

JIDERR

CICS cannot connect to a log stream referenced by the specified journal name, for one of the following reasons:

- The log stream does not exist and cannot be created dynamically using default model definitions.
- The log stream is a DASD-only log stream to which a CICS region in another MVS image is currently connected.

Default action: terminate the task abnormally.

LENGERR

the aggregate length of the journal record, comprising the user data (FROM, JTYPE, and PREFIX data) and the CICS header data, is too great to fit the maximum block size allowed for the log stream.

Default action: terminate the task abnormally.

NOJBUFSP

the journal buffers are logically full (that is, the current buffer has insufficient space for this journal record, and I/O is in progress on the alternate buffer).

Default action: CICS suspends 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. (You can override the default action by the NOSUSPEND option.)

NOTAUTH

a resource security check has failed on the JOURNALNAME(data-value) resource.

Default action: terminate the task abnormally.

NOTOPEN

occurs in any of the following situations:

- The command cannot be executed because the specified journal has been explicitly disabled by the user.
- The request could not be satisfied because the specified journal was defined using a journal model that maps it onto the logstream that is being used as the system log for this CICS system. The error is detected when trying to connect to the logstream and results in a definition for the JOURNALNAME being installed and set to 'failed'.

Default action: terminate the task abnormally.

Examples

The following example shows how to write synchronous journal output and wait for the output operation to be completed:

```
EXEC CICS WRITE
      JOURNALNAME('ACCTSJNL')
      JTYPEID('XX')
      FROM(KEYDATA)
      FLENGTH(40000)
      PREFIX(PROGNAME)
      PFXLENG(6)
      WAIT
```

The following example shows how to write deferred (asynchronous) user recovery data to the CICS system log:

```
EXEC CICS WRITE
      JOURNALNAME('DFHLOG')
      JTYPEID('UR')
      FROM(COMDATA)
      FLENGTH(10)
      REQID(ENTRYID)
```

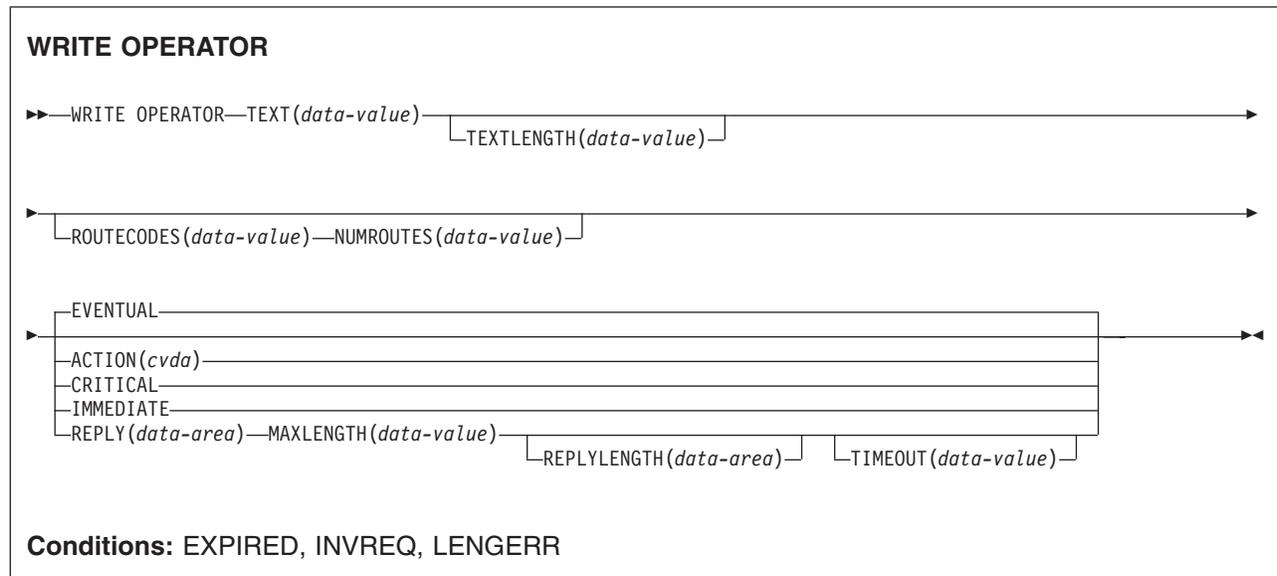
WRITE JOURNALNUM

Create a journal record.

This command is supported for compatibility with earlier releases of CICS. It is superseded by the WRITE JOURNALNAME command, which you are recommended to use instead.

WRITE OPERATOR

Write a message on the system console.



Description

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 until the console operator explicitly deletes it or CICS terminates.

The action code is identical with the 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)**.

Retained messages 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.

Options

ACTION(*cvda*)

specifies an action code to be associated with this message. 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.

If the data value begins with DFHnnnn or DFHaannnn, the message is treated as a CICS message and is reformatted accordingly.

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 the COBOL2 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.

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 no room to reformat the data 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 by the OPERTIM system initialization parameter. You can only specify TIMEOUT if you have also specified REPLY.

Conditions**EXPIRED**

RESP2 values:

7 TIMEOUT has occurred before the operator's reply was received.

Default action: return the exception condition to the application.

INVREQ

RESP2 values:

1 The TEXTLENGTH value is not valid.

2 The NUMROUTES value is not valid.

3 The ROUTECODES value is not valid.

4 The MAXLENGTH value is not valid.

5 The TIMEOUT value is not valid.

6 The ACTION value is not valid.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

8 The reply was longer than MAXLENGTH, and has been truncated.

Default action: terminate the task abnormally.

WRITEQ TD

Write data to transient data queue.

WRITEQ TD

▶▶—WRITEQ TD—QUEUE(*name*)—FROM(*data-area*)—┬──LENGTH(*data-value*)──┬──SYSID(*systemname*)──┬──▶▶

Conditions: DISABLED, INVREQ, IOERR, ISCINVREQ, LENGERR, LOCKED, NOSPACE, NOTAUTH, NOTOPEN, QIDERR, SYSIDERR

Description

WRITEQ TD writes transient data to a predefined symbolic destination.

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 named queue must have been defined to CICS.

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 queue is assumed to be on a remote system whether or not it is defined as remote. Otherwise the transient data queue definition is used to find out whether the data set is on a local or a remote system.

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 RECORDSIZE and associated formations specified in the TDQUEUE resource definition. 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.

LOCKED

occurs when the request cannot be performed because use of the queue has been restricted owing to a unit of work failing in-doubt. This can happen on any request for a logically-recoverable queue defined with WAIT(YES) and WAITACTION(REJECT) in the TDQUEUE resource definition.

Specify WAIT(YES) and WAITACTION(Queue) in the TDQUEUE resource definition if you want the transaction to wait.

Default action: terminate the task abnormally.

NOSPACE

occurs if no more space exists on the intrapartition or extrapartition queue, or the relative byte address (RBA) for an intrapartition queue would exceed 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.

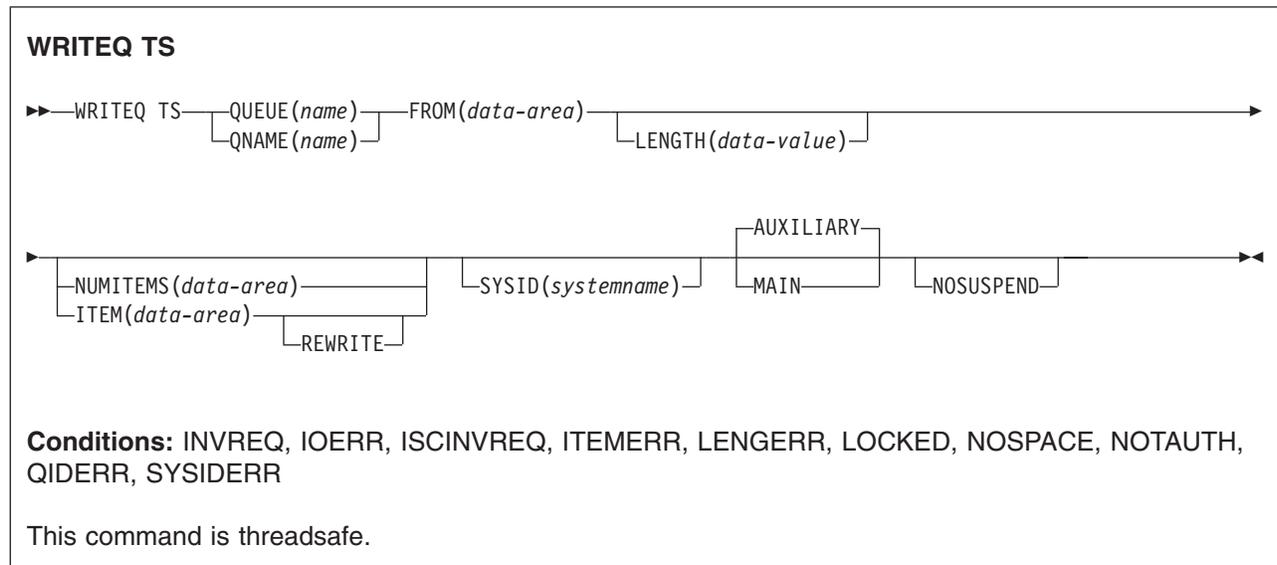
Examples

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 TS

Write data to a temporary storage queue.



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 Application Programming Guide* for more information about transaction affinities.

Description

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 or main storage 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.) If, however, space is not available in the temporary storage data set or main storage, and the NOSUSPEND option has been specified, the NOSPACE condition is raised.

Options

AUXILIARY

specifies that the temporary storage queue is on a direct access storage device in auxiliary storage. This is the default value for the first write.

This option is ignored:

- for an existing queue,
- if a TSMODEL resource definition with a matching prefix is installed in the system.

- if the AUXILIARY option is specified for a temporary storage data queue that resides in a temporary storage pool.

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).

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 (for example), 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 in CICS commands” on page 8.

MAIN

specifies that the temporary storage queue is in main storage.

This option is ignored:

- for an existing queue,
- if a TSMODEL resource definition with a matching prefix is installed in the system.
- if the MAIN option is specified for a temporary storage data sharing queue that resides in a temporary storage pool.

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 or in main storage to satisfy the WRITEQ TS request. Instead, the NOSPACE condition is raised.

Note, however, that if a HANDLE CONDITION for NOSPACE is active when the command is executed, this also overrides the default action, and control is passed to the user label supplied in the HANDLE CONDITION. This takes precedence over the NOSUSPEND option but is, of course, negated by either NOHANDLE or RESP.

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 name has less than 8 characters, you must still use an 8-character field, padded with blanks if necessary. 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.

QNAME (*name*)

an alternative to QUEUE, QNAME specifies the symbolic name (1–16 characters) of the queue to be written to. If the name has less than 16 characters, you must still use a 16-character field, padded with blanks if necessary. If the queue name appears in the TST, and the entry is marked as remote, or if the QNAME is described by a TSMODEL resource definition which identifies a remote system, 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 and shared queues only) specifies the system name (1–4 characters) identifying the remote system or shared queue pool to which the request is directed.

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

RESP2 values:

- 5** There is an irrecoverable input/output error for a shared 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.

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.

Default action: terminate the task abnormally.

LOCKED

RESP2 values:

- 0** The request cannot be performed because use of the queue has been restricted owing to a unit of work failing in-doubt.

Default action: terminate the task abnormally.

NOSPACE

occurs when the NOSUSPEND option is specified and there is no space for the data in:

- The auxiliary temporary storage data set
- The temporary storage pool list structure

Also occurs if there is no space and there is an active HANDLE CONDITION for NOSPACE.

Default action: ignore the condition.

NOTAUTH

RESP2 values:

- 101** 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 storage
- Auxiliary storage
- Temporary storage pool

Default action: terminate the task abnormally.

SYSIDERR

RESP2 values:

- 4** The CICS region in which the temporary storage command is executed fails to connect to the TS server managing the TS pool that supports the referenced temporary storage queue. (For example, this can happen if the CICS region is not authorized to access the temporary storage server).

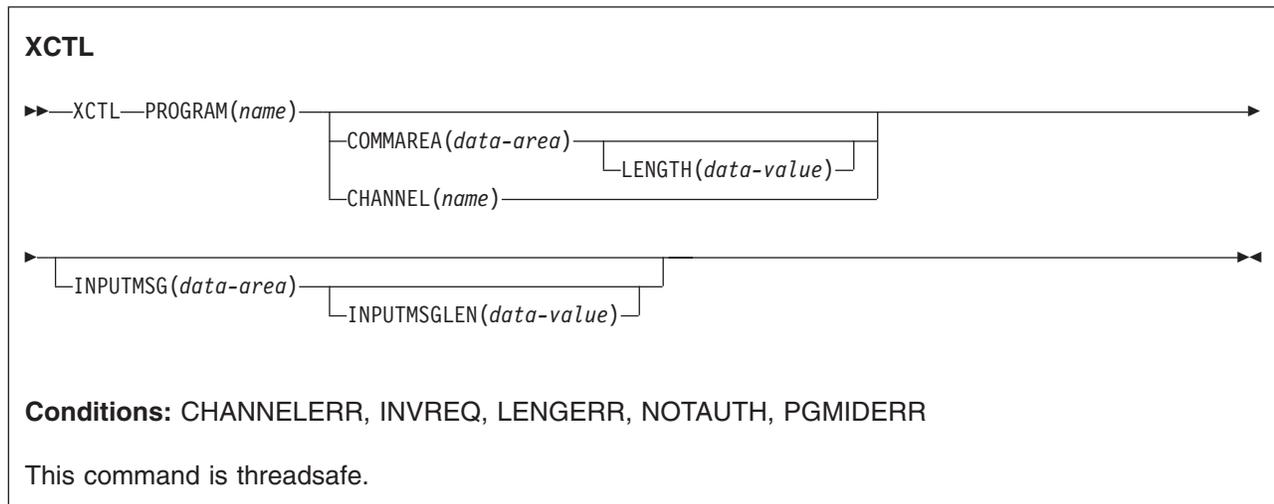
SYSIDERR can also occur if the temporary storage server has not been started, or because the server has failed (or been stopped) while CICS continues executing. Also occurs in any of the following situations:

- 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).
- When the link to the remote system is closed.

Default action: terminate the task abnormally.

XCTL

Transfer program control.



Description

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.

Options

CHANNEL(*name*)

specifies the name (1–16 characters) of a channel that is to be made available to the invoked program. The acceptable characters are A-Z a-z 0-9 \$ @ # / % & ? ! : | " = ~ , ; < > . - and _ . Leading and embedded blank characters are not permitted. If the name supplied is less than 16 characters, it is padded with trailing blanks up to 16 characters.

Channel names are always in EBCDIC. The allowable set of characters for channel names, listed above, includes some characters that do not have the same representation in all EBCDIC code pages. We therefore recommend that, if channels are to be shipped between regions, the characters used in naming them should be restricted to A-Z a-z 0-9 & : = , ; < > . - and _ .

The program that issues the XCTL command may:

- Have created the channel by means of one or more PUT CONTAINER CHANNEL commands
- Specify its current channel, by name
- Name a non-existent channel, in which case a new, empty, channel is created

COMMAREA(*data-area*)

specifies a communication area to be made available to the invoked program. In this option the contents of the data-area are 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 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 in CICS commands” on page 8.

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.

Conditions**CHANNELERR**

RESP2 values:

- 1 The name specified on the CHANNEL option contains an illegal character or combination of characters.

INVREQ

RESP2 values:

- 8 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.
- 29 EXEC XCTL is not allowed in a GLUE or TRUE.
- 30 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.
- 50 The Language Environment options specified in DFHJVMRO are too long.
- 200 An XCTL command with the INPUTMSG option is issued in a program invoked by DPL.

Default action: terminate the task abnormally.

LENGERR

RESP2 values:

- 11 LENGTH is less than 0 or greater than 32763.
- 26 The COMMAREA address passed was zero, but LENGTH was non-zero.
- 27 INPUTMSGLEN was less than 0 or greater than 32767.
- 28 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.

#

Default action: terminate the task abnormally.

NOTAUTH

RESP2 values:

101 A resource security check has failed on PROGRAM(name).

Default action: terminate the task abnormally.

PGMIDERR

RESP2 values:

1 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.

2 The program is disabled.

3 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

9 The installed program definition is for a remote program.

21 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.

22 The model returned by the program autoinstall control program was not defined in the PPT table, or was not enabled.

23 The program autoinstall control program returned invalid data.

24 Define for the program failed due to autoinstall returning an invalid program name or definition.

Default action: terminate the task abnormally.

Examples

The following example shows how to request a transfer of control to an application program called PROG2:

```
EXEC CICS XCTL PROGRAM('PROG2')
```

Appendix. Detailed reference information for the CICS API commands

This section contains detailed reference information which applies to more than one of the CICS API commands.

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 change the contents of any of them other than through an EXEC CICS command.

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.

At midnight, if EIBTIME has the value of 0240000+, the value of EIBDATE is
the day that has just ended. If EIBTIME has the value of 0000000+, the value
of EIBDATE is the day that is just beginning.

```
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

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 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 HANDLE
020E	POP HANDLE
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
 1008 START ATTACH
 1008 START BREXIT
 100A RETRIEVE
 100C CANCEL

 1202 WAIT EVENT
 1204 ENQ
 1206 DEQ
 1208 SUSPEND

 1402 WRITE JOURNALNUM
 1404 WAIT JOURNALNUM
 1406 WRITE JOURNALNAME
 1408 WAIT JOURNALNAME

 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

2004 DEFINE COUNTER
2006 GET COUNTER
2008 UPDATE COUNTER
200A DELETE COUNTER
200C REWIND COUNTER
200E QUERY COUNTER
2014 DEFINE DCOUNTER
2016 GET DCOUNTER
2018 UPDATE DCOUNTER
201A DELETE DCOUNTER
201C REWIND DCOUNTER
201E QUERY DCOUNTER

3402 DEFINE ACTIVITY
3404 DEFINE PROCESS
3406 RUN ACTIVITY
3408 RUN ACQPROCESS
340E ACQUIRE PROCESS
3410 ACQUIRE ACTIVITYID
3412 DELETE CONTAINER
3414 GET CONTAINER
3416 PUT CONTAINER
3418 RESET ACTIVITY
341A CHECK ACTIVITY
341C CANCEL ACTIVITY
341E CANCEL ACQPROCESS
3420 SUSPEND ACTIVITY
3422 SUSPEND ACQPROCESS
3424 RESUME ACTIVITY
3426 RESUME ACQPROCESS
3428 DELETE ACTIVITY
342A LINK ACQPROCESS
342C LINK ACTIVITY
342E CANCEL ACQACTIVITY
3430 RUN ACQACTIVITY
3432 LINK ACQACTIVITY
3434 SUSPEND ACQACTIVITY
3436 RESUME ACQACTIVITY
3438 CHECK ACQPROCESS
343A CHECK ACQACTIVITY
343C RESET ACQPROCESS

3602 DEFINE INPUT EVENT
3602 DEFINE COMPOSITE EVENT
3604 DELETE EVENT
3608 ADD SUBEVENT
360A REMOVE SUBEVENT
360E TEST EVENT
3610 RETRIEVE REATTACH EVENT

3612 RETRIEVE SUBEVENT
 3614 DEFINE TIMER
 3616 DELETE TIMER
 3618 CHECK TIMER
 361A FORCE TIMER

3802 WEB RECEIVE
 3804 WEB SEND
 3806 WEB READ
 3808 WEB STARTBROWSE
 380A WEB READNEXT
 380C WEB ENDBROWSE
 380E WEB WRITE HTTPHEADER
 3810 WEB EXTRACT
 3814 WEB RETRIEVE
 3816 WEB PARSE URL
 3818 WEB OPEN
 381A WEB CLOSE
 381C WEB CONVERSE

3C02 DOCUMENT CREATE
 3C04 DOCUMENT INSERT
 3C06 DOCUMENT RETRIEVE
 3C08 DOCUMENT SET

3E0E EXTRACT TCPIP
 3E10 EXTRACT CERTIFICATE

4802 ENTER TRACENUM
 4804 MONITOR

4A02 ASKTIME ABSTIME
 4A04 FORMATTIME
 4A06 CONVERTTIME

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

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.

EIBFN	EIBRCODE	Condition
02 ..	E0	INVREQ
04 ..	04	EOF

EIBFN	EIBRCODE	Condition
04 ..	10	EODS
04 ..	C1	EOF
04 ..	C2	ENDINPT
04 ..	D0	SYSIDERR (see note 1)
04 ..	D2	SESSIONERR (see note 2)
04 ..	D3	SYSBUSY (see note 3)
04 ..	D4	SESSBUSY
04 ..	D5	NOTALLOC
04 ..	E0	INVREQ (see note 4)
04 ..	E1	LENGERR (see note 5)
04 ..	E3	WRBRK
04 ..	E4	RDATT
04 ..	E5	SIGNAL
04 ..	E6	TERMIDERR
04 ..	E7	NOPASSBKRD
04 ..	E8	NOPASSBKWR
04 ..	EA	IGREQCD
04 ..	EB	CBIDERR
04 ..	EC	PARTNERIDERR
04 ..	ED	NETNAMEIDERR
04 ..	F1	TERMERR
04 20	EOC
04 40	INBFMH
04 F6	NOSTART
04 F7	NONVAL
06 ..	01	FILENOTFOUND
06 ..	02	ILLOGIC (see note 6)
06 ..	03	LOCKED
06 ..	05	RECORDBUSY
06 ..	08	INVREQ
06 ..	0C	NOTOPEN
06 ..	0D	DISABLED
06 ..	0F	ENDFILE
06 ..	80	IOERR (see note 6)
06 ..	81	NOTFND
06 ..	82	DUPREC
06 ..	83	NOSPACE
06 ..	84	DUPKEY
06 ..	85	SUPPRESSED
06 ..	86	LOADING
06 ..	D0	SYSIDERR (see note 1)
06 ..	D1	ISINVREQ
06 ..	D6	NOTAUTH
06 ..	E1	LENGERR
08 ..	01	QZERO
08 ..	02	QIDERR
08 ..	04	IOERR
08 ..	08	NOTOPEN
08 ..	10	NOSPACE
08 ..	C0	QBUSY
08 ..	D0	SYSIDERR (see note 1)
08 ..	D1	ISINVREQ

EIBFN	EIBRCODE	Condition
08 ..	D6	NOTAUTH
08 ..	D7	DISABLED
08 ..	E0	INVREQ
08 ..	E1	LENGERR
0A ..	01	ITEMERR
0A ..	02	QIDERR
0A ..	04	IOERR
0A ..	08	NOSPACE
0A ..	20	INVREQ
0A ..	D0	SYSDERR (see note 1)
0A ..	D1	ISCINVREQ
0A ..	D6	NOTAUTH
0A ..	E1	LENGERR
0C ..	E1	LENGERR
0C ..	E2	NOSTG
0E ..	01	PGMIDERR
0E ..	D6	NOTAUTH
0E ..	D9	RESUNAVAIL
0E ..	DA	CHANNELERR
0E ..	E0	INVREQ
0E ..	E1	LENGERR
0E ..	F1	TERMERR
10 ..	01	ENDDATA
10 ..	04	IOERR
10 ..	11	TRANSIDERR
10 ..	12	TERMIDERR
10 ..	20	EXPIRED
10 ..	81	NOTFND
10 ..	D0	SYSDERR (see note 1)
10 ..	D1	ISCINVREQ
10 ..	D6	NOTAUTH
10 ..	D8	USERIDERR
10 ..	D9	RESUNAVAIL
10 ..	DA	CHANNELERR
10 ..	E1	LENGERR
10 ..	E9	ENVDEFERR
10 ..	FF	INVREQ
12 ..	32	ENQBUSY
12 ..	E0	INVREQ
12 ..	E1	LENGERR
14 ..	01	JIDERR
14 ..	02	INVREQ
14 ..	05	NOTOPEN
14 ..	06	LENGERR
14 ..	07	IOERR
14 ..	09	NOJBUFSP
14 ..	D6	NOTAUTH

EIBFN	EIBRCODE	Condition
16 ..	01	ROLLEDBACK
18 ..	01	INVREQ
18 ..	02	RETPAGE
18 ..	04	MAPFAIL
18 ..	08	INVMPSZ (see note 7)
18 ..	20	INVERRTERM
18 ..	40	RTESOME
18 ..	80	RTEFAIL
18 ..	E1	LENGERR
18 ..	E3	WRBRK
18 ..	E4	RDATT
18 02	PARTNFAIL
18 04	INVPARTN
18 08	INVPARTNSET
18 10	INVLDC
18 20	UNEXPIN
18 40	IGREQCD
18 80	TSIOERR
18 01	OVERFLOW
18 04	EODS
18 08	EOC
18 10	IGREQID
1A ..	E0	INVREQ
1A ..	04	DSSTAT
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	SPOLERR
56 5A	NODEIDERR

Note:

1. When SYSIDERR occurs, further information is provided in bytes 1 and 2 of EIBRCODE, as shown in Figure 6.

.. 04 00	request was for a function that is not valid.
.. 04 04	no session available and NOQUEUE.
.. 04 08	modename not found (for APPC only).
.. 04 0C	modename not valid (for APPC only).
.. 04 10	task canceled or timed out during allocation (for APPC only).
.. 04 14	mode group is out of service (for APPC only).
.. 04 18	close - DRAIN=ALL (for APPC only).
.. 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.

Figure 6. Bytes 1 and 2 of EIBRCODE for SYSIDERR

Further information about SYSIDERR can be found in the *CICS Intercommunication Guide*.

2. When SESSIONERR occurs, further information is provided in bytes 1 and 2 of EIBRCODE, as shown in Figure 7.

.. 08	session out of service
.. 0C xx	session definition error
.. 0C 00	name not found
.. 0C 0C	profile not found.

Figure 7. Bytes 1 and 2 of EIBRCODE for SESSIONERR

Further information about SESSIONERR can be found in the *CICS Intercommunication Guide*.

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 as shown in Figure 8 on page 757.

.. .. . 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).

Figure 8. Byte 3 of EIBRCODE for SYSBUSY

Further information about SYSBUSY can be found in the *CICS Intercommunication Guide*.

4. When INVREQ occurs during terminal control operations, further information is provided in bytes 1 or 3 of EIBRCODE as shown in Figure 9.

.. 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.
.. .. . 2C	Command invalid in DPL environment.

Figure 9. Bytes 1 or 3 of EIBRCODE for INVREQ

5. When LENGERR occurs during terminal control operations, further information is provided in byte 1 of EIBRCODE, as shown in Figure 10 on page 758.

```

.. 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.

```

Figure 10. Byte 1 of EIBRCODE for LENGERR

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 shown in Figure 11.
where:

```

.. xx xx xx xx .. BDAM response.
.. xx .. . . . . . VSAM return code.
.. . . xx .. . . . VSAM error code.

```

Figure 11. EIBRCODE for ILLOGIC or IOERR

byte 3 =
VSAM problem determination code (ILLOGIC only)

byte 4 =
VSAM component code (ILLOGIC only)

Details of these response codes are given in the *DFSMS Macro Instructions for Data Sets* manual for VSAM, and the *DFSMS/MVS V1.3 Using Data Sets (SC26-4922)* for BDAM.

7. When INVMPsz occurs during BMS operations, byte 3 of field EIBRCODE contains the terminal code as shown in Figure 12.

```

.. . . . xx .. . . terminal code.

```

Figure 12. Byte 3 of EIBRCODE for INVMPsz

These are the same as the mapset suffixes shown in Table 26 on page 815.

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	58	SESSIONERR
01	ERROR	59	SYSBUSY
02	RDATT	60	SESSBUSY
03	WRBRK	61	NOTALLOC
04	EOF	62	CBIDERR
05	EODS	63	INVEXITREQ
06	EOC	64	INVPARTNSET
07	INBFMH	65	INVPARTN
08	ENDINPT	66	PARTNFAIL
09	NONVAL	69	USERIDERR
10	NOSTART	70	NOTAUTH
11	TERMIDERR	72	SUPPRESSED
12	FILENOTFOUND	80	NOSPOOL
13	NOTFND	81	TERMERR
14	DUPREC	82	ROLLEDBACK
15	DUPKEY	83	END
16	INVREQ	84	DISABLED
17	IOERR	85	ALLOCERR
18	NOSPACE	86	STRELERR
19	NOTOPEN	87	OPENERR
20	ENDFILE	88	SPOLBUSY
21	ILLOGIC	89	SPOLERR
22	LENGERR	90	NODEIDERR
23	QZERO	91	TASKIDERR
24	SIGNAL	92	TCIDERR
25	QBUSY	93	DSNNOTFOUND
26	ITEMERR	94	LOADING
27	PGMIDERR	95	MODELIDERR
28	TRANSIDERR	96	OUTDESCRERR
29	ENDDATA	97	PARTNERIDERR
31	EXPIRED	98	PROFILEIDERR
32	RETPAGE	99	NETNAMEIDERR

#

	No. Condition	No. Condition
	33 RTEFAIL	100 LOCKED
	34 RTESOME	101 RECORDBUSY
#	35 TSIOERR	102 UOWNOTFOUND
	36 MAPFAIL	103 UOWLNOTFOUND
	37 INVERRTERM	104 LINKABEND
	38 INVMPSTZ	105 CHANGED
	39 IREQID	106 PROCESSBUSY
	40 OVERFLOW	107 ACTIVITYBUSY
	41 INVLDC	108 PROCESSERR
	42 NOSTG	109 ACTIVITYERR
	43 JIDERR	110 CONTAINERERR
	44 QIDERR	111 EVENTERR
	45 NOJBUFSP	112 TOKENERR
	46 DSSTAT	113 NOTFINISHED
	47 SELNERR	114 POOLERR
	48 FUNCERR	115 TIMERERR
	49 UNEXPIN	116 SYMBOLERR
	50 NOPASSBKRD	117 TEMPLATERR
	51 NOPASSBKWR	121 RESUNAVAIL
	53 SYSIDERR	122 CHANNELERR
	54 ISCINVREQ	123 CCSIDERR
#	55 ENQBUSY	124 TIMEDOUT
	56 ENVDEFERR	
	57 IREQCD	

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. If a program uses DPL to link to a program in another CICS region, an EIBRESP2 from the remote region is not returned to the program doing the DPL.

For programs written in C or C++, any value passed via the *exit* or *return* function is saved in EIBRESP2. This means that when DPL is used to link to a C or C++ program in a remote region, this value is not returned to the linking program.

```
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 eibrldb;
PL/I:   CHAR(1)
ASM:    CL1
```

EIBRSRCE

contains the symbolic identifier of the resource being accessed by the latest executed command as shown in Table 16

Table 16. Symbolic identifier of resource being accessed

Command type	Resource	Length
BMS	Map name	7
File control	File name	8
Interval control	Transaction name	4
Journal control	Journal number	H
Journal control	Journal name	8
Program control	Program name	8
Temporary storage control	TS queue name	8 or 16
Terminal control	Terminal name; LU name; LU6.1 session or APPC convid	4
Transient data control	TD queue name	4

Note:

1. H= halfword binary.
2. Identifiers less than eight characters in length are padded on the right with blanks.
3. Identifiers greater than eight characters in length will be truncated.

