CICS® Transaction Server for VSE/ESA™

Enhancements Guide

Release 1

GC34-5763-03

CICS® Transaction Server for VSE/ESA™

Enhancements Guide

Release 1

GC34-5763-03

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 263.

Third Edition (June 2002)

This edition applies to Release 1 Modification Level 1 of the IBM licensed program CICS Transaction Server for VSE/ESA, program number 5648-054, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Consult the latest edition of the applicable IBM system bibliography for current information on this product.

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

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

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

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

Contents

Preface

What this book is about

This book provides a brief summary of the changes made in the Application Programming Interface, the CICS® Resource Definitions Online, the CICS System Initialization parameters, Trace Entries and the Web Support since Release 1 of CICS Transaction Server for VSE/ESA®.

How to use this book

This book is intended to show what CICS facilities are available to enable you to use your CICS system as a non-SNA server. Read Part 1, “Introduction” on page 1 for general information, and for guidance about which other parts of the book to consult.

What you need to know to understand this book

Within this book, reference is made to Release 1. The function described in this book has been added since the release of CICS Transaction Server for VSE/ESA and can be identified within the code as 1.1.1.

This book assumes that you are familiar with CICS, either as a system administrator or as a system or application programmer. Some parts of the book assume additional knowledge about CICS and other products.

Notes on terminology

When the term “CICS” is used without any qualification in this book, it refers to the CICS element of IBM® CICS Transaction Server for VSE/ESA.

In this release, the CICS Web interface is split into the Listener support for TCPIPSERVICE, and the protocol support for HTTP. This book now refers to the HTTP protocol support as ″**CICS Web support**″. Within the product code, the term ″**CICS Web interface**″ remains synonymous with ″**CICS Web support**″.

In this release, there are two ways of coding Web application programs.

1. **Commarea**-style applications are those that take a communication area containing an HTTP request as input, and build an HTTP response in the communication area.
2. **Web API** applications use the new WEB and DOCUMENT application programming interface to process the inbound HTTP request and build the response.

Summary of Changes

Changes for Third Edition (June 2002)

The Third Edition of this manual contains these main changes:

- The procedure you follow to configure your Web browsers (CICS clients) for client authentication has been changed. For details, see “Step 7: Configure for Client Authentication (Optional)” on page 79.
- The transaction `EXEC CICS EXTRACT CERTIFICATE` has been removed for your use. This is due to a conflict with the client authentication programs. However, as soon as the conflict has been resolved, this transaction will be made available for your use.
- Various minor corrections have been made to this manual.

Changes for Second Edition (December 2001)

The Second Edition of this manual contained these main changes:

- You can configure the CICS Transaction Server to use Secure Sockets Layer (SSL) security. For details, see Part 3, “Using Secure Sockets Layer (SSL)” on page 69.
- The CICS Transaction Server supports the TCP/IP protocol for connections to the CICS Transaction Gateway, which enables the CICS Universal Client to use the *External Call Interface (ECI) via TCP/IP*. For details, see Part 4, “Using the CICS Transaction Gateway with ECI” on page 87.

Part 1. Introduction

Chapter 1. Introduction

The CICS for Transaction Server for VSE/ESA Version 1 Enhancements Guide is intended to document those changes made to CICS TS for VSE/ESA in support of CICS Web support, the 3270 Bridge facility, and SSL. This book also documents the CICS Transaction Server support for TCP/IP-protocol connections to the CICS Transaction Gateway, which enables the CICS Universal Client to use the *External Call Interface (ECI) via TCP/IP*.

This book shows those areas of CICS that have been enhanced, either directly to support these two new functions, or where other changes have been made.

Part 2. CICS Web support

CICS Web support

Chapter 2. Configuring CICS Web support

Note

This chapter is taken from the *CICS Transaction Server for VSE/ESA, Internet Guide*, SC34-5765 (September 2000). It has been updated here for Secure Sockets Layer (SSL) support. You might find useful information in the other chapters of the *Internet Guide* manual, such as:

- ″Introduction to CICS Web support″
- ″Planning for CICS Web support″
- ″Writing CICS programs to process HTTP requests″
- ″Security for CICS Web support″

This chapter provides an overview of how to configure CICS Web support. Table 1 is a checklist of what you need to do.

Table 1. Configuring CICS Web support

Task	See...	Task completed?
Specifying the appropriate system initialization (SIT) parameters.	"Controlling web support with system initialization parameters"	YES / NO
Creating the necessary resource definitions.	"Defining resources to CICS" on page 8	YES / NO
Reserving ports for CICS Web support	"Reserving ports for CICS Web support" on page 13	YES / NO
Specifying a name server (optional).	"Specifying a name server" on page 14	YES / NO
Enabling lightpen support (optional).	"Enabling lightpen support" on page 14	YES / NO
Running the sample application to test CICS Web support.	"Running the sample application" on page 15	YES / NO
Use SSL support (optional).	Chapter 9, "Configuring CICS to use SSL" on page 75	YES / NO

Controlling web support with system initialization parameters

CICS Web support is controlled initially by *system initialization parameters*. When CICS is running, you can make changes using CEMT and CEDA.

There are four CICS system initialization parameters relating to CICS Web support, and a further three parameters relating to SSL:

- If you are using Web 3270 support, you can use the WEBDELAY parameter to fix:
  - The length of time, in minutes, after which a Web task and its associated data is marked for deletion if no activity takes place on it.
  - The frequency, in minutes, with which the garbage collection transaction CWBG is run to delete the marked tasks and their data.
- The TCPIP parameter specifies whether CICS TCPIP services are to be activated at CICS startup. The default is NO, meaning that HTTP services cannot

be enabled, and you cannot use any TCPIPSERVICE resources defined with CEDA. If TCPIP is set to YES, HTTP services can be enabled and can then process work.

- If you intend to use Secure Sockets Layer (SSL) as described in Chapter 9, “Configuring CICS to use SSL” on page 75, there are *three* further CICS system initialization parameters that you must use:
  – the ENCRYPTION parameter to specify the level of encryption you want for TCP/IP connections using SSL.
  – the KEYFILE parameter to specify the name of the VSE Keyring Library on the VSE/ESA host that you wish to use with CICS.
  – the SSLDELAY parameter to specify the length of time for which CICS retains session IDs for secure socket connections.

  (You must also set parameter TCPIP to YES).

For details of all the system initialization parameters described here, see “The system initialization parameter descriptions” on page 176.

Defining resources to CICS

This section describes the resources needed to configure CICS Web support. It contains these topics:

- “CICS supplied resource definitions”
- “DOCTEMPLATE definitions”
- “TCPIPSERVICE definitions” on page 10
- “TRANSACTION definitions for extra alias transactions” on page 10
- “PROGRAM definitions for user-replaceable programs” on page 11
- “Setting up a sub-library for the template manager” on page 11
- “Defining a conversion table” on page 12

CICS supplied resource definitions

CICS Web support provides an RDO group defining the CICS resources used by the interface. The following definitions are in the locked group DFHWEB:

- Transactions required by CICS Web support (for example, CWBA and CWXN)
- Programs supplied with the CICS Web support

To change these definitions, you must copy them to your own RDO group and modify them there.

Sample CICS Web TCPIPSERVICE definitions are provided in the locked group DFH$SOT. To change these definitions, you must copy them to your own group and change them there.

The group DFH$WBSN contains the resource definitions for the security sample programs described in the chapter ″Security for CICS Web support″ of the *CICS Transaction Server for VSE/ESA, Internet Guide*, SC34-5765.

DOCTEMPLATE definitions

DOCTEMPLATE definitions allow you to perform variable substitution on documents in a manner similar to that done by BMS for 3270 screens. Templates can contain HTML, or binary data such as images. The data within the template is retrieved whenever a call is made for the template by means of an EXEC CICS DOCUMENT CREATE or EXEC CICS DOCUMENT INSERT command. The template can reside in any of the following places:

- VSE/ESA sub-library.
- CICS auxiliary temporary storage.
- CICS extrapartition transient data.
- CICS load module.
- CICS file.
- Exit program.

See the Chapter 14, “CEDA DEFINE DOCTEMPLATE” on page 109 for details of how to define a DOCTEMPLATE, and information about programming with documents and the associated EXEC CICS DOCUMENT commands.

VSE/ESA sub-library

You can use any editor you like to create the templates as members of this sub-library. The record format must be F (fixed). The templates can contain sequence numbers as follows:

- F format, and LRECL 80: the sequence numbers must be in record positions 73 through 80.

In any other case, there must be no sequence numbers in the records. The template manager decides whether there are sequence numbers by looking at the first logical record of a member of the VSE/ESA sub-library, so members that are only partially sequenced might be interpreted incorrectly. The default library name is DFHHTML. The sub-library name must be DFHDOC. The members must have a member type of HTML.

CICS temporary storage

Define one TSQUEUE for each template. The document handler domain returns an error if a request for a template is made to a non-existent TSQUEUE.

CICS transient data

Define an extrapartition TDQUEUE for each template. If you use an intrapartition transient data queue, your data is lost as soon as it has been read. If you use an extrapartition data queue, you must reset the queue after reading it.

CICS load module

Compile and link-edit a data-only load module. For example, an Assembler CSECT could contain a PROLOG containing your own control information, an ENTRY statement, any number of DC statements containing the HTML you want to output (you must put your own linefeeds in), and an END statement. CICS assumes that the entry point of the load module delimits the start of the template.

CICS file

This can be any CICS-controlled file.

Exit program

This is called whenever a request is made for the template. CICS passes a commarea to the exit program which is mapped by the following copybooks:

- DFHDHTXD (Assembler)
- DFHDHTXH (C)
- DFHDHTXL (PL/I)
- DFHDHTXO (COBOL)

The commarea contains the address (`dhtx_buffer_ptr`) and length (`dhtx_buffer_len`) of a CICS-supplied buffer in which the EXITPGM must return the template. The actual length of the template must be returned in `dhtx_template_len`. If the template to be returned is longer than `dhtx_buffer_len`, the template must be truncated to length `dhtx_buffer_len` and the EXITPGM must set the length required in `dhtx_template_len`. The EXITPGM is then called again with a larger buffer.

TCPIPSERVICE definitions

For HTTP requests to be submitted directly to CICS, you need one or more TCPIPSERVICE resources to be installed.

The TCPIPSERVICE definition allows you to define which TCP/IP services are to use CICS internal Sockets support. The internal CICS service that can be defined is CICS Web support.

The TCPIPSERVICE definition allows you to manage these internal CICS interfaces, with CICS listening on multiple ports, with different flavors of CICS Web support on different ports.

You must install and open a TCPIPSERVICE definition for each port on which CICS is to listen for incoming HTTP requests. You can create your own TCPIPSERVICE definition, or copy the HTTPNSSL definitions from the DFH$SOT group into your own group and modify them to meet your system requirements.

The important attributes for a Web TCPIPSERVICE are:

- The STATUS must be OPEN
- The TRANSACTION to be attached by CICS when new work arrives on the specified port must be CWXN or a user-defined alias of CWXN, which must invoke DFHWBXN as the initial program.
- The port on which CICS is to listen
- The backlog of requests to be processed which TCP/IP for VSE/ESA is to allow
- The name of the analyzer user-replaceable module to be driven for TCPIPSERVICE
- The IP address on which CICS is to listen for incoming requests.
- The TS queue name. This is the 6–character prefix of TS queue names generated by CICS Web support when writing inbound and outbound data to temporary storage. If no prefix is supplied on the definitions, the default name of DFHWEB is used to generate TS queue names.
- The name of a certificate contained in the VSE Keyring Library, that TCP/IP services should use.
- The level of SSL to be used (either no SSL support, SSL support with server authentication only, or SSL support with server authentication *and* client authentication).

For more information on defining Web TCPIPSERVICEs, see Chapter 16, “CEDA DEFINE TCPIPSERVICE” on page 119.

TRANSACTION definitions for extra alias transactions

Two CICS transactions are provided with CICS Web support:

Web attach transaction (CWXN).

This CICS-supplied transaction invokes the analyzer program. It establishes the context in which the alias transaction CWBA is to run, and issues the appropriate ATTACH command. When CWXN is defined as the TRANSACTION on the TCPIPSERVICE definition, it is started by the sockets listener task CSOL when a new connection request is received on the port specified on the TCPIPSERVICE definition. If the HTTP 1.0 Keep-Alive header has been sent by the Web browser, CWXN remains in the system after the alias has been attached, and attaches new alias transactions to process further HTTP requests received from browser. If Keep-Alive has not been specified, CWXN terminates after the alias has been attached.

Alias transaction CWBA.

An alias transaction is a CICS-supplied transaction that is started by the Web attach transaction (CWXN) to process a single request. Many instances of the alias transaction can be active in a CICS system at the same time, each processing a different request. The alias transaction runs the CICS-supplied alias program that calls the CICS program. If you wish, you may set up additional transaction definitions for alias transactions, each using the CICS-supplied alias program.

You may want to use other alias transaction names for various reasons:

- Auditing
- Resource and command checking
- Allocating initiation priorities
- Allocating database plan selection
- Assigning different runaway values to different CICS programs

If you do want to use other alias transaction names, you must copy the definition of CWBA, making the necessary changes. The definition of CWBA is as follows:

```
DEFINE TRANSACTION(CWBA)   GROUP(DFHWEB)
       PROGRAM(DFHWBA)     TWASIZE(0)
       PROFILE(DFHCICST)   STATUS(ENABLED)
       TASKDATALOC(ANY)    TASKDATAKEY(USER)
       RUNAWAY(SYSTEM)     SHUTDOWN(DISABLED)
       PRIORITY(1)         TRANCLASS(DFHTCL00)
       DTIMOUT(NO)         INDOUBT(BACKOUT)
       SPURGE(YES)         TPURGE(NO)
       RESSEC(NO)          CMDSEC(NO)
```

You cannot change the program name in this definition. Only the CICS-supplied alias program DFHWBA can be used. All the extra alias transactions must be local transactions.

PROGRAM definitions for user-replaceable programs

Each incoming request is serviced by a CICS program that provides transaction processing services, and by two other user-replaceable programs, an analyzer (required) and a converter (optional).

If you are not using autoinstall for programs, you must define all the user-replaceable programs you use. If you are using autoinstall for programs, you do not need to define the converters. In any case analyzers must be defined with EXECKEY(CICS). All the user-replaceable programs must be local to the system in which CICS Web support is operating.

Setting up a sub-library for the template manager

If you use the HTML template manager for constructing HTTP responses, you may provide a VSE/ESA sub-library to hold the templates. You can use any editor you like to create the templates as members of this data set. The record format must be F (fixed). The templates can contain sequence numbers as follows:

- F format, and LRECL 80: the sequence numbers must be in record positions 73 through 80.

In any other case, there must be no sequence numbers in the records. The template manager decides whether there are sequence numbers by looking at the first logical record of a member of the VSE/ESA sub-library, so members that are only partially sequenced might be interpreted incorrectly.

Any VSE/ESA library can be used to specify sub-library member templates, as specified in the DOCTEMPLATE definition. If you are using the template manager (DFHWBTL) or the Web bridge (DFHWBTTA), references to templates that are not defined and installed as DOCTEMPLATE definitions are resolved as members of the library specified as DFHHTML. The sub-library name must be DFHDOC. The members must have a member type of HTML.

Defining a conversion table

If you have commarea-style Web applications which do not use the Web API, or you are using CICS Web support to run a terminal-oriented transaction, you need to create or modify a DFHCNV table for data conversion to allow CICS to deal with incoming requests. The use of the DFHCNV macro for defining the table is described in *CICS Family: Communicating from CICS on System/390*. There are two kinds of data conversion performed in CICS Web support:

Conversion of the HTTP header information. This information is always transmitted as ASCII data using the ISO 8859-1 (Latin-1) character set.
This is the base character set for HTTP and HTML. This data has to be translated into EBCDIC. The conversion template name that the server controller supplies to the DFHCCNV program, which does the translation, is DFHWBHH.

Conversion of the HTTP user data.
This information is transmitted in the code page of the HTTP client, and can be translated into EBCDIC if required. The conversion template name is supplied by the analyzer. If the request is not an HTTP request, all the request is translated using the name supplied by the analyzer.

You might also refer to the sample conversion table DFHCNV contained in VSE Library 59.

For data conversion of the HTTP headers, you need to create a conversion template as follows:

```
DFHCNV TYPE=ENTRY,                                          *
      RTYPE=PC,                                             *
      CLINTCP=8859-1,                                       *
      SRVERCP=037,                                          *
      RNAME=DFHWBHH,                                        *
      USREXIT=NO
DFHCNV TYPE=SELECT,OPTION=DEFAULT
DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=CHARACTER,DATALEN=32767,   *
      LAST=YES
```

In the TYPE=ENTRY macro, the RNAME parameter must be DFHWBHH. The code page specifications CLINTCP and SRVERCP will get the HTTP request headers translated from ASCII to EBCDIC, and the HTTP response headers translated from EBCDIC to ASCII. The TYPE=SELECT and TYPE=FIELD macros must be coded exactly as shown.

For each name that the analyzer might specify for translating user data in the request from the client code page into EBCDIC, and for translating the user data in the response from EBCDIC to the client code page, you need to create a conversion template as follows:

```
DFHCNV TYPE=ENTRY,                                          *
      RTYPE=PC,                                             *
      CLINTCP=8859-1,                                       *
      SRVERCP=037,                                          *
      RNAME=DFHWBUD,                                        *
      USREXIT=NO
DFHCNV TYPE=SELECT,OPTION=DEFAULT
DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=CHARACTER,DATALEN=32767,   *
      LAST=YES
```

In the TYPE=ENTRY macro, the CLINTCP parameter must specify the code page of the client, and the RNAME parameter must specify the name that the analyzer will supply. The sample entry above supports translation of user data in the request from ASCII to EBCDIC, and of the user data in the response from EBCDIC to ASCII, for the default analyzer, which uses the name DFHWBUD. You may code the TYPE=SELECT and TYPE=FIELD macros in any way that is appropriate to the format of the user data that the client sends.

You may use the TYPE=INITIAL macro to set defaults for some of the values specified in these samples, as explained in *CICS Family: Communicating from CICS on System/390*.

The following sample shows a complete definition of the conversion templates for use with a Web browser using a Japanese double-byte character set. The code page 932 is one of several code pages for Japanese Web browsers, and 931 is one of the corresponding System/390® code pages. This sample can be used with the default analyzer. The sample must be assembled using the High-Level Assembler before being used.

```
DFHCNV TYPE=INITIAL
DFHCNV TYPE=ENTRY,RTYPE=PC,RNAME=DFHWBHH,USREXIT=NO,          *
      SRVERCP=037,CLINTCP=8859-1
DFHCNV TYPE=SELECT,OPTION=DEFAULT
DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=CHARACTER,DATALEN=32767,   *
      LAST=YES
DFHCNV TYPE=ENTRY,RTYPE=PC,RNAME=DFHWBUD,USREXIT=NO,          *
      CLINTCP=932,SRVERCP=931
DFHCNV TYPE=SELECT,OPTION=DEFAULT
DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=CHARACTER,DATALEN=32767,   *
      LAST=YES
DFHCNV TYPE=FINAL
END
```

.

Configuring TCP/IP for VSE/ESA

This section describes the changes you must make to TCP/IP for VSE/ESA as part of configuring CICS Web support.

Reserving ports for CICS Web support

You are recommended to reserve as many ports as you need for CICS Web support, and to ensure that CICS Web support has exclusive use of those ports.

Application programmers may use port numbers from 256 to 32 767 for nonstandard servers. For VSE/ESA, be careful when choosing port numbers less than 1024 — the ″well-known″ ports.

The maximum length of any queue of requests for a TCP/IP port on which a program is listening is controlled by TCP/IP for VSE/ESA, and is currently defined to be equivalent to a maximum of 10. This cannot be modified.

Identifying the TCP/IP server

You must identify the TCP/IP for VSE/ESA server by specifying a name for its IP address, otherwise CICS Web Support will not initialize successfully. You use the TCP command DEFINE NAME to do so. For example:

```
DEFINE
NAME,NAME=WINVSE.HURSLEY.IBM.COM,IPADDR=9.20.101.97
```

where NAME is a meaningful host name, and IPADDR specifies your server IP address.

If you do not do this, you will see the following error message during CICS start-up:

```
DFHSO0117 applid Unable to determine the TCP/IP host name.
Language Environment return code X'00000458', return code X'00000000'.
TCP/IP services are unavailable.
```

Specifying a name server

If you want full CICS function (that is, if you want to use DFH$WBSN and DFHWBENV), CICS Web support needs to access a name server during its operation. You set the name server in TCP/IP for VSE/ESA using the TCP command SET DNS1. For example:

```
SET DNS1=n.n.n.n
```

where *n.n.n.n* is the dotted decimal address of the name server.

To override the default search, you can code individual name, IP address pairs using the TCP command DEFINE NAME. For example:

```
DEFINE NAME,NAME=name,IPaddr=n.n.n.n
```

If the name server lookup fails when CICS runs:

- The security sample program DFH$WBSN does not execute correctly.
- The environment variables program DFHWBENV does not return a connection name in SERVER_NAME, but the dotted decimal address of the connection, and it also returns a null string for REMOTE_HOST.

Enabling lightpen support

To enable selector pen processing over the CICS Web support 3270 bridge, you must define a bridge facility with lightpen support enabled. To do this, follow these steps:

1. Copy the following definitions to a new group. Unless all applications running on the CICS system require lightpen support, you should also rename both definitions:
   - The CICS-supplied bridge facility CBRF, in group DFHTERM.
   - Its default TYPETERM, DFHLU2, in group DFHTYPE.
2. In the TYPETERM definition, change the LIGHTPEN option under ″DEVICE PROPERTIES″ to YES.

3. In the TERMINAL definition, change the TYPETERM parameter to point to the new TYPETERM.
4. Install the definitions in the CICS region.
5. If you have created a new bridge facility definition, update the PROFILE definition of the 3270 transaction which you are going to run with CICS Web support, so that the bridge facility will be modelled on the new TERMINAL/TYPETERM definition:
   a. Identify the PROFILE that the transaction uses by using CEDA to view the PROFILE parameter of the TRANSACTION definition.
   b. If the profile is a CICS-supplied profile, make a copy of it to your own group and rename it.
   c. Alter the new PROFILE and enter the name of your new bridge facility in the FACILITYLIKE parameter.
   d. Alter your TRANSACTION definition to use the new PROFILE definition.

Running the sample application

A sample application DFH$WB1A is provided to help you test the operation of CICS Web support. From a suitable Web browser, enter a URL that connects to CICS Web support with absolute path `/CICS/CWBA/DFH$WB1A`. The response displays the message "`DFH$WB1A on system` *xxxxxxxx* `successfully invoked through the CICS Web support`." with *xxxxxxxx* replaced by the application ID of the CICS system in which CICS Web support is running.

Chapter 3. EXEC CICS DOCUMENT

DOCUMENT CREATE

Create a document.

DOCUMENT CREATE

```
►►──DOCUMENT──CREATE──DOCTOKEN(data-area)──┬─────────────────────────────────────────────┬──►
                                           └─┬─┬─FROM(data-area)───┬──LENGTH(data-area)─┬─┘
                                             │ ├─TEXT(data-area)───┤                    │
                                             │ └─BINARY(data-area)─┘                    │
                                             ├─FROMDOC(data-area)───────────────────────┤
                                             └─TEMPLATE(name)───────────────────────────┘

►──┬──────────────────────────────────────────────┬──┬─────────────────────┬──┬──────────────────────┬──►◄
   └─┤ Symlist ├──┤ listln ├──┤ delim ├──┤ unesc ├─┘  └─DOCSIZE(data-area)──┘  └─HOSTCODEPAGE(name)──┘
```

symlist:

```
├──SYMBOLLIST(data-area)──────────────────────────────────────────────────────┤
```

listln:

```
├──LISTLENGTH(data-value)─────────────────────────────────────────────────────┤
```

delim:

```
├──┬──────────────────────────┬───────────────────────────────────────────────┤
   └─DELIMITER(data-value)────┘
```

unesc:

```
├──┬─────────────┬────────────────────────────────────────────────────────────┤
   └─UNESCAPED───┘
```

Conditions: INVREQ, NOTFND, SYMBOLERR, TEMPLATERR

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 substition. 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.

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.

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 application supplied buffer.

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. 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.

The name is case-sensitive. It can contain only uppercase and lowercase letters, numbers, and underscores (″_″). 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. Symbols in the symbol list are separated by the & character. For example:

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

The following restrictions apply to the characters allowed in the value in the symbol list:

- A percent sign may be followed by two hexadecimal digits. When the value is put into the symbol table, the 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. If you want a percent sign in the value in the symbol table, you may put %25 in the symbol list. 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. Note that this restriction does not apply if the UNESCAPED keyword has been specified.
- An ampersand is not allowed. If you want an ampersand in the value in the symbol table, you must put %26 in the value in the symbol list. Note that this restriction does not apply if the DELIMITER keyword is used to specify a delimiter other than ampersand.
- When the value is put into the symbol table, a plus sign is interpreted as a space. If you want a plus sign in the value in the symbol table, you must put %2B in the value in the symbol list. Note that this restriction does not apply if the UNESCAPED keyword has been specified.
- If you want a space in the value in the symbol table, the value in your symbol list may contain a space, a plus sign, or %20. Note that this restriction does not apply if the UNESCAPED keyword has been specified.

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.

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 substition. The data will be marked as requiring conversion to the client code page when the document is sent.

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.

Conditions

INVREQ

RESP2 values are:

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

8 The value specified for DELIMITER is not valid.

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.

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. This condition can also occur for umatched Shift-Out Shift-In pairs.

DOCUMENT INSERT

Insert document objects.

DOCUMENT INSERT

```
►►──DOCUMENT──INSERT──DOCTOKEN(data-area)──┬─┬─TEXT(data-area)───┬──LENGTH(data-value)─┬──────────►
                                           │ ├─BINARY(data-area)─┤                     │
                                           │ └─FROM(data-area)───┘                     │
                                           ├─SYMBOL(name)──────────────────────────────┤
                                           ├─TEMPLATE(name)────────────────────────────┤
                                           ├─FROMDOC(data-area)────────────────────────┤
                                           └─BOOKMARK(name)────────────────────────────┘

►──┬─────────────────────┬──┬────────────────────┬──┬────────────────────────┬──────────────►◄
   └─DOCSIZE(data-value)─┘  └─HOSTCODEPAGE(name)─┘  └─AT(name)──┬──────────┬─┘
                                                                └─TO(name)─┘
```

Conditions: DUPREC, INVREQ, NOTFND, TEMPLATERR

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.

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.

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.

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.

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.

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 specifed 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.

DOCUMENT RETRIEVE

Copy a document into the application's own buffer.

DOCUMENT RETRIEVE

```
►►──DOCUMENT──RETRIEVE──DOCTOKEN(data-area)──INTO(data-area)──LENGTH(data-value)──────────►

►─MAXLENGTH(data-value)──┬──────────────────────┬──┬──────────┬───────────────────────────►◄
                         └─CLNTCODEPAGE(name)─┘  └─DATAONLY─┘
```

Conditions: LENGERR, NOTFND

Description

DOCUMENT RETRIEVE allows the application to obtain a copy of the document in its own buffer, which it can then manipulate directly. The document is managed by CICS, and the application does not have direct access to the buffer containing the contents of the document. The document exists only for the duration of the current transaction, so the application must retrieve the document and store it if the document is to exist over transaction boundaries. The retrieved document can be used as a basis for a new document by using the FROM option of the DOCUMENT CREATE command.

When the document is retrieved, CICS inserts tags into the document contents to identify the bookmarks and to delimit the blocks that do not require codepage conversion. To request a copy without tags, specify DATAONLY. The extracted document can also be converted into a single client codepage by using the CLNTCODEPAGE option.

Options

CLNTCODEPAGE(*name***)**
specifies the name of the client codepage to which the data should be converted. The name can be up to 40 characters in length; if it is shorter than 40 characters, it must be padded on the right with blanks.

DATAONLY
specifies that the data should be retrieved without any imbedded tags.

DOCTOKEN(*data-area***)**
specifies the 16-byte symbolic name of the document to be retrieved.

INTO(*data-area***)**
specifies the buffer that is to contain the copy of the document content.

LENGTH(*data-value***)**
specifies the length, as a fullword binary value, of the amount of data being returned to the application.

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.

DOCUMENT SET

Add symbols and values to symbol table.

DOCUMENT SET

```
►►──DOCUMENT──SET──DOCTOKEN(data-area)──┬─SYMBOL(name)──VALUE(data-area)────────────────┬──────────►
                                        └─SYMBOLLIST(data-area)──┬─────────────────────┬┘
                                                                 └─DELIMITER(data-value)┘

►─LENGTH(data-value)──┬───────────┬────────────────────────────────────────────────────────────►◄
                      └─UNESCAPED─┘
```

Conditions: NOTFND, SYMBOLERR

Description

DOCUMENT SET allows the application to add symbols and their associated values to the symbol table. If the symbol being added already exists in the table, it is replaced by the new definition.

Options

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.

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.

SYMBOLLIST(*data-area*)
specifies a buffer which contains a symbol list. 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.

The name is case-sensitive. It can contain only uppercase and lowercase letters, numbers, and underscores (″_″). 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). Symbols in the symbol list are separated by the & character. For example:

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

The following restrictions apply to the characters allowed in the value in the symbol list:

- A percent sign may be followed by two hexadecimal digits. When the value is put into the symbol table, the percent sign and the two hexadecimal digits following it are interpreted as the EBCDIC equivalent of the single ASCII characcter denoted by the two digits. If you want a percent sign in the value in the symbol table, you may put %25 in the symbol list. 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. Note that this restriction does not apply if the UNESCAPED keyword has been specified.
- An ampersand is not allowed. If you want an ampersand in the value in the symbol table, you must put %26 in the value in the symbol list. Note that this restriction does not apply if the DELIMITER keyword is used to specify a delimiter other than ampersand.
- When the value is put into the symbol table, a plus sign is interpreted as a space. If you want a plus sign in the value in the symbol table, you must put %2B in the value in the symbol list. Note that this restriction does not apply if the UNESCAPED keyword has been specified.
- If you want a space in the value in the symbol table, the value in your symbol list may contain a space, a plus sign, or %20. Note that this restriction does not apply if the UNESCAPED keyword has been specified.

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 as %2B are not converted to single byte values.

VALUE(*data-area*)
specifies an area containing the value to be associated with the SYMBOL.

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.

Chapter 4. EXEC CICS EXTRACT

EXTRACT TCPIP

Obtain information about TCPIP characteristics of the current transaction.