```
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

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
```

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 RDO TYPETERM resource definition. The second byte of the data area contains a model number in character form, as set by the TERMMODEL attribute of the TYPETERM resource definition.. TYPETERM is described in the *CICS Resource Definition Guide*.

The codes are listed here as both bit patterns and hexadecimal values.

Code		Meaning
.... ...1	X'01'	7770 System 7
.... ...1.	X'02'	Console
.... 1...	X'08'	Sequential disk
...1 ...1.	X'12'	Magnetic tape
...1 .1..	X'14'	Card reader or line printer
...1 1...	X'18'	Spooling system printer
...1 1..1	X'19'	Spooling internal reader
...1 1.1.	X'1A'	Hard-copy terminals
..1.	X'20'	Model 33/35 TWX
..1. ...1	X'21'	Teletypewriter
..1. ...1.	X'22'	1050
..1. .1..	X'24'	2740
..1. 1...	X'28'	2741 Correspondence
..1. 1.1.	X'2A'	2741 EBCDIC
..1. 1.11	X'2B'	Video terminals
.1..	X'40'	2260 local
.1.. ...1	X'41'	2260 remote
.1.. 1...	X'48'	1053
.1.. 1.1.	X'4A'	2265
.1.. 11..	X'4C'	

Code		Meaning
.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 1..1	X'99'	3277 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
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

Code		Meaning
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
....	X'00'	None
.... ...1	X'01'	Terminal facility indicator
.... ..1.	X'02'	KCP macro file mask
.... .1..	X'04'	Interval control indicator
.... 1...	X'08'	Destination control indicator
...1	X'10'	AID facility mask
111.	X'E0'	reserved

National language codes

Language codes are held as one character for NATLANG and NATLANGINUSE, and three characters for LANGUAGECODE and LANGINUSE.

Table 17. 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 18. 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

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 32 767. See the *CICS 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.

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.

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

CICS TS 3.1 does not support local TCAM terminals. However, it does support transaction routing or function shipping from TCAM terminals attached to a remote, pre-CICS TS 3.1, terminal-owning region, to which the terminals are connected by the DCB (not ACB) interface of TCAM.

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.

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 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 46.

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 3270 logical unit or a 3650 host-conversational (3270) logical unit, it is a printer defined by the RDO TERMINAL options PRINTER and ALTPRINTER, or by a printer supplied by the autoinstall user program.

For a 3270-display logical unit with the printer adapter feature (PRINTADAPTER(YES) option on RDO TYPETERM), 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 3270 logical unit to be eligible, it must have been specified by RDO TERMINAL options PRINTER and ALTPRINTER or by a printer supplied by the

autoinstall user program, and it must have the correct buffer capacity. If the copy feature is also specified (COPY(YES)) on RDO TYPETERM definition 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.

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. (See the *SAA Common Programming Interface-Resource Recovery Reference*, SC31-6821, for more details.)

CICS 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.

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 Application Programming 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*.

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

Table 19. Restricted API commands

ADDRESS	ISSUE
ALLOCATE	PURGE MESSAGE
ASSIGN	RECEIVE
CONNECT PROCESS	ROUTE
CONVERSE	SEND
EXTRACT PROCESS	SIGNOFF
FREE CONVID	SIGNON
HANDLE AID	WAIT TERMINAL

API commands and distributed program link

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.

Table 20. Summary of the CICS API by functional area

<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: These 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

Table 20. Summary of the CICS API by functional area (continued)

Functional area	Command	Supported?
BMS commands	ASSIGN COLOR PAGENUM DESTCOUNT PARTNPAGE INPARTN PARTNS LDCMNEM PARTNSET LDCNUM PS MAPCOLUMN MAPHEIGHT MAPLINE MAPWIDTH OPCLASS PURGE MESSAGE RECEIVE MAP PARTN ROUTE SEND CONTROL MAP PAGE PARTNSET TEXT TEXT MAPPED TEXT NOEDIT	NO
Built-in functions	BIF DEEDIT	YES
Condition handling	HANDLE CONDITION IGNORE CONDITION PUSH HANDLE POP HANDLE	YES
Console support	WRITE OPERATOR	YES
Diagnostic services	DUMP ENTER TRACE	YES

Table 20. Summary of the CICS API by functional area (continued)

Functional area	Command	Supported?
Environmental services	ASSIGN ABCODE ABDUMP ABPROGRAM APPLID ASRAINRPT ASRAKEY ASRAPSW ASRAREGS ASRASPC ASRASTG CMDSEC CWALENG INITPARM INITPARMLENG INVOKINGPROG NETNAME OPERKEYS OPID ORGABCODE PRINSYSID PROGRAM RESSEC RESTART RETURNPROG STARTCODE SYSID TASKPRIORITY TCTUALENG TRANPRIORITY TWALENG USERID	YES
Environmental services (continued)	ASSIGN ALTSCRNHT MAPLINE ALTSCRNWD MAPWIDTH APLKYBD MSRCONTROL APLTEXT NATLANGINUSE BTRANS NEXTTRANSID COLOR NUMTAB DEFSCRNHT OPCLASS DEFSCRNWD OPSECURITY DELIMITER OUTLINE DESTCOUNT PAGENUM DESTID PARTNPAGE DESTIDLENG PARTNS DSSCS PARTNSET DS3270 PS EWASUPP QNAME EXTDS SCRNHT FACILITY SCRNWD FCI SIGDATA GCHARS SOSI GCODES STATIONID GMMI TELLERID HILIGHT TERMCODE INPARTN TERMPRIORITY KATAKANA TEXTKYBD LANGINUSE TEXTPRINT LDCMNEM UNATTEND LDCNUM USERNAME MAPCOLUMN USERPRIORITY MAPHEIGHT VALIDATION	NO

Table 20. Summary of the CICS API by functional area (continued)

Functional area	Command	Supported?
Environmental services (continued)	ADDRESS ACEE COMMAREA CWA EIB TCTUA TWA	YES
File control	DELETE ENDBR READ READNEXT READPREV RESETBR REWRITE STARTBR UNLOCK WRITE	YES
Interval control	ASKTIME CANCEL DELAY FORMATTIME POST RETRIEVE START WAIT EVENT	YES
Journaling	WRITE JOURNALNAME WAIT JOURNALNAME WRITE JOURNALNUM WAIT JOURNALNUM	YES
Monitoring	MONITOR POINT	YES
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	SYNCPPOINT	YES

Table 20. Summary of the CICS API by functional area (continued)

Functional area	Command	Supported?
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

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 21 on page 780 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 22 on page 781 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, HILIGHT, 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 21. 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
DFHBM DAR	Dark
DFHBMFSE	MDT set
DFHBM PRF	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
DFHBM CUR	Field containing cursor flagged
DFHBM EC	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
DFHBLINK	Blink
DFHREVRS	Reverse video

Table 21. Standard attribute and printer control character list, DFHBMSCA (continued)

Constant	Meaning
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
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

Notes:

¹ For text processing only. Use for constructing embedded set attribute orders in user text.

² Cannot be used in set attribute orders.

Table 22. 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)

Table 22. Bit map for attributes (continued)

prot	a/n	hi	spd	ndp	mdt	ebcd	ascii	char
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	'

Table 23. Key to attributes and settings in Bit Map

Code	Meaning
a/n	Automatic skip or numeric
ascii	American National Standard Code for Information Interchange
char	Graphic character equivalent to hex code
ebcd	Extended binary coded decimal interchange code
hi	High intensity
H	High
mdt	modified data tag
ndp	nondisplay print
N	Numeric
prot	Protected
P	Protected
spd	Selector pen detectable
S	Automatic skip
U	Unprotected
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. See below for the meaning of each bit. The constants supplied in DFHMSRCA are listed in Table 24 on page 783.

Table 24. 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.

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.

STATE VALUE

Modifies state to on or off if the corresponding bit is set on in STATE MASK.

INDICATOR MASK

Performs a similar function to STATE MASK, but for indicators.

- 0 Light 1 (Green)
- 1 Light 1 (Green)
- 2 Light2 (Yellow)
- 3 Audible alarm 1 (Long buzz)
- 4 Audible alarm 2 (Short buzz)

INDICATOR VALUE

Performs a similar function to STATE VALUE.

Attention identifier constants, DFHAID

The standard attention identifier list, DFHAID, simplifies testing the contents of the EIBAID field. Table 25 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.

Table 25. Standard list DFHAID

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.

BMS macros

The syntax of each BMS macro is defined, 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 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. However, a map can have the same name as a mapset.

Before CICS can load a physical map, it requires an installed resource definition for the map object. You can either use program autoinstall to create the definition when the mapset is first used, or define a mapset in the CSD using the DEFINE MAPSET resource definition.

You assemble a BMS mapset definition to generate either a symbolic description map or a physical map. The physical map is a structured data area used at

execution time to build the data stream for the terminal. The symbolic map is a series of data structures which you copy into your program at compile time so you can refer to the fields in the map by name.

For programming information about the autoinstall user program, see the *CICS Customization Guide*.

DFHMSD

The DFHMSD macro defines a mapset.

DFHMDI

The DFHMDI macro defines a map within the mapset defined by the previous DFHMSD macro. A map contains zero or more fields.

DFHMDF

The DFHMDF macro defines a field within a map defined by the previous DFHMDI macro.

Related concepts

“Partition set definition” on page 786

Related reference

“DFHMDF” on page 788

“DFHMDI” on page 798

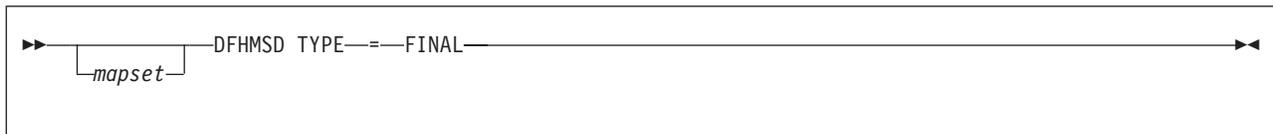
Map definition.

“DFHMSD” on page 807

Mapset definition.

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.

ADS descriptor

Physical maps produced by CICS Transaction Server for z/OS will also include an ADS descriptor in the output load module. This is provided to allow interpretation of the BMS Application Data Structure (the structure used by the application program for the data in SEND and RECEIVE MAP requests), without requiring your program to include the relevant DSECT or copybook at compile time.

The ADS descriptor contains a header with general information about the map, and a field descriptor for every field that appears in the ADS (corresponding to every named field in the map definition macro).

The ADS descriptor is generated for all maps. You can use the DSECT option to select the long form of the ADS, where all fields are aligned on 4-byte boundaries. The long form of the ADS is required by the 3270 Bridge when an interface to MQSeries® is used.

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.

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 Customization Guide*.)

DFHPDI

A partition set contains one or more partitions. Each partition is defined by coding a partition definition macro.

Related concepts

“Mapset, map, and field definition” on page 784

Related reference

“DFHPDI” on page 818

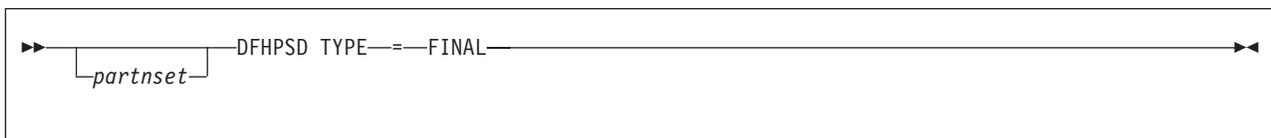
Partition definition.

“DFHPSD” on page 819

Partition set definition.

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.

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 DFHMDI or DFHMSD macro.

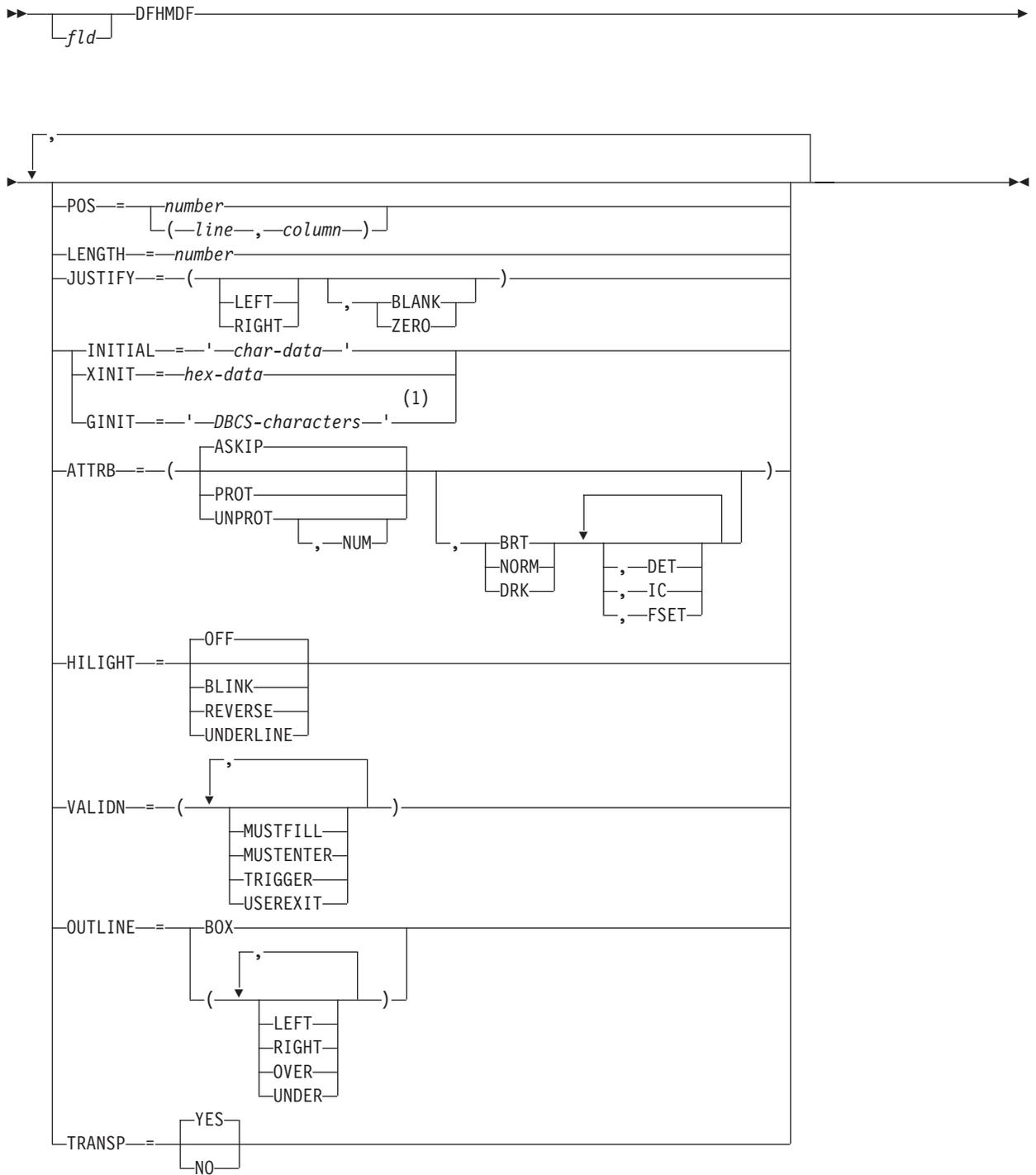
Briefly, a group of fields in a map would appear as follows in the map definition:

```
MAPSET DFHMSD....
      .
      .
MAP   DFHMDI....
      .
      .
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,...
      DFHMSD 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.

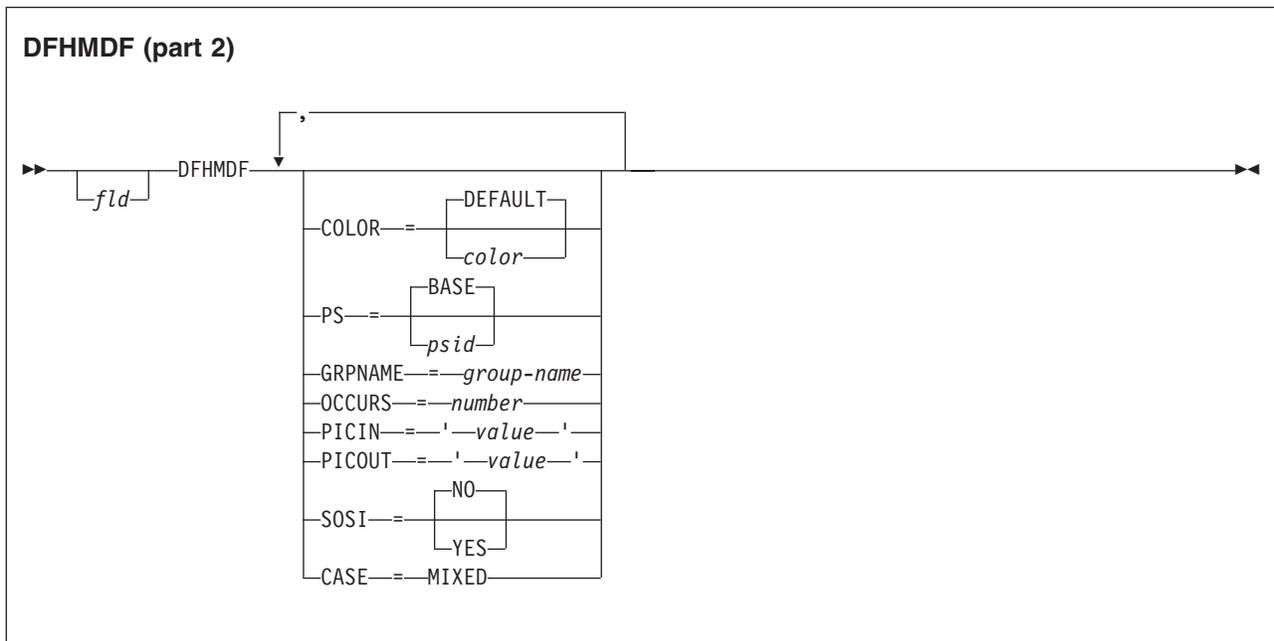
DFHMDF

DFHMDF (part 1)



Notes:

- 1 DBCS characters start with a shift-out character X'0E' and end with a shift-in character X'0F'.



Description

The DFHMDF macro defines a field within a map defined by the previous DFHMDF 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 Application Programming Guide*. If “fld” is 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. However, a map can have the same name as a mapset.

Before CICS can load a physical map, you must define a physical map using an RDO transaction with the MAPSET attribute.

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 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 the terminal definition specifies katakana support (KATAKANA(YES) option on RDO TYPETERM 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 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). 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.

The rules for defining group names are the same as for defining field names. See the *CICS Application Programming Guide* for details.

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.

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 RDO TYPETERM option HIGHLIGHT(YES).

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.

In general LENGTH can be omitted if PICIN or PICOUT is specified, unless PICOUT defines a COBOL picture containing a currency symbol that will be replaced by a currency sign of length greater than 1. LENGTH 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 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  DFHMDF  TYPE=DSECT,
        LANG=COBOL,
        MODE=INOUT
MAP    DFHMDF  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'
          DFHMDS 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.

```

Valid picture values for COBOL input maps are:

A P S V X 9 / and (

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 (

Refer to the appropriate language reference manual for the correct syntax of the PICTURE attribute.

Note: PL/I 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 currency string enclosed by less than (<) and greater than (>) symbols. For example:

```

DECLARE
  USPRICE PICTURE '$99.99',
  UKPRICE PICTURE '<£>99.99',
  EUPRICE PICTURE '<EUR>99.99';

```

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.

Valid picture values for COBOL output maps are:

A B E P S V X Z 0 9 , . + - \$
CR DB / and (

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 (

Refer to the appropriate language reference manual for the correct syntax of the PICTURE attribute.

Note: PL/I 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 currency string enclosed by less than (<) and greater than (>) symbols. For example:

```
DECLARE
    USPRICE PICTURE '$99.99',
    UKPRICE PICTURE '<£>99.99',
    EUPRICE PICTURE '<EUR>99.99';
```

Note: 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 ECPRICE 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 DFHMDI 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 DFHMDI macro or the DFHMSD 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.

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 that:

- validation is to be used on an 8775 terminal
- this field can be processed by the BMS global user exits

This overrides any VALIDN operand on the DFHMDI macro or 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 Application Programming Guide*.

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 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.

XINIT

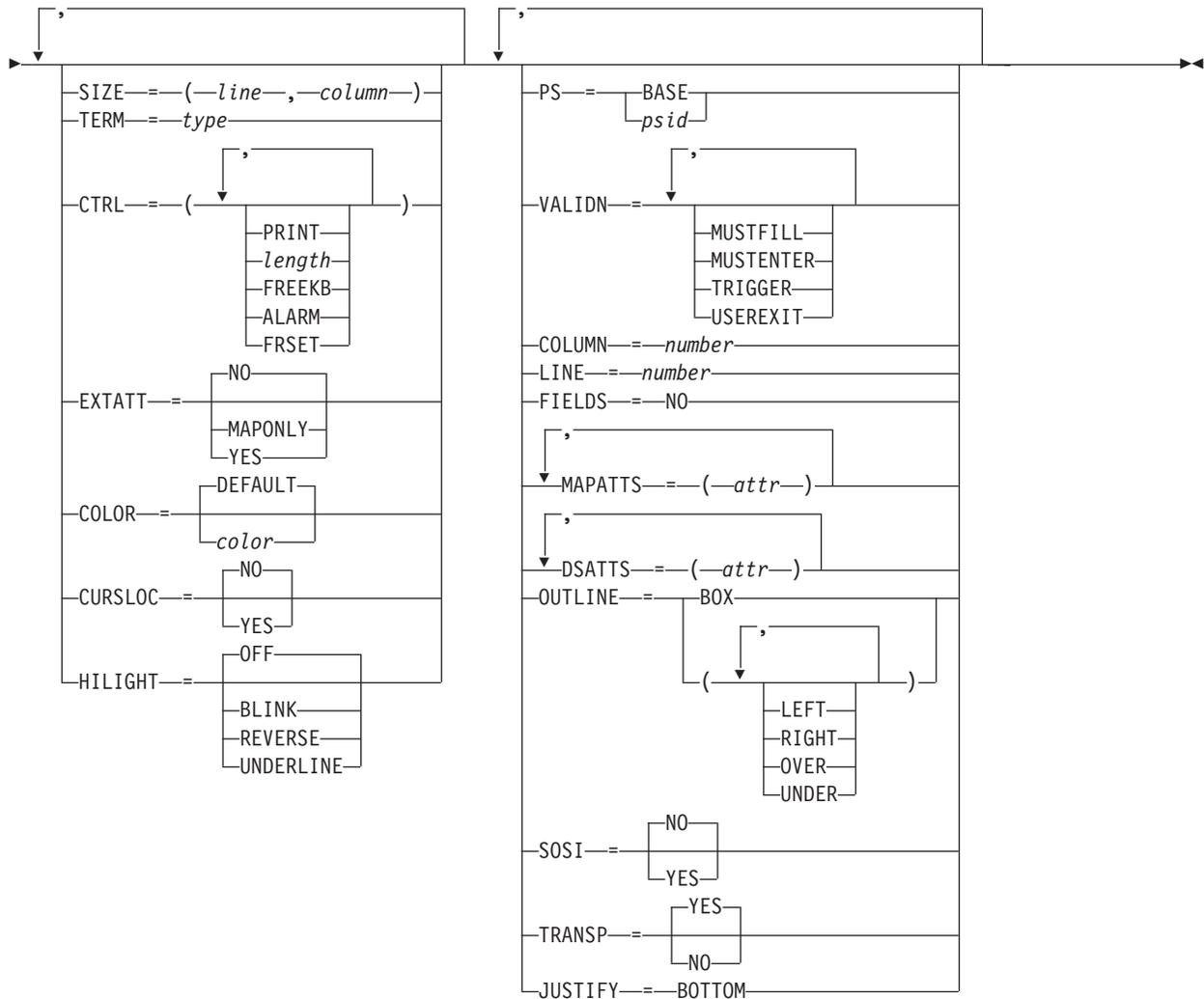
see INITIAL, earlier in the list. Only one of GINIT, INITIAL, or XINIT may be specified.

DFHMDI

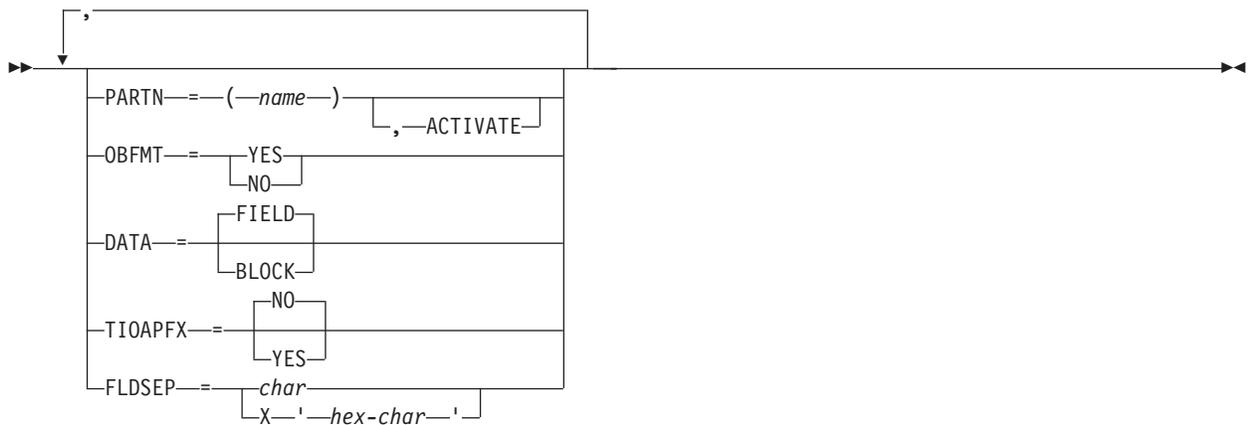
Map definition.

DFHMDI Minimum BMS

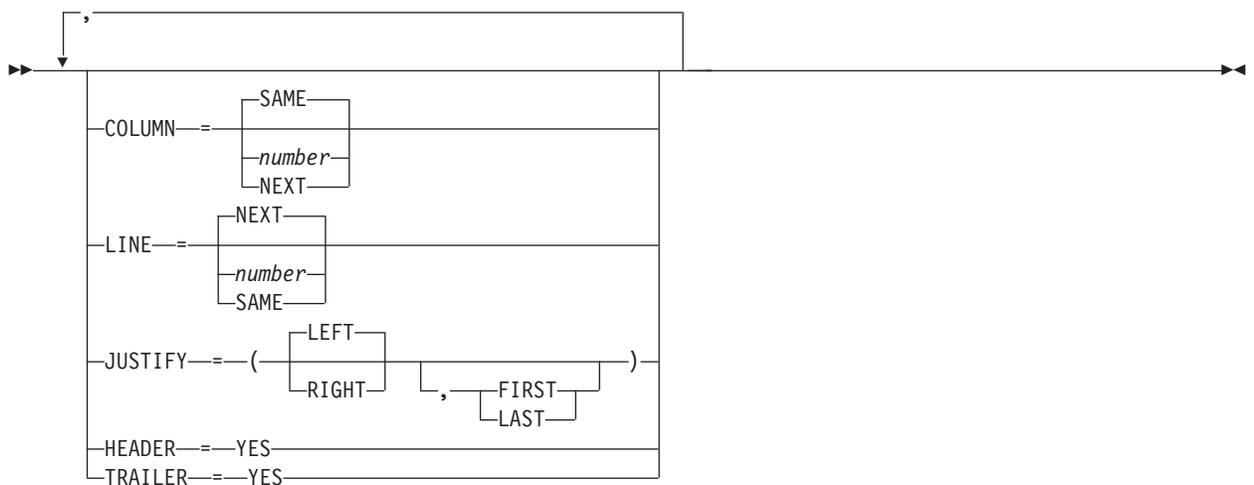
►► *map*—DFHMDI



DFHMDI Standard BMS



DFHMDI 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.

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 the RDO TYPETERM definition option COLOR(YES)>

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, 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.

Note: CTRL cannot be specified in the DFHMDI and DFHMSD macros in the same mapset.

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 “BMS-related constants” on page 779).

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 ...
```

Note:

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 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 (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 “BMS-related constants” on page 779).

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 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,

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 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 line that is completely empty.

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.

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 Application Programming Guide*.

The OBFMT operand overrides the OBFMT operand on the DFHMSD 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 DFHMSD 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 DFHMSD 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.

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.
- The map is to be used to send or receive data by way of the CICS 3270 Web Bridge.

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 DFHMDS 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 that:

- validation is to be used on an 8775 terminal
- this field can 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 Application Programming Guide*.

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 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.

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.)

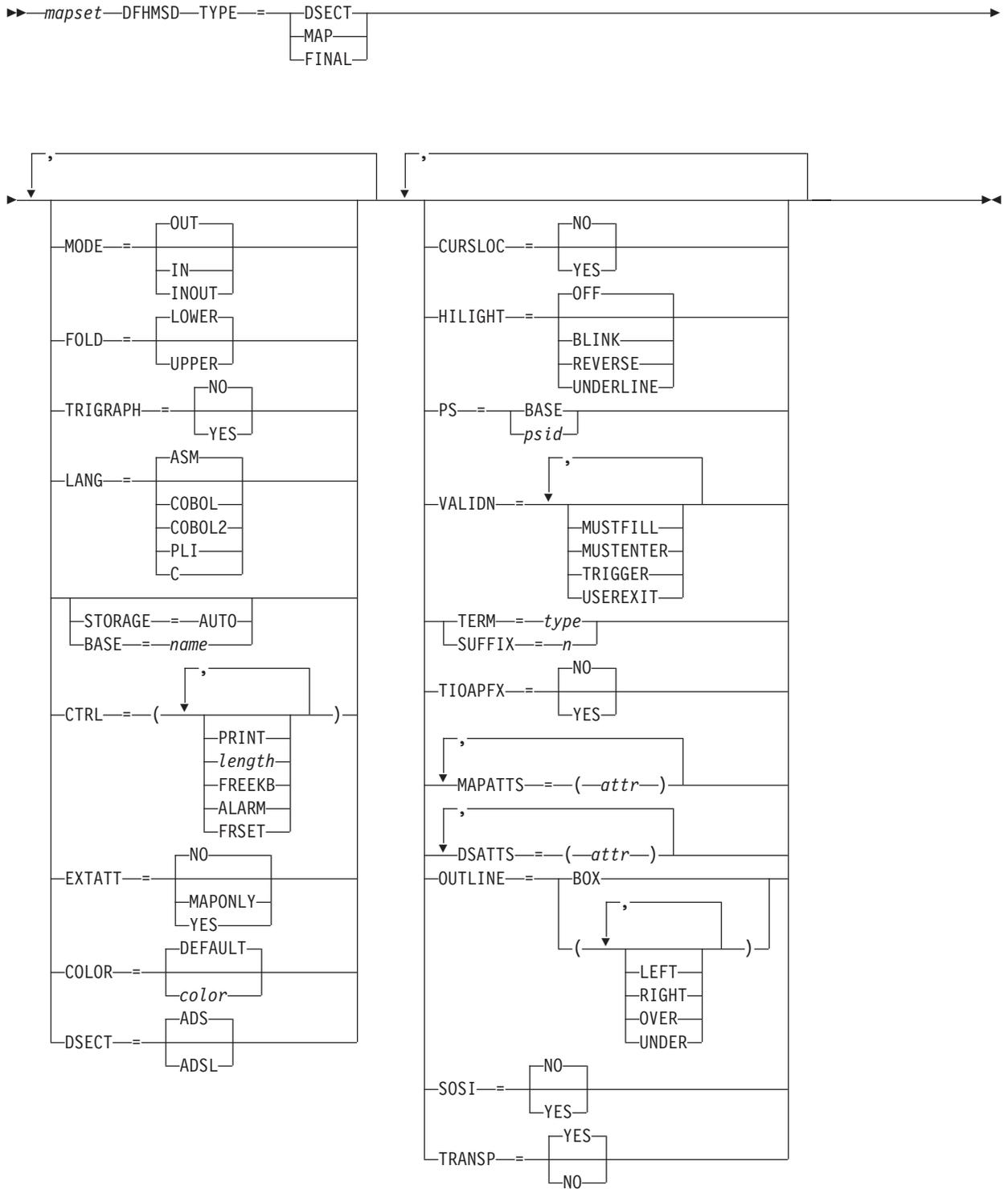
You should ensure that the names of maps, and names of fields within a mapset (or within multiple mapsets that are copied into one application program) are unique. However, a map can have the same name as a mapset.

Before CICS can load a physical map, you must define a physical map using an RDO DEFINE MAPSET command.

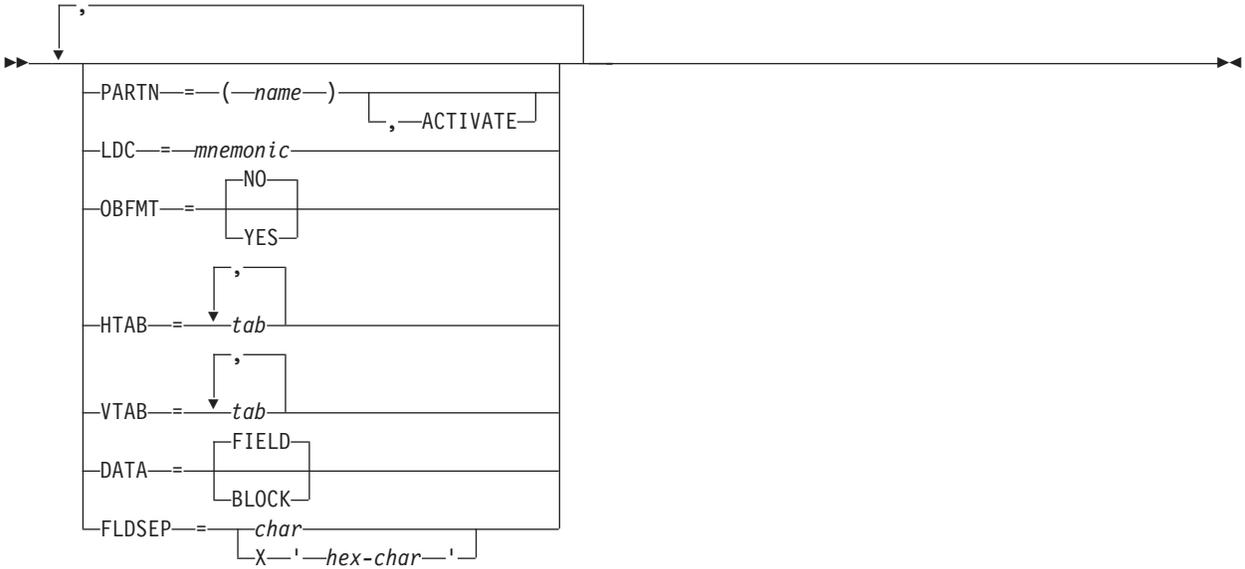
DFHMSD

Mapset definition.

DFHMSD Minimum BMS



DFHMSD 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 of the mapset. Normally, the name is up to 7 characters in length. However, if the mapset is used to generate HTML templates, and contains more than 36 maps, the name must not exceed 6 characters in length.

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. However, a map can have the same name as a mapset.

Before CICS can load a physical map, you must define a physical map using an RDO DEFINE MAPSET command.

You assemble a BMS mapset definition to generate either a symbolic description map or a physical map. The *CICS Application Programming Guide* tells you how to assemble and catalog the maps.

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 the RDO TYPETERM definition COLOR(YES) option.

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 DFHMDS 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.

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 “BMS-related constants” on page 779).