EXTRACT TCPIP

```
►►──EXTRACT──TCPIP──┬───────────────────────────────────────────────┬────────────────────►
                    └─CLIENTNAME(data-area)──CNAMELENGTH(data-area)─┘

►──┬───────────────────────────────────────────────┬─────────────────────────────────────►
   └─SERVERNAME(data-area)──SNAMELENGTH(data-area)─┘

►──┬─CLIENTADDR(data-area)──CADDRLENGTH(data-area)─┬──┬──────────────────────────┬──────►
   ├─CLIENTADDRNU(data-area)───────────────────────┤  └─TCPIPSERVICE(data-area)─┘
   ├─SERVERADDR(data-area)──SADDRLENGTH(data-area)─┤
   └─SERVERADDRNU(data-area)───────────────────────┘

►──┬───────────────────────┬──┬──────────────────────┬───────────────────────────────────►◄
   └─PORTNUMBER(data-area)─┘  └─PORTNUMNU(data-area)─┘
```

Conditions: INVREQ, LENGERR

Options

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. If the data exceeds the buffer length, a LENGERR condition is raised and the data is truncated.

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.

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:

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.

Chapter 5. EXEC CICS START

START BREXIT

Start a task in the 3270 bridge environment and associate it with the named bridge exit.

START BREXIT

```
►►──START──┬─BREXIT───────┬──TRANSID(name)──┬──────────────────────────────────────────────┬──────────►
           └─BREXIT(name)─┘                 └─BRDATA(data-area)──BRDATALENGTH(data-value)──┘

►──┬──────────────────┬──────────────────────────────────────────────────────────────────────────────►◄
   └─USERID(data-value)─┘
```

Conditions: INVREQ,LENGERR, NOTAUTH, PGMIDERR, TRANSIDERR, USERIDERR

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 the *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.

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)
⋮
```

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 transaction identifier specified in a START BREXIT command has not been defined to CICS.

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.

Chapter 6. EXEC CICS TS

DELETEQ TS

Delete a temporary storage queue.

DELETEQ TS

```
►►──DELETEQ TS──┬─QUEUE(name)─┬──┬─────────────────────┬──────────────────────────────────────►◄
                └─QNAME(name)─┘  └─SYSID(systemname)─┘
```

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

Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS 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.

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.

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

Default action: terminate the task abnormally.

READQ TS

Read data from temporary storage queue.

READQ TS

```
►►──READQ TS──┬─QUEUE(name)─┬──┬─INTO(data-area)─┬──┬───────────────────┬──┬─────────────────────┬──►
              └─QNAME(name)─┘  └─SET(ptr-ref)────┘  └─LENGTH(data-area)─┘  └─NUMITEMS(data-area)─┘

   ┌─NEXT──────────────┐
►──┼───────────────────┼──┬───────────────────┬─────────────────────────────────────────────────►◄
   └─ITEM(data-value)──┘  └─SYSID(systemname)─┘
```

Conditions: INVREQ, IOERR, ISCINVREQ, ITEMERR, LENGERR, NOTAUTH, QIDERR, SYSIDERR

Note for dynamic transaction routing

Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS 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.

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.

QNAME(*name*)
an alternative to QUEUE, QNAME specifies the symbolic name (1–16 characters) of the queue to be read from.

SET(*ptr-ref*)
specifies the pointer reference that is set to the address of the retrieved data. The pointer reference, unless changed by other commands or statements, is valid until the next READQ TS command or the end of task.

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

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

SYSID(*systemname*)
(remote 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
occurs when there is an irrecoverable input/output error.

Default action: terminate the task abnormally.

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

Default action: terminate the task abnormally.

ITEMERR
occurs in any of the following situations:

- The item number specified is invalid (that is, outside the range of item numbers written to the queue).
- An attempt is made to read beyond the end of the queue using the NEXT (default) option.

Default action: terminate the task abnormally.

LENGERR

occurs when the length of the stored data is greater than the value specified by the LENGTH option.

This condition only applies to the INTO option and cannot occur with SET.

Default action: terminate the task abnormally.

NOTAUTH

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

Default action: terminate the task abnormally.

QIDERR

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

Default action: terminate the task abnormally.

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.

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
```

WRITEQ TS

Write data to a temporary storage queue.

WRITEQ TS

```
►►──WRITEQ TS──┬─QUEUE(name)─┬──FROM(data-area)──┬─────────────────────┬──────────────────────────►
               └─QNAME(name)─┘                   └─LENGTH(data-value)─┘

                                                           ┌─AUXILIARY─┐
►──┬───────────────────────────┬──┬─────────────────────┬──┼───────────┼──┬─────────────┬──────────►◄
   ├─NUMITEMS(data-area)───────┤  └─SYSID(systemname)─┘  └─MAIN──────┘  └─NOSUSPEND─┘
   └─ITEM(data-area)─┬─────────┬┘
                     └─REWRITE─┘
```

Conditions: INVREQ, IOERR, ISCINVREQ, ITEMERR, LENGERR, NOSPACE, NOTAUTH, QIDERR, SYSIDERR

> **Note for dynamic transaction routing**
> Using this command could create inter-transaction affinities that adversely affect the use of dynamic transaction routing. See the *CICS 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 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, and the NOSUSPEND option has been specified, or there is an active HANDLE CONDITION for NOSPACE, 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.

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.

MAIN
specifies that the temporary storage queue is in main storage.

This option is ignored for an existing queue.:

If you use the MAIN option to write data to a temporary storage queue on a remote system, the data is stored in main storage if the remote system is accessed by the CICS multiregion operation (MRO) facility. If these conditions are not met, the data is stored in auxiliary storage.

If the system is MRO and MAIN is specified, the queue is not recoverable and SYNCPOINT ROLLBACK does not function.

NOSUSPEND
specifies that the application program is not to be suspended if there is insufficient space in the temporary storage data set to satisfy the WRITEQ TS request. Instead, the NOSPACE condition is raised.

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.

This does not apply to temporary storage queues in main storage (MAIN option).

NUMITEMS(*data-area*)
specifies a halfword binary field into which CICS stores a number indicating how many items there are now in the queue, after the WRITEQ TS command is executed.

If the record starts a new queue, the item number assigned is 1; subsequent item numbers follow on sequentially. NUMITEMS is not valid if REWRITE is specified.

QUEUE(*name*)
specifies the symbolic name (1–8 characters) of the queue to be written to. If the queue name appears in the TST, and the entry is marked as remote, the request is shipped to a remote system. The name must be unique within the

CICS system. Do not use X'FA' through X'FF', or **, or $$, or DF, as the first character of the name; these characters are reserved for CICS use. The name cannot consist solely of binary zeros.

QNAME(*name*)
an alternative to QUEUE, QNAME specifies the symbolic name (1–16 characters) of the queue to be written to. If the queue name appears in the TST, and the entry is marked as remote, the request is shipped to a remote system. The name must be unique within the CICS system. Do not use X'FA' through X'FF', or **, or $$, or DF, as the first character of the name; these characters are reserved for CICS use. The name cannot consist solely of binary zeros.

REWRITE
specifies that the existing record in the queue is to be overwritten with the data provided. If the REWRITE option is specified, the ITEM option must also be specified. If the specified queue does not exist, the QIDERR condition occurs. If the correct item within an existing queue cannot be found, the ITEMERR condition occurs and the data is not stored.

SYSID(*systemname*)
specifies the system name (1–4 characters) identifying the remote system 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
occurs when there is an irrecoverable input/output error.

Default action: terminate the task abnormally.

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

Default action: terminate the task abnormally.

ITEMERR
occurs in any of the following situations:

- The item number specified 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.

NOSPACE
occurs when the NOSUSPEND option is specified and there is no space for the data in the auxiliary temporary storage data set.

Also occurs if there is no space and there is an active HANDLE CONDITION for NOSPACE.

Default action: ignore the condition.

NOTAUTH
occurs when a resource security check has failed on QUEUE(name), or QNAME(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

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).
- The link to the remote system is closed.

Default action: terminate the task abnormally.

Chapter 7. EXEC CICS WEB

WEB ENDBROWSE FORMFIELD

Signals the end of a FORMFIELD browse.

WEB ENDBROWSE FORMFIELD

```
►►──WEB──ENDBROWSE──FORMFIELD──────────────────────────────────────────────────►◄
```

Conditions: INVREQ

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
occurs for the following conditions. RESP2 values are:

1 The command is being issued in a non-CICS Web 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

Signals the end of an HTTP header browse.

WEB ENDBROWSE HTTPHEADER

```
►►──WEB──ENDBROWSE──HTTPHEADER──────────────────────────────────────────────────────►◄
```

Conditions: INVREQ

Description

WEB ENDBROWSE HTTPHEADER terminates the browse of HTTP headers. No information is returned on the ENDBROWSE.

Conditions

INVREQ
occurs for the following conditions. RESP2 values are:

1 The command is being issued in a non-CICS Web interface 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 EXTRACT

Obtain information about inbound client request.

WEB EXTRACT

```
►►──WEB──EXTRACT──┬──────────────────────────────────────────────────┬──────────────────────►
                  └─HTTPMETHOD(data-area)──METHODLENGTH(data-value)─┘

►─┬────────────────────────────────────────────────┬──┬──────────────────────────────────────┬──►
  └─HTTPVERSION(data-area)──VERSIONLEN(data-value)─┘  └─PATH(data-area)──PATHLENGTH(data-value)─┘

►─┬─────────────────────────────────────────────────┬──┬──────────────────────┬──────────────►◄
  └─QUERYSTRING(data-area)──QUERYSTRLEN(data-value)─┘  └─REQUESTTYPE(cvda)─┘
```

Conditions: INVREQ, LENGERR

Description

WEB EXTRACT allows the application to obtain additional information about the inbound request from the client.

Note: This command can also be specified as EXTRACT WEB.

Options

HTTPMETHOD(data-area)
specifies a buffer to contain the HTTP method string on the request line of the inbound message.

HTTPVERSION(data-area)
specifies a buffer to contain the HTTP version string on the request line of the inbound message.

METHODLENGTH(data-value)
specifies the length of the buffer supplied on the HTTPMETHOD 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. The actual length of the HTTPMETHOD is returned in METHODLENGTH.

PATH(data-area)
specifies a buffer to contain the PATH specified in the request line of the inbound message.

PATHLENGTH(data-value)
specifies the length of the buffer supplied on the PATH 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. The actual length of the PATH is returned in PATHLENGTH.

QUERYSTRING(data-area)
specifies a buffer to contain the query string on the request line of the HTTP request. The query string is the value or values encoded after the question mark (?) delimiting the end of the path on the request line of an HTTP request. The data is passed to the application in its escaped form.

QUERYSTRLEN(data-value)
specifies the length of the buffer supplied on the QUERYSTRING option. On

returning from CICS, QUERYSTRLEN is set to the actual length of the data returned to the application. If the data exceeds the buffer length, a LENGERR condition is returned and the data is truncated.

REQUESTTYPE(cvda)
specifies the type of request received. The request type can be HTTP or NONHTTP.

Conditions

INVREQ
occurs for the following conditions. RESP2 values are:

1 The command is being issued in a non-CICS Web interface 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
occurs for the following conditions. RESP2 values are:

4 The length in METHODLENGTH is less than or equal to zero.

5 The length in VERSIONLEN is less than or equal to zero.

6 The length in PATHLENGTH is less than or equal to zero.

8 The length in QUERYSTRLEN is less than or equal to zero.

WEB READ FORMFIELD

Retrieve the value of a field from an HTML form.

WEB READ FORMFIELD

```
►►──WEB──READ──FORMFIELD(data-area)──NAMELENGTH(data-value)──VALUE(data-area)──────────────►

►─VALUELENGTH(data-area)──┬─────────────────────────────────────────────┬──────────────────►◄
                          └─CLNTCODEPAGE(name)──HOSTCODEPAGE(name)─┘
```

Conditions: INVREQ, LENGERR, NOTFND

Description

WEB READ FORMFIELD retrieves the value of a specific field from an HTML form, the name of which is given on the request in the FORMFIELD parameter. The form 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 codepage to be used when data is converted from the client codepage. If this is not specified, CICS obtains it from the charset parameter on the Content-Type header of the HTTP request for the current CICS task. If the Content-Type header is not present, CICS uses codepage 819 (ISO-8859–1). If you specify CLNTCODEPAGE you must also specify HOSTCODEPAGE. Valid values for CLNTCODEPAGE are the CICS-supported CLINTCP and SRVERCP values described in *CICS Family: Communicating from CICS on System/390*. Mapping is performed between the IANA Coded Character Set values used in HTTP requests to the CLINTCP values known to CICS; these values are listed in the *CICS Internet Guide*.

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(name)
specifies the 8–character name of the host codepage to be used when the forms data is converted from the ASCII codepage, it was received in, into the EBCDIC, CICS processes it in. If this is not specified the default (037) is used. If you specify HOSTCODEPAGE you must also specify CLNTCODEPAGE.

NAMELENGTH(data-value)
specifies the length, as a fullword binary value, of the form field name.

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 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. VALUELENGTH contains the actual length of the data returned.

Conditions

INVREQ
RESP2 values are:

1 The command is being issued in a non-CICS Web application.

3 The command is being issued for a non-HTTP request.

11 The client codepage cannot be found.

12 The server codepage cannot be found.

13 No forms data has been supplied in the body of the HTTP request.

14 The codepage combination for client and server is invalid.

LENGERR
RESP2 values are:

1 The length in VALUELENGTH is less than or equal to zero.

2 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

```
►►──WEB──READ──HTTPHEADER(data-area)──NAMELENGTH(data-value)──VALUE(data-area)──────────►

►─VALUELENGTH(data-value)────────────────────────────────────────────────────────────►◄
```

Conditions: INVREQ, LENGERR, NOTFND

Description

WEB READ HTTPHEADER extracts HTTP header information.

Options

HTTPHEADER(data-area)
specifies the name of the HTTP header.

NAMELENGTH(data-area)
specifies the length, as a fullword binary value, of the HTTP header name.

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, 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. The actual length of the VALUE is returned in VALUELENGTH.

Conditions

INVREQ
RESP2 values are:

1 The command is being issued in a non-CICS Web interface application.

3 The command is being issued for a non-HTTP request.

LENGERR
RESP2 values are:

1 The length in VALUELENGTH is less than or equal to zero.

2 The header value has been truncated because the receiving buffer is too small.

NOTFND
RESP2 values are:

1 The header with the given name could not be found.

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

Description

WEB READNEXT FORMFIELD retrieves the next name-value pair in an HTML form.

Options

FORMFIELD(data-area)
specifies the buffer that contains the form field. The case of the name is the same 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. NAMELENGTH contains the actual length of the data returned.

VALUE(data-area)
specifies the buffer that contains 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 contains 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 VALUELENGTH contains the actual length of the data returned.

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 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 expected format.

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

```
►►──WEB──READNEXT──HTTPHEADER(data-area)──NAMELENGTH(data-value)──VALUE(data-area)──────────►

►─VALUELENGTH(data-value)──────────────────────────────────────────────────────────────────►◄
```

Conditions: ENDFILE, INVREQ, LENGERR

Description

WEB READNEXT HTTPHEADER retrieves the next HTTP header in the list of headers.

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, 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. The actual length of the HTTPHEADER name is returned in NAMELENGTH.

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, 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. The actual length of the VALUE is returned in VALUELENGTH.

Conditions

ENDFILE
The end of the list of HTTPHEADERS has been reached.

INVREQ
RESP2 values are:

1 The command is being issued in a non-CICS Web interface 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.

WEB RECEIVE

Receive an HTTP request.