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 ...
```

Note:

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 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 “BMS-related constants” on page 779).

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 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.

DSECT

specifies whether the copybook generated supports the normal or long form of the application data structure (ADS).

ADS (default) requests mapping of the normal form of the ADS.

ADSL

requests mapping of the long form of the ADS, where all fields are aligned on 4-byte boundaries. This form is required by the 3270 Bridge interface to MQSeries. See the *CICS External Interfaces Guide* for more information about the 3270 Bridge.

This option requires LANG=C.

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.

HILIGHT

specifies the default highlighting attribute for all fields in all maps in a mapset. This is overridden by the HILIGHT operand of the DFHMDI, which is in turn 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 HIGHLIGHT operand is ignored unless the terminal supports highlighting, as indicated by HIGHLIGHT(YES) on the RDO TYPETERM definition.

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 DFHMDF 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 Application Programming Guide*.

The OBFMT operand on DFHMSD is overridden by the OBFMT operand on DFHMDF.

YES specifies that all maps within this mapset can be used in outboard formatting, except those for which OBFMT=NO is specified in the DFHMDF macro.

NO specifies that no maps within this mapset can be used in outboard formatting, except those for which OBFMT=YES is specified in DFHMDF.

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.

The **PS** operand is ignored unless the terminal supports programmed symbols, as indicated by **PROGSYMBOLS(YES)** on the **RDO TYPETERM** definition.

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 on the `ALTSUFFIX` attribute of a `TYPETERM` definition, 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 26.

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 Resource Definition Guide*.

Table 26. BMS terminal types

Type	Suffix	Notes
CRLP	A	Card-reader-in/line-printer-out
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	p	All interactive LUs including 3790 full-function LU and SCS printer LUs (3270 and 3790).

Table 26. BMS terminal types (continued)

Type	Suffix	Notes
2980	Q	
2980-4	R	
3270	blank	Default if TERM omitted. Same as ALL; used when no need to distinguish between models.
3601	U	
3653	V	Plus host-conv (3653) LU.
3650UP	W	Plus interpreter LU.
3650/3270	X	Plus host-conv (32700 LU.
BCHLU/3770B	Y	Plus all batch and BDI LUs.
ALL (all of the above)	blank	

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 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 Installation Guide*.

VALIDN

specifies that:

- validation is to be used on an 8775 terminal
- this field can 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 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 Application Programming Guide*.

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 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.

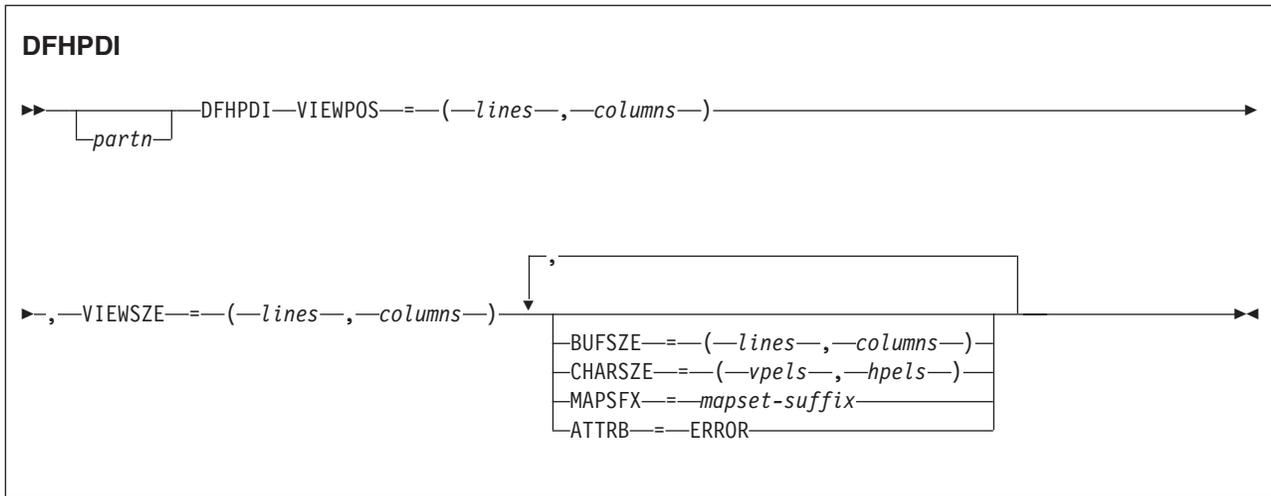
Note: The USEREXIT specification is totally unconnected with the field validation extended attribute as defined in the 3270 datastream architecture.

VTAB

specifies one or more tab positions for use with interactive and batch logical units and SCS printers having vertical forms control.

DFHPDI

Partition definition.



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.

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 TYPETERM option ERRHILIGHT is honored, but the LASTLINE option is ignored.

BUFSZE(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.

CHARSZE(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 CHARSZE 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. 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 TYPETERM 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.

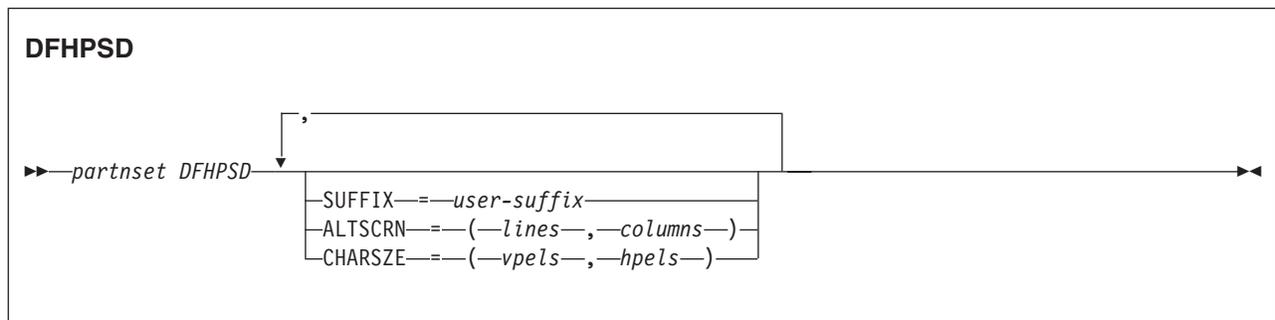
Note: The information given here on positioning viewports is necessarily brief. For more information you should consult the component description for the device you are using.

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 TYPETERM 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

Partition set definition.



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.

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 TYPETERM option ALTSCREEN. 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 CHARSIZE 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 TYPETERM option ALTSUFFIX. 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.

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.

Bibliography

The CICS Transaction Server for z/OS library

The published information for CICS Transaction Server for z/OS is delivered in the following forms:

The CICS Transaction Server for z/OS Information Center

The CICS Transaction Server for z/OS Information Center is the primary source of user information for CICS Transaction Server. The Information Center contains:

- Information for CICS Transaction Server in HTML format.
- Licensed and unlicensed CICS Transaction Server books provided as Adobe Portable Document Format (PDF) files. You can use these files to print hardcopy of the books. For more information, see “PDF-only books.”
- Information for related products in HTML format and PDF files.

One copy of the CICS Information Center, on a CD-ROM, is provided automatically with the product. Further copies can be ordered, at no additional charge, by specifying the Information Center feature number, 7014.

Licensed documentation is available only to licensees of the product. A version of the Information Center that contains only unlicensed information is available through the publications ordering system, order number SK3T-6945.

Entitlement hardcopy books

The following essential publications, in hardcopy form, are provided automatically with the product. For more information, see “The entitlement set.”

The entitlement set

The entitlement set comprises the following hardcopy books, which are provided automatically when you order CICS Transaction Server for z/OS, Version 3 Release 1:

Memo to Licensees, GI10-2559
CICS Transaction Server for z/OS Program Directory, GI10-2586
CICS Transaction Server for z/OS Release Guide, GC34-6421
CICS Transaction Server for z/OS Installation Guide, GC34-6426
CICS Transaction Server for z/OS Licensed Program Specification, GC34-6608

You can order further copies of the following books in the entitlement set, using the order number quoted above:

CICS Transaction Server for z/OS Release Guide
CICS Transaction Server for z/OS Installation Guide
CICS Transaction Server for z/OS Licensed Program Specification

PDF-only books

The following books are available in the CICS Information Center as Adobe Portable Document Format (PDF) files:

CICS books for CICS Transaction Server for z/OS

General

CICS Transaction Server for z/OS Program Directory, GI10-2586
CICS Transaction Server for z/OS Release Guide, GC34-6421
CICS Transaction Server for z/OS Migration from CICS TS Version 2.3, GC34-6425

CICS Transaction Server for z/OS Migration from CICS TS Version 1.3,
GC34-6423

CICS Transaction Server for z/OS Migration from CICS TS Version 2.2,
GC34-6424

CICS Transaction Server for z/OS Installation Guide, GC34-6426

Administration

CICS System Definition Guide, SC34-6428

CICS Customization Guide, SC34-6429

CICS Resource Definition Guide, SC34-6430

CICS Operations and Utilities Guide, SC34-6431

CICS Supplied Transactions, SC34-6432

Programming

CICS Application Programming Guide, SC34-6433

CICS Application Programming Reference, SC34-6434

CICS System Programming Reference, SC34-6435

CICS Front End Programming Interface User's Guide, SC34-6436

CICS C++ OO Class Libraries, SC34-6437

CICS Distributed Transaction Programming Guide, SC34-6438

CICS Business Transaction Services, SC34-6439

Java Applications in CICS, SC34-6440

JCICS Class Reference, SC34-6001

Diagnosis

CICS Problem Determination Guide, SC34-6441

CICS Messages and Codes, GC34-6442

CICS Diagnosis Reference, GC34-6899

CICS Data Areas, GC34-6902

CICS Trace Entries, SC34-6443

CICS Supplementary Data Areas, GC34-6905

Communication

CICS Intercommunication Guide, SC34-6448

CICS External Interfaces Guide, SC34-6449

CICS Internet Guide, SC34-6450

Special topics

CICS Recovery and Restart Guide, SC34-6451

CICS Performance Guide, SC34-6452

CICS IMS Database Control Guide, SC34-6453

CICS RACF Security Guide, SC34-6454

CICS Shared Data Tables Guide, SC34-6455

CICS DB2 Guide, SC34-6457

CICS Debugging Tools Interfaces Reference, GC34-6908

CICSplex SM books for CICS Transaction Server for z/OS

General

CICSplex SM Concepts and Planning, SC34-6459

CICSplex SM User Interface Guide, SC34-6460

CICSplex SM Web User Interface Guide, SC34-6461

Administration and Management

CICSplex SM Administration, SC34-6462

CICSplex SM Operations Views Reference, SC34-6463

CICSplex SM Monitor Views Reference, SC34-6464

CICSplex SM Managing Workloads, SC34-6465

CICSplex SM Managing Resource Usage, SC34-6466

CICSplex SM Managing Business Applications, SC34-6467

Programming

CICSplex SM Application Programming Guide, SC34-6468

CICSplex SM Application Programming Reference, SC34-6469

Diagnosis

CICSplex SM Resource Tables Reference, SC34-6470
CICSplex SM Messages and Codes, GC34-6471
CICSplex SM Problem Determination, GC34-6472

CICS family books

Communication

CICS Family: Interproduct Communication, SC34-6473
CICS Family: Communicating from CICS on System/390, SC34-6474

Licensed publications

The following licensed publications are not included in the unlicensed version of the Information Center:

CICS Diagnosis Reference, GC34-6899
CICS Data Areas, GC34-6902
CICS Supplementary Data Areas, GC34-6905
CICS Debugging Tools Interfaces Reference, GC34-6908

Other CICS books

The following publications contain further information about CICS, but are not provided as part of CICS Transaction Server for z/OS, Version 3 Release 1.

<i>Designing and Programming CICS Applications</i>	SR23-9692
<i>CICS Application Migration Aid Guide</i>	SC33-0768
<i>CICS Family: API Structure</i>	SC33-1007
<i>CICS Family: Client/Server Programming</i>	SC33-1435
<i>CICS Transaction Gateway for z/OS Administration</i>	SC34-5528
<i>CICS Family: General Information</i>	GC33-0155
<i>CICS 4.1 Sample Applications Guide</i>	SC33-1173
<i>CICS/ESA 3.3 XRF Guide</i>	SC33-0661

Books from related libraries

MVS

See the following books:

z/OS MVS Initialization and Tuning Guide, SC28-1751
z/OS MVS Initialization and Tuning Reference, SC28-1752
z/OS MVS JCL User's Guide, GC28-1758
z/OS MVS System Commands, GC28-1781

Systems Network Architecture

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

Sessions between Logical Units, GC20-1868.

SQL

For information about executing SQL see the following books:

DB2 Universal Database for OS/390 and z/OS: Application Programming and SQL Guide, SC26-9933
DB2 Universal Database for OS/390 and z/OS: SQL Reference, SC26-9944.

Other related books

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

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/ESE 3.3 IBM 3270 Data Stream Device Guide, SC33-0232

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

IBM 4700/3600/3630 Guide, SC33-0233

Accessibility

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

You can perform most tasks required to set up, run, and maintain your CICS system in one of these ways:

- using a 3270 emulator logged on to CICS
- using a 3270 emulator logged on to TSO
- using a 3270 emulator as an MVS system console

IBM Personal Communications provides 3270 emulation with accessibility features for people with disabilities. You can use this product to provide the accessibility features you need in your CICS system.

Index

Special characters

- > 32K COMMAREAs (channels)
 - ASSIGN command 50
 - CHANNEL option of RETURN command 483
 - CHANNEL option of XCTL command 742
 - DELETE CONTAINER (CHANNEL) command 139
 - GET CONTAINER (CHANNEL) command 253
 - MOVE CONTAINER (CHANNEL) command 366
 - PUT CONTAINER (CHANNEL) command 380
 - START CHANNEL command 603

Numerics

- 16MB line 262
- 2260 Display Station
 - CONVERSE command 101
 - RECEIVE command 443
 - SEND command 520
- 2265 Display Station
 - CONVERSE command 101
 - RECEIVE command 443
 - SEND command 520
- 2980 general banking terminal system
 - RECEIVE/SEND commands 520
- 2980 General Banking Terminal System
 - DFH2980 structure 445
 - output control 445
 - output to common buffer 445
 - passbook control 444
 - RECEIVE/SEND commands 444
- 3270 information display system (TCAM supported) 521
 - logical unit 508
- 3270 Information Display System 102, 332 (TCAM supported) 446
 - logical unit 89, 301, 434
- 3600 finance communication system
 - 3614 logical unit 510
- 3600 Finance Communication System
 - 3601 logical unit 90, 435, 509
 - 3614 logical unit 91, 436
 - pipeline logical units 508
- 3630 Plant Communication System
 - RECEIVE command 435
 - SEND command 509
- 3650 Host Command Processor
 - CONVERSE command 93
- 3650 Logical Units
 - RECEIVE command 437
- 3650 Store System
 - host conversational
 - LU 3270 511
 - LU 3653 511
 - interpreter logical unit 91, 310, 317, 436, 510
- 3650/3680 Full-function Logical Unit
 - RECEIVE command 438
 - SEND command 514

- 3650/3680 Store System
 - host command processor LU 512
- 3680 Host Command Processor
 - CONVERSE command 93
- 3680 Programmable Store System
 - host command processor LU 512
- 3740 Data Entry System 308, 309
- 3767 communication terminal
 - interactive logical unit 512
- 3767 Communication Terminal
 - interactive logical unit 93, 437
- 3770 data communication system
 - batch logical unit 513
- 3770 Data Communication System
 - batch logical unit 94, 438
- 3770 Full Function Logical Unit
 - RECEIVE command 438
 - SEND command 514
- 3770 interactive logical unit
 - RECEIVE command 437
- 3770 Interactive Logical Unit
 - SEND command 512
- 3790 communication system
 - 3270-display logical unit 514
 - full-function logical unit 513
 - SCS printer logical unit 514
- 3790 Communication System
 - 3270-display logical unit 95, 447
 - Full Function Logical Unit 94
 - full-function logical unit 438

A

- ABCODE option
 - ABEND command 27
 - ASSIGN command 47
 - CHECK ACQPROCESS command 76
 - CHECK ACTIVITY command 79
 - INQUIRE ACTIVITYID command 280
- ABDUMP option
 - ASSIGN command 47
- ABEND command 27
- ABEND exit, reactivating 273
- abend support commands 19
- abnormal termination, task 273
- ABPROGRAM option
 - ASSIGN command 47
 - CHECK ACQPROCESS command 76
 - CHECK ACTIVITY command 79
 - INQUIRE ACTIVITYID command 280
- absolute expressions 7
- ABSTIME option
 - ASKTIME command 45
 - CONVERTTIME command 108
 - FORMATTIME command 206
 - INQUIRE TIMER command 288
- access to system information
 - ADDRESS command 34

- access to system information (*continued*)
 - ADDRESS SET command 36
 - ASSIGN command 46
 - CICS storage areas 34, 36
- ACCUM option
 - SEND CONTROL command 526
 - SEND MAP command 531
 - SEND TEXT command 547
- ACEE option
 - ADDRESS command 34
- ACQACTIVITY option
 - CANCEL (BTS) command 70
 - CHECK ACTIVITY command 79
 - DELETE CONTAINER (BTS) command 137
 - FORCE TIMER command 203
 - GET CONTAINER (BTS) command 250
 - LINK ACTIVITY command 354
 - PUT CONTAINER (BTS) command 377
 - RESUME command 473
 - RUN command 502
 - SUSPEND (BTS) command 623
- ACQPROCESS option
 - CANCEL (BTS) command 70
 - CHECK ACQPROCESS command 76
 - DELETE CONTAINER (BTS) command 137
 - FORCE TIMER command 203
 - GET CONTAINER (BTS) command 250
 - LINK ACQPROCESS command 351
 - PUT CONTAINER (BTS) command 378
 - RESET ACQPROCESS command 464
 - RESUME command 473
 - RUN command 502
 - SUSPEND (BTS) command 623
- ACQUIRE command 29
- ACTION option
 - WEB CONVERSE command 658
 - WEB SEND command (Client) 706
 - WEB SEND command (Server) 699
 - WRITE OPERATOR command 732
- ACTIVE mode, of an activity 281
- activities
 - destruction of 135
 - modes 281
 - processing states 281
- ACTIVITY option
 - ASSIGN command 47
 - CANCEL (BTS) command 70
 - CHECK ACTIVITY command 79
 - DEFINE ACTIVITY command 109
 - DELETE ACTIVITY command 135
 - DELETE CONTAINER (BTS) command 137
 - GET CONTAINER (BTS) command 250
 - GETNEXT ACTIVITY command 265
 - INQUIRE ACTIVITYID command 280
 - LINK ACTIVITY command 354
 - PUT CONTAINER (BTS) command 378
 - RESET ACTIVITY command 466
 - RESUME command 473
 - RUN command 502
 - SUSPEND (BTS) command 623
- activity-related commands
 - ACQUIRE 29
 - CANCEL (BTS) 70
 - CHECK ACQPROCESS 76
 - CHECK ACTIVITY 78
 - DEFINE ACTIVITY 109
 - DEFINE PROCESS 119
 - DELETE ACTIVITY 135
 - INQUIRE ACTIVITYID 280
 - INQUIRE PROCESS 287
 - LINK ACQPROCESS 350
 - LINK ACTIVITY 353
 - RESET ACQPROCESS 464
 - RESET ACTIVITY 466
 - RESUME 473
 - RUN 500
 - STARTBROWSE ACTIVITY 614
 - SUSPEND (BTS) 623
- ACTIVITYBUSY condition
 - ACQUIRE command 30
 - CANCEL (BTS) command 70
 - CHECK ACTIVITY command 80
 - DELETE ACTIVITY command 135
 - LINK ACTIVITY command 354
 - RESET ACTIVITY command 466
 - RESUME command 473
 - RUN command 502
 - SUSPEND (BTS) command 623
- ACTIVITYERR condition
 - ACQUIRE command 30
 - CANCEL (BTS) command 70
 - CHECK ACTIVITY command 80
 - DEFINE ACTIVITY command 110
 - DELETE ACTIVITY command 135
 - DELETE CONTAINER (BTS) command 138
 - GET CONTAINER (BTS) command 251
 - GETNEXT ACTIVITY command 266
 - INQUIRE ACTIVITYID command 282
 - INQUIRE CONTAINER command 284
 - INQUIRE EVENT command 286
 - INQUIRE TIMER command 289
 - LINK ACTIVITY command 354
 - MOVE CONTAINER (BTS) command 364
 - PUT CONTAINER (BTS) command 378
 - RESET ACTIVITY command 466
 - RESUME command 473
 - RUN command 503
 - STARTBROWSE ACTIVITY command 615
 - STARTBROWSE CONTAINER command 617
 - STARTBROWSE EVENT command 618
 - SUSPEND (BTS) command 623
- ACTIVITYID option
 - ACQUIRE command 30
 - ASSIGN command 47
 - DEFINE ACTIVITY command 109
 - GETNEXT ACTIVITY command 265
 - GETNEXT PROCESS command 271
 - INQUIRE ACTIVITYID command 280
 - INQUIRE CONTAINER command 283
 - INQUIRE EVENT command 285
 - INQUIRE PROCESS command 287

ACTIVITYID option (*continued*)
 INQUIRE TIMER command 288
 STARTBROWSE ACTIVITY command 614
 STARTBROWSE CONTAINER command 616
 STARTBROWSE EVENT command 618

ACTPARTN option
 SEND CONTROL command 526
 SEND MAP command 531
 SEND TEXT command 547

ADD SUBEVENT command 32

ADDRESS command 34

ADDRESS SET command 36

address, cursor 772

ADS descriptor 785

ADS value
 DFHMSD 812

ADSL value
 DFHMSD 812

AFTER option
 DEFINE TIMER command 122
 POST command 372
 ROUTE command 495
 START command 592

AID option
 RECEIVE MAP MAPPINGDEV command 455

ALARM option
 SEND CONTROL command 526
 SEND MAP command 531
 SEND MAP MAPPINGDEV command 538
 SEND TEXT command 547
 SEND TEXT NOEDIT command 557

ALARM value
 DFHMDI 801
 DFHMSD 810

ALIGNED attribute
 PL/I 6

ALL option
 SEND PAGE command 541

ALLOCATE (APPC) command 37

ALLOCATE (LUTYPE6.1) command 41

ALLOCATE (MRO) command 43

ALLOCERR condition
 SPOOLCLOSE command 573
 SPOOLOPEN INPUT command 575
 SPOOLOPEN OUTPUT command 579
 SPOOLREAD command 582
 SPOOLWRITE command 585

ALTER option
 QUERY SECURITY command 387

ALTERNATE option
 CONVERSE (non-VTAM) command 102
 CONVERSE (VTAM) command 95
 SEND (non-VTAM) command 521
 SEND (VTAM) command 515
 SEND CONTROL command 526
 SEND MAP command 531
 SEND TEXT command 547
 SEND TEXT NOEDIT command 557

ALTSCRN operand
 DFHPSD 819

ALTSCRNHT option
 ASSIGN command 48

ALTSCRNWD option
 ASSIGN command 48

AND option
 DEFINE COMPOSITE EVENT command 113

ANYKEY option
 HANDLE AID command 275

APLKYBD option
 ASSIGN command 48

APLTEXT option
 ASSIGN command 48

APPC basic conversations
 commands 19

APPC logical unit
 acquiring session to 37
 initiating conversation with 83
 returning mapped sessions to CICS 211
 sending and receiving 86

APPC mapped conversations
 abending 293
 commands 19
 ensuring transmission of accumulated data 639
 extracting attributes of 188
 informing partner of error 315
 issuing a positive response 299
 receiving data 432
 requesting change of direction 336
 retrieving values from attach header 197
 returning sessions to CICS 211
 sending data 505

application performance, monitoring 360

APPLID option
 ASSIGN command 48

argument values
 assembler language 6
 C 4
 COBOL 3
 PL/I 5

AS option
 MOVE CONTAINER (BTS) command 364
 MOVE CONTAINER (CHANNEL) command 367

ASA option
 SPOOLOPEN OUTPUT command 578

ASIS option
 CONVERSE (non-VTAM) command 102
 CONVERSE (VTAM) command 95
 RECEIVE (non-VTAM) command 447
 RECEIVE (VTAM) command 439
 RECEIVE MAP command 451
 RECEIVE PARTN command 458
 SEND (non-VTAM) command 521

ASKIP value
 DFHMDF 790

ASKTIME command 45

ASRAINTRPT option
 ASSIGN command 48

ASRAKEY option
 ASSIGN command 48

ASRAPSW option
 ASSIGN command 48

- ASRAREGS option
 - ASSIGN command 49
- ASRASPC option
 - ASSIGN command 49
- ASRASTG option
 - ASSIGN command 49
- assembler language
 - argument values 6
 - LENGTH option default 8
 - program exit 12
 - register contents 11
 - translated code 11
- ASSIGN command 46
- asynchronous interrupt 769
- ASYNCHRONOUS option
 - RUN command 502
- AT option
 - DEFINE TIMER command 122
 - DOCUMENT INSERT command 155
 - POST command 372
 - ROUTE command 496
 - START command 592
- attach
 - start a task 598
- ATTACHID option
 - BUILD ATTACH (LUTYPE6.1) command 62
 - BUILD ATTACH (MRO) command 65
 - CONVERSE (non-VTAM) command 102
 - CONVERSE (VTAM) command 95
 - EXTRACT ATTACH (LUTYPE6.1) command 180
 - EXTRACT ATTACH (MRO) command 184
 - SEND (non-VTAM) command 521
 - SEND (VTAM) command 515
- attention identifier (AID) 275
- ATTRB operand
 - DFHMDF 790
 - DFHPDI 818
- attributes
 - control character list, DFHBMSCA 779
- AUTHENTICATE option
 - EXTRACT TCPIP command 199
- authentication commands 19
- AUTOPAGE option
 - SEND PAGE command 541
- AUXILIARY option
 - WRITEQ TS command 737

B

- back out to a syncpoint 626
- BASE operand
 - DFHMSD 810
- BASE value
 - DFHMDF 797
 - DFHMDI 806
 - DFHMSD 814
- basic mapping support (BMS)
 - ADS descriptor 785
 - commands 20
 - completing a logical message 541
 - deleting a logical message 375
 - basic mapping support (BMS) *(continued)*
 - determining input partition 458
 - field definition macro 785, 789
 - full BMS
 - RECEIVE MAP command 451
 - RECEIVE PARTN command 458
 - SEND CONTROL command 525
 - SEND MAP command 530
 - SEND PAGE 541
 - SEND PARTNSET 545
 - SEND TEXT command 546
 - SEND TEXT MAPPED 553
 - SEND TEXT NOEDIT 556
 - full-function BMS
 - PURGE MESSAGE 375
 - map definition macro 785, 800
 - mapping input data 451
 - mapping input data with MAPPINGDEV 455
 - mapset definition macro 785, 809
 - minimum BMS
 - RECEIVE MAP command 451
 - RECEIVE MAP MAPPINGDEV command 455
 - SEND CONTROL command 525
 - SEND MAP command 530
 - SEND MAP MAPPINGDEV command 538
 - partition definition macro 786, 818
 - partition set definition macro 786, 819
 - related constants 779
 - routing a logical message 495
 - sending previously mapped data 553
 - sending user-defined data stream 556
 - standard BMS
 - RECEIVE MAP command 451
 - RECEIVE PARTN command 458
 - SEND CONTROL command 525
 - SEND MAP command 530
 - SEND PARTNSET 545
 - SEND TEXT command 546
 - batch data interchange (BDI)
 - add record to data set 297
 - commands 20
 - conditions 318
 - delete a record from data set 311
 - read record from data set 327
 - request next record number 318
 - send data to output device 333
 - terminate data set 295, 306
 - update a record in data set 329
 - wait for function completion 339
 - batch logical unit, 3770 94, 438, 513
 - BELOW option
 - GETMAIN command 262
 - BIF DEEDIT command 60
 - big COMMAREAs (channels)
 - ASSIGN command 50
 - DELETE CONTAINER (CHANNEL) command 139
 - big COMMAREAs, channels 139, 253, 366, 380, 483, 603, 742
 - BINARY option
 - DOCUMENT INSERT command 155

- BLANK value
 - DFHMDF 793
- BLINK value
 - DFHMDF 792
 - DFHMDI 803
 - DFHMSD 812
- BLOCK value
 - DFHMDI 802
 - DFHMSD 811
- BOOKMARK option
 - DOCUMENT INSERT command 155
- BOTTOM value
 - DFHMDI 804
- BRDATA option
 - START BREXIT command 600
- BRDATALENGTH option
 - START BREXIT command 601
- BREXIT option
 - START BREXIT command 600
- bridge (3270)
 - start a task 600
- BRIDGE option
 - ASSIGN command 50
- browse operation
 - ending 168
 - read next record 403
 - read previous record 414
 - reset starting point 468
 - starting 608
- BROWSETOKEN option
 - ENDBROWSE ACTIVITY command 170
 - ENDBROWSE CONTAINER command 171
 - ENDBROWSE EVENT command 172
 - ENDBROWSE PROCESS command 173
 - GETNEXT ACTIVITY command 265
 - GETNEXT CONTAINER command 267
 - GETNEXT EVENT command 269
 - GETNEXT PROCESS command 271
 - STARTBROWSE ACTIVITY command 615
 - STARTBROWSE CONTAINER command 616
 - STARTBROWSE EVENT command 618
 - STARTBROWSE PROCESS command 620
- browsing commands
 - ENDBROWSE ACTIVITY 170
 - ENDBROWSE CONTAINER 171
 - ENDBROWSE PROCESS 173
 - GETNEXT ACTIVITY 265
 - GETNEXT CONTAINER 267
 - GETNEXT EVENT 269
 - GETNEXT PROCESS 271
 - INQUIRE ACTIVITYID 280
 - INQUIRE CONTAINER 283
 - INQUIRE EVENT 285
 - INQUIRE PROCESS 287
 - INQUIRE TIMER 288
 - STARTBROWSE ACTIVITY 614
 - STARTBROWSE CONTAINER 616
 - STARTBROWSE PROCESS 620
- BRT value
 - DFHMDF 790
- BTRANS option
 - ASSIGN command 50
- BTS commands
 - ACQUIRE 29
 - ADD SUBEVENT 32
 - CANCEL (BTS) 70
 - CHECK ACQPROCESS 76
 - CHECK ACTIVITY 78
 - CHECK TIMER 81
 - DEFINE ACTIVITY 109
 - DEFINE COMPOSITE EVENT 112
 - DEFINE INPUT EVENT 118
 - DEFINE PROCESS 119
 - DEFINE TIMER 122
 - DELETE ACTIVITY 135
 - DELETE CONTAINER (BTS) 137
 - DELETE EVENT 142
 - DELETE TIMER 144
 - ENDBROWSE ACTIVITY 170
 - ENDBROWSE CONTAINER 171
 - ENDBROWSE EVENT 172
 - ENDBROWSE PROCESS 173
 - FORCE TIMER 203
 - GET CONTAINER (BTS) 250
 - GETNEXT ACTIVITY 265
 - GETNEXT CONTAINER 267
 - GETNEXT EVENT 269
 - GETNEXT PROCESS 271
 - INQUIRE ACTIVITYID 280
 - INQUIRE CONTAINER 283
 - INQUIRE EVENT 285
 - INQUIRE PROCESS 287
 - INQUIRE TIMER 288
 - LINK ACQPROCESS 350
 - LINK ACTIVITY 353
 - MOVE CONTAINER (BTS) 363
 - PUT CONTAINER (BTS) 377
 - REMOVE SUBEVENT 463
 - RESET ACQPROCESS 464
 - RESET ACTIVITY 466
 - RESUME 473
 - RETRIEVE REATTACH EVENT 479
 - RETRIEVE SUBEVENT 481
 - RUN 500
 - STARTBROWSE ACTIVITY 614
 - STARTBROWSE CONTAINER 616
 - STARTBROWSE EVENT 618
 - STARTBROWSE PROCESS 620
 - SUSPEND (BTS) 623
 - TEST EVENT 627
- BUFFER option
 - GDS RECEIVE command 242
 - RECEIVE (non-VTAM) command 447
 - RECEIVE (VTAM) command 439
- BUFSZE operand
 - DFHPDI 818
- BUILD ATTACH (LUTYPE6.1) command 62
- BUILD ATTACH (MRO) command 65
- built-in functions
 - commands 20

C

C language

- ADDRESS COMMAREA 35
- ADDRESS EIB 35
- argument values 4
- LENGTH option default 5
- translated code 10
- CADDRLENGTH option
 - EXTRACT TCPIP command 199
- CANCEL (BTS) command 70
- CANCEL command 68
- CANCEL option
 - ABEND command 27
 - HANDLE ABEND command 274
- CANCELLING mode, of an activity 281
- CARD option
 - ISSUE ABORT command 295
 - ISSUE END command 306
 - ISSUE SEND command 333
 - ISSUE WAIT command 339
- CASE operand
 - DFHMDF 791
- CBIDERR condition
 - ALLOCATE (APPC) command 39
 - ALLOCATE (LUTYPE6.1) command 42
 - CONVERSE (non-VTAM) command 106
 - CONVERSE (VTAM) command 98
 - EXTRACT ATTACH (LUTYPE6.1) command 183
 - EXTRACT ATTACH (MRO) command 186
 - SEND (non-VTAM) command 524
 - SEND (VTAM) command 518
- CBUFF option
 - SEND (non-VTAM) command 521
- CCSIDERR condition
 - GET CONTAINER (CHANNEL) command 255
 - PUT CONTAINER (CHANNEL) command 382
 - SOAPFAULT ADD command 566
- CERTIFICATE option
 - EXTRACT CERTIFICATE command 192
 - WEB OPEN command 673
- CHANGE PASSWORD command 73
- CHANGE TASK command 75
- CHANGED condition
 - DELETE command 130
 - REWRITE command 491
- CHANGETIME option
 - VERIFY PASSWORD command 637
- channel commands
 - CHANNEL option of RETURN command 483
 - CHANNEL option of XCTL command 742
 - DELETE CONTAINER (CHANNEL) 139
 - GET CONTAINER (CHANNEL) 253
 - MOVE CONTAINER (CHANNEL) 366
 - PUT CONTAINER (CHANNEL) 380
 - START CHANNEL 603
- Channel commands 21
- CHANNEL option
 - ASSIGN command 50
 - DELETE CONTAINER (CHANNEL) command 139
 - GET CONTAINER (CHANNEL) command 253
 - LINK command 344

CHANNEL option (*continued*)

- MOVE CONTAINER (CHANNEL) command 367
- PUT CONTAINER (CHANNEL) command 380
- RETURN command 483
- START TRANSID (CHANNEL) command 604
- XCTL command 742
- CHANNELERR condition
 - DELETE CONTAINER (CHANNEL) command 139
 - GET CONTAINER (CHANNEL) command 255
 - LINK command 346
 - MOVE CONTAINER (CHANNEL) command 367
 - PUT CONTAINER (CHANNEL) command 383
 - RETURN command 486
 - START TRANSID (CHANNEL) command 605
 - XCTL command 743
- channels
 - ASSIGN command 50
- channels as large COMMAREAs 139, 253, 366, 380, 483, 603, 742
- CHARACTERSET option
 - WEB CONVERSE command 663
 - WEB RECEIVE command (Server) 688
 - WEB SEND command (Client) 707
 - WEB SEND command (Server) 699
- CHARSZ operand
 - DFHPDI 818
 - DFHPSD 820
- CHECK ACQPROCESS command 76
- CHECK ACTIVITY command 78
- CHECK TIMER command 81
- CHUNKING option
 - WEB SEND command (Client) 707
 - WEB SEND command (Server) 699
- CICS business transaction services (BTS) commands 20
- CICS Web Interface (CWI) commands
 - DOCUMENT CREATE 151
 - DOCUMENT INSERT 155
 - DOCUMENT RETRIEVE 158
 - DOCUMENT SET 160
 - EXTRACT CERTIFICATE 192
- CICS Web support commands
 - CONVERSE WEB 657
 - WEB CLOSE 654
 - WEB CONVERSE 657
 - WEB ENDBROWSE FORMFIELD 666
 - WEB ENDBROWSE HTTPHEADER 667
 - WEB EXTRACT 668
 - WEB OPEN 673
 - WEB PARSE URL 677
 - WEB READ FORMFIELD 680
 - WEB READ HTTPHEADER 682
 - WEB READNEXT FORMFIELD 684
 - WEB READNEXT HTTPHEADER 686
 - WEB RECEIVE 688
 - WEB RECEIVE (Client) 693
 - WEB RETRIEVE 697
 - WEB SEND (Client) 706
 - WEB SEND (Server) 698
 - WEB STARTBROWSE FORMFIELD 713
 - WEB STARTBROWSE HTTPHEADER 715

CICS Web support commands *(continued)*
 WEB WRITE HTTPHEADER 716
 CICS DATAKEY option
 GETMAIN command 263
 CIPHERS option
 WEB OPEN command 673
 CLASS option
 SPOOL OPEN INPUT command 574
 SPOOL OPEN OUTPUT command 578
 CLEAR option
 HANDLE AID command 275
 client requests
 extracting information 192
 CLIENTADDR option
 EXTRACT TCPIP command 199
 CLIENTADDRNU option
 EXTRACT TCPIP command 199
 CLIENTCONV option
 WEB CONVERSE command 663
 WEB RECEIVE command (Client) 694
 WEB SEND command (Client) 708
 CLIENTNAME option
 EXTRACT TCPIP command 200
 CLNTCODEPAGE option
 DOCUMENT RETRIEVE command 158
 WEB READ FORMFIELD command 680
 WEB RECEIVE command (Server) 689
 WEB SEND command (Server) 700
 WEB STARTBROWSE FORMFIELD command 713
 CLOSESTATUS option
 WEB CONVERSE command 658
 WEB SEND command (Client) 708
 WEB SEND command (Server) 700
 CLRPARTN option
 HANDLE AID command 275
 CMDSEC option
 ASSIGN command 50
 CNAMELENGTH option
 EXTRACT TCPIP command 200
 CNOTCOMPL option
 SEND (non-VTAM) command 522
 SEND (VTAM) command 515
 COBOL
 argument values 3
 translated code 10
 CODEPAGE option
 WEB OPEN command 674
 CODEREG operand 14
 COLOR operand
 DFHMDF 792
 DFHMDI 800
 DFHMSD 810
 COLOR option
 ASSIGN command 50
 COLUMN operand
 DFHMDI 801
 column value
 DFHMDI 806
 command language translator
 translated code 11
 commands
 format, arguments 1
 scheduling 23
 security 23
 spool 23
 TCP/IP 23
 temporary storage control 24
 COMMAREA option
 ADDRESS command 34
 LINK command 344
 RETURN command 484
 XCTL command 742
 common buffer, output to, 2980 445
 common programming interface communications (CPI communications) 774
 COMMONNAME option
 EXTRACT CERTIFICATE command 193
 COMMONNAMLEN option
 EXTRACT CERTIFICATE command 193
 COMPAREMAX option
 GET COUNTER command 256
 GET DCOUNTER command 256
 UPDATE COUNTER command 632
 UPDATE DCOUNTER command 632
 COMPAREMIN option
 GET COUNTER command 257
 GET DCOUNTER command 257
 UPDATE COUNTER command 633
 UPDATE DCOUNTER command 633
 COMPLETE mode, of an activity 281
 COMPLETE option
 DUMP TRANSACTION command 163
 COMPOSITE option
 GETNEXT EVENT command 269
 INQUIRE EVENT command 285
 COMPSTATUS option
 CHECK ACQPROCESS command 76
 CHECK ACTIVITY command 79
 INQUIRE ACTIVITYID command 280
 CONFIRM option
 GDS SEND command 245
 SEND (VTAM) command 515
 CONNECT PROCESS command 83
 CONSISTENT option
 READ command 393
 READNEXT command 406
 READPREV command 417
 CONSOLE option
 ISSUE ABORT command 295
 ISSUE END command 306
 ISSUE SEND command 333
 ISSUE WAIT command 339
 console support commands 21
 constants
 AID values, DFHAID 784
 attribute values, DFHBMSCA 779
 for 3270 attributes 779
 for examining EIBAID field 784
 for MSR control values 782
 for printer format controls 779
 MSR control, DFHtex read 782

constants (*continued*)
 printer control values, DFHBMSCA 779

container commands
 DELETE CONTAINER (BTS) 137
 DELETE CONTAINER (CHANNEL) 139
 ENDBROWSE CONTAINER 171
 GET CONTAINER (BTS) 250
 GET CONTAINER (CHANNEL) 253
 GETNEXT CONTAINER 267
 INQUIRE CONTAINER 283
 MOVE CONTAINER (BTS) 363
 MOVE CONTAINER (CHANNEL) 366
 PUT CONTAINER (BTS) 377
 PUT CONTAINER (CHANNEL) 380
 STARTBROWSE CONTAINER 616

CONTAINER option
 DELETE CONTAINER (BTS) command 137
 DELETE CONTAINER (CHANNEL) command 139
 GET CONTAINER (BTS) command 250
 GET CONTAINER (CHANNEL) command 253
 GETNEXT CONTAINER command 267
 INQUIRE CONTAINER command 283
 MOVE CONTAINER (BTS) command 364
 MOVE CONTAINER (CHANNEL) command 367
 PUT CONTAINER (BTS) command 378
 PUT CONTAINER (CHANNEL) command 381

CONTAINERERR condition
 DELETE CONTAINER (BTS) command 138
 DELETE CONTAINER (CHANNEL) command 139
 GET CONTAINER (BTS) command 251
 GET CONTAINER (CHANNEL) command 255
 INQUIRE CONTAINER command 284
 MOVE CONTAINER (BTS) command 364
 MOVE CONTAINER (CHANNEL) command 367
 PUT CONTAINER (BTS) command 378
 PUT CONTAINER (CHANNEL) command 383

context-switching
 described 350, 353, 500

CONTROL option
 QUERY SECURITY command 387

CONVDATA option
 GDS CONNECT PROCESS command 223
 GDS EXTRACT ATTRIBUTES command 226
 GDS FREE command 230
 GDS ISSUE ABEND command 232
 GDS ISSUE CONFIRMATION command 234
 GDS ISSUE ERROR command 236
 GDS ISSUE PREPARE command 238
 GDS ISSUE SIGNAL command 240
 GDS RECEIVE command 242
 GDS SEND command 245
 GDS WAIT command 248

CONVERSE (2260) command 101
 CONVERSE (3270 display) command 102
 CONVERSE (3270 logical) command 89
 CONVERSE (3600-3601) command 90
 CONVERSE (3600-3614) command 91
 CONVERSE (3650 interpreter) command 91
 CONVERSE (3650-3270) command 92
 CONVERSE (3650-3653) command 92
 CONVERSE (3650-3680) command 93

CONVERSE (3767) command 93
 CONVERSE (3770) command 94
 CONVERSE (3790 3270-display) command 95
 CONVERSE (3790 full-function or inquiry) command 94
 CONVERSE (APPC) command 86
 CONVERSE (LUTYPE2/LUTYPE3) command 87
 CONVERSE (LUTYPE4) command 87
 CONVERSE (LUTYPE6.1) command 88
 CONVERSE (MRO) command 101
 CONVERSE (non-VTAM default) command 100
 CONVERSE (SCS) command 88
 CONVERSE (VTAM default) command 86
 CONVERSE option
 ISSUE LOAD command 317
 CONVERSE WEB command 657
 converse with terminal or LU 769
 CONVERTTIME command 107

CONVID option
 CONNECT PROCESS command 83
 CONVERSE (non-VTAM) command 103
 CONVERSE (VTAM) command 96
 EXTRACT ATTACH (LUTYPE6.1) command 180
 EXTRACT ATTACH (MRO) command 184
 EXTRACT ATTRIBUTES (APPC) command 188
 EXTRACT ATTRIBUTES (MRO) command 190
 EXTRACT PROCESS command 197
 FREE (APPC) command 211
 FREE (LUTYPE6.1) command 213
 FREE (MRO) command 214
 GDS ALLOCATE command 219
 GDS CONNECT PROCESS command 223
 GDS EXTRACT ATTRIBUTES command 226
 GDS EXTRACT PROCESS command 228
 GDS FREE command 230
 GDS ISSUE ABEND command 232
 GDS ISSUE CONFIRMATION command 234
 GDS ISSUE ERROR command 236
 GDS ISSUE PREPARE command 238
 GDS ISSUE SIGNAL command 240
 GDS RECEIVE command 242
 GDS SEND command 245
 GDS WAIT command 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION command 299
 ISSUE ERROR command 315
 ISSUE PREPARE command 322
 ISSUE SIGNAL (APPC) command 336
 ISSUE SIGNAL (LUTYPE6.1) command 338
 POINT command 369
 RECEIVE (VTAM) command 439
 SEND (VTAM) command 515
 WAIT CONVID command 639
 WAIT TERMINAL command 650

copy displayed information 772

copybooks
 DFHAID 784
 DFHBMSCA 779
 DFHEIBLK 12
 DFHMSRCA 782

COUNTER option
 DEFINE COUNTER command 115
 DEFINE DCOUNTER command 115
 DELETE COUNTER command 140
 GET COUNTER command 257
 QUERY COUNTER command 384
 REWIND COUNTER command 487
 UPDATE COUNTER command 633

COUNTRY option
 EXTRACT CERTIFICATE command 193

COUNTRYLEN option
 EXTRACT CERTIFICATE command 193

CPI communications (SAA) 774
 create a journal record 341, 726

CTLCHAR option
 CONVERSE (non-VTAM) command 103
 CONVERSE (VTAM) command 96
 ISSUE COPY (3270 logical) command 301
 SEND (non-VTAM) command 522
 SEND (VTAM) command 516

CTRL operand
 DFHMDI 801
 DFHMSD 810

CURRENT option
 SEND PAGE command 542

CURSLOC operand
 DFHMDI 802
 DFHMSD 811

cursor address 772

CURSORS option
 RECEIVE MAP MAPPINGDEV command 455
 SEND CONTROL command 526
 SEND MAP command 531
 SEND MAP MAPPINGDEV command 538
 SEND TEXT command 547

cursor position
 terminal control 772

CVDA (CICS-value data area)
 argument values 3
 command format 3
 passing and receiving 16

CVDA options

ACTION
 WRITE OPERATOR command 732

ALTER
 QUERY SECURITY command 387

ASRAKEY
 ASSIGN command 48

ASRASPC
 ASSIGN command 49

CONTROL
 QUERY SECURITY command 387

LOGMESSAGE
 QUERY SECURITY command 387

MAXLIFETIME
 DEQ 150
 ENQ 175

PURGEABILITY
 WAIT EXTERNAL 644
 WAITCICS 653

CVDA options (continued)

READ
 QUERY SECURITY command 387

STATE 97, 105, 440, 449, 517, 523
 ALLOCATE (APPC) 39
 ALLOCATE (MRO) 43
 CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 EXTRACT ATTRIBUTES (MRO) 190
 FREE (APPC) 211
 FREE (MRO) 214
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 226
 GDS FREE 230
 GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 234
 GDS ISSUE ERROR 236
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243
 GDS SEND 246
 GDS WAIT 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION 299
 ISSUE ERROR 315
 ISSUE PREPARE 322
 ISSUE SIGNAL (APPC) 336
 WAIT CONVID 639

UPDATE
 QUERY SECURITY command 389

CVDA values

ALLOCATED
 ALLOCATE (APPC) 39
 ALLOCATE (MRO) 43
 CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 EXTRACT ATTRIBUTES (MRO) 190
 FREE (APPC) 211
 FREE (MRO) 214
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 226
 GDS FREE 230
 GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 234
 GDS ISSUE ERROR 236
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243
 GDS SEND 246
 GDS WAIT 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION 299
 ISSUE ERROR 315
 ISSUE PREPARE 322
 ISSUE SIGNAL (APPC) 336
 RECEIVE (MRO) command 449
 RECEIVE (VTAM) command 440
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517

CVDA values *(continued)*

ALLOCATED *(continued)*
 WAIT CONVID 639
 ALTERABLE
 QUERY SECURITY command 387
 BASESPACE
 ASSIGN command 49
 CHUNKEND
 WEB SEND command (Client) 708
 WEB SEND command (Server) 700
 CHUNKNO
 WEB SEND command (Client) 707
 WEB SEND command (Server) 700
 CHUNKYES
 WEB SEND command (Client) 707
 WEB SEND command (Server) 700
 CICSEXECKEY
 ASSIGN command 48
 CLICONVERT
 WEB CONVERSE command 664
 WEB RECEIVE command (Client) 694
 WEB SEND command (Client) 708
 CLOSE
 WEB CONVERSE command 659
 WEB SEND command (Client) 708
 WEB SEND command (Server) 701
 CONFFREE
 CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 FREE (APPC) 211
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 226
 GDS FREE 230
 GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 234
 GDS ISSUE ERROR 236
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243
 GDS SEND 246
 GDS WAIT 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION 299
 ISSUE ERROR 315
 ISSUE PREPARE 322
 ISSUE SIGNAL (APPC) 336
 RECEIVE (VTAM) command 440
 SEND (VTAM) command 517
 WAIT CONVID 639
 CONFRECEIVE
 CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 FREE (APPC) 211
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 226
 GDS FREE 230
 GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 234
 GDS ISSUE ERROR 236

CVDA values *(continued)*

CONFRECEIVE *(continued)*
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243
 GDS SEND 246
 GDS WAIT 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION 299
 ISSUE ERROR 315
 ISSUE PREPARE 322
 ISSUE SIGNAL (APPC) 336
 RECEIVE (VTAM) command 440
 SEND (VTAM) command 517
 WAIT CONVID 639
 CONFSEND
 CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 FREE (APPC) 211
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 226
 GDS FREE 230
 GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 234
 GDS ISSUE ERROR 236
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243
 GDS SEND 246
 GDS WAIT 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION 299
 ISSUE ERROR 315
 ISSUE PREPARE 322
 ISSUE SIGNAL (APPC) 336
 RECEIVE (VTAM) command 440
 SEND (VTAM) command 517
 WAIT CONVID 639
 CRITICAL
 WRITE OPERATOR command 732
 CTRLABLE
 QUERY SECURITY command 387
 DELETE
 WEB CONVERSE command 660, 710
 EVENTUAL
 WEB SEND command (Server) 699
 WRITE OPERATOR command 732
 EXPECT
 WEB CONVERSE command 658
 WEB SEND command (Client) 706
 FREE
 CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 EXTRACT ATTRIBUTES (MRO) 190
 FREE (APPC) 211
 FREE (MRO) 214
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 226
 GDS FREE 230

CVDA values (continued)

FREE (continued)

GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 234
 GDS ISSUE ERROR 236
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243
 GDS SEND 246
 GDS WAIT 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION 299
 ISSUE ERROR 315
 ISSUE PREPARE 322
 ISSUE SIGNAL (APPC) 336
 RECEIVE (MRO) command 449
 RECEIVE (VTAM) command 441
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517
 WAIT CONVID 639

GET

WEB CONVERSE command 660
 WEB SEND command (Client) 710

HEAD

WEB CONVERSE command 660
 WEB SEND command (Client) 710

HTTP

WEB EXTRACT command 671
 WEB OPEN command 675

HTTPNO

WEB EXTRACT command 671
 WEB RECEIVE command (Server) 691

HTTPS

WEB EXTRACT command 671
 WEB OPEN command 675

HTTPYES

WEB EXTRACT command 671
 WEB RECEIVE command (Server) 691

IMMEDIATE

WEB SEND command (Server) 699
 WRITE OPERATOR command 732

LOG

QUERY SECURITY command 387

NOCLICONVERT

WEB CONVERSE command 664
 WEB RECEIVE command (Client) 694
 WEB SEND command (Client) 709

NOCLOSE

WEB CONVERSE command 659
 WEB SEND command (Client) 708
 WEB SEND command (Server) 701

NOINCONVERT

WEB CONVERSE command 664

NOLOG

QUERY SECURITY command 387

NONCICS

ASSIGN command 48

NOOUTCONVERT

WEB CONVERSE command 664

NOSRVCONVERT

WEB RECEIVE command (Server) 691

CVDA values (continued)

NOSRVCONVERT (continued)

WEB SEND command (Server) 703
 NOTALTERABLE
 QUERY SECURITY command 387
 NOTAPPLIC
 ASSIGN command 48, 49
 NOTCTRLABLE
 QUERY SECURITY command 387
 NOTPURGEABLE
 WAIT EXTERNAL 644
 WAITCICS 653
 NOTREADABLE
 QUERY SECURITY command 387
 NOTUPDATABLE
 QUERY SECURITY command 389
 OPTIONS
 WEB CONVERSE command 660
 WEB SEND command (Client) 710

PENDFREE

CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 EXTRACT ATTRIBUTES (MRO) 190
 FREE (APPC) 211
 FREE (MRO) 214
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 226
 GDS FREE 230
 GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 234
 GDS ISSUE ERROR 236
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243
 GDS SEND 246
 GDS WAIT 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION 299
 ISSUE ERROR 315
 ISSUE PREPARE 322
 ISSUE SIGNAL (APPC) 336
 RECEIVE (MRO) command 449
 RECEIVE (VTAM) command 441
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517
 WAIT CONVID 639

PENDRECEIVE

CONNECT PROCESS 84
 EXTRACT ATTRIBUTES (APPC) 188
 FREE (APPC) 211
 GDS ALLOCATE 220
 GDS CONNECT PROCESS 224
 GDS EXTRACT ATTRIBUTES 227
 GDS FREE 230
 GDS ISSUE ABEND 232
 GDS ISSUE CONFIRMATION 235
 GDS ISSUE ERROR 236
 GDS ISSUE PREPARE 238
 GDS ISSUE SIGNAL 240
 GDS RECEIVE 243

CVDA values (continued)

PENDRECEIVE (continued)

GDS SEND 246
GDS WAIT 248
ISSUE ABEND command 293
ISSUE CONFIRMATION 299
ISSUE ERROR 315
ISSUE PREPARE 322
ISSUE SIGNAL (APPC) 336
RECEIVE (VTAM) command 441
SEND (VTAM) command 517
WAIT CONVID 639

PURGEABLE

WAIT EXTERNAL 644
WAITCICS 653

PUT

WEB CONVERSE command 660
WEB SEND command (Client) 710

READABLE

QUERY SECURITY command 387

RECEIVE

CONNECT PROCESS 84
CONVERSE command (non-VTAM) 105
EXTRACT ATTRIBUTES (APPC) 188
EXTRACT ATTRIBUTES (MRO) 190
FREE (APPC) 211
FREE (MRO) 214
GDS ALLOCATE 220
GDS CONNECT PROCESS 224
GDS EXTRACT ATTRIBUTES 227
GDS FREE 231
GDS ISSUE ABEND 232
GDS ISSUE CONFIRMATION 235
GDS ISSUE ERROR 236
GDS ISSUE PREPARE 238
GDS ISSUE SIGNAL 240
GDS RECEIVE 243
GDS SEND 246
GDS WAIT 248
ISSUE ABEND command 293
ISSUE CONFIRMATION 299
ISSUE ERROR 315
ISSUE PREPARE 322
ISSUE SIGNAL (APPC) 336
RECEIVE (MRO) command 449
RECEIVE (VTAM) command 441
SEND (non-VTAM) command 523
SEND (VTAM) command 517
WAIT CONVID 639

RFC1123

FORMATTIME command 207

ROLLBACK

CONNECT PROCESS 84
CONVERSE command (non-VTAM) 105
EXTRACT ATTRIBUTES (APPC) 188
EXTRACT ATTRIBUTES (MRO) 190
FREE (APPC) 211
FREE (MRO) 214
GDS ALLOCATE 220
GDS CONNECT PROCESS 224
GDS EXTRACT ATTRIBUTES 227

CVDA values (continued)

ROLLBACK (continued)

GDS FREE 231
GDS ISSUE ABEND 232
GDS ISSUE CONFIRMATION 235
GDS ISSUE ERROR 237
GDS ISSUE PREPARE 239
GDS ISSUE SIGNAL 241
GDS RECEIVE 243
GDS SEND 246
GDS WAIT 249
ISSUE ABEND command 293
ISSUE CONFIRMATION 299
ISSUE ERROR 315
ISSUE PREPARE 322
ISSUE SIGNAL (APPC) 336
RECEIVE (MRO) command 449
RECEIVE (VTAM) command 441
SEND (non-VTAM) command 523
SEND (VTAM) command 517
WAIT CONVID 639

SEND

CONNECT PROCESS 84
CONVERSE command (non-VTAM) 105
EXTRACT ATTRIBUTES (APPC) 188
EXTRACT ATTRIBUTES (MRO) 190
FREE (APPC) 211
FREE (MRO) 214
GDS ALLOCATE 220
GDS CONNECT PROCESS 224
GDS EXTRACT ATTRIBUTES 227
GDS FREE 231
GDS ISSUE ABEND 233
GDS ISSUE CONFIRMATION 235
GDS ISSUE ERROR 237
GDS ISSUE PREPARE 239
GDS ISSUE SIGNAL 241
GDS RECEIVE 243
GDS SEND 246
GDS WAIT 249
ISSUE ABEND command 293
ISSUE CONFIRMATION 299
ISSUE ERROR 315
ISSUE PREPARE 322
ISSUE SIGNAL (APPC) 336
RECEIVE (MRO) command 449
RECEIVE (VTAM) command 441
SEND (non-VTAM) command 523
SEND (VTAM) command 517
WAIT CONVID 639

SRVCONVERT

WEB RECEIVE command (Server) 690
WEB SEND command (Server) 702

SUBSPACE

ASSIGN command 49

SYNCFREE

CONNECT PROCESS 84
CONVERSE command (non-VTAM) 105
EXTRACT ATTRIBUTES (APPC) 188
EXTRACT ATTRIBUTES (MRO) 190
FREE (APPC) 211

CVDA values (continued)

SYNCFREE (continued)

FREE (MRO) 214
GDS ALLOCATE 220
GDS CONNECT PROCESS 224
GDS EXTRACT ATTRIBUTES 227
GDS FREE 231
GDS ISSUE ABEND 233
GDS ISSUE CONFIRMATION 235
GDS ISSUE ERROR 237
GDS ISSUE PREPARE 239
GDS ISSUE SIGNAL 241
GDS RECEIVE 243
GDS SEND 246
GDS WAIT 249
ISSUE ABEND command 293
ISSUE CONFIRMATION 299
ISSUE ERROR 315
ISSUE PREPARE 322
ISSUE SIGNAL (APPC) 336
RECEIVE (MRO) command 449
RECEIVE (VTAM) command 441
SEND (non-VTAM) command 523
SEND (VTAM) command 517
WAIT CONVID 639

SYNCRECEIVE

CONNECT PROCESS 84
CONVERSE command (non-VTAM) 105
EXTRACT ATTRIBUTES (APPC) 188
EXTRACT ATTRIBUTES (MRO) 190
FREE (APPC) 211
FREE (MRO) 214
GDS ALLOCATE 220
GDS CONNECT PROCESS 224
GDS EXTRACT ATTRIBUTES 227
GDS FREE 231
GDS ISSUE ABEND 233
GDS ISSUE CONFIRMATION 235
GDS ISSUE ERROR 237
GDS ISSUE PREPARE 239
GDS ISSUE SIGNAL 241
GDS RECEIVE 243
GDS SEND 246
GDS WAIT 249
ISSUE ABEND command 293
ISSUE CONFIRMATION 299
ISSUE ERROR 316
ISSUE PREPARE 322
ISSUE SIGNAL (APPC) 336
RECEIVE (MRO) command 449
RECEIVE (VTAM) command 441
SEND (non-VTAM) command 523
SEND (VTAM) command 517
WAIT CONVID 639

SYNCSEND

CONNECT PROCESS 84
CONVERSE command (non-VTAM) 105
EXTRACT ATTRIBUTES (APPC) 188
EXTRACT ATTRIBUTES (MRO) 190
FREE (APPC) 212
FREE (MRO) 214

CVDA values (continued)

SYNCSEND (continued)

GDS ALLOCATE 220
GDS CONNECT PROCESS 224
GDS EXTRACT ATTRIBUTES 227
GDS FREE 231
GDS ISSUE ABEND 233
GDS ISSUE CONFIRMATION 235
GDS ISSUE ERROR 237
GDS ISSUE PREPARE 239
GDS ISSUE SIGNAL 241
GDS RECEIVE 243
GDS SEND 246
GDS WAIT 249
ISSUE ABEND command 293
ISSUE CONFIRMATION 299
ISSUE ERROR 316
ISSUE PREPARE 322
ISSUE SIGNAL (APPC) 336
RECEIVE (MRO) command 449
RECEIVE (VTAM) command 441
SEND (non-VTAM) command 523
SEND (VTAM) command 517
WAIT CONVID 639

TASK

DEQ 150
ENQ 175

TRACE

WEB CONVERSE command 660
WEB SEND command (Client) 710

UOW

DEQ 150
ENQ 175

UPDATABLE

QUERY SECURITY command 389

USEREXECKEY

ASSIGN command 48

CWA option

ADDRESS command 35

CWALENG option

ASSIGN command 50

D

data

passing to new tasks 591

DATA operand

DFHMDI 802

DFHMSD 811

DATA option

FREEMAIN command 216

data sets

add records to 297

interrogating 325

processing termination 306

read records from 327

update records 329

data tables

CICS/user-maintained/coupling facility

DELETE command 128

ENDBR command 168

- data tables *(continued)*
 - CICS/user-maintained/coupling facility *(continued)*
 - READ command 391
 - READNEXT command 403
 - READPREV command 414
 - RESETBR command 468
 - REWRITE command 490
 - STARTBR command 608
 - UNLOCK command 628
 - WRITE command 719
- data to output device, sending 333
- data-area argument
 - CICS command format 3
 - definition 1
- data-value argument
 - CICS command format 3
 - definition 1
- data, deleting
 - file control records 128
 - named counter 140
 - temporary storage queues 147
 - transient data queues 145
- DATA1 option
 - MONITOR command 360
- DATA2 option
 - MONITOR command 360
- DATALENGTH option
 - INQUIRE CONTAINER command 284
 - LINK command 344
- DATAONLY option
 - DOCUMENT RETRIEVE command 158
 - SEND MAP command 532
 - SEND MAP MAPPINGDEV command 539
- DATAPointer option
 - FREEMAIN command 217
- DATAREG operand 14
- DATASTR option
 - BUILD ATTACH (LUTYPE6.1) command 62
 - BUILD ATTACH (MRO) command 65
 - EXTRACT ATTACH (LUTYPE6.1) command 180
 - EXTRACT ATTACH (MRO) command 184
- DATATYPE option
 - PUT CONTAINER (CHANNEL) command 381
- DATE option
 - FORMATTIME command 206
- DATEFORM option
 - FORMATTIME command 206
- DATESEP option
 - FORMATTIME command 206
- DATESTRING option
 - CONVERTTIME command 107
 - FORMATTIME command 206
- DAYCOUNT option
 - FORMATTIME command 206
- DAYOFMONTH option
 - DEFINE TIMER command 123
 - FORMATTIME command 206
- DAYOFWEEK option
 - FORMATTIME command 207
- DAYOFYEAR option
 - DEFINE TIMER command 123
- DAYS option
 - DEFINE TIMER command 123
- DAYSLEFT option
 - VERIFY PASSWORD command 637
- DCOUNTER option
 - DELETE DCOUNTER command 140
 - GET DCOUNTER command 257
 - QUERY DCOUNTER command 384
 - REWIND DCOUNTER command 487
 - UPDATE DCOUNTER command 633
- DDMMYY option
 - FORMATTIME command 207
- DDMMYYYY option
 - FORMATTIME command 207
- DEBKEY option
 - READ command 393
 - STARTBR command 609
- DEBREC option
 - READ command 393
 - STARTBR command 609
- DEFAULT option
 - CONVERSE (non-VTAM) command 103
 - CONVERSE (VTAM) command 96
 - SEND (non-VTAM) command 522
 - SEND (VTAM) command 516
 - SEND CONTROL command 526
 - SEND MAP command 532
 - SEND TEXT command 547
 - SEND TEXT NOEDIT command 557
- DEFINE ACTIVITY command 109
- DEFINE COMPOSITE EVENT command 112
- DEFINE COUNTER command 114
- DEFINE DCOUNTER command 114
- DEFINE INPUT EVENT command 118
- DEFINE PROCESS command 119
- DEFINE TIMER command 122
- DEFRESP option
 - CONVERSE (non-VTAM) command 103
 - CONVERSE (VTAM) command 96
 - ISSUE ADD command 297
 - ISSUE ERASE command 311
 - ISSUE REPLACE command 329
 - ISSUE SEND command 333
 - SEND (non-VTAM) command 522
 - SEND (VTAM) command 516
- DEFSCRNHT option
 - ASSIGN command 50
- DEFSCRNWD option
 - ASSIGN command 50
- DELAY command 125
- delay processing, task 125
- DELETE ACTIVITY command 135
- DELETE command 128
- DELETE CONTAINER (BTS) command 137
- DELETE CONTAINER (CHANNEL) command 139
- DELETE COUNTER command 140
- DELETE DCOUNTER command 140
- DELETE EVENT command 142
- delete loaded program 461
- DELETE option
 - SPOOLCLOSE command 572

- delete records
 - batch data interchange records 311
- DELETE TIMER command 144
- DELETEQ TD command 145
- DELETEQ TS command 147
- deleting data
 - named counter 140
 - temporary storage queues 147
 - transient data queues 145
- DELIMITER option
 - ASSIGN command 50
- DEQ command 149
- dequeue from resource 149
- DEST option
 - CONVERSE (non-VTAM) command 103
 - SEND (non-VTAM) command 522
- DESTCOUNT option
 - ASSIGN command 50
- DESTID option
 - ASSIGN command 51
 - ISSUE ABORT command 295
 - ISSUE ADD command 297
 - ISSUE END command 306
 - ISSUE ERASE command 311
 - ISSUE NOTE command 318
 - ISSUE QUERY command 325
 - ISSUE REPLACE command 329
 - ISSUE SEND command 334
 - ISSUE WAIT command 339
- DESTIDLENG option
 - ASSIGN command 51
 - ISSUE ABORT command 295
 - ISSUE ADD command 297
 - ISSUE END command 306
 - ISSUE ERASE command 311
 - ISSUE NOTE command 318
 - ISSUE QUERY command 325
 - ISSUE REPLACE command 329
 - ISSUE SEND command 334
 - ISSUE WAIT command 339
- destruction of activities 135
- DET value
 - DFHMDF 790
- DFH2980 structure 445
- DFHAID attention identifier list 784
- DFHBMSCA, standard attribute and printer control
 - character list, BMS 779
- DFHEAI interface processor 12
- DFHECALL macro 11
- DFHEIBLK copybook 12
- DFHEICAL macro, use DFHECALL 11
- DFHEIEND macro 11
- DFHEIENT macro
 - CODEREG 14
 - DATAREG 14
 - defaults 14
 - description 11
 - EIBREG 14
- DFHEIGBL macro 11
- DFHEIPLR symbolic register 15
- DFHEIRET macro 12
- DFHEISTG macro 12
- DFHMDF macro 788
- DFHMDI macro 798
- DFHMIRS 346
- DFHMUSD macro 807
- DFHMSRCA, MSR control value constants 782
- DFHPDI macro 818
- DFHPSD macro 819
- DFHRESP, built-in function 9
- DFHVALUE, translator routine 16
- diagnostic services commands 21
- DISABLED condition
 - DELETE command 130
 - DELETEQ TD command 145
 - READ command 397
 - READQ TD command 425
 - STARTBR command 611
 - UNLOCK command 629
 - WRITE command 721
 - WRITEQ TD command 734
- disconnect a switched line 769
- display-device operations
 - attention identifier (AID) 772
 - attention identifier list, DFHAID 784
 - copy displayed information 772
 - cursor address 772
 - erase all unprotected fields 772
 - input operation without data 772
 - pass control on receipt of an AID 275, 279
 - print displayed information 771
 - standard attribute and printer control character list, DFHBMSCA 779
 - terminal 771
- distributed program link (DPL) 774
- DOCSIZE option
 - DOCUMENT INSERT command 156
- DOCTOKEN option
 - DOCUMENT RETRIEVE command 158
 - DOCUMENT SET command 161
 - WEB CONVERSE command 659
 - WEB RETRIEVE command 697
 - WEB SEND command (Client) 709
 - WEB SEND command (Server) 701
- document
 - adding symbols to symbol table 160
 - creating 151
 - DOCUMENT CREATE command 151
 - DOCUMENT INSERT command 155
 - DOCUMENT option
 - DOCUMENT INSERT command 156
 - DOCUMENT RETRIEVE command 158
 - document services
 - commands 21
 - DOCUMENT SET command 160
 - DORMANT mode, of an activity 281
 - DPL, distributed program link 774
 - DRK value
 - DFHMDF 791
 - DS3270 option
 - ASSIGN command 51

- DSATTS operand
 - DFHMDI 803
 - DFHMSD 812
- DSECT operand
 - DFHMSD 812
- DSECT value
 - DFHMSD 816
- DSSCS option
 - ASSIGN command 51
- DSSTAT condition
 - ISSUE RECEIVE command 328
- DUMP TRANSACTION command 163
- DUMPCODE option
 - DUMP TRANSACTION command 163
- DUMPID option
 - DUMP TRANSACTION command 163
- DUPKEY condition
 - DELETE command 130
 - READ command 397
 - READNEXT command 410
 - READPREV command 420
- DUPREC condition
 - REWRITE command 491
 - WRITE command 721
- dynamic allocation 577
- dynamic storage, extensions 13

E

- ECADDR option
 - WAIT EVENT command 641
- ECBLIST option
 - WAIT EXTERNAL command 644
 - WAITCICS command 652
- EDF, execution diagnostic facility 596, 606
- EIB fields
 - EIBAID 745
 - EIBATT 745
 - EIBCALEN 745
 - EIBCOMPL 745
 - EIBCONF 746
 - EIBCPOSN 746
 - EIBDATE 746
 - EIBDS 746
 - EIBEOC 747
 - EIBERR 747
 - EIBERRCD 747
 - EIBFMH 747
 - EIBFN 747
 - EIBFREE 752
 - EIBNODAT 752
 - EIBRCODE 752
 - EIBRECV 758
 - EIBREQID 759
 - EIBRESP 759
 - EIBRESP2 760
 - EIBRLDBK 760
 - EIBRSRCE 761
 - EIBSIG 761
 - EIBSYNC 761
 - EIBSYNRB 761

- EIB fields (*continued*)
 - EIBTASKN 762
 - EIBTIME 762
 - EIBTRMID 762
 - EIBTRNID 762
- EIB option
 - ADDRESS command 35
- EIBAID 772
 - examining contents of field 784
- EIBREG operand 14
- END condition
 - GETNEXT ACTIVITY command 266
 - GETNEXT CONTAINER command 267
 - GETNEXT EVENT command 270
 - GETNEXT PROCESS command 271
 - RETRIEVE REATTACH EVENT command 480
 - RETRIEVE SUBEVENT command 482
- ENDACTIVITY option
 - RETURN command 484
- ENDBR command 168
- ENDBROWSE ACTIVITY command 170
- ENDBROWSE CONTAINER command 171
- ENDBROWSE EVENT command 172
- ENDBROWSE PROCESS command 173
- ENDDATA condition
 - RETRIEVE command 477
- ENDFILE condition
 - READNEXT command 410
 - READPREV command 420
 - SPOOLREAD command 582
 - WEB READNEXT FORMFIELD command 684
 - WEB READNEXT HTTPHEADER command 686
- ENDFILE option
 - ISSUE ENDOUTPUT command 309
- ENDINPT condition
 - RECEIVE (non-VTAM) command 450
- ENDOUTPUT option
 - ISSUE ENDFILE command 308