WEB RECEIVE

```
►►──WEB──RECEIVE──┬─INTO(data-area)─┬──LENGTH(data-area)──MAXLENGTH(data-value)──┬─────────────┬──────►
                  └─SET(ptr-ref)────┘                                             └─TYPE(cvda)──┘

►──┬──────────────┬──┬──────────────────────────────────────────┬──────────────────────────────────►◄
   └─NOTRUNCATE───┘  └─CLNTCODEPAGE(name)──HOSTCODEPAGE(name)───┘
```

Conditions: INVREQ, LENGERR

Description

WEB RECEIVE receives the body of an HTTP Request from the CICS Web interface or Business Logic Interface into an application-supplied buffer.

Options

CLNTCODEPAGE(name)
specifies the 40-character name of the codepage to be used when data is converted from the client codepage. If this is not specified, data is not converted, and is passed to the application in the form in which it was received over the socket connection. If you specify CLNTCODEPAGE you must also specify HOSTCODEPAGE. Valid values for CLNTCODEPAGE are the CICS-supported CLINTCP and SRVERCP values described in *CICS Family: Communicating from CICS on System/390*. Mapping is performed between the IANA Coded Character Set values used in HTTP requests to the CLINTCP values known to CICS; these values are listed in the *CICS Internet Guide*.

HOSTCODEPAGE(name)
specifies the 8-character name of the host codepage to be used when data is converted from the ASCII codepage in which it was received into the EBCDIC codepage in which CICS will process it. If this is not specified, data is not converted, and is passed to the application in the form in which it was received over the socket connection.If you specify HOSTCODEPAGE you must also specify CLNTCODEPAGE.

INTO(data-area)
specifies the buffer that is to contain the data being received.

LENGTH(data-area)
specifies a fullword binary data area which will be set to the length of the data received.

MAXLENGTH(data-value)
specifies the maximum amount, as a fullword binary value, of data that CICS is to recover, if INTO is specified.

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.

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.

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 a value of HTTP or NONHTTP to identify whether the data received is an HTTP or non-HTTP message.

Conditions

INVREQ
RESP2 values:

1 the command is issued in a non-CICS Web interface application.

LENGERR
occurs if data is discarded by CICS because its length exceeds the maximum the program accepts. RESP2 values:

1 The length is negative.

2 The length is positive and exceeds the program’s maximum.

WEB RETRIEVE

Retrieve a document token.

WEB RETRIEVE

```
►►──WEB──RETRIEVE──DOCTOKEN(data-area)──────────────────────────────────►◄
```

Conditions: INVREQ

Description

The WEB RETRIEVE command retrieves the DOCTOKEN of the document which was sent using an earlier WEB SEND command.

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 interface application.

2 A WEB SEND command has not been issued.

WEB SEND

Select a document for delivery.

WEB SEND

```
►►──WEB──SEND──DOCTOKEN(name)──┬────────────────────┬──┬─────────────────────────┬──────────────────►
                               └─CLNTCODEPAGE(name)─┘  └─STATUSCODE(data-value)──┘

►──┬──────────────────────────────────────────────┬────────────────────────────────────────────────►◄
   └─STATUSTEXT(data-area)──LENGTH(data-value)────┘
```

Conditions: INVREQ, NOTFND

Description

WEB SEND selects a document for delivery by the CICS Web interface or the Business Logic Interface.

Options

CLNTCODEPAGE(name)
specifies the 40-character name of the ASCII codepage in which the data is to be delivered to the browser. If this is not specified, data is not converted, and is sent to the browser in the form in which the application supplied it. Valid values for CLNTCODEPAGE are the CICS-supported CLINTCP and SRVERCP values described in *CICS Family: Communicating from CICS on System/390*. Mapping is performed between the IANA Coded Character Set values used in HTTP requests and the CLINTCP values known to CICS; these values are listed in the *CICS Internet Guide*.

DOCTOKEN(name)
specifies the 16-byte symbolic name of the document to be sent.

LENGTH(data-value)
specifies the length, as a fullword binary value, of the string supplied on the STATUSTEXT option.

STATUSCODE(data-value)
is a binary halfword value to be inserted on the status line. The code is intended to be a HTTP status code and should conform to the rules for HTTP status codes. However, CICS does not validate the code and it is the application's responsibility to ensure that the value is valid.

STATUSTEXT(data-area)
specifies a data-area containing text describing the value of the status code. The status text is intended to be a human-readable value for the status code.

Conditions

INVREQ
occurs for the following conditions. RESP2 values are:

1 The command is being issued in a non-CICS Web interface application.

NOTFND
occurs for the following conditions. RESP2 values are:

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

7 The specified CLNTCODEPAGE is not recognised or is incorrectly specified.

WEB STARTBROWSE FORMFIELD

Signal the start of an HTML form field browse.

WEB STARTBROWSE FORMFIELD

```
►►──WEB──STARTBROWSE──FORMFIELD──┬────────────────────────────────────────────┬──────────────────────────►◄
                                 └─CLNTCODEPAGE(name)──HOSTCODEPAGE(name)─┘
```

Conditions: INVREQ

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 codepage to be used when data is converted from the client codepage. If this is not specified, CICS obtains it from the charset parameter on the Content-Type header of the HTTP request for the current CICS task. If the Content-Type header is not present, CICS used codepage 819 (ISO-8859–1). If you specify CLNTCODEPAGE you must also specify HOSTCODEPAGE. Valid values for CLNTCODEPAGE are the CICS-supported CLINTCP and SRVERCP values described in *CICS Family: Communicating from CICS on System/390*. Mapping is performed between the IANA Coded Character Set values used in HTTP requests to the CLINTCP values known to CICS; these values are listed in the *CICS Internet Guide*.

HOSTCODEPAGE(name)
specifies the 8–character name of the host codepage used when data is converted from the ASCII codepage, it was received in, into the EBCDIC, CICS processes it in. If this is not specified the default (037) is used. If you specify HOSTCODEPAGE you must also specify CLNTCODEPAGE.

Conditions

INVREQ
occurs for the following conditions. RESP2 values are:

1 The command is being issued in a non-CICS Web application.

3 The command is being issued for a non-HTTP request.

5 There is already a WEB STARTBROWSE in progress.

11 The client codepage cannot be found.

12 The server codepage cannot be found.

13 No forms data has been supplied in the body of the HTTP request.

14 The codepage combination for client and server is invalid.

WEB STARTBROWSE HTTPHEADER

Signal the start of an HTTP header browse.

WEB STARTBROWSE HTTPHEADER

```
►►──WEB──STARTBROWSE──HTTPHEADER──────────────────────────────────────────────►◄
```

Conditions: INVREQ

Description

WEB STARTBROWSE HTTPHEADER signals the start of a browse of the HTTP header information.

Conditions

INVREQ
occurs for the following conditions. RESP2 values are:

1 The command is being issued in a non-CICS Web interface application.

3 The command is being issued for a non-HTTP request.

WEB WRITE

Build HTTP header information.

WEB WRITE

```
►►──WEB──WRITE──HTTPHEADER(data-area)──NAMELENGTH(data-value)──VALUE(data-area)──────────────►

►─VALUELENGTH(data-value)──────────────────────────────────────────────────────────────────►◄
```

Conditions: INVREQ

Description

WEB WRITE allows the application to add HTTP header information to the response.

Options

HTTPHEADER(data-area)
specifies the name of the HTTP header to build. It is a string of text containing the name of the header. The name should conform to the standards specified in the current RFC for the HTTP protocol. No validation of the information is done by CICS. If the header already exists, it is replaced with the new information.

NAMELENGTH(data-value)
specifies the length, as a fullword binary value, of the HTTP header name.

VALUE(data-area)
specifies the value of the HTTP header. It is a text string containing the value of the header.

VALUELENGTH(data-area)
specifies the length, as a fullword binary value, of the HTTP header value.

Conditions

INVREQ
The command is being issued in a non-CICS Web interface environment.

Part 3. Using Secure Sockets Layer (SSL)

Chapter 8. Introduction to Secure Sockets Layer (SSL)

One of the major concerns when providing commercial services on the Internet is providing for transaction security and communications security.

Information exchanges are secure if all the following are true:
- Messages are confidential.
- The information exchange has integrity.
- Both sender and receiver are accountable.
- Both parties in the exchange can be authenticated.

Secure Sockets Layer (usually referred to as simply *SSL*) is a security protocol that was developed by Netscape Communications Corporation, along with RSA Data Security, Inc. The primary goal of the SSL protocol is to provide a private channel between communicating applications that ensures privacy of data, authentication of the partners, and integrity.

SSL is a protocol layer implemented on top of the standard TCP/IP socket API, which has security implemented within it. There are APIs for implementing SSL support for most languages (C language, Java, and so on).

This chapter consists of these sections:

Overview of SSL

SSL is a **handshake protocol** developed to provide security and privacy over the Internet. The SSL protocol uses encryption and authentication to ensure:

Privacy

The data to be exchanged between the client and the server is encrypted, so that only the intended recipient can read it. SSL uses public key encryption as a secure mechanism to distribute a secret key between the server and the client. Public key encryption is a technique that uses a pair of asymmetric keys for encryption and decryption. With SSL, one or more keys (public key, private key, session keys) are passed between the client and the server, using public key cryptography. These keys are then used to encrypt and decrypt all traffic along the SSL connection. This encryption protects the data from other parties trying to eavesdrop, as no other parties will have the secret key needed to decrypt the data. This ensures that private information, such as a credit card number, is transferred securely.

Integrity

The message transport includes a message integrity check based on a secure hashing algorithm. This algorithm is performed when the message is sent, and again when it is received. If the two hash values do not match, the receiver is warned that the message may have been tampered with.

Authentication

When a client establishes a connection with CICS, it may be required to authenticate its details to the server. The authentication mechanism is based on the exchange of digital certificates (X.509v3 certificates). These digital certificates contain information about an entity, such as the system name and public key, and the server's digital signature. Digital certificates are issued by a Certificate Authority, and encrypted using the Certificate Authority's private key. If you can decrypt the certificate using the Certificate Authority's public key, you know that the information contained within the certificate can be trusted (that is, that the certificate really does belong to whoever claims to own it).

SSL and the Web

The HTTPS protocol is a variant of HTTP for handling secure Web transactions. Most current browsers support the HTTPS URL access method to connect to HTTP servers that use SSL. A secure connection is made with a URL such as

```
https://www.company.com
```

If you use the HTTPS protocol without specifying a port number, a default port number of 443 is assumed.

Encryption and Keys

The SSL protocol operates between the application layer and the TCP/IP layer. This allows it to encrypt the data stream itself, which can then be transmitted securely, using any of the application layer protocols. Two encryption techniques are used:

- Public key cryptography standard (PKCS), which encrypts and decrypts certificates during the SSL handshake. Encryption keys are created in pairs, a public key and its associated private key. Data encrypted with a given public key can be decrypted only with the associated private key; this means that data is readable by only the intended recipient. Data encrypted with a given private key can be decrypted only with the associated public key; this means that authentication data is assured to originate from the owner of the private key.
- A mutually agreed symmetric encryption technique, such as DES (data encryption standard), or triple DES, is used in the data transfer following the handshake.

PKCS, as used by SSL, works briefly as follows:

1. A private and public key pair is requested, usually as part of a certificate application (see "Authentication and Certificates" on page 73 for details).
2. As part of the certificate creation, a private key and public key are created by means of an algorithm based on two random numbers. The private and public keys which result are related to each other such that:
   - **It is not feasible to deduce the value of the private key from the public key, nor the public key from the private key**

     The private key is stored securely, and is not made known to anyone but its owner. The public key can be made freely available to any user, with no risk of compromising the security of the private key.
   - **Information encrypted using the public key can be decrypted only with the private key**

     Information can be encrypted by any user, and sent securely to the holder of the private key. A third party cannot use the public key to read the information.
   - **Information encrypted using the private key can be decrypted only with the public key**

Only the holder of the private key can encrypt information that can be decrypted with the public key. A third party cannot pass as the sender of the information.

Authentication and Certificates

To make an environment secure, you must be sure that any communication is with ″trusted″ sites whose identity you can be sure of. SSL uses *certificates* for authentication — these are digitally signed documents which bind the public key to the identity of the private key owner. Authentication happens at connection time, and is independent of the application or the application protocol. Authentication involves making sure that sites with which you communicate are who they claim to be. With SSL, authentication is performed by an exchange of certificates, which are blocks of data in a format described in ITU-T standard X.509. The X.509 certificates are digitally signed by an external authority known as a certificate authority.

Certificates are digitally signed using the public-key encryption technique. The signature is created by partially encrypting the certificate with the certificate authority's private key. A user of the certificate is assured of the origin of the certificate when it is successfully decrypted by the certificate authority's public key.

Client authentication

When SSL is used, the *server certificate* which is used for *server authentication*, is mandatory. However, the *client certificate* which is used for *client authentication*, is optional: some clients may not support client certicates; other may not have certificates installed. Servers can decide whether to require client authentication for a connection.

The Role of Certificate Authorities

In order that one system can be assured that a certificate received from another system is genuine, a trusted third party that can vouch for the certificate is needed. Therefore *Certificate authorities* are independent bodies who act as the trusted third parties, by issuing certificates for use by others. Before issuing a certificate, a certificate authority will examine the credentials of the person or organisation that has requested the certificate. When the certificate has been issued, information about it is held on a publicly accessible repository. Users can consult the repository to check the status and validity of any certificates received.

Certificate authorities issue several levels of security certificates for different purposes. For example:

- Secure e-mail
- Client authentication
- Server authentication

Chapter 9. Configuring CICS to use SSL

This chapter explains how to configure CICS to use Secure Sockets Layer (SSL).

This chapter contains these sections:

- “Hardware prerequisites”
- “Software prerequisites”
- “Step 1: Configure the VSE/ESA Host”
- “Step 2: Decide Which Authorization Level You Require” on page 76
- “Step 3: Define the System Initialization Parameters” on page 76
- “Step 4: Define a TCPIPSERVICE Resource” on page 76
- “Step 5: Download the Client KeyRing File” on page 77
- “Step 6: Copy a Server Certificate Into Client Keyring File” on page 78
- “Step 7: Configure for Client Authentication (Optional)” on page 79
- “Currently-Supported SSL Cipher Suites” on page 85

Alternative Methods of Configuring for SSL!

Alternative methods for creating and managing certificates on the VSE/ESA host and on Web clients, are provided on the *VSE/ESA e-business Connectors and Utilities* home page, whose URL is:

```
http://www.ibm.com/servers/eserver/zseries/os/vse/support/vseconn/conmain.htm
```

Hardware prerequisites

To use SSL with VSE/ESA, you do not require any additional hardware.

Software prerequisites

These are the software prerequisites for using SSL:

- VSE/ESA Version 2 Release 6

Note!

Also check the VSE/ESA homepage for any APARs that you might need to install. You can find the VSE/ESA homepage at:

```
http://www.ibm.com/servers/eserver/zseries/os/vse/
```

Step 1: Configure the VSE/ESA Host

You should ensure that all steps described in the following chapter have been completed:

> “Configuring the VSE/ESA Host for SSL” in the *VSE/ESA e-business Connectors, User’s Guide*, SC33-6719.

Step 2: Decide Which Authorization Level You Require

You can decide to use:
- Server authentication only (which is mandatory).
- Server authentication *and* client authentication.

Server authentication is the process by which the CICS client (Web browser) can check that a server (in this case the VSE/ESA host) is “who he claims to be”, during the SSL “handshake”.

Client authentication is the process by which the server (in this case the VSE/ESA host) can check that the CICS client (Web browser) is “who he claims to be”, during the SSL “handshake”. See also “Authentication and Certificates” on page 73 for further details.