- English and katakana characters, mixed 102, 452, 459
- ENQ command 174
- ENQBUSY condition
 - ENQ command 176
- ensuring terminal operation has completed 650
- ENTER option
 - HANDLE AID command 275
- ENTER TRACEID command
 - monitoring aspects replaced by MONITOR command 360
 - tracing aspects replaced by ENTER TRACENUM command 178
- ENTER TRACENUM command 178
- ENTRY option
 - LOAD command 357
- entry to assembler-language program 11
- ENTRYNAME option
 - MONITOR command 361
- ENVDEFERR condition
 - RETRIEVE command 477
- environment services
 - commands 21

EOC condition
 ALLOCATE (LUTYPE6.1) command 42
 CONVERSE (non-VTAM) command 106
 CONVERSE (VTAM) command 99
 ISSUE RECEIVE command 328
 RECEIVE (non-VTAM) command 450
 RECEIVE (VTAM) command 441
 RECEIVE MAP command 453
 RECEIVE PARTN command 459
 WAIT TERMINAL command 650
 EODS condition
 CONVERSE (VTAM) command 99
 ISSUE RECEIVE command 328
 RECEIVE (VTAM) command 441
 RECEIVE MAP command 453
 RECEIVE PARTN command 460
 EOF condition
 CONVERSE (non-VTAM) command 106
 RECEIVE (non-VTAM) command 450
 EQUAL option
 READ command 393
 RESETBR command 469
 STARTBR command 609
 equated symbols 7
 erase all unprotected fields 772
 ERASE option
 CONVERSE (non-VTAM) command 103
 CONVERSE (VTAM) command 96
 SEND (non-VTAM) command 522
 SEND (VTAM) command 516
 SEND CONTROL command 526
 SEND MAP command 532
 SEND MAP MAPPINGDEV command 539
 SEND TEXT command 547
 SEND TEXT NOEDIT command 557
 ERASEAUP option
 SEND CONTROL command 526
 SEND MAP command 532
 SEND MAP MAPPINGDEV command 539
 ERRTERM option
 ROUTE command 496
 ESDS (entry-sequenced data set)
 DELETE command 131
 READ 395
 STARTBR command 610
 WRITE command 719
 ESM
 ACEE pointer 34
 QUERY SECURITY command, NOTFND
 condition 389
 QUERY SECURITY command, RESCLASS
 option 388
 RESTYPE values 389
 USERNAME 58
 ESM, external security manager 596, 606
 ESMREASON option
 CHANGE PASSWORD command 73
 SIGNON command 562
 VERIFY PASSWORD command 637
 ESMRESP option
 CHANGE PASSWORD command 73
 ESMRESP option (*continued*)
 SIGNON command 562
 VERIFY PASSWORD command 637
 EVENT option
 ADD SUBEVENT command 32
 DEFINE ACTIVITY command 109
 DEFINE COMPOSITE EVENT command 113
 DEFINE INPUT EVENT command 118
 DEFINE TIMER command 123
 DELETE EVENT command 142
 GETNEXT EVENT command 269
 INQUIRE ACTIVITYID command 281
 INQUIRE EVENT command 285
 INQUIRE TIMER command 288
 REMOVE SUBEVENT command 463
 RETRIEVE REATTACH EVENT command 480
 RETRIEVE SUBEVENT command 481
 TEST EVENT command 627
 event-related commands
 CHECK TIMER 81
 DEFINE COMPOSITE EVENT 112
 DEFINE INPUT EVENT 118
 DEFINE TIMER 122
 DELETE EVENT 142
 DELETE TIMER 144
 ENDBROWSE EVENT 172
 FORCE TIMER 203
 GETNEXT EVENT 269
 INQUIRE EVENT 285
 INQUIRE TIMER 288
 REMOVE SUBEVENT 463
 RETRIEVE REATTACH EVENT 479
 RETRIEVE SUBEVENT 481
 STARTBROWSE EVENT 618
 TEST EVENT 627
 EVENTERR condition
 ADD SUBEVENT command 32
 DEFINE ACTIVITY command 110
 DEFINE COMPOSITE EVENT command 113
 DEFINE INPUT EVENT command 118
 DEFINE TIMER command 124
 DELETE EVENT command 142
 INQUIRE EVENT command 286
 LINK ACQPROCESS command 351
 LINK ACTIVITY command 354
 REMOVE SUBEVENT command 463
 RETRIEVE SUBEVENT command 482
 RUN command 503
 TEST EVENT command 627
 events, timer
 control area, timer 371
 monitoring point 360
 waiting for 641
 EVENTTYPE option
 GETNEXT EVENT command 269
 INQUIRE EVENT command 285
 RETRIEVE REATTACH EVENT command 480
 RETRIEVE SUBEVENT command 481
 EWASUPP option
 ASSIGN command 51

- examples
 - using the ABEND command 27
 - using the ADDRESS SET command 36
 - using the ASKTIME command 45
 - using the ASSIGN command 59
 - using the BIF DEEDIT command 60
 - using the DELAY command 126
 - using the DELETE command 134
 - using the DEQ command 150
 - using the DUMP TRANSACTION command 168
 - using the ENQ command 177
 - using the ENTER TRACENUM command 179
 - using the FORMATIME command 208
 - using the FREEMAIN command 217
 - using the GETMAIN command 264
 - using the HANDLE ABEND command 274
 - using the HANDLE AID command 276
 - using the HANDLE CONDITION command 278
 - using the LINK command 349
 - using the LOAD command 359
 - using the MONITOR command 361
 - using the POST command 374
 - using the READ command 402
 - using the READQ TD command 427
 - using the READQ TS command 430
 - using the RELEASE command 462
 - using the RETRIEVE command 477
 - using the REWRITE command 494
 - using the START ATTACH command 598
 - using the START BREXIT command 602
 - using the START command 590
 - using the WAIT EVENT command 642
 - using the WAIT EXTERNAL command 645
 - using the WAIT JOURNALNAME command 647
 - using the WAITCICS command 654
 - using the WRITE command 725
 - using the WRITE JOURNALNAME command 729
 - using the WRITEQ TD command 736
 - using the XCTL command 744
- EXCEPTION option
 - ENTER TRACENUM command 178
- exception support commands 22
- exclusive control release, UNLOCK command 628
- EXEC CICS command format 1
- execution diagnostic facility (EDF) 596, 606
- exit from ASM program 12
- exit, abnormal termination recovery 273
- expiration time, notification when reached 371
- EXPIRED condition
 - DELAY command 126
 - POST command 373
 - WRITE OPERATOR command 733
- EXPIRYTIME option
 - VERIFY PASSWORD command 637
- EXTATT operand
 - DFHMDI 803
 - DFHMSD 812
- EXTDS option
 - ASSIGN command 51
- external security manager (ESM) 387, 596, 606
- EXTRACT ATTACH (LUTYPE6.1) command 180

- EXTRACT ATTACH (MRO) command 184
- EXTRACT ATTRIBUTES (APPC) command 188
- EXTRACT ATTRIBUTES (MRO) command 190
- EXTRACT CERTIFICATE command 192
- EXTRACT LOGONMSG command 195
- EXTRACT PROCESS command 197
- EXTRACT TCPIP command 199
- EXTRACT TCT command 202

F

- FACILITY option
 - ASSIGN command 51
- FACILITYTKN option
 - RUN command 502
- FCI option
 - ASSIGN command 51, 765
- FCT option
 - DUMP TRANSACTION command 164
- field
 - extracting information 680
 - field definition macro, BMS 785
- FIELD option
 - BIF DEEDIT command 60
- field separator operand 803, 812
- FIELD value
 - DFHMDI 802
 - DFHMSD 811
- FIELDS operand
 - DFHMDI 803
- file control
 - commands 22
 - deleting VSAM records 128
 - end browse operation 168
 - read next record 403
 - read previous record 414
 - release exclusive control 628
 - specify start for browse 608
 - update a record 490
 - writing new record 719
- FILE option
 - DELETE command 128
 - ENDBR command 168
 - READ command 393
 - READNEXT command 406
 - READPREV command 417
 - RESETBR command 469
 - REWRITE command 490
 - STARTBR command 609
 - UNLOCK command 628
 - WRITE command 720
- filename
 - definition 4, 5, 6, 7
- filename argument, CICS command format 3
- FILENOTFOUND condition
 - DELETE command 130
 - ENDBR command 169
 - READ command 398
 - READNEXT command 410
 - READPREV command 420
 - RESETBR command 470

FILENOTFOUND condition (*continued*)
 REWRITE command 491
 STARTBR command 611
 UNLOCK command 629
 WRITE command 721
 FIRESTATUS option
 GETNEXT EVENT command 270
 INQUIRE EVENT command 285
 TEST EVENT command 627
 FIRST value
 DFHMDI 804
 FLDSEP operand
 DFHMDI 803
 DFHMSD 812
 FLENGTH option
 DUMP TRANSACTION command 164
 fullword alternative to LENGTH 767
 GDS RECEIVE command 242
 GDS SEND command 245
 GET CONTAINER (BTS) command 251
 GET CONTAINER (CHANNEL) command 253
 GETMAIN command 263
 LOAD command 357
 PUT CONTAINER (BTS) command 378
 PUT CONTAINER (CHANNEL) command 382
 RECEIVE (non-VTAM) command 448
 RECEIVE (VTAM) command 439
 SEND (non-VTAM) command 522
 SEND (VTAM) command 516
 SPOOLWRITE command 585
 WRITE JOURNALNAME command 726
 FMH option
 CONVERSE (non-VTAM) command 103
 CONVERSE (VTAM) command 96
 SEND (non-VTAM) command 522
 SEND (VTAM) command 516
 START command 593
 FMHPARM option
 SEND MAP command 532
 SEND PAGE command 542
 SEND TEXT command 548
 FOLD operand
 DFHMSD 812
 FOR option
 DELAY command 125
 FORCE TIMER command 203
 form field
 extracting information 680
 FORMATTIME command 205
 FORMFEED option
 SEND CONTROL command 526
 SEND MAP command 532
 SEND MAP MAPPINGDEV command 539
 SEND TEXT command 548
 FORMFIELD option
 WEB READ FORMFIELD command 680
 WEB READNEXT FORMFIELD command 684
 WEB STARTBROWSE FORMFIELD command 713
 FREE (APPC) command 211
 FREE (LUTYPE6.1) command 213
 FREE (MRO) command 214
 FREE command 210
 free main storage 216
 FREEKB option
 SEND CONTROL command 527
 SEND MAP command 532
 SEND MAP MAPPINGDEV command 539
 SEND TEXT command 548
 SEND TEXT NOEDIT command 557
 FREEKB value
 DFHMDI 801
 DFHMSD 810
 FREEMAIN command 216
 FROM option
 CONVERSE (non-VTAM) command 103
 CONVERSE (VTAM) command 96
 DUMP TRANSACTION command 164
 ENTER TRACENUM command 179
 GDS SEND command 246
 ISSUE ADD command 297
 ISSUE PASS command 320
 ISSUE REPLACE command 329
 ISSUE SEND command 334
 PUT CONTAINER (BTS) command 378
 PUT CONTAINER (CHANNEL) command 382
 RECEIVE MAP command 452
 RECEIVE MAP MAPPINGDEV command 456
 REWRITE command 490
 SEND (non-VTAM) command 522
 SEND (VTAM) command 516
 SEND MAP command 532
 SEND MAP MAPPINGDEV command 539
 SEND TEXT command 548
 SEND TEXT MAPPED command 553
 SEND TEXT NOEDIT command 557
 SPOOLWRITE command 585
 START ATTACHcommand 599
 START command 593
 WEB CONVERSE command 659
 WEB SEND command (Client) 709
 WEB SEND command (Server) 701
 WRITE command 720
 WRITE JOURNALNAME command 726
 WRITEQ TD command 734
 WRITEQ TS command 738
 FROMACTIVITY option
 MOVE CONTAINER (BTS) command 364
 FROMCCSID option
 PUT CONTAINER (CHANNEL) command 382
 FROMDOC option
 DOCUMENT INSERT command 156
 FROMFLENGTH option
 CONVERSE (non-VTAM) command 103
 CONVERSE (VTAM) command 96
 fullword alternative to FROMLENGTH 767
 FROMLENGTH option
 CONVERSE (non-VTAM) command 103
 CONVERSE (VTAM) command 96
 ENTER TRACENUM command 179
 fullword length alternative (FROMFLENGTH) 767
 WEB CONVERSE command 659
 WEB SEND command (Client) 709

FROMLENGTH option (*continued*)
 WEB SEND command (Server) 701
 FROMPROCESS option
 MOVE CONTAINER (BTS) command 364
 FRSET option
 SEND CONTROL command 527
 SEND MAP command 533
 SEND MAP MAPPINGDEV command 539
 FRSET value
 DFHMDI 801
 DFHMSD 810
 FSET value
 DFHMDF 791
 Full Function Logical Unit, 3790 94, 438, 513
 FULLDATE option
 FORMATTIME 207
 fullword length option 767
 FUNCERR condition
 ISSUE ABORT command 296
 ISSUE ADD command 298
 ISSUE END command 307
 ISSUE ERASE command 312
 ISSUE NOTE command 318
 ISSUE QUERY command 325
 ISSUE REPLACE command 330
 ISSUE SEND command 334
 ISSUE WAIT command 340

G

GCHARS option
 ASSIGN command 52
 GCODES option
 ASSIGN command 52
 GDS (generalized data stream) 19
 GDS ALLOCATE command 219
 GDS ASSIGN command 222
 GDS CONNECT PROCESS command 223
 GDS EXTRACT ATTRIBUTES command 226
 GDS EXTRACT PROCESS command 228
 GDS FREE command 230
 GDS ISSUE ABEND command 232
 GDS ISSUE CONFIRMATION command 234
 GDS ISSUE ERROR command 236
 GDS ISSUE PREPARE command 238
 GDS ISSUE SIGNAL command 240
 GDS RECEIVE command 242
 GDS SEND command 245
 GDS WAIT command 248
 General Banking Terminal System (2980 General
 Banking Terminal System) 444
 generalized data stream (GDS) 19
 generic applid, XRF 48
 GENERIC option
 DELETE command 129
 READ command 393
 RESETBR command 469
 STARTBR command 609
 GET CONTAINER (BTS) command 250
 GET CONTAINER (CHANNEL) command 253
 GET COUNTER command 256

GET DCOUNTER command 256
 get main storage 261
 GETMAIN command 261
 GETNEXT ACTIVITY command 265
 GETNEXT CONTAINER command 267
 GETNEXT EVENT command 269
 GETNEXT PROCESS command 271
 GINIT operand
 DFHMDF 792
 GMMI option
 ASSIGN command 52
 GROUPID option
 SIGNON command 562
 GRPNAME operand
 DFHMDF 792
 GTEQ option
 READ command 394
 RESETBR command 469
 STARTBR command 609

H

HANDLE ABEND command 273
 HANDLE AID command 275
 HANDLE CONDITION command 277
 header
 browsing 666, 713
 retrieve next 684
 HEADER operand
 DFHMDI 803
 HEADER option
 SEND TEXT command 548
 hhmms argument, CICS command format 3
 HILIGHT operand
 DFHMDF 792
 DFHMDI 803
 DFHMSD 812
 HILIGHT option
 ASSIGN command 52
 HOLD option
 LOAD command 357
 HONEOM option
 SEND CONTROL command 527
 SEND MAP command 533
 SEND TEXT command 548
 SEND TEXT NOEDIT command 557
 host command processor LU, 3650/3680 512
 host conversational LU 3650
 (3270) 92, 511
 (3653) 92, 511
 HOST option
 WEB EXTRACT command 669
 WEB OPEN command 674
 WEB PARSE URL command 677
 HOSTCODEPAGE option
 WEB READ FORMFIELD command 680
 WEB RECEIVE command (Server) 689
 WEB SEND command (Server) 701
 WEB STARTBROWSE FORMFIELD command 713
 HOSTLENGTH option
 WEB EXTRACT command 669

HOSTLENGTH option *(continued)*
 WEB OPEN command 674
 WEB PARSE URL command 677

HOURS option
 DEFINE TIMER command 123
 DELAY command 125
 POST command 372
 ROUTE command 496
 START command 593

HTAB operand
 DFHMSD 813

HTTPHEADER option
 WEB READ HTTPHEADER command 682
 WEB READNEXT HTTPHEADER command 686
 WEB WRITE HTTPHEADER command 717

HTTPMETHOD option
 WEB EXTRACT command 670

HTTPRNUM option
 WEB OPEN command 674

HTTPVERSION option
 WEB EXTRACT command 670

HTTPVNUM option
 WEB OPEN command 674

I

IC value
 DFHMDF 791

IGNORE CONDITION command 279

IGREQCD condition
 CONVERSE (VTAM) command 99
 ISSUE SEND command 334
 SEND (VTAM) command 518
 SEND CONTROL command 528
 SEND MAP command 536
 SEND PAGE command 543
 SEND TEXT command 551
 SEND TEXT MAPPED command 554
 SEND TEXT NOEDIT command 558

IGREQID condition
 ROUTE command 498
 SEND CONTROL command 529
 SEND MAP command 536
 SEND TEXT command 551
 SEND TEXT MAPPED command 554
 SEND TEXT NOEDIT command 558

ILLOGIC condition
 DELETE command 131
 EIBRCODE 758
 ENDBR command 169
 ENDBROWSE ACTIVITY command 170
 ENDBROWSE CONTAINER command 171
 ENDBROWSE PROCESS command 173
 GETNEXT ACTIVITY command 266
 GETNEXT CONTAINER command 267
 GETNEXT PROCESS command 271
 INQUIRE PROCESS command 287
 READ command 398
 READNEXT command 410
 READPREV command 420
 RESETBR command 470

ILLOGIC condition *(continued)*
 REWRITE command 492
 SPOOLOPEN INPUT command 575
 SPOOLOPEN OUTPUT command 580
 SPOOLREAD command 583
 STARTBR command 611
 UNLOCK command 629
 WEB EXTRACT command 672
 WEB STARTBROWSE HTTPHEADER
 command 715
 WRITE command 721

IMMEDIATE option
 RETURN command 484

implicit SPOOLCLOSE 574

INBFMH condition
 CONVERSE (non-VTAM) command 106
 CONVERSE (VTAM) command 99
 RECEIVE (non-VTAM) command 450
 RECEIVE (VTAM) command 441

INCREMENT option
 GET COUNTER command 257
 GET DCOUNTER command 257
 REWIND COUNTER command 487
 REWIND DCOUNTER command 487

INITIAL mode, of an activity 281

INITIAL operand
 DFHMDF 793

initialize main storage 261

initiate a task 591

INITIMG option
 GETMAIN command 263

INITPARM option
 ASSIGN command 52

INITPARMLEN option
 ASSIGN command 52

INPARTN option
 ASSIGN command 52
 RECEIVE MAP command 452

input operation without data 772

INPUTEVENT option
 LINK ACQPROCESS command 351
 LINK ACTIVITY command 354
 RUN command 502

INPUTMSG option
 LINK command 344
 RETURN command 485
 XCTL command 743

INPUTMSGLEN option
 LINK command 345
 RETURN command 485
 XCTL command 743

INQUIRE ACTIVITYID command 280

INQUIRE CONTAINER command 283

INQUIRE EVENT command 285

INQUIRE PROCESS command 287

INQUIRE TIMER command 288

interactive logical units 93, 437, 512

interface processor DFHEAI 12

interpreter logical unit, 3650
 CONVERSE command 91
 ISSUE EODS command 310

interpreter logical unit, 3650 *(continued)*
 ISSUE LOAD command 317
 RECEIVE command 436
 SEND (VTAM) command 510
 interrogate a data set 325
 interval control
 ASKTIME options 45
 cancel interval control command 68
 CANCEL options 69
 commands 22
 DELAY options 125
 delay processing of task 125
 FORMATTIME options 206
 notification when specified time expires 371
 request current time of day 45
 retrieve data stored for task 475
 start a task 588
 wait for event to occur 641
 INTERVAL option
 DELAY command 125
 POST command 372
 ROUTE command 496
 START command 593
 INTO option
 CONVERSE (non-VTAM) command 104
 CONVERSE (VTAM) command 96
 DOCUMENT RETRIEVE command 158
 EXTRACT LOGONMSG command 195
 GDS RECEIVE command 243
 GET CONTAINER (BTS) command 251
 GET CONTAINER (CHANNEL) command 254
 ISSUE RECEIVE command 327
 READ command 394
 READNEXT command 406
 READPREV command 417
 READQ TD command 424
 READQ TS command 428
 RECEIVE (non-VTAM) command 448
 RECEIVE (VTAM) command 439
 RECEIVE MAP command 452
 RECEIVE MAP MAPPINGDEV command 456
 RECEIVE PARTN command 459
 RETRIEVE command 476
 SPOOLREAD command 582
 WEB CONVERSE command 661
 WEB RECEIVE command (Client) 694
 WEB RECEIVE command (Server) 689
 INTOCCSID option
 GET CONTAINER (CHANNEL) command 254
 INVALIDCOUNT option
 VERIFY PASSWORD command 637
 INVERRTERM condition
 ROUTE command 498
 INVITE option
 GDS SEND command 246
 SEND (non-VTAM) command 522
 SEND (VTAM) command 516
 INVLDC condition
 ROUTE command 498
 SEND CONTROL command 529
 SEND MAP command 536
 INVLDC condition *(continued)*
 SEND TEXT command 551
 INVMP SZ condition
 EIBRCODE byte 3 758
 RECEIVE MAP command 453
 RECEIVE MAP MAPPINGDEV command 457
 SEND MAP command 536
 SEND MAP MAPPINGDEV command 540
 INVOKE WEBSERVICE command 290
 INVOKINGPROG option
 ASSIGN command 52
 INVPARTN condition
 RECEIVE MAP command 453
 RECEIVE PARTN command 460
 SEND CONTROL command 529
 SEND MAP command 536
 SEND TEXT command 551
 SEND TEXT NOEDIT command 559
 INVPARTNSET condition
 SEND PARTNSET command 545
 INVREQ condition
 ACQUIRE command 30
 ADD SUBEVENT command 32
 ALLOCATE (APPC) command 39
 ALLOCATE (LUTYPE6.1) command 42
 ALLOCATE (MRO) command 44
 ASSIGN command 58
 CANCEL (BTS) command 70
 CHANGE PASSWORD command 74
 CHANGE TASK command 75
 CHECK ACQPROCESS command 77
 CHECK ACTIVITY command 80
 CHECK TIMER command 81
 CONNECT PROCESS command 84
 CONVERSE (VTAM) command 99
 CONVERTTIME command 108
 DEFINE ACTIVITY command 110
 DEFINE COMPOSITE EVENT command 113
 DEFINE INPUT EVENT command 118
 DEFINE PROCESS command 120
 DEFINE TIMER command 124
 DELAY command 126
 DELETE ACTIVITY command 135
 DELETE command 131
 DELETE CONTAINER (BTS) command 138
 DELETE CONTAINER (CHANNEL) command 139
 DELETE COUNTER command 116, 141, 385
 DELETE DCOUNTER command 116
 DELETE EVENT command 142
 DELETE TIMER command 144
 DELETEQ TD command 145
 DELETEQ TS command 147
 DEQ command 150
 DUMP TRANSACTION command 166
 EIBRCODE bytes 1-3 757
 ENDBR command 169
 ENQ command 176
 ENTER TRACENUM command 179
 EXTRACT ATTACH (LUTYPE6.1) command 183
 EXTRACT ATTACH (MRO) command 187
 EXTRACT ATTRIBUTES (APPC) command 189

INVREQ condition *(continued)*

EXTRACT CERTIFICATE command 194
 EXTRACT PROCESS command 198
 EXTRACT TCPIP command 201
 EXTRACT TCT command 202
 FORCE TIMER command 203
 FORMATTIME command 208
 FREE (APPC) command 212
 FREE (LUTYPE6.1) command 213
 FREE (MRO) command 215
 FREEMAIN command 217
 GET CONTAINER (BTS) command 251
 GET CONTAINER (CHANNEL) command 255
 HANDLE AID command 276
 INQUIRE EVENT command 286
 INQUIRE TIMER command 289
 ISSUE ABEND command 294
 ISSUE ABORT command 296
 ISSUE ADD command 298
 ISSUE CONFIRMATION command 300
 ISSUE END command 307
 ISSUE ENDFILE command 308
 ISSUE ENDOUTPUT command 309
 ISSUE EODS command 310
 ISSUE ERASE command 312
 ISSUE ERASEAUP command 313
 ISSUE ERROR command 316
 ISSUE NOTE command 319
 ISSUE PASS command 321
 ISSUE PREPARE command 323
 ISSUE PRINT command 324
 ISSUE QUERY command 325
 ISSUE RECEIVE command 328
 ISSUE REPLACE command 330
 ISSUE SEND command 334
 ISSUE SIGNAL (APPC) command 337
 ISSUE WAIT command 340
 LINK ACQPROCESS command 351
 LINK ACTIVITY command 354
 LINK command 346
 LOAD command 358
 MONITOR command 361
 MOVE CONTAINER (BTS) command 364
 MOVE CONTAINER (CHANNEL) command 368
 POP HANDLE command 370
 POST command 373
 PURGE MESSAGE command 375
 PUT CONTAINER (BTS) command 378
 PUT CONTAINER (CHANNEL) command 383
 QUERY SECURITY command 389
 READ command 398
 READNEXT command 410
 READPREV command 420
 READQ TD command 425
 READQ TS command 429
 RECEIVE (non-VTAM) command 450
 RECEIVE MAP command 453
 RECEIVE MAP MAPPINGDEV command 457
 RECEIVE PARTN command 460
 RELEASE command 461
 REMOVE SUBEVENT command 463

INVREQ condition *(continued)*

RESET ACQPROCESS command 464
 RESET ACTIVITY command 466
 RESETBR command 470
 RESUME command 474
 RETRIEVE command 477
 RETRIEVE REATTACH EVENT command 480
 RETRIEVE SUBEVENT command 482
 RETURN command 486
 REWRITE command 492
 ROUTE command 498
 RUN command 503
 SEND (non-VTAM) command 524
 SEND CONTROL command 529
 SEND MAP command 536
 SEND MAP MAPPINGDEV command 540
 SEND PAGE command 543
 SEND PARTNSET command 545
 SEND TEXT command 551
 SEND TEXT MAPPED command 554
 SEND TEXT NOEDIT command 559
 SIGNOFF command 560
 SIGNON command 563
 SOAPFAULT ADD command 566
 SOAPFAULT CREATE command 570
 SOAPFAULT DELETE command 571
 SPOOLCLOSE command 573
 SPOOLOPEN INPUT command 575
 SPOOLOPEN OUTPUT command 580
 SPOOLREAD command 583
 SPOOLWRITE command 586
 START ATTACH command 599
 START BREXIT command 601
 START command 595
 START TRANSID (CHANNEL) command 605
 STARTBR command 611
 STARTBROWSE EVENT command 618
 SUSPEND (BTS) command 624
 SYNCPOINT command 625
 SYNCPOINT ROLLBACK command 626
 TEST EVENT command 627
 UNLOCK command 629
 VERIFY PASSWORD command 637
 WAIT CONVID command 639
 WAIT EVENT command 641
 WAIT EXTERNAL command 644
 WAIT TERMINAL command 650
 WAITCICS command 653
 WEB CONVERSE command 664
 WEB ENDBROWSE FORMFIELD command 666
 WEB ENDBROWSE HTTPHEADER command 667
 WEB EXTRACT command 672
 WEB OPEN command 676
 WEB PARSE URL command 678
 WEB READ FORMFIELD command 681
 WEB READ HTTPHEADER command 683
 WEB READNEXT FORMFIELD command 684
 WEB READNEXT HTTPHEADER command 686
 WEB RECEIVE command (Client) 696
 WEB RECEIVE command (Server) 691
 WEB RETRIEVE command 697

INVREQ condition (*continued*)

- WEB SEND command (Client) 711
- WEB SEND command (Server) 703
- WEB STARTBROWSE FORMFIELD command 714
- WEB STARTBROWSE HTTPHEADER command 715
- WEB WRITE HTTPHEADER command 718
- WRITE command 722
- WRITE JOURNALNAME command 728
- WRITE OPERATOR command 733
- WRITEQ TD command 734
- WRITEQ TS command 739
- XCTL command 743

IOERR condition

- ACQUIRE command 30
- CANCEL (BTS) command 71
- CHECK ACTIVITY command 80
- CHECK TIMER command 81
- DEFINE ACTIVITY command 110
- DEFINE PROCESS command 121
- DELETE ACTIVITY command 135
- DELETE command 131
- DELETE CONTAINER (BTS) command 138
- DUMP TRANSACTION command 166
- EIBRCODE 758
- ENDBR command 169
- GET CONTAINER (BTS) command 252
- GETNEXT ACTIVITY command 266
- GETNEXT PROCESS command 271
- INQUIRE CONTAINER command 284
- INQUIRE EVENT command 286
- INQUIRE TIMER command 289
- LINK ACQPROCESS command 351
- LINK ACTIVITY command 355
- MOVE CONTAINER (BTS) command 365
- PUT CONTAINER (BTS) command 379
- READ command 399
- READNEXT command 411
- READPREV command 421
- READQ TD command 425
- READQ TS command 429
- RESET ACQPROCESS command 464
- RESET ACTIVITY command 467
- RESETBR command 471
- RESUME command 474
- RETRIEVE command 477
- REWRITE command 492
- RUN command 503
- START command 595
- STARTBR command 611
- STARTBROWSE CONTAINER command 617
- STARTBROWSE EVENT command 619
- STARTBROWSE PROCESS command 620
- SUSPEND (BTS) command 624
- UNLOCK command 630
- WEB CONVERSE command 665
- WEB OPEN command 676
- WEB RECEIVE command (Client) 696
- WEB SEND command (Client) 712
- WEB SEND command (Server) 705
- WRITE command 722

IOERR condition (*continued*)

- WRITE JOURNALNAME command 728
- WRITEQ TD command 734
- WRITEQ TS command 739

IOERR option

- WAIT JOURNALNAME command 647

ISCINVREQ condition

- CANCEL command 69
- DELETE command 132
- DELETEQ TD command 145
- DELETEQ TS command 148
- ENDBR command 169
- READ command 399
- READNEXT command 411
- READPREV command 421
- READQ TD command 426
- READQ TS command 429
- RESETBR command 471
- REWRITE command 492
- START command 595
- START TRANSID (CHANNEL) command 605
- STARTBR command 612
- UNLOCK command 630
- WRITE command 722
- WRITEQ TD command 735
- WRITEQ TS command 739

ISSUE ABEND command 293

ISSUE ABORT command 295

ISSUE ADD command 297

ISSUE CONFIRMATION command 299

ISSUE COPY (3270 logical) command 301

ISSUE COPY command

- general information 772

ISSUE DISCONNECT (default) command 303

ISSUE DISCONNECT (LUTYPE6.1) command 305

ISSUE DISCONNECT command

- general information 769

ISSUE END command 306

ISSUE ENDFILE command 308

ISSUE ENDOUTPUT command 309

ISSUE EODS command 310

ISSUE ERASE command 311

ISSUE ERASEAUP command 313

- general information 772

ISSUE ERROR command 315

ISSUE LOAD command 317

ISSUE NOTE command 318

ISSUE PASS command 320

ISSUE PREPARE command 322

ISSUE PRINT command 324

- general information 771

ISSUE QUERY command 325

ISSUE RECEIVE command 327

ISSUE REPLACE command 329

ISSUE RESET command 332

ISSUE SEND command 333

ISSUE SIGNAL (APPC) command 336

ISSUE SIGNAL (LUTYPE6.1) command 338

ISSUE SIGNAL command

- general information 769

ISSUE WAIT command 339

- ISSUER option
 - EXTRACT CERTIFICATE command 193
- ITEM option
 - READQ TS command 428
 - WRITEQ TS command 738
- ITEMERR condition
 - READQ TS command 430
 - WRITEQ TS command 740
- IUTYPE option
 - BUILD ATTACH (LUTYPE6.1) command 63
 - BUILD ATTACH (MRO) command 66
 - EXTRACT ATTACH (LUTYPE6.1) command 181
 - EXTRACT ATTACH (MRO) command 185

J

- JIDERR condition
 - WRITE JOURNALNAME command 728
- JIDERR option
 - WAIT JOURNALNAME command 647
- JOURNAL command 341
- journal control
 - create a journal record 341
- journal record, creating 726
- journaling commands 22
- JOURNALNAME option
 - WAIT JOURNALNAME command 646
 - WRITE JOURNALNAME command 726
- JTYPEID option
 - WRITE JOURNALNAME command 727
- JUSFIRST option
 - SEND TEXT command 548
- JUSLAST option
 - SEND TEXT command 549
- JUSTIFY operand
 - DFHMDF 793
 - DFHMDI 804
- JUSTIFY option
 - SEND TEXT command 549

K

- katakana and English characters, mixed 102, 459
- KATAKANA option
 - ASSIGN command 53
- katakana terminals
 - CONVERSE (3270 logical) command 95
 - CONVERSE (LUTYPE2/LUTYPE3) command 95
 - CONVERSE command (3270 display) 102
 - CONVERSE command (3600 BTAM) 102
 - CONVERSE command (3735) 102
 - CONVERSE command (3740) 102
 - CONVERSE command (System/3) 102
 - CONVERSE command (System/7) 102
 - RECEIVE (non-VTAM) command 447
 - RECEIVE (VTAM) command 439
 - RECEIVE MAP command 452
 - RECEIVE PARTN command 459
 - SEND (non-VTAM) command 521
- KEEP option
 - SPOOLCLOSE command 572

- KEYLENGTH option
 - DELETE command 129
 - ISSUE ERASE command 311
 - ISSUE REPLACE command 329
 - READ command 394
 - READNEXT command 406
 - READPREV command 417
 - RESETBR command 469
 - STARTBR command 609
 - WRITE command 720
- KEYNUMBER option
 - ISSUE ERASE command 311
 - ISSUE REPLACE command 329
- keyword length 767

L

- L40, L64, or L80 options
 - SEND CONTROL command 527
 - SEND MAP command 534
 - SEND TEXT command 549
 - SEND TEXT NOEDIT command 557
- label argument, CICS command format 3
- LABEL option
 - HANDLE ABEND command 274
- LANG operand
 - DFHMDS 813
- LANGINUSE option
 - ASSIGN 53
 - SIGNON command 562
- language codes 766
- LANGUAGECODE option
 - SIGNON command 562
- large COMMAREAs, channels 139, 253, 366, 380, 483, 603, 742
- LAST option
 - GDS SEND command 246
 - SEND (non-VTAM) command 522
 - SEND (VTAM) command 516
 - SEND CONTROL command 527
 - SEND MAP command 533
 - SEND PAGE command 542
 - SEND TEXT command 549
 - SEND TEXT MAPPED command 553
 - SEND TEXT NOEDIT command 557
- LAST value
 - DFHMDI 804
- LASTUSETIME option
 - VERIFY PASSWORD command 637
- LDC operand
 - DFHMDS 813
- LDC option
 - CONVERSE (VTAM) command 97
 - ROUTE command 496
 - SEND (VTAM) command 517
 - SEND CONTROL command 527
 - SEND MAP command 533
 - SEND TEXT command 549
- LDCMNEM option
 - ASSIGN command 53

LDCNUM option
 ASSIGN command 53

LEAVEKB option
 CONVERSE (non-VTAM) command 104
 RECEIVE (non-VTAM) command 448
 SEND (non-VTAM) command 523

LEFT value
 DFHMDf 793
 DFHMdI 804

LENGERR condition
 BIF DEEDIT command 60
 CONNECT PROCESS command 85
 CONVERSE (non-VTAM) command 106
 CONVERSE (VTAM) command 99
 DEQ command 150
 EIBRCODE byte 1 757
 ENQ command 176
 ENTER TRACENUM command 179
 EXTRACT CERTIFICATE command 194
 EXTRACT PROCESS command 198
 EXTRACT TCPIP command 201
 GET CONTAINER (BTS) command 252
 GET CONTAINER (CHANNEL) command 255
 GETMAIN command 264
 ISSUE COPY (3270 logical) command 301
 ISSUE PASS command 321
 ISSUE RECEIVE command 328
 LINK command 347
 LOAD command 358
 PUT CONTAINER (CHANNEL) command 383
 QUERY SECURITY command 389
 READ command 399
 READNEXT command 411
 READPREV command 421
 READQ TD command 426
 READQ TS command 430
 RECEIVE (non-VTAM) command 450
 RECEIVE (VTAM) command 441
 RECEIVE PARTN command 460
 RETRIEVE command 477
 RETURN command 486
 REWRITE command 492
 SEND (non-VTAM) command 524
 SEND (VTAM) command 518
 SEND TEXT command 552
 SOAPFAULT ADD command 566
 SOAPFAULT CREATE command 570
 SPOOLOPEN OUTPUT command 580
 SPOOLREAD command 583
 SPOOLWRITE command 586
 START ATTACH command 599
 START BREXITcommand 601
 START command 596
 WEB CONVERSE command 665
 WEB EXTRACT command 672
 WEB OPEN command 676
 WEB PARSE URL command 678
 WEB READ FORMFIELD command 681
 WEB READ HTTPHEADER command 683
 WEB READNEXT FORMFIELD command 685
 WEB READNEXT HTTPHEADER command 687

LENGERR condition (*continued*)
 WEB RECEIVE command (Client) 696
 WEB RECEIVE command (Server) 691
 WEB SEND command (Client) 712
 WEB STARTBROWSE FORMFIELD command 714
 WEB WRITE HTTPHEADER command 718
 WRITE command 723
 WRITE JOURNALNAME command 728
 WRITE OPERATOR command 733
 WRITEQ TD command 735
 WRITEQ TS command 740
 XCTL command 743

LENGERR option
 DOCUMENT RETRIEVE command 159

LENGTH operand
 DFHMDf 793

LENGTH option
 BIF DEEDIT command 60
 built-in function 60
 default (assembler language) 8
 default (C) 5
 default (PL/I) 6
 DEQ command 149
 DOCUMENT RETRIEVE command 158
 DOCUMENT SET command 161
 DUMP TRANSACTION command 164
 ENQ command 175
 EXTRACT CERTIFICATE command 193
 EXTRACT LOGONMSG command 195
 fullword length alternative (FLENGTH) 767
 GETMAIN command 263
 ISSUE ADD command 297
 ISSUE PASS command 320
 ISSUE RECEIVE command 327
 ISSUE REPLACE command 330
 ISSUE SEND command 334
 LINK command 345
 LOAD command 358
 READ command 394
 READNEXT command 407
 READPREV command 417
 READQ TD command 424
 READQ TS command 428
 RECEIVE (non-VTAM) command 448
 RECEIVE (VTAM) command 439
 RECEIVE MAP command 452
 RECEIVE MAP MAPPINGDEV command 456
 RECEIVE PARTN command 459
 RETRIEVE command 476
 RETURN command 485
 REWRITE command 491
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517
 SEND MAP command 534
 SEND MAP MAPPINGDEV command 539
 SEND TEXT command 549
 SEND TEXT MAPPED command 553
 SEND TEXT NOEDIT command 557
 START ATTACHcommand 599
 START command 593
 WEB RECEIVE command (Client) 694

LENGTH option (*continued*)
 WEB RECEIVE command (Server) 689
 WEB SEND command (Server) 702
 WRITE command 720
 WRITEQ TD command 734
 WRITEQ TS command 738
 XCTL command 743
 LENGTH value
 DFHMDI 801
 DFHMSD 810
 LENGTHLIST option
 DUMP TRANSACTION command 164
 LEVEL option
 GETNEXT ACTIVITY command 265
 LIGHTPEN option
 HANDLE AID command 275
 LINE operand
 DFHMDI 804
 LINE option
 SPOOLWRITE command 585
 line value
 DFHMDI 806
 line,column value
 DFHMDF 797
 LINEADDR option
 CONVERSE (non-VTAM) command 104
 SEND (non-VTAM) command 523
 LINK ACQPROCESS command 350
 LINK ACTIVITY command 353
 LINK command 342
 link to program expecting return 342
 LIST option
 ROUTE command 497
 literal constants 7
 LLID option
 GDS RECEIVE command 243
 LOAD command 357
 load programs, tables, or maps 357
 LOADING condition
 DELETE command 132
 READ command 399
 READNEXT command 412
 STARTBR command 612
 WRITE command 723
 LOCALITY option
 EXTRACT CERTIFICATE command 193
 LOCALITYLEN option
 EXTRACT CERTIFICATE command 193
 LOCKED condition
 ACQUIRE command 30
 CANCEL (BTS) command 71
 CHECK ACTIVITY command 80
 DELETE ACTIVITY command 136
 DELETE command 132
 DELETE CONTAINER (BTS) command 138
 DELETEQ TD command 145
 DELETEQ TS command 148
 GET CONTAINER (BTS) command 252
 LINK ACTIVITY command 355
 MOVE CONTAINER (BTS) command 365
 PUT CONTAINER (BTS) command 379
 LOCKED condition (*continued*)
 READ command 400
 READNEXT command 412
 READPREV command 422
 READQ TD command 426
 RESET ACQPROCESS command 464
 RESET ACTIVITY command 467
 RESUME command 474
 REWRITE command 493
 RUN command 503
 SUSPEND (BTS) command 624
 WRITE command 723
 WRITEQ TD command 735
 WRITEQ TS command 740
 logical device code (LDC) 90, 509
 logical messages, BMS
 completing a logical message 541
 full BMS
 ROUTE command 495
 purging a logical message 375
 routing a logical message 495
 LOGMESSAGE option
 QUERY SECURITY command 387
 LOGMODE option
 ISSUE PASS command 320
 LOGONLOGMODE option
 ISSUE PASS 321
 LU (logical unit)
 3270 Information Display System 89, 301, 434, 508
 3270 SCS Printer 88, 507
 3270-Display, LUTYPE2 87, 433, 506
 3270-Display, LUTYPE3 433, 506
 3600 (3601) 90, 435, 509
 3600 (3614) 91, 436, 510
 3600 pipeline 435, 508
 3650 host conversational (3270) 92, 511
 3650 host conversational (3653) 92, 511
 3650 interpreter 91, 310, 317, 436, 510
 3650/3680 host command processor 512
 3770 batch 94, 438, 513
 3790 (3270-display) 95, 447, 514
 3790 (3270-printer) 515
 3790 full-function 438, 513
 3790 full-function or inquiry 94
 3790 SCS printer 514
 batch 94, 438, 513
 conversing with (CONVERSE) 769
 interactive 93
 reading data from 327, 767
 writing data to 297, 768
 LUNAME option
 ISSUE PASS command 321
 LUTYPE2, 3270-Display LU 87, 433, 506
 LUTYPE3, 3270-Display LU 433, 506
 LUTYPE4
 logical unit 87, 433, 506
 LUTYPE6.1 logical unit
 acquiring a session 41
 communicating on LUTYPE6.1 session 88
 converting 8-character names to 4 characters 202
 disconnecting 305

LUTYPE6.1 logical unit (*continued*)
 getting information about 369
 receiving data 434
 requesting change of direction 338
 retrieving values from an LUTYPE6.1 header 180
 sending data 507
 specifying values for an MRO attach header 65
 specifying values for LUTYPE6.1 attach header 62

M

macros, BMS, summary 784
 magnetic slot reader (MSR) 782
 MAIN option
 WRITEQ TS command 738
 main storage 261
 map definition macro, BMS 785, 800
 MAP option
 RECEIVE MAP command 452
 RECEIVE MAP MAPPINGDEV command 456
 SEND MAP command 534
 SEND MAP MAPPINGDEV command 540
 MAP value
 DFHMSD 816
 MAPATTS operand
 DFHMDI 805
 DFHMSD 813
 MAPCOLUMN option
 ASSIGN command 53
 MAPFAIL condition
 RECEIVE MAP command 453
 RECEIVE MAP MAPPINGDEV command 457
 MAPHEIGHT option
 ASSIGN command 53
 MAPLINE option
 ASSIGN command 53
 MAPONLY option
 SEND MAP command 534
 SEND MAP MAPPINGDEV command 540
 MAPONLY value
 DFHMDI 803
 DFHMSD 812
 MAPPINGDEV option
 RECEIVE MAP MAPPINGDEV command 456
 SEND MAP MAPPINGDEV command 540
 maps, loading 357
 mapset definition macro (DFHMSD) 785, 809
 MAPSET option
 RECEIVE MAP command 452
 RECEIVE MAP MAPPINGDEV command 456
 SEND MAP command 534
 SEND MAP MAPPINGDEV command 540
 MAPSFX operand
 DFHPDI 819
 MAPWIDTH option
 ASSIGN command 53
 MASSINSERT option
 WRITE command 720
 MAXDATALEN option
 EXTRACT TCPIP command 200

MAXFLENGTH option
 CONVERSE (non-VTAM) command 104
 CONVERSE (VTAM) command 97
 fullword alternative to MAXLENGTH 767
 GDS RECEIVE command 243
 RECEIVE (non-VTAM) command 448
 RECEIVE (VTAM) command 440
 SPOOLREAD command 582
 MAXIMUM option
 DEFINE COUNTER command 115
 DEFINE DCOUNTER command 115
 QUERY COUNTER command 384
 QUERY DCOUNTER command 384
 MAXLENGTH option
 CONVERSE (non-VTAM) command 104
 CONVERSE (VTAM) command 97
 DOCUMENT RETRIEVE command 159
 fullword length alternative (MAXFLENGTH) 767
 RECEIVE (non-VTAM) command 448
 RECEIVE (VTAM) command 440
 WEB CONVERSE command 661
 WEB RECEIVE command (Client) 694
 WEB RECEIVE command (Server) 689
 WRITE OPERATOR command 732
 MAXLIFETIME option
 DEQ command 150
 ENQ command 175
 MAXPROCLN option
 EXTRACT PROCESS command 197
 GDS EXTRACT PROCESS command 228
 MCC option
 SPOOLOPEN OUTPUT command 578
 MEDIATYPE option
 WEB CONVERSE command 659, 662
 WEB RECEIVE command (Client) 695
 WEB SEND command (Client) 709
 WEB SEND command (Server) 702
 METHOD option
 WEB CONVERSE command 660
 WEB SEND command (Client) 709
 METHODLENGTH option
 WEB EXTRACT command 670
 MINIMUM option
 DEFINE COUNTER command 115
 DEFINE DCOUNTER command 115
 QUERY COUNTER command 385
 QUERY DCOUNTER command 385
 MINUTES option
 DEFINE TIMER command 123
 DELAY command 126
 POST command 373
 ROUTE command 497
 START command 593
 MMDDYY option
 FORMATTIME command 207
 MMDDYYYY option
 FORMATTIME command 207
 MODE operand
 DFHMSD 813
 MODE option
 CHECK ACQPROCESS command 77

MODE option (*continued*)
 CHECK ACTIVITY command 79
 INQUIRE ACTIVITYID command 281
 model codes (terminal) 763
 MODENAME option
 GDS ALLOCATE command 219
 modes, of an activity
 ACTIVE 281
 CANCELLING 281
 COMPLETE 281
 DORMANT 281
 INITIAL 281
 MONITOR command 360
 monitoring application performance 360
 monitoring commands 22
 MONTH option
 DEFINE TIMER command 123
 MONTHOFYEAR option
 FORMATTIME command 207
 MOVE CONTAINER (BTS) command 363
 MOVE CONTAINER (CHANNEL) command 366
 MSR (magnetic slot reader)
 control byte values and constants 782
 DFHMSRCA, 782
 MSR option
 SEND CONTROL command 527
 SEND MAP command 534
 SEND TEXT command 549
 MSRCONTROL option
 ASSIGN command 53
 multi region operation (MRO) commands
 ALLOCATE 43
 BUILD ATTACH 65
 CONVERSE command 101
 EXTRACT ATTACH 184
 EXTRACT ATTRIBUTES 190
 FREE 214
 RECEIVE 443
 SEND 519
 multiple base registers 14
 MUSTENTER value
 DFHMDF 798
 DFHMDI 807
 DFHMDS 817
 MUSTFILL value
 DFHMDF 798
 DFHMDI 807
 DFHMDS 817

N

name argument, CICS command format 3
 NAME option
 WAIT EVENT command 641
 WAIT EXTERNAL command 644
 WAITCICS command 652
 named counter
 define named counter 114
 delete named counter 140
 query named counter 384
 named counter server commands 22
 named counter server, GET command 256
 named counter server, REWIND command 487
 named counter server, UPDATE command 632
 NAMELENGTH option
 WEB READ FORMFIELD command 681
 WEB READ HTTPHEADER command 682
 WEB READNEXT FORMFIELD command 684
 WEB READNEXT HTTPHEADER command 686
 WEB STARTBROWSE FORMFIELD command 713
 WEB WRITE HTTPHEADER command 717
 national language codes 766
 NATLANG option
 SIGNON command 562
 NATLANGINUSE option
 ASSIGN command 53
 SIGNON command 562
 NETNAME option
 ASSIGN command 54
 EXTRACT TCT command 202
 NETNAMEIDERR condition
 ALLOCATE (APPC) command 39
 new tasks, passing data to 591
 NEWPASSWORD option
 CHANGE PASSWORD command 73
 SIGNON command 562
 NEXT option
 READQ TS command 429
 NEXT value
 DFHMDI 801, 805
 NEXTTRANSID option
 ASSIGN command 54
 NLEOM option
 ROUTE command 497
 SEND MAP command 534
 SEND TEXT command 549
 NO value
 DFHMDI 803, 805, 806
 DFHMDS 812, 813, 816
 NOAUTOPAGE option
 SEND PAGE command 542
 NOCC option
 SPOOL OPEN OUTPUT command 578
 NOCHECK option
 DEFINE PROCESS command 119
 START command 593
 NODATA option
 GET CONTAINER (BTS) command 251
 GET CONTAINER (CHANNEL) command 254
 NODE option
 SPOOL OPEN OUTPUT command 578
 NODEIDERR condition
 SPOOL OPEN OUTPUT command 580
 NODUMP option
 ABEND command 27
 NOFLUSH option
 SEND MAP command 535
 NOHANDLE option
 deactivating HANDLE CONDITION command 278
 option 9
 overriding HANDLE AID 9

NOJBUFSP condition
 WRITE JOURNALNAME command 728

NONVAL condition
 ISSUE LOAD command 317

NOPASSBKRD condition
 RECEIVE (non-VTAM) command 450

NOPASSBKWR condition
 SEND (non-VTAM) command 524

NOQUEUE option
 ALLOCATE (APPC) command 38
 ALLOCATE (LUTYPE6.1) command 41
 ALLOCATE (MRO) command 43
 GDS ALLOCATE command 219

NOQUIESCE
 ISSUE PASS command 321

NORM value
 DFHMDF 791

NOSPACE condition
 DUMP TRANSACTION command 167
 REWRITE command 493
 WRITE command 723
 WRITEQ TD command 735
 WRITEQ TS command 740

NOSPOOL condition
 SPOOLCLOSE command 573
 SPOOLOPEN INPUT command 575
 SPOOLOPEN OUTPUT command 580
 SPOOLREAD command 583
 SPOOLWRITE command 586

NOSTART condition
 ISSUE LOAD command 317

NOSTG condition
 DUMP TRANSACTION command 167
 GETMAIN command 264
 SPOOLCLOSE command 573
 SPOOLOPEN INPUT command 575
 SPOOLOPEN OUTPUT command 580
 SPOOLREAD command 583
 SPOOLWRITE command 586

NOSUSPEND option
 ALLOCATE (APPC) 37
 ALLOCATE (LUTYPE6.1) command 41
 DELETE 129
 ENQ command 176
 GETMAIN command 263
 READ command 395
 READNEXT command 408
 READPREV command 418
 READQ TD command 425
 REWRITE command 491
 WRITE command 720
 WRITE JOURNALNAME command 727
 WRITEQ TS command 738

NOTALLOC condition
 CONNECT PROCESS command 85
 CONVERSE (non-VTAM) command 106
 CONVERSE (VTAM) command 99
 EXTRACT ATTACH (LUTYPE6.1) command 183
 EXTRACT ATTACH (MRO) command 187
 EXTRACT ATTRIBUTES (APPC) command 189
 EXTRACT ATTRIBUTES (MRO) command 191

NOTALLOC condition (*continued*)
 EXTRACT LOGONMSG command 196
 EXTRACT PROCESS command 198
 EXTRACT TCT command 202
 FREE (APPC) command 212
 FREE (LUTYPE6.1) command 213
 FREE (MRO) command 215
 FREE command 210
 ISSUE ABEND command 294
 ISSUE CONFIRMATION command 300
 ISSUE COPY (3270 logical) command 301
 ISSUE DISCONNECT (LUTYPE6.1) command 305
 ISSUE ENDFILE command 308
 ISSUE ENDOUTPUT command 309
 ISSUE EODS command 310
 ISSUE ERASEAUP command 313
 ISSUE ERROR command 316
 ISSUE LOAD command 317
 ISSUE PASS command 321
 ISSUE PREPARE command 323
 ISSUE PRINT command 324
 ISSUE SIGNAL (APPC) command 337
 ISSUE SIGNAL (LUTYPE6.1) command 338
 POINT command 369
 RECEIVE (non-VTAM) command 450
 RECEIVE (VTAM) command 441
 SEND (non-VTAM) command 524
 SEND (VTAM) command 518
 WAIT CONVID command 640
 WAIT SIGNAL command 649
 WAIT TERMINAL command 650

NOTAUTH condition
 ACQUIRE command 30
 CANCEL (BTS) command 71
 CANCEL command 69
 CHANGE PASSWORD command 74
 DEFINE ACTIVITY command 111
 DEFINE PROCESS command 121
 DELETE command 132
 DELETEQ TD command 146
 DELETEQ TS command 148
 ENDBR command 169
 HANDLE ABEND command 274
 INQUIRE ACTIVITYID command 282
 INQUIRE CONTAINER command 284
 INQUIRE EVENT command 286
 INQUIRE PROCESS command 287
 INQUIRE TIMER command 289
 LINK ACQPROCESS command 351
 LINK ACTIVITY command 355
 LINK command 347
 LOAD command 358
 READ command 400
 READNEXT command 412
 READPREV command 422
 READQ TD command 426
 READQ TS command 430
 RELEASE command 462
 RESET ACQPROCESS command 464
 RESET ACTIVITY command 467
 RESETBR command 471

NOTAUTH condition *(continued)*
 REWRITE command 493
 RUN command 503
 SIGNON command 563
 SPOOLOPEN INPUT command 575
 START ATTACH command 599
 START BREXIT command 601
 START command 596
 START TRANSID (CHANNEL) command 606
 STARTBR command 612
 STARTBROWSE ACTIVITY command 615
 STARTBROWSE CONTAINER command 617
 STARTBROWSE EVENT command 619
 STARTBROWSE PROCESS command 620
 UNLOCK command 630
 VERIFY PASSWORD command 638
 WEB CONVERSE command 665
 WEB OPEN command 676
 WEB SEND command (Client) 712
 WRITE command 724
 WRITE JOURNALNAME command 728
 WRITEQ TD command 735
 WRITEQ TS command 740
 XCTL command 744

NOTFND condition
 CANCEL command 69
 DELETE command 132
 DELETE COUNTER command 259, 488, 634
 QUERY SECURITY command 389
 READ command 400
 READNEXT command 412
 READPREV command 422
 RESETBR command 471
 REWRITE 493
 SOAPFAULT ADD command 567
 SOAPFAULT DELETE command 571
 SPOOLCLOSE command 573
 SPOOLOPEN INPUT command 575
 SPOOLOPEN OUTPUT command 581
 SPOOLREAD command 583
 STARTBR command 612
 WEB CONVERSE command 665
 WEB OPEN command 676
 WEB READ FORMFIELD command 681
 WEB READ HTTPHEADER command 683
 WEB RECEIVE command (Server) 691
 WEB SEND command (Client) 712
 WEB SEND command (Server) 704
 WEB STARTBROWSE FORMFIELD command 714

NOTFND option
 DOCUMENT RETRIEVE command 159

NOTOPEN condition
 DELETE command 133
 DUMP TRANSACTION command 167
 READ command 401
 READQ TD command 426
 SPOOLCLOSE command 573
 SPOOLOPEN INPUT command 576
 SPOOLOPEN OUTPUT command 581
 SPOOLREAD command 584
 SPOOLWRITE command 586

NOTOPEN condition *(continued)*
 STARTBR command 613
 UNLOCK command 630
 WEB CLOSE command 656
 WEB CONVERSE command 664
 WEB ENDBROWSE HTTPHEADER command 667
 WEB EXTRACT command 672
 WEB READ HTTPHEADER command 683
 WEB READNEXT HTTPHEADER command 687
 WEB RECEIVE command (Client) 696
 WEB SEND command (Client) 711
 WEB STARTBROWSE HTTPHEADER
 command 715
 WEB WRITE HTTPHEADER command 718
 WRITE command 724
 WRITE JOURNALNAME command 728
 WRITEQ TD command 735

NOTOPEN option
 WAIT JOURNALNAME command 647

NOTRUNCATE option
 CONVERSE (non-VTAM) command 104
 CONVERSE (VTAM) command 97
 RECEIVE (non-VTAM) command 449
 RECEIVE (VTAM) command 440
 WEB CONVERSE command 662
 WEB RECEIVE command (Client) 695
 WEB RECEIVE command (Server) 690

NOWAIT option
 ISSUE ADD command 297
 ISSUE ERASE command 311
 ISSUE REPLACE command 330
 ISSUE SEND command 334

NUM value
 DFHMDF 791
 number value
 DFHMDF 797

NUMBER value
 DFHMDI 801, 805

NUMCIPHERS option
 WEB OPEN command 675

NUMEVENTS option
 WAIT EXTERNAL command 644
 WAITCICS command 653

NUMITEMS option
 READQ TS command 429
 WRITEQ TS command 738

NUMREC option
 DELETE command 129
 ISSUE ADD command 298
 ISSUE ERASE command 312
 ISSUE REPLACE command 330

NUMROUTES option
 WRITE OPERATOR command 732

NUMSEGMENTS option
 DUMP TRANSACTION command 164

NUMTAB option
 ASSIGN command 54

O

- OBFMT operand
 - DFHMDI 805
 - DFHMSD 813
- OCCURS operand
 - DFHMDF 794
- OFF value
 - DFHMDF 792
 - DFHMDI 803
 - DFHMSD 812
- OIDCARD option
 - SIGNON command 563
- ON option
 - DEFINE TIMER command 123
- OPCLASS option
 - ASSIGN command 54
 - ROUTE command 497
- OPENERR condition
 - DUMP TRANSACTION command 167
 - SPOOLOPEN INPUT command 576
 - SPOOLOPEN OUTPUT command 581
- OPERID option
 - HANDLE AID command 275
- OPERKEYS option
 - ASSIGN command 54
- OPERPURGE option
 - SEND PAGE command 542
- OPID option
 - ASSIGN command 54
- OPSECURITY option
 - ASSIGN command 54
- options
 - BMS 451, 455, 547
 - length 767
- OPTIONS(MAIN)
 - in PL/I 11
- OR option
 - DEFINE COMPOSITE EVENT command 113
- ORGABCODE option
 - ASSIGN command 54
- ORGANIZATION option
 - EXTRACT CERTIFICATE command 193
- ORGANIZATLEN option
 - EXTRACT CERTIFICATE command 193
- ORGUNIT option
 - EXTRACT CERTIFICATE command 193
- ORGUNITLEN option
 - EXTRACT CERTIFICATE command 193
- OUTDESCR option
 - SPOOLOPEN OUTPUT command 578
- OUTDESCRERR condition
 - SPOOLOPEN OUTPUT command 581
- OUTLINE operand
 - DFHMDF 794
 - DFHMDI 805
 - DFHMSD 814
- OUTLINE option
 - ASSIGN command 54
- OUTPARTN option
 - SEND CONTROL command 527
 - SEND MAP command 535

- OUTPARTN option (*continued*)
 - SEND TEXT command 550
 - SEND TEXT NOEDIT command 558
- output control, 2980 General Banking Terminal System 445
- output to common buffer, 2980 445
- OVERFLOW condition
 - SEND MAP command 537
- OWNER option
 - EXTRACT CERTIFICATE command 193

P

- PA1-PA3 option
 - HANDLE AID command 275
- PAGE option
 - SPOOLWRITE command 585
- PAGENUM option
 - ASSIGN command 54
- PAGING option
 - SEND CONTROL command 528
 - SEND MAP command 535
 - SEND TEXT command 550
 - SEND TEXT MAPPED command 554
 - SEND TEXT NOEDIT command 558
- partition definition macro (DFHPDI) 786, 818
- partition set definition macro (DFHPSD) 786, 819
- PARTN operand
 - DFHMDI 805
 - DFHMSD 814
- PARTN option
 - RECEIVE PARTN command 459
- PARTNER option
 - ALLOCATE(APPC) command 39
 - CONNECT PROCESS command 83
 - GDS ALLOCATE command 220
 - GDS CONNECT PROCESS command 223
- PARTNERIDERR condition
 - ALLOCATE (APPC) command 39
 - CONNECT PROCESS command 85
- PARTNFAIL condition
 - RECEIVE MAP command 453
- PARTNPAGE option
 - ASSIGN command 55
- PARTNS option
 - ASSIGN command 55
- PARTNSET option
 - ASSIGN command 55
- PASSBK option
 - RECEIVE (non-VTAM) command 449
 - SEND (non-VTAM) command 523
- passbook control, 2980 444
- passing a session 320
- passing control
 - expecting return (LINK) 342
 - on receipt of an AID (HANDLE AID command) 275
 - on receipt of an AID (IGNORE AID) 279
 - without return (XCTL) 742
- passing data to new tasks 591
- PASSWORD option
 - CHANGE PASSWORD command 73

PASSWORD option (*continued*)
 SIGNON command 563
 VERIFY PASSWORD command 637

PATH option
 WEB CONVERSE command 660
 WEB EXTRACT command 670
 WEB PARSE URL command 678
 WEB SEND command (Client) 710

PATHLENGTH option
 WEB CONVERSE command 660
 WEB EXTRACT command 670
 WEB PARSE URL command 678
 WEB SEND command (Client) 710

PCT option
 DUMP TRANSACTION command 165

performance, application, monitoring 360

PF1–24 option
 HANDLE AID command 275

PFXLENG option
 WRITE JOURNALNAME command 727

PGMIDERR condition
 HANDLE ABEND command 274
 LINK ACQPROCESS command 352
 LINK ACTIVITY command 355
 LINK command 347
 LOAD command 358
 RELEASE command 462
 START BREXIT command 601
 XCTL command 744

PICIN operand
 DFHMDF 794

PICOUT operand
 DFHMDF 796

pipeline logical units 435, 508

PIPLENGTH option
 CONNECT PROCESS command 83
 EXTRACT PROCESS command 197
 GDS CONNECT PROCESS command 223
 GDS EXTRACT PROCESS command 228

PIPLIST option
 CONNECT PROCESS command 83
 EXTRACT PROCESS command 197
 GDS CONNECT PROCESS command 224
 GDS EXTRACT PROCESS command 228

PL/I language
 argument values 5
 LENGTH option default 6
 PROCEDURE statement 11
 STAE option 27
 translated code 11

plus 32K COMMAREAs (channels)
 ASSIGN command 50
 CHANNEL option of RETURN command 483
 CHANNEL option of XCTL command 742
 DELETE CONTAINER (CHANNEL) command 139
 GET CONTAINER (CHANNEL) command 253
 MOVE CONTAINER (CHANNEL) command 366
 PUT CONTAINER (CHANNEL) command 380
 START CHANNEL command 603

POINT command 369

POINT option
 MONITOR command 361

pointer-ref argument, CICS command format 3

pointer-value argument, CICS command format 3

POOL option
 DEFINE COUNTER command 115
 DEFINE DCOUNTER command 115
 DELETE COUNTER command 140
 DELETE DCOUNTER command 140
 GET COUNTER command 257
 GET DCOUNTER command 257
 QUERY COUNTER command 385
 QUERY DCOUNTER command 385
 REWIND COUNTER command 488
 REWIND DCOUNTER command 488
 UPDATE COUNTER command 633
 UPDATE DCOUNTER command 633

POP HANDLE command 370

PORTNUMBER option
 EXTRACT TCPIP command 200
 WEB EXTRACT command 670
 WEB OPEN command 675
 WEB PARSE URL command 678

PORTNUMNU option
 EXTRACT TCPIP command 200

POS operand 787
 DFHMDF 797

POST command 371

posting timer-event control area 371

PPT option
 DUMP TRANSACTION command 165

PREDICATE option
 GETNEXT EVENT command 270
 INQUIRE EVENT command 286

PREFIX option
 WRITE JOURNALNAME command 727

PRINCONVID option
 GDS ASSIGN command 222

PRINSYSID option
 ASSIGN command 55
 GDS ASSIGN command 222

print displayed information 771

PRINT option
 ISSUE ABORT command 295
 ISSUE END command 306
 ISSUE SEND command 334
 ISSUE WAIT command 339
 SEND CONTROL command 528
 SEND MAP command 535
 SEND MAP MAPPINGDEV command 540
 SEND TEXT command 550
 SEND TEXT NOEDIT command 558
 SPOOLOPEN OUTPUT command 579

PRINT value
 DFHMDF 801
 DFHMDF 810

printer control character list, DFHBMSCA 779

priority of task, changing 75

PRIORITY option
 CHANGE TASK command 75

PRIVACY option
 EXTRACT TCPIP command 200

PROCESS option
 ACQUIRE command 30
 ASSIGN command 55
 BUILD ATTACH (LUTYPE6.1) command 63
 BUILD ATTACH (MRO) command 66
 DEFINE PROCESS command 119
 DELETE CONTAINER (BTS) command 137
 EXTRACT ATTACH (LUTYPE6.1) command 181
 EXTRACT ATTACH (MRO) command 185
 GET CONTAINER (BTS) command 251
 GETNEXT PROCESS command 271
 INQUIRE ACTIVITYID command 281
 INQUIRE CONTAINER command 284
 INQUIRE PROCESS command 287
 PUT CONTAINER (BTS) command 378
 STARTBROWSE ACTIVITY command 615
 STARTBROWSE CONTAINER command 617

PROCESSBUSY condition
 ACQUIRE command 31
 CANCEL (BTS) command 71
 DELETE CONTAINER (BTS) command 138
 GET CONTAINER (BTS) command 252
 LINK ACQPROCESS command 352
 PUT CONTAINER (BTS) command 379
 RESET ACQPROCESS command 465
 RUN command 503

PROCESSERR condition
 ACQUIRE command 31
 CANCEL (BTS) command 71
 DEFINE PROCESS command 121
 GETNEXT PROCESS command 271
 INQUIRE CONTAINER command 284
 INQUIRE PROCESS command 287
 LINK ACQPROCESS command 352
 RESET ACQPROCESS command 465
 RESUME command 474
 RUN command 504
 STARTBROWSE ACTIVITY command 615
 STARTBROWSE CONTAINER command 617
 STARTBROWSE PROCESS command 620
 SUSPEND (BTS) command 624

processing state, of an activity
 ACTIVE 281
 CANCELLING 281
 COMPLETE 281
 DORMANT 281
 INITIAL 281

processing task, control delay of 125

PROCESSTYPE option
 ACQUIRE command 30
 ASSIGN command 55
 DEFINE PROCESS command 120
 INQUIRE ACTIVITYID command 281
 INQUIRE CONTAINER command 284
 INQUIRE PROCESS command 287
 STARTBROWSE ACTIVITY command 615
 STARTBROWSE CONTAINER command 617
 STARTBROWSE PROCESS command 620

PROCLENGTH option
 CONNECT PROCESS command 83
 EXTRACT PROCESS command 197
 GDS CONNECT PROCESS command 224
 GDS EXTRACT PROCESS command 228

PROCNAME option
 CONNECT PROCESS command 83
 EXTRACT PROCESS command 198
 GDS CONNECT PROCESS command 224
 GDS EXTRACT PROCESS command 228

PROFILE option
 ALLOCATE (APPC) command 39
 ALLOCATE (LUTYPE6.1) command 41
 ALLOCATE (MRO) command 43

program control
 commands 23
 deleting loaded program 461
 LINK command options 344
 linking to another program 342
 load a program, table, or map 357
 returning program control 483
 transfer program control 742

PROGRAM option
 ASSIGN command 55
 DEFINE ACTIVITY command 110
 DEFINE PROCESS command 120
 DUMP TRANSACTION command 165
 HANDLE ABEND command 274
 INQUIRE ACTIVITYID command 281
 ISSUE LOAD command 317
 LINK command 345
 LOAD command 358
 RELEASE command 461
 XCTL command 743

PROT value
 DFHMD 791

PROTECT option
 START command 593

PS operand
 DFHMD 797
 DFHMDI 805
 DFHMSD 814

PS option
 ASSIGN command 55

PSEUDOBIN option
 CONVERSE (non-VTAM) command 104
 RECEIVE (non-VTAM) command 449
 SEND (non-VTAM) command 523

psid value
 DFHMD 797
 DFHMDI 806
 DFHMSD 814

PUNCH option
 SPOOL OPEN OUTPUT command 579

PURGE MESSAGE command 375

PURGEABILITY option
 WAIT EXTERNAL command 644
 WAITCICS command 653

PUSH HANDLE command 376

PUT CONTAINER (BTS) command 377

PUT CONTAINER (CHANNEL) command 380

Q

QBUSY condition
 READQ TD command 426
QIDERR condition
 DELETEQ TD command 146
 DELETEQ TS command 148
 QUERY SECURITY command 390
 READQ TD command 426
 READQ TS command 430
 WRITEQ TD command 735
 WRITEQ TS command 740
QNAME option
 ASSIGN command 55
 DELETEQ TS command 147
 READQ TS command 429
 WRITEQ TS command 739
QUERY COUNTER command 384
QUERY DCOUNTER command 384
QUERY SECURITY command 387
QUERYSTRING option
 WEB EXTRACT command 671
 WEB PARSE URL command 678
 WEB SEND command 661
 WEB SEND command (Client) 710
QUERYSTRLEN option
 WEB EXTRACT command 671
 WEB PARSE URL command 678
 WEB SEND command 661
 WEB SEND command (Client) 711
QUEUE option
 BUILD ATTACH (LUTYPE6.1) command 64
 BUILD ATTACH (MRO) command 66
 DELETEQ TD command 145
 DELETEQ TS command 147
 EXTRACT ATTACH (LUTYPE6.1) command 181
 EXTRACT ATTACH (MRO) command 185
 READQ TD command 425
 READQ TS command 429
 RETRIEVE command 476
 START command 593
 WRITEQ TD command 734
 WRITEQ TS command 739
QZERO condition
 READQ TD command 426

R

RBA option
 DELETE command 129
 READ command 395
 READNEXT command 408
 READPREV command 418
 RESETBR command 469
 STARTBR command 610
 WRITE command 720
RDATT condition
 CONVERSE (non-VTAM) command 106
 RECEIVE (non-VTAM) command 450
 RECEIVE MAP command 454
reactivate an ABEND exit 273

READ command 391
READ option
 QUERY SECURITY command 387
reading records
 batch data interchange 327
 browsing, next 403
 browsing, previous (VSAM) 414
 file control 391
 from temporary storage queue 428
 from terminal or LU 767
 from transient data queue 424
READNEXT command 403
READPREV command 414
READQ TD command 424
READQ TS command 428
RECEIVE (2260) command 443
RECEIVE (2980) command 444
RECEIVE (3270 display) command 446
RECEIVE (3270 logical) command 434
RECEIVE (3600 pipeline) command 435
RECEIVE (3600-3601) command 435
RECEIVE (3600-3614) command 436
RECEIVE (3650) command 436
RECEIVE (3767) command 437
RECEIVE (3770) command 438
RECEIVE (3790 3270-display) command 447
RECEIVE (3790 full-function or inquiry) command 438
RECEIVE (APPC) command 432
RECEIVE (LUTYPE2/LUTYPE3) command 433
RECEIVE (LUTYPE4) command 433
RECEIVE (LUTYPE6.1) command 434
RECEIVE (MRO) command 443
RECEIVE (non-VTAM) command 442
RECEIVE (VTAM default) command 432
RECEIVE command
 input operation without data 772
 read from terminal or logical unit 767
RECEIVE MAP command 451
RECEIVE MAP MAPPINGDEV command 455
RECEIVE PARTN command 458
RECFM option
 BUILD ATTACH (LUTYPE6.1) command 64
 BUILD ATTACH (MRO) command 67
 EXTRACT ATTACH (LUTYPE6.1) command 182
 EXTRACT ATTACH (MRO) command 185
RECORDBUSY condition
 DELETE command 133
 READ command 401
 READNEXT command 412
 READPREV command 422
 REWRITE command 493
 WRITE command 724
RECORDLENGTH option
 SPOOL OPEN OUTPUT command 579
records
 deleting VSAM 128
 reading 327, 391
 release exclusive control 628
 requesting next number 318
 updating 329, 490
 writing new 719

- records (*continued*)
 - writing new (adding) 297
- REDUCE option
 - GET COUNTER command 258
 - GET DCOUNTER command 258
- register contents in assembler language 11
- relative byte address (RBA) 129
- RELEASE command 461
- RELEASE option
 - SEND PAGE command 542
- relocatable expression 7
- REMOVE SUBEVENT command 463