For applications such as flight booking systems, where the number and identity of potential clients is unknown, client authentication is probably not practical. In this case, *server authentication only* would be performed. However, you must carefully consider which level of authentication you require before making a decision.

If you decide that:
- *Server authentication only* is required, perform Steps 3 to 6 below, ensuring you set `SSL(YES)` in the TCPIPSERVICE profile.
- *Server and client authentication* is required, perform Steps 3 to 7 below, ensuring you set `SSL(CLIENTAUTH)` in the TCPIPSERVICE profile.

(TCPIPCSERVICE is described in “Step 4: Define a TCPIPSERVICE Resource”).

Step 3: Define the System Initialization Parameters

There are *four* system initialization parameters relating to SSL:
- ENCRYPTION={WEAK|**NORMAL**|STRONG}, described on page 195 of the chapter “CICS system initialization parameters”.
- KEYFILE=*name*, described on page 204 of the chapter “CICS system initialization parameters”.
- SSLDELAY=**600**|number, described on page 225 of the chapter “CICS system initialization parameters”.
- TCPIP{**NO**|YES}, described on page 231 of the chapter “CICS system initialization parameters”.

Step 4: Define a TCPIPSERVICE Resource

There are three TCPIPSERVICE resource definition attributes relating to SSL:
1. CERTIFICATE(*certificate-label*)
2. PORTNUMBER
3. SSL(NO|YES|CLIENTAUTH)

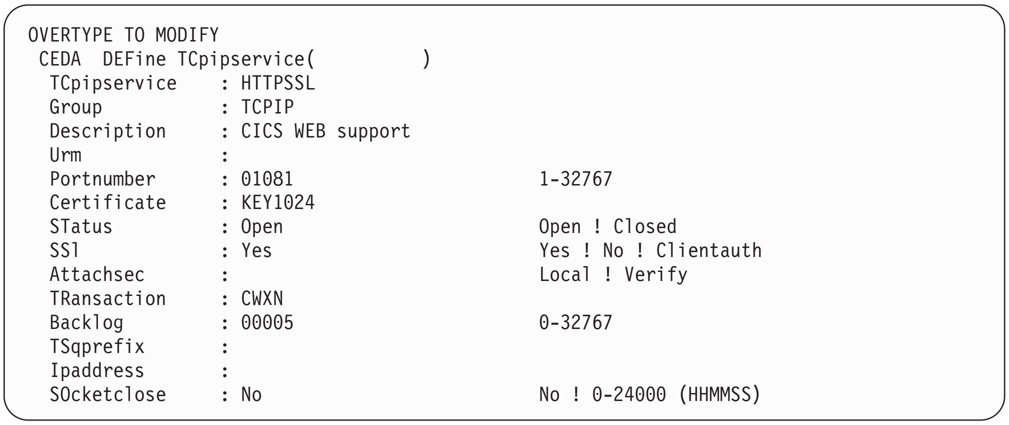
See Chapter 16, “CEDA DEFINE TCPIPSERVICE” on page 119 for a description of the above three attributes.

Note: If you do not enter a PORTNUMBER in your CICS client (Web browser), your CICS client will expect the default PORTNUMBER **443** (see “SSL and the Web” on page 72).

You must activate the TCPIPSERVICE definition either by:

- Specifying `STATUS(OPEN)` and installing the definition.
- Installing the definition and later using a `CEMT SET TCPIPSERVICE OPEN` command (described in Chapter 16, “CEDA DEFINE TCPIPSERVICE” on page 119.

Figure 1 shows an example of the entries you make when defining a TCPIPSERVICE definition for use with SSL:



```
OVERTYPE TO MODIFY
 CEDA  DEFine TCpipservice(          )
  TCpipservice    : HTTPSSL
  Group           : TCPIP
  Description     : CICS WEB support
  Urm             :
  Portnumber      : 01081                        1-32767
  Certificate     : KEY1024
  STatus          : Open                         Open ! Closed
  SSl             : Yes                          Yes ! No ! Clientauth
  Attachsec       :                              Local ! Verify
  TRansaction     : CWXN
  Backlog         : 00005                        0-32767
  TSqprefix       :
  Ipaddress       :
  SOcketclose     : No                           No ! 0-24000 (HHMMSS)
```

Figure 1. Example of a TCPIPSERVICE resource definition for SSL

Step 5: Download the Client KeyRing File

You can download the IBM-supplied *client keyring file* with the name **KeyRing.pfx** from the VSE/ESA homepage which you can find at:

`http://www.ibm.com/servers/eserver/zseries/os/vse/`

into each CICS client (Web browser).

Note: Before starting to download the client keyring file, you might wish to check the recommended Web browser settings described under “Step 7.5: Import Your Client Keyring File” on page 83.

The client keyring file contains a sample:

1. Client certificate, which includes a public key. This public key is one of a key pair – the matching private key (SAMPLE.PRVK) is installed on the VSE/ESA host during the configuration of the VSE/ESA host for SSL.
2. Root certificate, issued by the Thawte Corporation at **www.thawte.com**.

The client keyring file is initially protected by the password ‘ssltest’.

To:

- change the password,
- manage the certificates stored in the client keyring file,

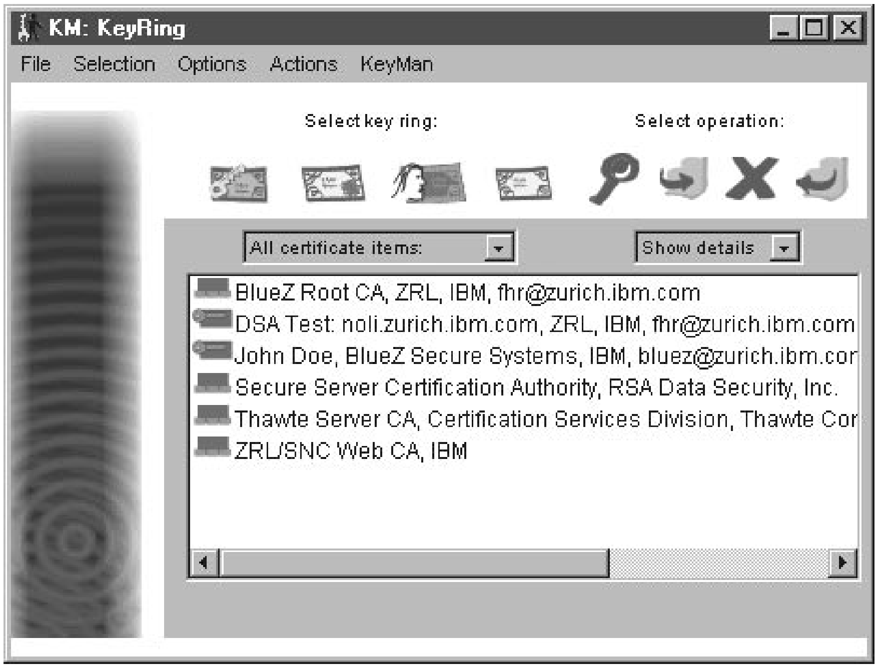
you can use:

1. The *IBM KeyMan Professional* tool (which you can download from **http://www.alphaworks.ibm.com/tech/keyman**).
2. The *ikeyman* tool (contained in the JSSE package of the *IBM Java Development Kit 1.3 for Windows*).

3. The **keytool.exe** (part of the Java Development Kit 1.3, from IBM and Sun Microsystems).

Note: The description contained in this manual is based upon the use of the *IBM KeyMan Professional* tool.

Figure 2 shows how the contents of the client keyring file might look, after server certificates and root certificates have been obtained and stored.



BlueZ Root CA, ZRL, IBM, fhr@zurich.ibm.com
DSA Test: noli.zurich.ibm.com, ZRL, IBM, fhr@zurich.ibm.com
John Doe, BlueZ Secure Systems, IBM, bluez@zurich.ibm.cor
Secure Server Certification Authority, RSA Data Security, Inc.
Thawte Server CA, Certification Services Division, Thawte Cor
ZRL/SNC Web CA, IBM

Figure 2. Example of a Client Keyring File

Step 6: Copy a Server Certificate Into Client Keyring File

A *server certificate* includes a public key, and is required so that the CICS client (Web browser) can perform *server authentication*.

Decide if you wish to use the IBM-supplied sample server certificate with each CICS client, or if instead you wish to use your *own* server certificate. Depending upon your decision, you must copy into the client keyring file (**KeyRing.pfx**) either:

- The IBM-supplied sample server certificate (`SAMPLE.CERT`).
- Your own server certificate.

If a CICS client does not contain a copy of a server certificate and communicates with the CICS Transaction Server running on the VSE/ESA host, the user will be prompted to decide if a copy of the server certificate should be stored on the CICS client. You can therefore either:

- Provide each CICS client with a copy of the server certificate before starting.
- Let each CICS client take a copy during the first processing that takes place.

Note: You can use a tool such as the IBM KeyMan (which you can download from **http://www.alphaworks.com**) to copy a server certificate into the client keyring file stored on a CICS client.

The IBM-provided sample server certificate enables your CICS client to start communicating immediately with the CICS Transaction Server running on the VSE/ESA host, using SSL *server authentication* support. You can use this sample server certificate for learning and testing purposes.

Step 7: Configure for Client Authentication (Optional)

This section describes the steps you must take to configure a CICS client (Web browser) for *client authentication*.

Digital certificates can be either root certificates, server certificates, or client certificates. If client authentication is required (in addition to server authentication), a *client certificate* is provided by clients to authenticate the client to the server.

Notes:

1. If you do not plan to implement client authentication, you can skip this section.
2. The description and pictures contained in this section are based upon the *IBM KeyMan* tool.
3. Instead of using the IBM Keyman tool, you could use (for example) the *ikeyman* tool (contained in the JSSE package of the *IBM Java Development Kit 1.3 for Windows*), or the **keytool.exe** (part of the Java Development Kit 1.3, from IBM and Sun Microsystems).

To configure a CICS client for client authentication, you should follow these steps:

- “Step 7.1: Generate a Key Pair”
- “Step 7.2: Generate and Store a Client Certificate” on page 80
- “Step 7.3: Import the CA’s Root Certificate into the Client Keyring File” on page 82
- “Step 7.5: Import Your Client Keyring File” on page 83
- “Step 7.4: File and Save Your Client Keyring File” on page 83
- “Step 7.6: Create Access Rights for CICS Clients to Use Host Resources” on page 84

Before beginning, you must ensure that you have set `SSL(CLIENTAUTH)` in the TCPIPSERVICE profile, as described in “Step 4: Define a TCPIPSERVICE Resource” on page 76.

Step 7.1: Generate a Key Pair

During the initial “handshake”, SSL uses a “key pair” consisting of:

- a private key
- a public key

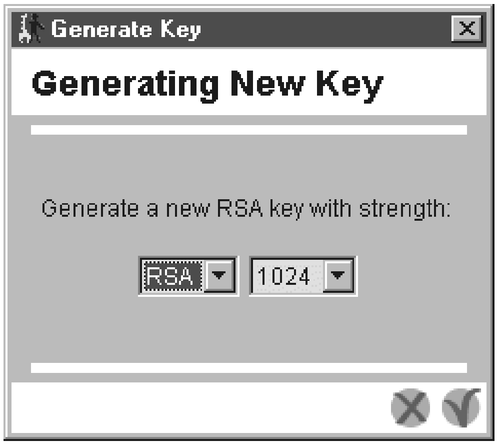
This description of how to generate a key pair uses the IBM *KeyMan* tool to generate a key pair (private and public keys). The public key will be later stored on a *client certificate*. You can obtain KeyMan from:

```
http://www.alphaworks.ibm.com/
```

However you can, of course, use another tool of your choice to generate a key pair and store them in the *client keyring file* (described in “Step 5: Download the Client KeyRing File” on page 77).

These are the steps you take to generate a key pair:

1. Start the KeyMan program. Open an existing client keyring file or create a new client keyring file (with PKCS#12 token) by clicking the appropriate icon in the first window.
2. In the *KM: Keyring* window, from the Menu pull-down select **Actions** and **Generate key**. The window shown in Figure 3 is displayed.



Generate Key

Generating New Key

Generate a new RSA key with strength:

RSA 1024

Figure 3. "Generate Key" Window

3. Select the key length you require, and start the generation process. After some seconds, a window *KM: New* is displayed, showing the new entry for a key pair.

   Notes:

   a. Although the window has the title *Generate New Key*, there are (always) *two* keys that are generated: a private key and a public key.
   b. You must ensure that the key length you choose matches the length of the key used on the VSE/ESA host:
      - Cipher suites that use 512-bit handshaking can only be used with a 512-bit key that is stored on the VSE/ESA host.
      - Cipher suites that use 1024-bit handshaking can only be used with a 1024-bit key that is stored on the VSE/ESA host.

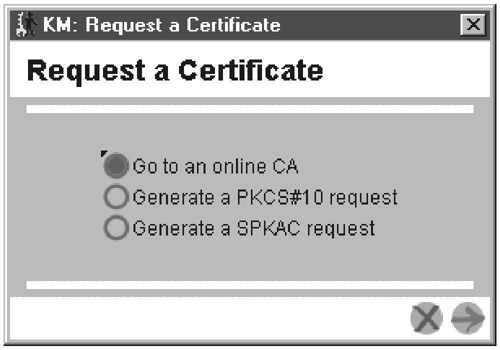
      Also see “Currently-Supported SSL Cipher Suites” on page 85 for further details.

Step 7.2: Generate and Store a Client Certificate

In this step, a client certificate is generated and then stored in the client keyring file. The client certificate will include the public key that was generated in the previous step.

To generate and store *your own* client certificate, you should:

1. Click on the key pair that was generated in the previous step, and from the menu pull-down select **Actions** — **Request Certificate**. The *KM: Request a Certificate* window is displayed, as shown in Figure 4 on page 81.


KM: Request a Certificate

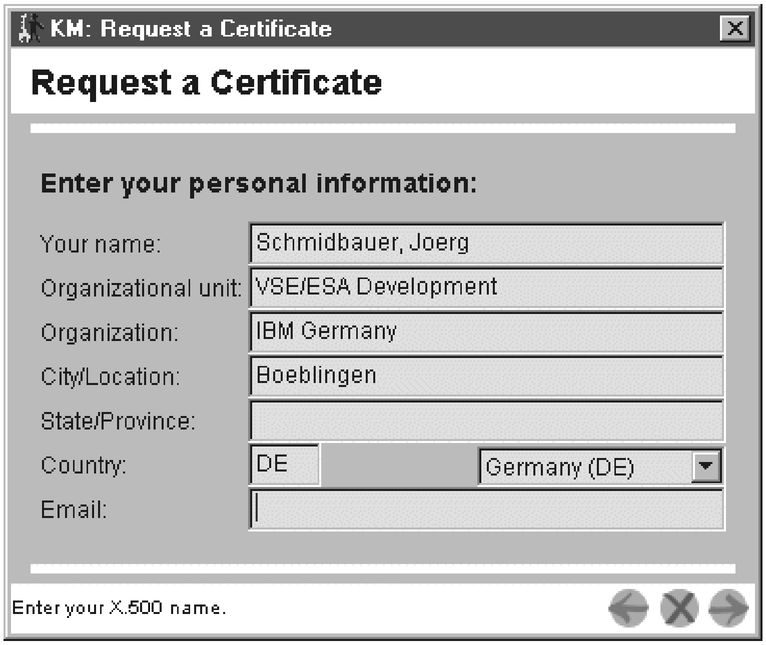
Request a Certificate

● Go to an online CA
○ Generate a PKCS#10 request
○ Generate a SPKAC request

Figure 4. "KM: Request a Certificate" Window – Choose Request Method

Now select **Go to an online CA**. Click **Next dialog page** (the arrow pointing to the right), and you are then given a choice of Certificate Authorities (CAs) from which you wish to obtain a certificate.