- REPEATABLE option
 - READ command 395
 - READNEXT command 408
 - READPREV command 418
- REPLY option
 - WRITE OPERATOR command 732
- REPLYLENGTH option
 - WRITE OPERATOR command 732
- REQID option
 - CANCEL command 69
 - DELAY command 126
 - ENDBR command 168
 - POST command 373
 - READNEXT command 408
 - READPREV command 419
 - RESETBR command 470
 - ROUTE command 497
 - SEND CONTROL command 528
 - SEND MAP command 535
 - SEND TEXT command 550
 - SEND TEXT MAPPED command 554
 - SEND TEXT NOEDIT command 558
 - START command 593
 - STARTBR command 610
 - WAIT JOURNALNAME command 646
 - WRITE JOURNALNAME command 727
- REQUESTTYPE option
 - WEB EXTRACT command 671
- RESCLASS option
 - QUERY SECURITY command 388
- RESET ACQPROCESS command 464
- RESET ACTIVITY command 466
- RESET option
 - HANDLE ABEND command 274
- reset start for browse 468
- RESETBR command 468
- RESID option
 - QUERY SECURITY command 388
- RESIDLENGTH option
 - QUERY SECURITY command 388
- RESOURCE option
 - BUILD ATTACH (LUTYPE6.1) command 64
 - BUILD ATTACH (MRO) command 67
 - DEQ command 150
 - ENQ command 176
 - ENTER TRACENUM command 179
 - EXTRACT ATTACH (LUTYPE6.1) command 182
 - EXTRACT ATTACH (MRO) command 186
- resource scheduling 149
- RESP
 - deactivating NOHANDLE 278
 - option 9
 - values in EIBRESP 759
- RESP2
 - EXPIRED in messages to console operators 733
 - INVREQ in messages to console operators 733
 - INVREQ in SIGNOFF command (Security control) 560
 - INVREQ in SIGNON (Security control) 563
 - INVREQ in WAIT EXTERNAL 644
 - INVREQ on WAITCICS 653
 - LENGERR in messages to console operators 733
 - NOTAUTH in SIGNON (Security control) 563
 - option 9
 - USERIDERR in SIGNON (Security control) 564
 - values in EIBRESP2 760
- RESSEC option
 - ASSIGN command 55
- RESTART option
 - ASSIGN command 56
- RESTYPE option
 - QUERY SECURITY command 388
- RESUME command 473
- RESUNAVAIL condition
 - LINK command 348
 - START command 596
 - START TRANSID (CHANNEL) command 606
- RETAIN option
 - SEND PAGE command 542
- RETCODE option
 - GDS ALLOCATE command 220
 - GDS ASSIGN command 222
 - GDS CONNECT PROCESS command 224
 - GDS EXTRACT ATTRIBUTES command 226
 - GDS EXTRACT PROCESS command 229
 - GDS FREE command 230
 - GDS ISSUE ABEND command 232
 - GDS ISSUE CONFIRMATION command 234
 - GDS ISSUE ERROR command 236
 - GDS ISSUE PREPARE command 238
 - GDS ISSUE SIGNAL command 240
 - GDS RECEIVE command 243
 - GDS SEND command 246
 - GDS WAIT command 248
- RETPAGE condition
 - SEND CONTROL command 529
 - SEND MAP command 537
 - SEND PAGE command 544
 - SEND TEXT command 552
- RETRIEVE command 475
- retrieve data stored for task 475
- RETRIEVE REATTACH EVENT command 479
- RETRIEVE SUBEVENT command 481
- RETURN command 483
- return program control 483
- RETURNPROG option
 - ASSIGN command 56
- REVERSE value
 - DFHMDF 792
 - DFHMDI 804

REVERSE value *(continued)*
 DFHMSD 813
 REWIND COUNTER command 487
 REWIND DCOUNTER command 487
 REWRITE command 490
 REWRITE option
 WRITEQ TS command 739
 RIDFLD option
 DELETE command 129
 ISSUE ADD command 298
 ISSUE ERASE command 312
 ISSUE NOTE command 318
 ISSUE REPLACE command 330
 READ command 395
 READNEXT command 408
 READPREV command 419
 RESETBR command 470
 STARTBR command 610
 WRITE command 720
 RIGHT value
 DFHMDF 793
 DFHMDI 804
 ROLLBACK option
 SYNCPOINT ROLLBACK command 626
 ROLLEDBACK condition
 LINK command 348
 SYNCPOINT command 625
 ROUTE command 495
 ROUTECODES option
 WRITE OPERATOR command 732
 RPROCESS option
 BUILD ATTACH (LUTYPE6.1) command 64
 BUILD ATTACH (MRO) command 67
 EXTRACT ATTACH (LUTYPE6.1) command 182
 EXTRACT ATTACH (MRO) command 186
 RRESOURCE option
 BUILD ATTACH (LUTYPE6.1) command 64
 BUILD ATTACH (MRO) command 67
 EXTRACT ATTACH (LUTYPE6.1) command 182
 EXTRACT ATTACH (MRO) command 186
 RRN option
 DELETE command 130
 ISSUE ADD command 298
 ISSUE ERASE command 312
 ISSUE NOTE command 318
 ISSUE REPLACE command 330
 READ command 396
 READNEXT command 409
 READPREV command 419
 RESETBR command 470
 STARTBR command 610
 WRITE command 721
 RTEFAIL condition
 ROUTE command 498
 RTERMID option
 RETRIEVE command 476
 START command 594
 RTESOME condition
 ROUTE command 499
 RTRANSID option
 RETRIEVE command 476

RTRANSID option *(continued)*
 START command 594
 RUN command 500

S

SAA (Systems Application Architecture)
 communications (CPI) 774
 Resource Recovery 773
 SADDRLENGTH option
 EXTRACT TCPIP command 200
 SAME value
 DFHMDI 801, 805
 schedule use of resource by task 149, 174
 scheduling commands 23
 SCHEME option
 WEB EXTRACT command 671
 WEB OPEN command 675
 SCHEMENAME option
 WEB PARSE URL command 678
 SCRNHHT option
 ASSIGN command 56
 SCRNRWD option
 ASSIGN command 56
 SCS (SNA character string)
 CONVERSE command 88
 SEND (VTAM) command 514
 SEND command 507
 SCS printer logical unit, 3790 514
 SECONDS option
 DEFINE TIMER command 123
 DELAY command 126
 POST command 373
 ROUTE command 497
 START command 594
 security commands 23
 SEGMENTLIST option
 DUMP TRANSACTION command 165
 SELNERR condition
 ISSUE ABORT command 296
 ISSUE ADD command 298
 ISSUE END command 307
 ISSUE ERASE command 312
 ISSUE NOTE command 319
 ISSUE QUERY command 326
 ISSUE REPLACE command 330
 ISSUE SEND command 335
 ISSUE WAIT command 340
 SEND (2260) command 520
 SEND (2980) command 520
 SEND (3270 display) command 521
 SEND (3270 logical) command 508
 SEND (3600 pipeline) command 508
 SEND (3600-3601) command 509
 SEND (3600-3614) command 510
 SEND (3650 interpreter) command 510
 SEND (3650-3270) command 511
 SEND (3650-3653) command 511
 SEND (3650-3680) command 512
 SEND (3767) command 512
 SEND (3770) command 513

SEND (3790 3270-display) command 514
 SEND (3790 3270-printer) command 515
 SEND (3790 full-function or inquiry) command 513
 SEND (3790 SCS) command 514
 SEND (APPC) command 505
 SEND (LUTYPE2/LUTYPE3) command 506
 SEND (LUTYPE4) command 506
 SEND (LUTYPE6.1) command 507
 SEND (MRO) command 519
 SEND (non-VTAM default) command 519
 SEND (SCS) command 507
 SEND (VTAM default) command 505
 send asynchronous interrupt 769
 SEND command
 write to terminal 768
 SEND CONTROL command 525
 SEND MAP command 530
 SEND MAP MAPPINGDEV command 538
 SEND PAGE command 541
 SEND PARTNSET command 545
 SEND TEXT command 546
 SEND TEXT MAPPED command 553
 SEND TEXT NOEDIT command 556
 sending data to output device 333
 sequential retrieval, browsing
 reading records 391
 SERIALNUM option
 EXTRACT CERTIFICATE command 193
 SERIALNUMLEN option
 EXTRACT CERTIFICATE command 194
 SERVADDRNU option
 EXTRACT TCPIP command 200
 SERVERADDR option
 EXTRACT TCPIP command 200
 SERVERCONV option
 WEB RECEIVE command (Server) 690
 WEB SEND command (Server) 702
 SERVERNAME option
 EXTRACT TCPIP command 200
 SESSBUSY condition
 ALLOCATE (LUTYPE6.1) command 42
 SESSION option
 ALLOCATE (LUTYPE6.1) command 41
 CONNECT PROCESS command 84
 CONVERSE (non-VTAM) command 104
 CONVERSE (VTAM) command 97
 EXTRACT ATTACH (LUTYPE6.1) command 182
 EXTRACT ATTACH (MRO) command 186
 EXTRACT ATTRIBUTES (MRO) command 190
 FREE (LUTYPE6.1) command 213
 FREE (MRO) command 214
 ISSUE DISCONNECT (LUTYPE6.1) command 305
 ISSUE SIGNAL (LUTYPE6.1) command 338
 POINT command 369
 RECEIVE (non-VTAM) command 449
 RECEIVE (VTAM) command 440
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517
 WAIT TERMINAL command 650
 session, passing 320
 SESSIONERR condition
 ALLOCATE (LUTYPE6.1) command 42
 EIBRCODE bytes 1-2 756
 SESSTOKEN option
 WEB CLOSE command 655
 WEB CONVERSE command 661
 WEB ENDBROWSE HTTPHEADER command 667
 WEB EXTRACT command 671
 WEB OPEN command 675
 WEB READ HTTPHEADER command 682
 WEB READNEXT HTTPHEADER command 686
 WEB RECEIVE command (Client) 695
 WEB SEND command (Client) 711
 WEB STARTBROWSE HTTPHEADER
 command 715
 WEB WRITE HTTPHEADER command 717
 SET option
 ADDRESS SET command 36
 CONVERSE (non-VTAM) command 104
 CONVERSE (VTAM) command 97
 EXTRACT LOGONMSG command 195
 GDS RECEIVE command 243
 GET CONTAINER (BTS) command 251
 GET CONTAINER (CHANNEL) command 254
 GETMAIN command 263
 INQUIRE CONTAINER command 284
 ISSUE RECEIVE command 327
 LOAD command 358
 POST command 373
 READ command 396
 READNEXT command 409
 READPREV command 419
 READQ TD command 425
 READQ TS command 429
 RECEIVE (non-VTAM) command 449
 RECEIVE (VTAM) command 440
 RECEIVE MAP command 452
 RECEIVE MAP MAPPINGDEV command 456
 RECEIVE PARTN command 459
 RETRIEVE command 476
 SEND CONTROL command 528
 SEND MAP command 535
 SEND MAP MAPPINGDEV command 540
 SEND PAGE command 542
 SEND TEXT command 550
 WEB CONVERSE command 662
 WEB READ FORMFIELD command 681
 WEB RECEIVE command (Client) 695
 WEB RECEIVE command (Server) 691
 SHARED option
 GETMAIN command 263
 SIGDATA option
 ASSIGN command 56
 SIGNAL condition
 CONVERSE (VTAM) command 99
 ISSUE CONFIRMATION command 300
 ISSUE DISCONNECT (default) command 303
 ISSUE ERROR command 316
 RECEIVE (VTAM) command 442
 SEND (VTAM) command 518
 WAIT SIGNAL command 649

SIGNAL condition (*continued*)
 WAIT TERMINAL command 650

SIGNOFF command 560

SIGNON command 561

single thread used with JES 574

SIT option
 DUMP TRANSACTION command 165

SIZE operand
 DFHMDI 806

SNAMELENGTH option
 EXTRACT TCPIP command 200

SOAPFAULT ADD command 565

SOAPFAULT CREATE command 568

SOAPFAULT DELETE command 571

SOSI operand
 DFHMDF 797
 DFHMDI 806
 DFHMSD 814

SOSI option
 ASSIGN command 56

SPCOMMAND
 RESID value not valid 389

SPOLBUSY condition
 SPOOLOPEN INPUT command 576
 SPOOLOPEN OUTPUT command 581

SPOLERR condition
 SPOOLOPEN INPUT command 576
 SPOOLREAD command 584
 SPOOLWRITE command 586

Spool commands 23

SPOOLCLOSE command 572

SPOOLCLOSE, implicit 574

SPOOLOPEN INPUT command 574

SPOOLOPEN OUTPUT 577

SPOOLREAD command 582

SPOOLWRITE command 585

SSLTYPE option
 EXTRACT TCPIP command 200

STAE option, PL/I 27

standard attribute and printer control character list, BMS (DFHBMSCA) 779

START ATTACH command 598

START CHANNEL command 603

START command 588, 600

STARTBR command 608

STARTBROWSE ACTIVITY command 614

STARTBROWSE CONTAINER command 616

STARTBROWSE EVENT command 618

STARTBROWSE PROCESS command 620

STARTCODE option
 ASSIGN command 56

STATE option
 ALLOCATE (APPC) command 39
 ALLOCATE (MRO) command 43
 CONNECT PROCESS command 84
 CONVERSE (non-VTAM) command 105
 CONVERSE (VTAM) command 97
 EXTRACT ATTRIBUTES (APPC) command 188
 EXTRACT ATTRIBUTES (MRO) command 190
 EXTRACT CERTIFICATE command 194
 FREE (APPC) command 211

STATE option (*continued*)
 FREE (MRO) command 214
 GDS ALLOCATE command 220
 GDS CONNECT PROCESS command 224
 GDS EXTRACT ATTRIBUTES command 226
 GDS FREE command 230
 GDS ISSUE ABEND command 232
 GDS ISSUE CONFIRMATION command 234
 GDS ISSUE ERROR command 236
 GDS ISSUE PREPARE command 238
 GDS ISSUE SIGNAL command 240
 GDS RECEIVE command 243
 GDS SEND command 246
 GDS WAIT command 248
 ISSUE ABEND command 293
 ISSUE CONFIRMATION command 299
 ISSUE ERROR command 315
 ISSUE PREPARE command 322
 ISSUE SIGNAL (APPC) command 336
 RECEIVE (non-VTAM) command 449
 RECEIVE (VTAM) command 440
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517
 WAIT CONVID command 639

STATELEN option
 EXTRACT CERTIFICATE command 194

STATIONID option
 ASSIGN command 57

STATUS option
 CHECK TIMER command 81
 INQUIRE TIMER command 288

STATUSCODE option
 WEB CONVERSE command 662
 WEB RECEIVE command (Client) 695
 WEB SEND command (Server) 703

STATUSLEN option
 WEB CONVERSE command 662
 WEB RECEIVE command (Client) 696
 WEB SEND command (Server) 703

STATUSTEXT option
 WEB CONVERSE command 662
 WEB RECEIVE command (Client) 695
 WEB SEND command (Server) 703

storage area length 47

storage control commands 23

STORAGE operand
 DFHMSD 814

STORAGE option
 DUMP TRANSACTION command 165

storage, dynamic 13

STRELERR condition
 SPOOLCLOSE command 573
 SPOOLOPEN INPUT command 576
 SPOOLOPEN OUTPUT command 581
 SPOOLREAD command 584
 SPOOLWRITE command 587

STRFIELD option
 CONVERSE (non-VTAM) command 105
 CONVERSE (VTAM) command 98
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517

STRINGFORMAT option
 FORMATTIME command 207
 stub, program 12
 SUBADDR option
 ISSUE ABORT command 296
 ISSUE END command 307
 ISSUE SEND command 334
 ISSUE WAIT command 340
 SUBEVENT option
 ADD SUBEVENT command 32
 DEFINE COMPOSITE EVENT command 113
 REMOVE SUBEVENT command 463
 RETRIEVE SUBEVENT command 482
 SUFFIX operand
 DFHMSD 815
 DFHPSD 820
 SUPPRESSED condition
 DUMP TRANSACTION command 167
 GET COUNTER command 260
 REWIND COUNTER command 489
 UPDATE COUNTER command 634
 WRITE command 724
 SUSPEND (BTS) command 623
 SUSPEND command 622
 SUSPSTATUS option
 CHECK ACQPROCESS command 77
 CHECK ACTIVITY command 80
 INQUIRE ACTIVITYID command 281
 switched line disconnection 769
 SYMBOL option
 DOCUMENT INSERT command 156
 DOCUMENT SET command 161
 SYMBOLERR condition
 DOCUMENT SET command 162
 symbolic register DFHEIPLR 15
 SYMBOLLIST option
 DOCUMENT SET command 152, 161
 synchronization levels
 basic conversations 229
 synchronize, action
 journal output (WAIT JOURNALNAME) 646
 terminal input/output 769
 SYNCHRONOUS option
 RUN command 502
 SYNCLEVEL option
 CONNECT PROCESS command 84
 EXTRACT PROCESS command 198
 GDS CONNECT PROCESS command 224
 GDS EXTRACT PROCESS command 229
 SYNCONRETURN option
 LINK command 345
 syncpoint
 backing out 626
 commands 23
 establishing 625
 SYNCPOINT command 625
 SYNCPOINT ROLLBACK command 626
 syntax notation 2
 SYSBUSY condition
 ALLOCATE (APPC) command 39
 ALLOCATE (LUTYPE6.1) command 42
 SYSBUSY condition (*continued*)
 ALLOCATE (MRO) command 44
 EIBRCODE byte 3 756
 SYSID option
 ALLOCATE (APPC) command 39
 ALLOCATE (LUTYPE6.1) command 42
 ALLOCATE (MRO) command 43
 ASSIGN command 57
 CANCEL command 69
 DELETE command 130
 DELETEQ TD command 145
 DELETEQ TS command 147, 739
 ENDBR command 168
 EXTRACT TCT command 202
 GDS ALLOCATE command 220
 LINK command 346
 READ command 396
 READNEXT command 409
 READPREV command 419
 READQ TD command 425
 READQ TS command 429
 RESETBR command 470
 REWRITE command 491
 START command 594
 START TRANSID (CHANNEL) command 604
 STARTBR command 610
 UNLOCK command 629
 WRITE command 721
 WRITEQ TD command 734
 SYSIDERR condition
 ALLOCATE (APPC) command 39
 ALLOCATE (LUTYPE6.1) command 42
 ALLOCATE (MRO) command 44
 CANCEL command 69
 DELETE command 134
 DELETEQ TD command 146
 DELETEQ TS command 148
 EIBRCODE bytes 1-2 756
 ENDBR command 169
 LINK command 348
 READ command 401
 READNEXT command 413
 READPREV command 422
 READQ TD command 426
 READQ TS command 430
 RESETBR command 471
 REWRITE command 494
 START command 596
 START TRANSID (CHANNEL) command 606
 STARTBR command 613
 UNLOCK command 630
 WRITE command 724
 WRITEQ TD command 735
 WRITEQ TS command 740
 systemname
 definition 4, 5, 6, 7
 systemname argument, CICS command format 3

T

- TABLES option
 - DUMP TRANSACTION command 165
- tables, loading 357
- task
 - initiation 591
- task control commands 23
- TASK option
 - DUMP TRANSACTION command 165
- task, abnormal termination 273
- task, delay processing of 125
- TASKDATALOC resource definition option 35
- TASKPRIORITY option
 - ASSIGN command 57
- TCAM-supported terminals and logical units 769
- TCP/IP services 23
- TCPIPSERVICE option
 - EXTRACT TCPIP command 201
- TCT option
 - DUMP TRANSACTION command 166
- TCTUA option
 - ADDRESS command 35
- TCTUALENG option
 - ASSIGN command 57
- teletypewriter
 - messages 770
 - programming 770
- TELLERID option
 - ASSIGN command 57
- TEMPLATE option
 - DOCUMENT INSERT command 156
- temporary storage control commands 24
- TERM operand
 - DFHMDI 806
 - DFHMSD 815
- TERMCODE option
 - ASSIGN command 57, 763
- TERMERR condition
 - CONNECT PROCESS command 85
 - CONVERSE (non-VTAM) command 106
 - CONVERSE (VTAM) command 100
 - ISSUE ABEND command 294
 - ISSUE CONFIRMATION command 300
 - ISSUE COPY (3270 logical) command 301
 - ISSUE DISCONNECT (default) command 304
 - ISSUE DISCONNECT (LUTYPE6.1) command 305
 - ISSUE EODS command 310
 - ISSUE ERASEAUP command 314
 - ISSUE ERROR command 316
 - ISSUE LOAD command 317
 - ISSUE PREPARE command 323
 - ISSUE PRINT command 324
 - ISSUE SIGNAL (APPC) command 337
 - ISSUE SIGNAL (LUTYPE6.1) command 338
 - LINK command 349
 - RECEIVE (non-VTAM) command 450
 - RECEIVE (VTAM) command 442
 - SEND (non-VTAM) command 524
 - SEND (VTAM) command 518
 - WAIT SIGNAL command 649
 - WAIT TERMINAL command 651
- TERMINAL option
 - DUMP TRANSACTION command 166
 - RECEIVE MAP command 453
 - SEND CONTROL command 528
 - SEND MAP command 536
 - SEND TEXT command 551
 - SEND TEXT MAPPED command 554
 - SEND TEXT NOEDIT command 558
- terminal control 767
 - commands 24
- terminal model codes 763
- terminal operator paging, initiate paging transaction 541
- TERMINAL option
 - DUMP TRANSACTION command 166
 - RECEIVE MAP command 453
 - SEND CONTROL command 528
 - SEND MAP command 536
 - SEND TEXT command 551
 - SEND TEXT MAPPED command 554
 - SEND TEXT NOEDIT command 558
- terminal type codes 763
- terminate data set processing
 - abnormal 295
 - normal 306
- TERMPRIORITY option
 - ASSIGN command 57
- TEST EVENT command 627
- TEXT option
 - DOCUMENT INSERT command 157
 - WRITE OPERATOR command 732
- TEXTKYBD option
 - ASSIGN command 57
- TEXTLENGTH option
 - WRITE OPERATOR command 733
- TEXTPRINT option
 - ASSIGN command 58
- threadsafe commands 17
- time of day, requesting 45
- TIME option
 - DELAY command 126
 - FORMATTIME command 207
 - POST command 373
 - ROUTE command 498
 - START command 594
- TIMEDOUT condition
 - WEB CONVERSE command 665
 - WEB OPEN command 676
 - WEB RECEIVE command (Client) 696
- TIMEOUT option
 - WRITE OPERATOR command 733
- TIMER option
 - CHECK TIMER command 81
 - DEFINE TIMER command 123
 - DELETE TIMER command 144
 - FORCE TIMER command 203
 - GETNEXT EVENT command 270
 - INQUIRE EVENT command 286
 - INQUIRE TIMER command 288
- timer-event control area 371
- TERMINAL option
 - EXTRACT TCT command 202
 - ISSUE COPY (3270 logical) command 301
 - START command 594
 - START TRANSID (CHANNEL) command 604
- TERMIDERR condition
 - START command 596
 - START TRANSID (CHANNEL) command 606
- terminal control 767
 - commands 24
- terminal model codes 763
- terminal operator paging, initiate paging transaction 541
- TERMINAL option
 - DUMP TRANSACTION command 166
 - RECEIVE MAP command 453
 - SEND CONTROL command 528
 - SEND MAP command 536
 - SEND TEXT command 551
 - SEND TEXT MAPPED command 554
 - SEND TEXT NOEDIT command 558
- terminal type codes 763
- terminate data set processing
 - abnormal 295
 - normal 306
- TERMPRIORITY option
 - ASSIGN command 57
- TEST EVENT command 627
- TEXT option
 - DOCUMENT INSERT command 157
 - WRITE OPERATOR command 732
- TEXTKYBD option
 - ASSIGN command 57
- TEXTLENGTH option
 - WRITE OPERATOR command 733
- TEXTPRINT option
 - ASSIGN command 58
- threadsafe commands 17
- time of day, requesting 45
- TIME option
 - DELAY command 126
 - FORMATTIME command 207
 - POST command 373
 - ROUTE command 498
 - START command 594
- TIMEDOUT condition
 - WEB CONVERSE command 665
 - WEB OPEN command 676
 - WEB RECEIVE command (Client) 696
- TIMEOUT option
 - WRITE OPERATOR command 733
- TIMER option
 - CHECK TIMER command 81
 - DEFINE TIMER command 123
 - DELETE TIMER command 144
 - FORCE TIMER command 203
 - GETNEXT EVENT command 270
 - INQUIRE EVENT command 286
 - INQUIRE TIMER command 288
- timer-event control area 371

TIMERERR condition
 CHECK TIMER command 81
 DEFINE TIMER command 124
 DELETE TIMER command 144
 FORCE TIMER command 204
 INQUIRE TIMER command 289

TIMESEP option
 FORMATTIME command 207

TIOAPFX operand
 DFHMDI 806
 DFHMSD 816

TITLE option
 ROUTE command 498

TO option
 DOCUMENT INSERT command 157

TOACTIVITY option
 MOVE CONTAINER (BTS) command 364

TOCHANNEL option
 MOVE CONTAINER (CHANNEL) command 367

TOLENGTH option
 CONVERSE (non-VTAM) command 105
 CONVERSE (VTAM) command 98
 fullword alternative to TOLENGTH 767
 SPOOLREAD command 582

TOKEN option
 DELETE command 130
 READ command 396
 READNEXT 409
 READPREV command 419
 REWRITE command 491
 SPOOLCLOSE command 572
 SPOOLOPEN INPUT command 574
 SPOOLOPEN OUTPUT command 579
 SPOOLREAD command 582
 SPOOLWRITE command 585
 UNLOCK command 629

TOKENERR condition
 ENDBROWSE ACTIVITY command 170
 ENDBROWSE CONTAINER command 171
 ENDBROWSE EVENT command 172
 ENDBROWSE PROCESS command 173
 GETNEXT ACTIVITY command 266
 GETNEXT CONTAINER command 268
 GETNEXT EVENT command 270
 GETNEXT PROCESS command 271
 WEB CONVERSE command 665
 WEB SEND command (Client) 712

TOLENGTH option
 CONVERSE (non-VTAM) command 105
 CONVERSE (VTAM) command 98
 fullword length alternative (TOLENGTH) 767
 WEB CONVERSE command 663

TOPROCESS option
 MOVE CONTAINER (BTS) command 364

TRACENUM option
 ENTER TRACENUM command 179

TRAILER operand
 DFHMDI 806

TRAILER option
 SEND PAGE command 543
 SEND TEXT command 551

TRANPRIORITY option
 ASSIGN command 58

transfer program control 742

TRANSID option
 CANCEL command 69
 DEFINE ACTIVITY command 110
 DEFINE PROCESS command 120
 INQUIRE ACTIVITYID command 281
 LINK command 346
 RETURN command 485
 SEND PAGE command 543
 START ATTACH command 599
 START BREXITcommand 601
 START command 594
 START TRANSID (CHANNEL) command 605

TRANSIDERR condition
 DEFINE ACTIVITY command 111
 DEFINE PROCESS command 121
 START ATTACH command 599
 START BREXIT command 601
 START command 596
 START TRANSID (CHANNEL) command 607

transient data commands 26

transient data control
 delete intrapartition queue 145
 read data from TD queue 424
 write data to TD queue 734

translated code 10

TRANSP operand
 DFHMDF 798
 DFHMDI 807
 DFHMSD 816

TRIGGER option
 HANDLE AID command 275

TRIGGER value
 DFHMDF 798
 DFHMDI 807
 DFHMSD 817

TRIGRAPH operand
 DFHMSD 816

TRT option
 DUMP TRANSACTION command 166

TSIOERR condition
 PURGE MESSAGE command 375
 SEND CONTROL command 529
 SEND MAP command 537
 SEND PAGE command 544
 SEND TEXT command 552
 SEND TEXT MAPPED command 554
 SEND TEXT NOEDIT command 559

TWA option
 ADDRESS command 35

TWALENG option
 ASSIGN command 58

type codes (terminal) 763

TYPE operand
 DFHMSD 816

TYPE option
 WEB RECEIVE command (Server) 691

U

- UNATTEND option
 - ASSIGN command 58
- UNCOMMITTED
 - READ command 396
- UNCOMMITTED option
 - READNEXT 409
 - READPREV command 420
- UNDERLINE value
 - DFHMDF 792
 - DFHMDI 804
 - DFHMSD 813
- UNEXPIN condition
 - ISSUE ABORT command 296
 - ISSUE ADD command 298
 - ISSUE END command 307
 - ISSUE ERASE command 312
 - ISSUE NOTE command 319
 - ISSUE QUERY command 326
 - ISSUE RECEIVE command 328
 - ISSUE REPLACE command 331
 - ISSUE SEND command 335
 - ISSUE WAIT command 340
 - RECEIVE MAP command 454
- UNLOCK command 628
- UNPROT value
 - DFHMDF 791
- UNTIL option
 - DELAY command 126
- UPDATE COUNTER command 632
- UPDATE DCOUNTER command 632
- UPDATE option
 - QUERY SECURITY command 389
 - READ command 397
 - READNEXT 410
 - READPREV command 420
- updating records
 - batch data interchange 329
 - file control 490
- URIMAP option
 - WEB EXTRACT command 671
 - WEB OPEN command 661, 675, 711
- URL option
 - WEB PARSE URL command 678
- URLLENGTH option
 - WEB PARSE URL command 678
- USERDATAKEY option
 - GETMAIN command 264
- USEREXIT value
 - DFHMDF 798
 - DFHMDI 807
 - DFHMSD 817
- USERID option
 - ASSIGN command 58
 - CHANGE PASSWORD command 73
 - DEFINE ACTIVITY command 110
 - DEFINE PROCESS command 120
 - EXTRACT CERTIFICATE command 194
 - INQUIRE ACTIVITYID command 282
 - SIGNON command 563
 - SPOOLOPEN INPUT command 575

- USERID option (*continued*)
 - SPOOLOPEN OUTPUT command 579
 - START BREXIT command 601
 - START command 595
 - START TRANSID (CHANNEL) command 605
 - VERIFY PASSWORD command 637
- USERIDERR condition
 - CHANGE PASSWORD command 74
 - SIGNON command 564
 - START BREXIT command 601
 - START command 596
 - START TRANSID (CHANNEL) command 607
 - VERIFY PASSWORD command 638
- USERNAME option
 - ASSIGN command 58
- USERPRIORITY option
 - ASSIGN command 58
- USING option
 - ADDRESS SET command 36

V

- VALIDATION option
 - ASSIGN command 58
- VALIDN operand
 - DFHMDF 798
 - DFHMDI 807
 - DFHMSD 817
- VALUE option
 - DEFINE COUNTER command 115
 - DEFINE DCOUNTER command 115
 - DOCUMENT SET command 162
 - GET COUNTER command 258
 - GET DCOUNTER command 258
 - QUERY COUNTER command 385
 - QUERY DCOUNTER command 385
 - UPDATE COUNTER command 633
 - UPDATE DCOUNTER command 633
 - WEB READ FORMFIELD command 681
 - WEB READ HTTPHEADER command 682
 - WEB READNEXT FORMFIELD command 684
 - WEB READNEXT HTTPHEADER command 686
 - WEB WRITE HTTPHEADER command 718
- VALUELENGTH option
 - WEB READ FORMFIELD command 681
 - WEB READ HTTPHEADER command 682
 - WEB READNEXT FORMFIELD command 684
 - WEB READNEXT HTTPHEADER command 686
 - WEB WRITE HTTPHEADER command 718
- VERIFY PASSWORD command 636
- VERSIONLEN option
 - WEB EXTRACT command 672
- VIEWPOS operand
 - DFHPDI 819
- VIEWSIZE operand
 - DFHPDI 819
- VOLUME option
 - ISSUE ABORT command 296
 - ISSUE ADD command 298
 - ISSUE END command 307
 - ISSUE ERASE command 312

VOLUME option *(continued)*
 ISSUE NOTE command 318
 ISSUE QUERY command 325
 ISSUE REPLACE command 330
 ISSUE SEND command 334
 ISSUE WAIT command 340

VOLUMELENG option
 ISSUE ABORT command 296
 ISSUE ADD command 298
 ISSUE END command 307
 ISSUE ERASE command 312
 ISSUE NOTE command 318
 ISSUE QUERY command 325
 ISSUE REPLACE command 330
 ISSUE SEND command 334
 ISSUE WAIT command 340

VSAM WRITE MASSINSERT
 DISABLED cannot occur 629
 NOTOPEN cannot occur 630
 terminate operation 628

VTAB operand
 DFHMSD 817

VTAM logon data, access to 195

W

WAIT CONVID (APPC) command 639
 WAIT EVENT command 641
 WAIT EXTERNAL command 643
 WAIT JOURNALNAME command 646
 WAIT JOURNALNUM command 648

WAIT option
 GDS SEND command 246
 ISSUE COPY (3270 logical) command 301
 ISSUE ERASEAUP command 313
 RETRIEVE command 476
 SEND (non-VTAM) command 523
 SEND (VTAM) command 517
 SEND command 768
 SEND CONTROL command 528
 SEND MAP command 536
 SEND TEXT command 551
 SEND TEXT MAPPED command 554
 SEND TEXT NOEDIT command 558
 terminal control 769
 WRITE JOURNALNAME command 727

WAIT SIGNAL command 649
 WAIT TERMINAL command 650
 general information 769

WAITCICS command 652

waits
 batch data interchange 339
 for event to occur 641
 terminal control operation 769

WEB CLOSE command 654
 WEB CONVERSE command 657
 WEB ENDBROWSE FORMFIELD command 666
 WEB ENDBROWSE HTTPHEADER command 667
 WEB EXTRACT command 668
 WEB OPEN command 673
 WEB PARSE URL command 677

WEB READ FORMFIELD command 680
 WEB READ HTTPHEADER command 682
 WEB READNEXT FORMFIELD command 684
 WEB READNEXT HTTPHEADER command 686
 WEB RECEIVE command (Client) 693
 WEB RECEIVE command (Server) 688
 WEB RETRIEVE command 697
 WEB SEND command (Client) 706
 WEB SEND command (Server) 698
 WEB STARTBROWSE FORMFIELD command 713
 WEB STARTBROWSE HTTPHEADER command 715

web support 26

WEB WRITE HTTPHEADER command 716

WPMEDIA option
 ISSUE ABORT command 296
 ISSUE END command 307
 ISSUE SEND command 334
 ISSUE WAIT command 340

WRAP option
 GET COUNTER command 258
 GET DCOUNTER command 258

WRBRK condition
 CONVERSE (non-VTAM) command 106
 SEND (non-VTAM) command 524
 SEND CONTROL command 529
 SEND MAP command 537
 SEND PAGE command 544
 SEND TEXT command 552
 SEND TEXT MAPPED command 554
 SEND TEXT NOEDIT command 559

WRITE command 719
 WRITE JOURNALNAME command 726
 WRITE JOURNALNUM command 730
 WRITE OPERATOR command 731
 critical action 732
 eventual action 732
 immediate action 732

WRITEQ TD command 734
 WRITEQ TS command 737

writing data
 to temporary storage queue 737
 to terminal or logical unit 768
 to transient data queue 734

writing records to data sets
 batch data interchange 297
 file control 719

X

XCTL command 742
 XINIT operand
 DFHMDF 798

XRF, generic applid 48

Y

YEAR option
 DEFINE TIMER command 124
 FORMATTIME command 208

YES value
 DFHMDF 803, 805, 806

YES value *(continued)*
DFHMSD 812, 813, 816
YYDDD option
FORMATTIME command 208
YYDDMM option
FORMATTIME command 208
YYMMDD option
FORMATTIME command 208
YYYYDDD option
FORMATTIME command 208
YYYYDDMM option
FORMATTIME command 208
YYYYMMDD option
FORMATTIME command 208

Z

ZERO value
DFHMDF 793

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Programming interface information

This book is intended to help you write application programs using EXEC CICS commands that obtain the services of CICS.

This book documents General-use Programming Interface and Associated Guidance Information provided by CICS.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of IBM or other companies.



Product Number: 5655-M15

SC34-6434-08



Spine information:



CICS Transaction Server for z/OS Application Programming Reference

Version 3
Release 1