2. Select the **Thawte Trial Server** and click **Next dialog page**. This implementation uses certificates provided by the Thawte Corporation.
3. You are now presented with a dialog into which you enter your personal details (Figure 5).


KM: Request a Certificate

Request a Certificate

Enter your personal information:

Your name:	Schmidbauer, Joerg
Organizational unit:	VSE/ESA Development
Organization:	IBM Germany
City/Location:	Boeblingen
State/Province:	
Country:	DE Germany (DE)
Email:	

Enter your X.500 name.

Figure 5. "KM: Request a Certificate" Window – Enter Personal Details

4. KeyMan now launches your default Web browser, and connects to **www.thawte.com**. Your certificate request has been automatically copied onto your Clipboard.
5. Now you must decide if you wish to:
   - purchase a client certificate from Thawte for production use.
   - obtain a ***personal*** certificate from Thawte for testing purposes, which is free of charge, and use this as a client certificate.

   After you have pasted your certificate request into an area provided by Thawte, your client certificate is generated and displayed. Now you must:

a. Copy this client certificate to the Clipboard, and then import it into the client keyring file (Figure 6).

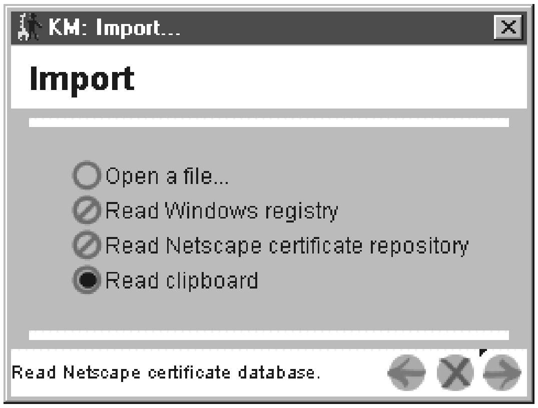
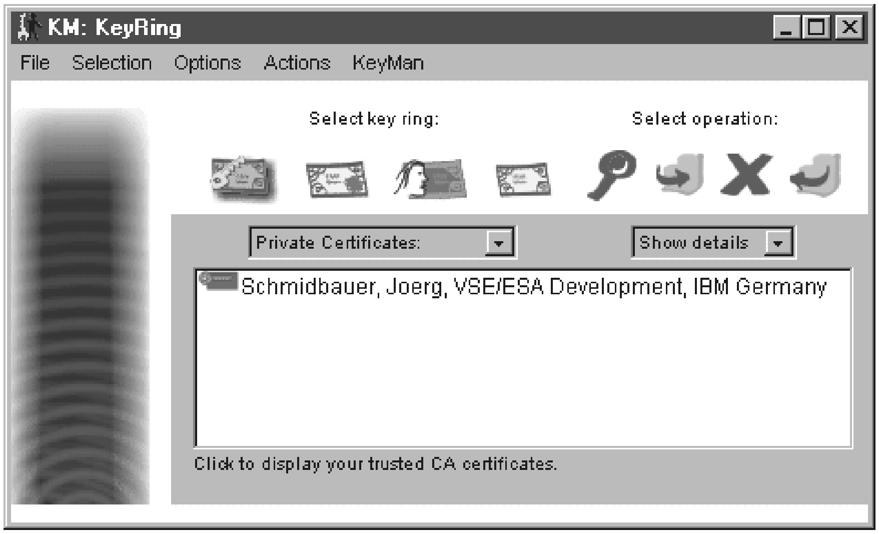


Figure 6. "KM: Import" Window – Import Certificate into Client Keyring File

b. Also from the Clipboard, paste the client certificate into an ASCII text file on your client workstation, then select **Save As** and give the file a name that you can later easily refer to. This file containing a client certificate will be required in "Step 7.6: Create Access Rights for CICS Clients to Use Host Resources" on page 84.

6. The client keyring file (Figure 7) now shows your client certificate. The public key that you generated in "Step 7.1: Generate a Key Pair" on page 79 has been merged into this client certificate.



Schmidbauer, Joerg, VSE/ESA Development, IBM Germany

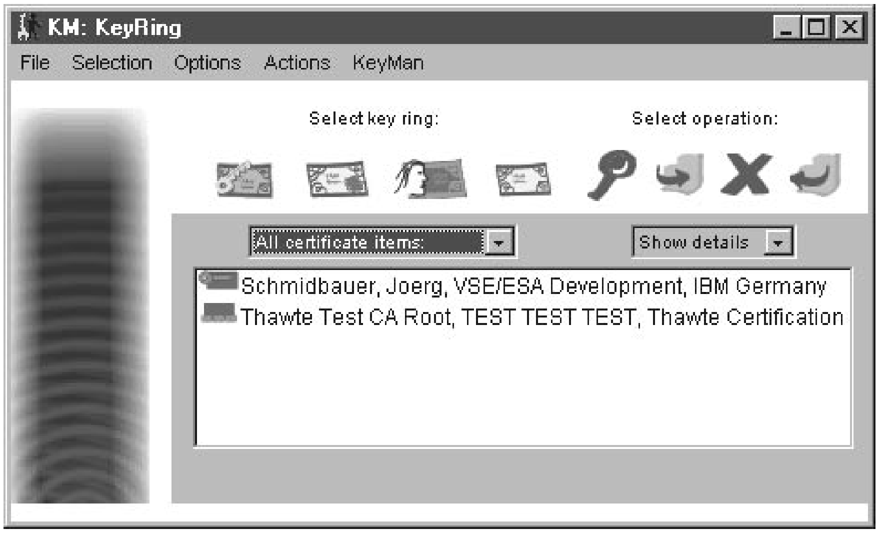
Figure 7. "KM: KeyRing" Window – Display Client Certificate

7. If you created a *new* client keyring file, save it by selecting **File** and **Save** from the menu pull-down. Then give the new client keyring file a password and file name.

Step 7.3: Import the CA's Root Certificate into the Client Keyring File

Now return to the Thawte selection panel, and click the option for obtaining the root certificate (that is related to your client certificate).

Ensure that this root certificate is copied to the Clipboard, and then import it into the client keyring file as shown in Figure 8.



Schmidbauer, Joerg, VSE/ESA Development, IBM Germany
Thawte Test CA Root, TEST TEST TEST, Thawte Certification

Figure 8. "KM: KeyRing" Window – Import Root Certificate

Step 7.4: File and Save Your Client Keyring File

After the root certifcate has been imported into your keyring file, you *must* save the keyring file by selecting **File** and **Save** from the menu pull-down.

You are also prompted to enter a password to protect this keyring file.

Step 7.5: Import Your Client Keyring File

In this step you import into your CICS client (Web browser) the client keyring file that contains both a client certificate and root certificate.

For *Microsoft Internet Explorer* Web browsers, you double-click **KeyRing.pfx** and the Internet Explorer is started. The *Certificate Manager Import Wizard* window is displayed, with general information. Now you should follow the instructions provided for adding the client keyring file to your certificate store. During the procedure, you will be requested to enter the:

- file you wish to import (for example `c:\vsecon\samples\KeyRing.pfx`).
- password that you specified when you saved the client keyring file in "Step 7.3: Import the CA's Root Certificate into the Client Keyring File" on page 82.
- *certificate store* (system area where certificates are stored). You can either define your own store, or let the system automatically select a store.

For *Netscape* Web browsers, you start your Web browser, and then Click **Security** from the menu bar. Under the Certificates column, you click **Yours** and then **Import a Certificate...**. Now you should follow the instructions for importing your client keyring file. During the procedure, you will be requested to enter a password with which you can protect the client keyring file.

Note: If you receive the message "The data cannot be decrypted because it was encrypted using an algorithm or key size which is not allowed by this configuration", then the options set in the client keyring file prevent this file from being imported into your Web browser. Check and amend the PKCS#12 options using, for example, the IBM *KeyMan* tool.

Step 7.6: Create Access Rights for CICS Clients to Use Host Resources

You can use the service functions provided with client authentication on the VSE/ESA host, to create the access rights that a CICS client has to VSE/ESA host resources. This is done using a VSE/ESA User-ID that is associated with the client certificate that the CICS client provides. A client-certificates/User-IDs mapping list is managed on the VSE/ESA host, where each client certificate in the mapping list is associated with a VSE/ESA User-ID.

To create the access rights for a CICS client, you must:

1. Paste the client certificate that you saved in a new file (in "Step 7.2: Generate and Store a Client Certificate" on page 80) into a job on the VSE/ESA host similar to the one shown in Figure 9. This job will catalog your client certificate into the VSE Keyring Library on the VSE/ESA host.

```
* $$ JOB JNM=CERTSTH1,CLASS=0,DISP=D
// JOB CATCERT
// ID USER=<uid>,PWD=<password>
*
* * The following is a sample job that contains a x509v3
* * client certificate in Base64 format.
*
// EXEC LIBR,PARM='MSHP'
A S=CRYPTO.KEYRING
CATALOG CERTSTH1.CCERT           DATA=YES REPLACE=YES EOD=/#
-----BEGIN CERTIFICATE-----
MIICnDCCAgWgAwIFAgIDBCQ/MA0GCSqGSIb3DQEBBAUAMIGHMQswCQYDVQQGEwJa
QTEiMCAGA1UECBMZRk9SIFRFU1RJTkcgUFVSUE9TRVMgT05MWTEdMBsGA1UEChMU
VGhhd3RlIENlcnRpZmljYXRpb24xFzAVBgNVBAsTDlRFU1QgVEVTVCBURVNUMRww
GgYDVQQDExNUaGF3dGUgVGVzdCBDQSBSb290MB4XDTAyMDMxOTE1MDgxMFoXDTAy
MDQwOTE1MDgxMFowdzEvMC0GA1UEAxMmSEggQnJvd3NlciAxMDI0LWJpdCBDbGll
bnQgQ2VydGlmaWNhdGUxFDASBgNVBAsTC0RldmVsb3BtZW50MQwwCgYDVQQKEwNJ
Qk0xEzARBgNVBAcTCkJvZWJsaW5nZW4xCEGJBgNVBAYTAkRFMIGfMA0GCSqGSIb3
DQEBAQUAA4GNADCBiQKBgQDKHGfEMZYRACD6ActwdgUvC9WAlcBGb9SYRkzpAiZ4
15lQAlo9f74eT7YOH63+EswfgZeOVO/+7nhPbW9u3FAC3BhSO8k4IDJjJGSRs390
NG/3foAIBtN0FE0q/yPt/txgPgTxrilSRTpFiM4CwOICcx/InEu1gMEKFSgdMeU/
GwIDAQABoyUwIzATBgNVHSUEDDAKBggrBgEFBQcDATAMBgNVHRMBAf8EAjAAMA0G
CSqGSIb3DQEBBAUAA4GBAF55Tg/E+yt9eOPCrU30DltFJwpFFFtyoA4SWfo2N4/v
F7FjaoP832LalYiWi9YyIrhVYuPR8qF4RJ3csWx/TN5FSNLz1SxnLsPwE2LoDby4
JrIgWPnHz8QCZJscbntjhi6DdbO1EtcI7woT0a2JytsyboIywiV7sPmljwepWRsW
-----END CERTIFICATE-----
/#
/*
/&
* $$ EOJ
```

Figure 9. Sample Job to Catalog Client Certificate into VSE Keyring Library

2. Submit the job.
3. Use the *Client Certificates/User-IDs dialog* on the VSE/ESA host to assign a VSE/ESA User ID to this client certificate.
4. Create access rights for the VSE/ESA User ID, giving it access to the VSE/ESA host resources you think are appropriate.

For details of the *Client Certificates/User-IDs dialog* and client-authentication service functions, refer to the chapter "Service Functions for Client Authentication" in the

VSE/ESA e-business Connectors, User's Guide, SC33-6719.

Currently-Supported SSL Cipher Suites

Table 2 shows the SSL cipher suites that are currently supported by SSL for VSE:

Table 2. Currently Supported SSL Cipher Suites

CICS Param	Hex Code	Cipher Suite	Hand-shaking	Encryp-tion
WEAK	01	SSL_RSA_WITH_NULL_MD5	512-bit	No
	02	SSL_RSA_WITH_NULL_SHA	512-bit	No
NORMAL	08	SSL_RSA_EXPORT_WITH_DES40_CBC_SHA	512-bit	40-bit
	09	SSL_RSA_WITH_DES_CBC_SHA	1024-bit	56-bit
STRONG	0A	SSL_RSA_WITH_3DES_EDE_CBC_SHA	1024-bit	168-bit
	62	RSA1024_EXPORT_DESCBC_SHA	1024-bit	56-bit

Notes:

1. The exportable ciphers 01 and 02 (also 62) require the SSL 3.0 handshaking. They cannot be used with TLS 1.0 handshaking. The other ciphers (08, 09, 0A) can be used with both SSL 3.0 and TLS 1.0, handshaking.
2. The NULL_MD5 (01) and NULL_SHA (02), and DES40_CBC_SHA (08) cipher suites require to have a 512-bit key on the VSE side. They cannot be used with a 1024-bit key.
3. The cipher suites that use 1024-bit handshaking, also require a 1024-bit key on the VSE/ESA host.

Part 4. Using the CICS Transaction Gateway with ECI

Chapter 10. Introduction to the ECI / CICS Transaction Gateway

This chapter provides an introduction to the CICS Transaction Gateway used with VSE/ESA, and the External Call Interface (ECI). It contains these sections:

- "How the ECI and CICS Transaction Gateway are used"
- "How the CICS Transaction Gateway accesses CICS" on page 90
- "The External Call Interface (ECI)" on page 90

How the ECI and CICS Transaction Gateway are used

The CICS Transaction Gateway provides a **CICS Java class library** that includes classes that provide an application programming interface (API), and are used to communicate between the Java gateway application and a Java application or applet.

The class *JavaGateway* is used to establish communication with the Gateway process, and uses Java's sockets protocol. The class *ECIRequest* is used to specify the External Call Interface (ECI) calls that are passed to the gateway.

The multithreaded architecture of the CICS Transaction Gateway enables a single Gateway to support multiple concurrently connected users.

The CICS Transaction Server supports the TCP/IP protocol for connections to the CICS Transaction Gateway, which enables the CICS Universal Client to use the External Call Interface (ECI) via TCP/IP. However, the External Presentation Interface (EPI) is *not* supported here.

Figure 10 shows how the CICS Transaction Gateway and ECI are used in a 3-tier environment.

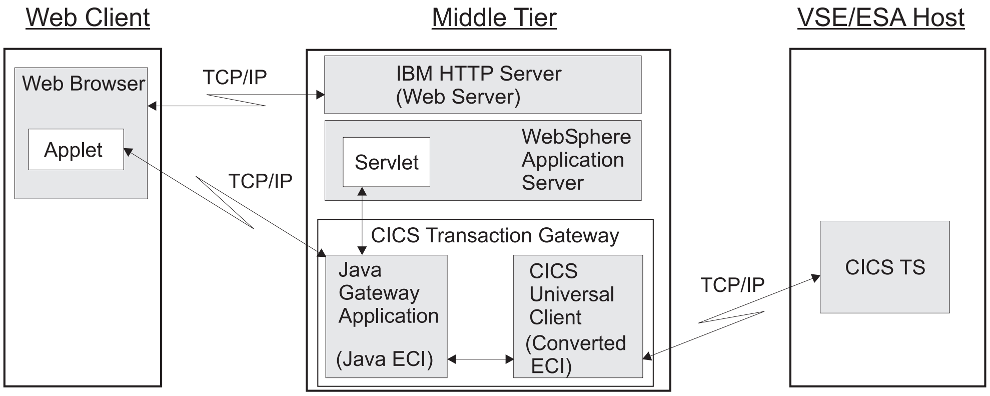


Figure 10. How the CICS Transaction Gateway and ECI are used

How the CICS Transaction Gateway accesses CICS

The flow of control when a Web browser calls CICS transaction processing facilities using the CICS Transaction Gateway (as shown in Figure 10 on page 89) is as follows:

1. The Web browser calls the Web server on the middle-tier, using HTTP (Hypertext Transfer Protocol) to get HTML pages.
2. When the browser, which is interpreting the HTML and presenting it on the screen, finds an applet tag, it calls the Web server on the middle-tier to get the applet and the classes that it needs. It then executes the applet.
3. An applet that is going to communicate with CICS creates a *JavaGateway* object. The creation of this object causes a call to the CICS Transaction Gateway long-running task on the middle-tier .
4. The applet creates an *ECIRequest* object to represent its request for a CICS program, and calls the flow method of the *JavaGateway* object, passing the instance of the *ECIRequest* object.
5. The CICS Transaction Gateway on the middle-tier receives the request, and calls the CICS program on the VSE/ESA host.
6. When the CICS program ends, the results are returned to the Web browser via the CICS Transaction Gateway on the middle-tier.

The flow of data when a Web browser calls CICS transaction processing facilities using the gateway, is as follows:

1. The Web browser acquires data for the CICS program from the end user.
2. The Web browser constructs a communication area for the CICS program that is to supply transaction processing services.
3. The CICS Transaction Gateway on the middle-tier receives the communication area and passes it to the CICS program on the VSE/ESA host. The contents of the communication area are translated from ASCII (in the gateway) to EBCDIC (in the CICS Transaction Server for VSE/ESA).
4. The CICS program on the VSE/ESA host supplies the transaction processing services, enquiring on and perhaps changing CICS resources. If the program ends normally, changes to recoverable resources are committed. If the program ends abnormally, the changes are backed out.
5. The communication area is translated from EBCDIC to ASCII, and returned to the gateway on the middle-tier, which forwards it to the Web browser.
6. The Web browser presents information to the end user.

The External Call Interface (ECI)

The ECI allows a non-CICS application to call a CICS program in a CICS server. The application can be connected to several servers at the same time, and it can have several program calls outstanding at the same time.

The CICS program cannot perform terminal I/O, but can access and update all other CICS resources. The same CICS program can be called by a non-CICS application using the ECI, or by a CICS program using EXEC CICS LINK. Data is exchanged between the two programs by means of a COMMAREA, in a similar way to CICS. The user can specify the length of the COMMAREA data to optimize performance.

Calls may be made synchronously or asynchronously. Synchronous calls return control when the called program completes, and the information returned is immediately available. Asynchronous calls return control without reference to the completion of the called program, and the application can ask to be notified when the information becomes available.

Calls may also be extended. That is, a single logical unit of work may cover two or more successive calls, though only one call can be active for each logical unit of work at any time. If it uses asynchronous calls, the application can manage many logical units of work concurrently.

The called program can update resources on its own system, it can use distributed program link (DPL) to call CICS programs on other systems, and it can access resources on other CICS systems by function shipping, by distributed transaction processing (DTP), or (in the CICS Transaction Server for VSE/ESA environment) by the front end programming interface (FEPI).

For more information on the external access interfaces, see *CICS Family: Client/Server Programming*.

Chapter 11. Configuring for ECI via TCP/IP

This chapter explains how to configure the CICS Transaction Server for VSE/ESA to use the CICS ECI (External Call Interface) via TCP/IP. It contains these sections:

- "Hardware prerequisites"
- "Software prerequisites"
- "Defining a TCPIPSERVICE Resource for ECI"
- "Displaying the ATTACHSEC Attribute" on page 94

Hardware prerequisites

None.

Software prerequisites

These are the software prerequisites for using CICS ECI via TCP/IP:

- VSE/ESA Version 2 Release 6 or later.
- The CICS Universal Client.

Defining a TCPIPSERVICE Resource for ECI

To use the ECI via TCP/IP support, you must install and activate *one or more* TCP/IP services for ECI use.

There are three TCPIPSERVICE resource definition attributes relating to ECI:

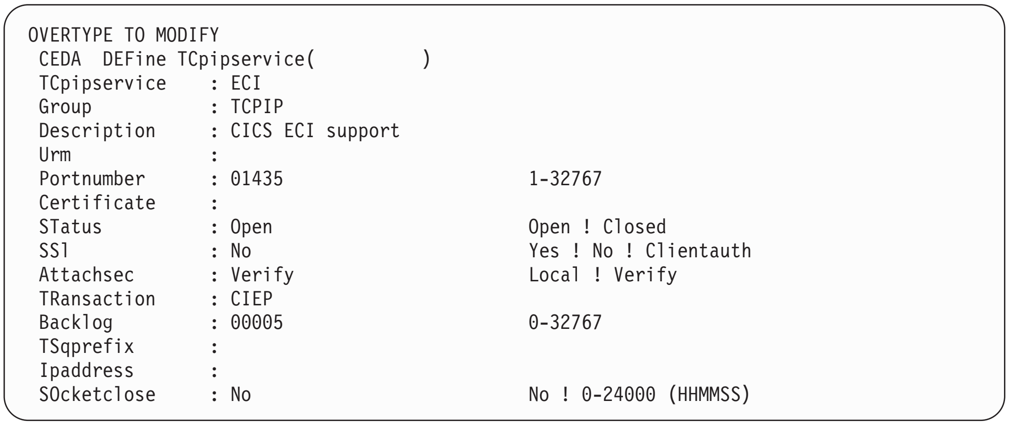
1. ATTACHSEC
2. PORTNUMBER
3. TRANSACTION

See Chapter 16, "CEDA DEFINE TCPIPSERVICE" on page 119 for a description of the above attributes (which are pre-defined in CICS TS systems that contain CICS Web Support).

You activate the TCPIPSERVICE definition either by:

- Specifying `STATUS(OPEN)` and installing the definition.
- Installing the definition and later using a `CEMT SET TCPIPSERVICE OPEN` command (described in Chapter 16, "CEDA DEFINE TCPIPSERVICE" on page 119.

Figure 11 on page 94 shows an example of the entries you make when defining a TCPIPSERVICE definition for use with ECI:



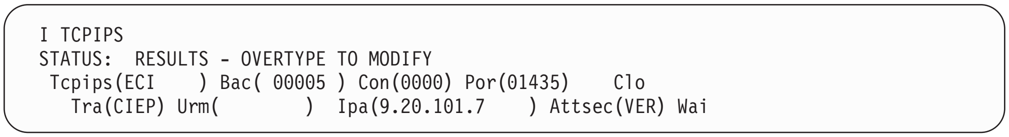
```
OVERTYPE TO MODIFY
 CEDA  DEFine TCpipservice(          )
 TCpipservice    : ECI
 Group           : TCPIP
 Description     : CICS ECI support
 Urm             :
 Portnumber      : 01435                       1-32767
 Certificate     :
 STatus          : Open                        Open ! Closed
 SSl             : No                          Yes ! No ! Clientauth
 Attachsec       : Verify                      Local ! Verify
 TRansaction     : CIEP
 Backlog         : 00005                       0-32767
 TSqprefix       :
 Ipaddress       :
 SOcketclose     : No                          No ! 0-24000 (HHMMSS)
```

Figure 11. Example of a TCPIPSERVICE resource definition for ECI

Related Section:
“CEMT INQUIRE/SET TCPIPSERVICE” on page 137

Displaying the ATTACHSEC Attribute

If you enter `CEMT I TCPIPSERVICE`, the ATTACHSEC setting will appear as follows:



```
 I TCPIPS
 STATUS:  RESULTS - OVERTYPE TO MODIFY
  Tcpips(ECI    ) Bac( 00005 ) Con(0000) Por(01435)    Clo
    Tra(CIEP) Urm(        )  Ipa(9.20.101.7    ) Attsec(VER) Wai
```

Figure 12. The CEMT INQ TCPIPSERV display.

Part 5. Using CICS documents

This part introduces CICS documents. It tells you what you need to consider when writing applications that use documents as a means of formatting information.

The document handler domain allows you to build up formatted data areas, known as documents. Some examples of how these formatted areas, or documents, can be used, are:

- Sending HTML data to be displayed by a Web browser.
- Creating standard formats for printing (for example, using your own letterhead, address, and so on).

Chapter 12. The DOCUMENT application programming interface

This section explains the function and use of the commands in the DOCUMENT application programming interface:

- EXEC CICS DOCUMENT CREATE
- EXEC CICS DOCUMENT INSERT
- EXEC CICS DOCUMENT RETRIEVE
- EXEC CICS DOCUMENT SET

Creating a document

To create an empty document, use the EXEC CICS DOCUMENT CREATE command. This has a mandatory DOCTOKEN parameter requiring a 16–byte data-area. The document handler domain uses the DOCTOKEN operand to return a token, which is used to identify the document on subsequent calls. The following example creates an empty document, and returns the token in the variable MYDOC:

```
EXEC CICS DOCUMENT CREATE
        DOCTOKEN(MYDOC)
```

To create a document with data, use the EXEC CICS DOCUMENT CREATE command in any of the following ways:

- Specify the BINARY parameter
- Specify the TEXT parameter
- Insert one document into another document
- Use document templates

The BINARY parameter

Use this parameter to add to the document the contents of a data-area that must not undergo conversion to a client code page when the data is sent.

```
EXEC CICS DOCUMENT CREATE
        DOCTOKEN(MYDOC1)
        BINARY(DATA-AREA)
```

The TEXT parameter

Use this parameter to add the specified contents to the document. For example, if you define a character string variable called DOCTEXT and initialise it to *This is an example of text to be added to a document*, you can use the following command to create a document consisting of this text string:

```
EXEC CICS DOCUMENT CREATE
        DOCTOKEN(MYDOC2)
        TEXT(DOCTEXT)
        LENGTH(53)
```

Inserting one document into another

To insert an existing documentinto a new document, you can use the EXEC CICS DOCUMENT CREATE command with the FROMDOC option. The following example does this:

```
EXEC CICS DOCUMENT CREATE
        DOCTOKEN(MYDOC3)
        FROMDOC(MYDOC2)
```

where MYDOC2 and MYDOC3 are 16–character variables. MYDOC2 must contain the token returned by a previous EXEC CICS DOCUMENT CREATE command.

This results in two identical documents, each containing the text *This is an example of text to be added to a document*.

Using document templates

Portions of the data which make up a document can be created off-line and then inserted directly into the document. These are known as **templates**; they are CICS resources, defined using RDO.

Templates can contain a mixture of static data with symbols embedded in the data, which are substituted at run time when the template is inserted into the document. An example of this is when a programmer creates HTML web pages using an HTML editor. The output from the HTML editor can then be made accessible to CICS Web Interface applications using templates.

Chapter 13. Programming with documents

This section covers the following topics:
- Symbols and symbol lists
- Embedded DOCTEMPLATE commands
- Using DOCTEMPLATEs in your application
- The lifespan of a document
- Retrieving the document without control information
- Using multiple calls to construct a document
- Bookmarks and inserting data
- Replacing data in the document
- Codepages and codepage conversion

Setting symbol values

The application program needs to define values for the symbols that will be substituted when the template is used. These values can be defined on the EXEC CICS DOCUMENT CREATE or the EXEC CICS DOCUMENT SET commands. The symbols that are set are associated with a particular document and cannot be used in a different document.

The DOCUMENT CREATE and DOCUMENT SET commands both take a SYMBOLLIST operand which allows several symbols to be defined in a single command. The SYMBOLLIST operand is a character string consisting of one or more definitions with single byte separators. By default, the separator is an ampersand, but you can override this by using the DELIMITER option of the DOCUMENT SET or DOCUMENT CREATE commands. A definition consists of a name, an equals sign, and a value. Here is an example:

```
  mytitle=New Authors&auth1=Halliwell Sutcliffe&auth2=Stanley
Weyman
```

This example defines three symbols. The first symbol called mytitle will have the value 'New Authors'. The second symbol called auth1 will have the value 'Halliwell Sutcliffe' and the last symbol called auth2 will contain the value 'Stanley Weyman'.

The following rules apply when setting symbols using a SYMBOLLIST. The name must contain only uppercase and lowercase letters, numbers and the special characters dollar ('$'), underscore ('_'), hyphen ('-'), pound ('#'), period ('.') and at sign ('@'). The name is case-sensitive, so uppercase letters are regarded as different from lowercase letters.

The values in the symbol list can contain any characters except the symbol separator (which defaults to an ampersand, but can be overridden by use of the DELIMITER option). The following restrictions on the use of the percent sign ("%") and the plus sign ("+") apply unless the UNESCAPED option of DOCUMENT CREATE or DOCUMENT SET has been specified.. A percent sign must be followed by two characters that are hexadecimal digits (that is, 0–9, a-f, and A-F). When the value is put into the symbol table, a plus sign is interpreted as a space, 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, and the remaining characters are left as they are. If you want a plus sign in the value in the symbol table, you must put %2B in the value in the symbol list. If you want a percent sign in the value in the symbol table, you must put in the value %25 in the symbol list. If you want an ampersand in the value in the symbol table, you must put %26 in the value in the symbol list. If you want a space in the value in the symbol table, the value in your symbol list may contain a space, a plus sign, or a %20.

The DOCUMENT SET command allows you to set individual symbol values with the SYMBOL and VALUE options. Ampersands have no special significance when used in the VALUE option. The restrictions on the use of the plus sign and percent sign for SYMBOLLISTS also apply to the VALUE option unless the UNESCAPED option of the DOCUMENT SET has been specified.

The following example shows you how you can pass symbol values to the document handler containing embedded plus signs, percent signs, and ampersands, none of which are to undergo unescape processing:

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
          DELIMITER('!')
          SYMBOLLIST('COMPANY=BLOGGS & SON!ORDER=NUTS+BOLTS')
          LISTLENGTH(37)
          UNESCAPED
```

Here the symbol COMPANY has a value of 'BLOGGS & SON', and the symbol ORDER has a value of 'NUTS+BOLTS'. The delimiter used in this example is '!', but it is best to use a non-printable character that does not appear in the symbol value. The use of the UNESCAPED option ensures that the plus sign in 'NUTS+BOLTS' does not get converted to a space.

Embedded template commands

The Document Handler recognises four commands which can be embedded in the template. Three of the commands follow the syntax rules for Server Side Include commands. A Server Side Include command starts with the characters left angle bracket, exclamation mark, hyphen, hyphen, pound followed by the command and it is terminated with the characters hyphen, hyphen, right angle bracket. For example:

```
(e.g. <!--#command -->).
```

The three commands that are supported are #set, #echo and #include.

The #set command is used to set the values of symbols and is useful for setting up default values for symbols. The #set command in the template will be ignored if the symbol has already been given a value using the EXEC CICS DOCUMENT SET command. If a previous #set command has been used to assign a value to the symbol, the value will be overriden. A symbol which has been assigned a value using the EXEC CICS DOCUMENT SET command can only be changed by issuing another EXEC CICS DOCUMENT SET command.

The #echo command identifies a symbol that must be substituted when the template is inserted into the document. The string containing the #echo command will be completely replaced by the value associated with the symbol. If no symbol has been defined with that name, the #echo command will remain in the output data. An alternative method to using the #echo command is to specify the symbol name, preceding it with an ampersand and terminating it with a semicolon. If we set a symbol called ASYM and give it a value of 'sample', the following two templates will give the same result after substitution.

```
Template 1:
   This is an example template.
   <!--#set var=ASYM value='sample'-->
   This is a <!--#echo var=ASYM--> symbol.
```

```
Template 2:
   This is an example template.
   <!--#set var=ASYM value='sample'-->
   This is a &ASYM; symbol.
Result of substitution:
   This is an example template.
   This is a sample symbol.
```

The #include command allows a template to be embedded within another template. Up to 32 levels of embedding are allowed.

Using templates in your application

If you have created a template and defined it to CICS, the following example shows how you can use the template to create the contents of a document. The following template is created and defined to CICS with the name ASampleTemplate.

```
<!--#set var=ASYM value='DFLTUSER'-->
This is a sample document which has been created by user
<!--#echo var=ASYM-->.
```

In the application program, you can define a 48-byte variable called TEMPLATENAME and initialize it to a value of ’ASampleTemplate’. Once again you must define a 16-byte field for the document token (in this example, ATOKEN). You can then issue the command to create the document.

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
          TEMPLATE(TEMPLATENAME)
```

This will result in a document being created with the content “ This is a sample document which has been created by user DFLTUSER.”.

To change the symbol to another value, you can issue the EXEC CICS DOCUMENT CREATE command with the SYMBOLLIST option:

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
          TEMPLATE(TEMPLATENAME)
          SYMBOLLIST('ASYM=Joe Soap')
          LISTLENGTH(13)
```

This will result in a document being created with the content “This is a sample document which has been created by user Joe Soap.”.

The lifespan of a document

Documents created by an application exist only for the length of the CICS task in which they are created. This means that when the last program in the CICS task returns control to CICS, all documents created during the task’s lifetime are deleted. It is the application’s responsibility to save a document before terminating if the document is going to be used in another task. You can obtain a copy of the document by using the EXEC CICS DOCUMENT RETRIEVE command. The application can then save this copy to a location of its choice, such as a temporary storage queue. The copy can then be used to recreate the document.

The following sequence of commands show how a document can be created, retrieved and stored on a temporary storage queue, assuming that the following variables have been defined and initialised in the application program:

- A 16-byte field ATOKEN to hold the document token

- A 20-byte buffer DOCBUF to hold the retrieved document
- A fullword binary field called FWORDLEN to hold the length of the data retrieved
- A halfword binary field called HWORDLEN to hold the length for the temporary storage WRITE command.

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
          TEXT('A sample document.')
          LENGTH(18)
EXEC CICS DOCUMENT RETRIEVE
          DOCTOKEN(ATOKEN)
          INTO(DOCBUF)
          LENGTH(FWORDLEN)
          MAXLENGTH(20)
EXEC CICS WRITEQ TS
          QUEUE('AQUEUE')
          FROM(DOCBUF)
          LENGTH(HWORDLEN)
```

You can now use the following sequence of commands to recreate the document in the same or another application.

```
EXEC CICS READQ TS
          QUEUE('AQUEUE')
          INTO(DOCBUF)
          LENGTH(HWORDLEN)
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
          FROM(DOCBUF)
          LENGTH(FWORDLEN)
```

When the document is retrieved, the data that is delivered to the application buffer is stored in a form which contains control information necessary to reconstruct an exact replica of the document. The document that is created from the retrieved copy is therefore identical to the original document. To help the application calculate the size of the buffer needed to hold a retrieved document, each document command which alters the size of the document has a DOCSIZE option. This is a fullword value which gives the maximum size that the buffer must be to contain the document when it is retrieved. This size is calculated to include all the control information and data. The size should not be taken as an accurate size of the document as the actual length delivered to the application can often be slightly smaller than this size. The length delivered will however never exceed the length in the DOCSIZE option.

The above example introduced the use of the FROM option on the DOCUMENT CREATE command. The data passed on the FROM option was the buffer returned to the application when the DOCUMENT RETRIEVE command was issued. It is possible for the application to supply data on the FROM option that did not originate from the DOCUMENT RETRIEVE command. When this happens, the document handler treats the data as a template and parses the data for template commands and symbols.

Retrieving the document without control information

The document data containing control information is only useful to an application that wishes to recreate a copy of the original document. It is possible to issue a DOCUMENT RETRIEVE command and ask for the control information to be omitted. The following command sequence uses the DATAONLY option on the

DOCUMENT RETRIEVE command to instruct the Document Handler to return only the data. This example assumes that the following variables have been defined and initialised in the application program:

- A 16-byte field ATOKEN to hold the document token
- A 20-byte buffer DOCBUF to hold the retrieved document
- A fullword binary field called FWORDLEN to hold the length of the data retrieved.

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
          TEXT('A sample document.')
          LENGTH(18)
EXEC CICS DOCUMENT RETRIEVE
          DOCTOKEN(ATOKEN)
          INTO(DOCBUF)
          LENGTH(FWORDLEN)
          MAXLENGTH(20)
          DATALONLY
```

When the commands have executed, the buffer DOCBUF will contain the string “A sample document.”.

Using multiple calls to construct a document

Once a document has been created, the contents can be extended by issuing one or more EXEC CICS DOCUMENT INSERT commands. The options on the EXEC CICS DOCUMENT INSERT command work in the same way as the equivalent options on the EXEC CICS DOCUMENT CREATE command. The following sequence of commands shows an empty document being created followed by two INSERT commands:

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Sample line 1. ')
          LENGTH(15)
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Sample line 2. ')
          LENGTH(15)
```

The document resulting from the above commands will contain:

```
Sample line 1. Sample line 2.
```

You can use the DOCUMENT RETRIEVE and DOCUMENT INSERT commands to insert a whole document into an existing document. The following variables must first be defined and initialized in the application program:

- A 16-byte field RTOKEN which contains the document token of the document to be retrieved
- A buffer DOCBUF of sufficient length to hold the retrieved document
- A fullword binary field called RETRIEVLEN to hold the length of the data retrieved
- A fullword binary field called MAXLEN to hold the maximum amount of data the buffer can receive, i.e. the length of DOCBUF
- A 16-byte field ITOKEN which contains the document token of the document that is being inserted into

The following sequence of commands shows a document indicated by RTOKEN being inserted into another document indicated by ITOKEN:

```
 EXEC CICS DOCUMENT RETRIEVE
           DOCTOKEN(RTOKEN)
           INTO(DOCBUF)
           LENGTH(RETRIEVLEN)
           MAXLENGTH(MAXLEN)

 EXEC CICS DOCUMENT INSERT
           DOCTOKEN(ITOKEN)
           FROM(DOCBUF)
           LENGTH(RETRIEVLEN)
```

The retrieved document is inserted at the end of the document specified in the DOCUMENT INSERT command, and all the control information of the retrieved document will be present in the second document. The LENGTH parameter of the DOCUMENT INSERT command must be equal to the value returned from the DOCUMENT RETRIEVE command into the field RETRIEVLEN.

The DOCUMENT INSERT command allows an operand called SYMBOL to be used to add blocks of data to the document. SYMBOL must contain the name of a valid symbol whose value has been set. The Document Handler inserts the value that is associated with the symbol into the document.

Bookmarks and inserting data

The sequence in which an application inserts data into a document might not reflect the desired sequence that the data should appear in the document. Bookmarks allow the application to insert blocks of data in any order yet still control the sequence of the data in the document. A bookmark is a label that the application inserts between blocks of data. Note: a bookmark cannot be inserted in the middle of a block of data.

The following example creates a document with two blocks of text and a bookmark:

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
          TEXT('Pre-bookmark text. ')
          LENGTH(19)
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          BOOKMARK('ABookmark       ')
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Post-bookmark text. ')
          LENGTH(20)
```

The document will now contain:

```
     Pre-bookmark text. <ABookmark>Post-bookmark text.
```

Note that the text <ABookmark> does not appear in the document content but serves merely as a pointer to that position in the document. To add data to this document, you can insert text at the bookmark as follows:

```
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Inserted at a bookmark. ')
          LENGTH(25)
          AT('ABookmark       ')
```

Logically, the data of the document will contain the following (Note that in this instance, only the data is being shown and not the position of the bookmark).

```
Pre-bookmark text. Inserted at a bookmark. Post-bookmark text.
```

If the AT option is omitted, the data is always appended to the end of the document. A special bookmark of 'TOP' can be used to insert data at the top of the document, making it unnecessary to define a bookmark which will mark the top of the document.

Replacing data in the document

The following example shows how data between two bookmarks can be replaced:

```
EXEC CICS DOCUMENT CREATE
          DOCTOKEN(ATOKEN)
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Initial sample text. ')
          LENGTH(21)
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          BOOKMARK('BMark1          ')
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Text to be replaced. ')
          LENGTH(21)
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          BOOKMARK('BMark2          ')
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Final sample text. ')
          LENGTH(19)
```

At this point the logical structure of the document will be as follows:

```
  Initial sample text. <BMark1>Text to be replaced. <BMark2>Final
  sample text.
```

You can now issue the command to replace the text between the two bookmarks, BMark1 and BMark2:

```
EXEC CICS DOCUMENT INSERT
          DOCTOKEN(ATOKEN)
          TEXT('Replacement Text. ')
          LENGTH(18)
          AT('BMark1          ')
          TO('BMark2          ')
```

The document now has the following logical structure:

```
   Initial sample text. <BMark1>Replacement Text.  <BMark2>Final
   sample text.
```

Codepages and codepage conversion

The documents that an application creates may be transmitted to systems running on other platforms, especially when applications running under the CICS Web interface use the Document Handler to generate Web pages. To assist the application with the problem of converting data from the codepages used on the host to the codepages used on the target system, the Document Handler allows the application to specify the codepages being used on each system. When the EXEC CICS DOCUMENT CREATE and EXEC CICS DOCUMENT INSERT commands are used, the TEXT, FROM, TEMPLATE and SYMBOL options can have a HOSTCODEPAGE option coded to indicate the codepage for that block of data, since the data being added with these options is seen as being textual data. Each block can be specified in a different codepage. When the EXEC CICS DOCUMENT

RETRIEVE command is issued, the CLNTCODEPAGE option tells the Document Handler to convert all the individual blocks from their respective host codepages into a single client codepage.

Part 6. CICS Resource Definitions Online Changes

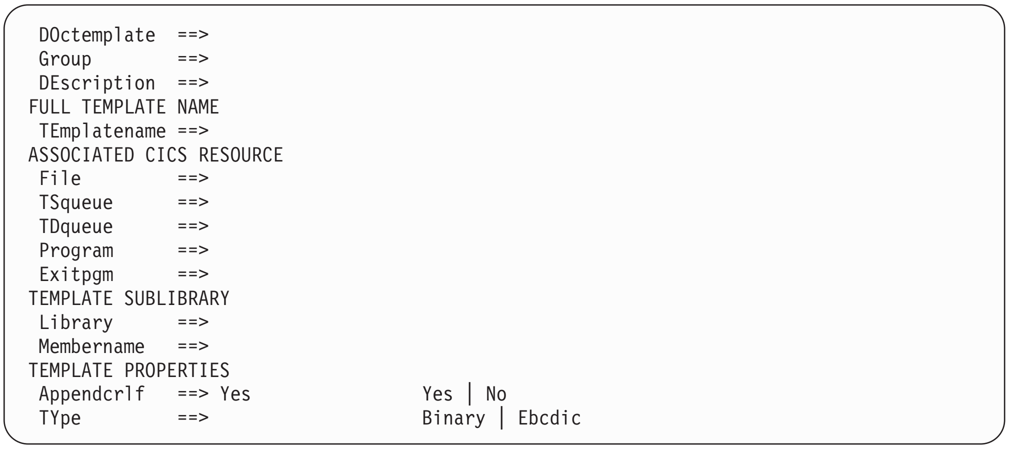
Chapter 14. CEDA DEFINE DOCTEMPLATE

Use this resource definition to define document templates to CICS. Document templates allow you to perform variable substitution on documents in a manner similar to that done by BMS for 3270 screens.

The template can reside in any one of the following places:

- VSE/ESA sub-library
- CICS temporary storage
- CICS transient data
- CICS load module
- exit program

Defining a DOCTEMPLATE



```
 DOctemplate  ==>
 Group        ==>
 DEscription  ==>
FULL TEMPLATE NAME
 TEmplatename ==>
ASSOCIATED CICS RESOURCE
 File         ==>
 TSqueue      ==>
 TDqueue      ==>
 Program      ==>
 Exitpgm      ==>
TEMPLATE SUBLIBRARY
 Library      ==>
 Membername   ==>
TEMPLATE PROPERTIES
 Appendcrlf   ==> Yes                Yes | No
 TYpe         ==>                    Binary | Ebcdic
```

Figure 13. The DEFINE panel for DOCTEMPLATE

Options

APPENDCRLF
specifies whether CICS is to delete trailing blanks from and append carriage-return line-feed to each logical record of the template.

LIBRARY
specifies the library name of the VSE/ESA library containing the template. The library applies only to a template of type VSE/ESA library. If a membername attribute is supplied without a value for library, the SOURCE LIBDEF chain is searched for the template. If a value for library is specified, the sub-library DFHDOC of that library is used to locate the template.

DESCRIPTION(*text*)
You can provide a description of the resource you are defining in this field. The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, ensure that for each left parenthesis there is a matching right one. For each single apostrophe in the text, code two apostrophes.

DOCTEMPLATE

DOCTEMPLATE
specifies the name of this DOCTEMPLATE definition. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 $ @ # . / - _ % & ¢ ? ! : | " = ¬ , ; < and >.

EXITPGM
specifies the exit program to be invoked when a request is made for this template. CICS passes a commarea to the exit program which is mapped by the following copybooks:

- DFHDHTXD (Assembler)
- DFHDHTXH (C)
- DFHDHTXL (PL/I)
- DFHDHTXO (COBOL)

FILE(*filename*)
specifies, for a template of type FILE, the 8-character name of the CICS file definition for the data set containing the template.

GROUP(*groupname*)
specifies the group name, which can be up to eight characters in length. The characters allowed are A-Z 0-9 @ # and $. Lowercase characters are treated as uppercase characters. Do not use group names beginning with DFH, because these characters are reserved for use by CICS.

Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see *CICS Resource Definition Guide*.

MEMBERNAME
specifies the name of the VSE/ESA library member containing the template. It applies only to a template of type VSE/ESA library. The VSE/ESA library member type of the template must be HTML.

PROGRAM
specifies the program in which the template data is stored. CICS loads the program and takes all data after the entrypoint to be the template.

TDQUEUE
specifies the name of the TD queue on which the template is stored.

TEMPLATENAME
specifies the extended Template-name by which the doctemplate is to be known outside the resource definition function.

TSQUEUE
specifies the name of the TS queue on which the template is stored.

TYPE(BINARY|EBCDIC)
specifies the format of the contents of the template.

BINARY
When the template is loaded from the template library, no parsing of the template’s contents is done.

EBCDIC
When the template is loaded from the template library, the contents are parsed as EBCDIC text.

Note: The fields EXITPGM, FILE, MEMBER, PROGRAM, TDQUEUE, and TSQUEUE define alternative sources of the template data. Only one of them may be specified on each DOCTEMPLATE definition.

Chapter 15. CEDA DEFINE PROFILE

You specify options that control the interactions between transactions and terminals or logical units, as a PROFILE. The PROFILE is a means of standardizing the use of such options as screen size and printer compatibility, and the use of such functions as message journaling and the node error program.

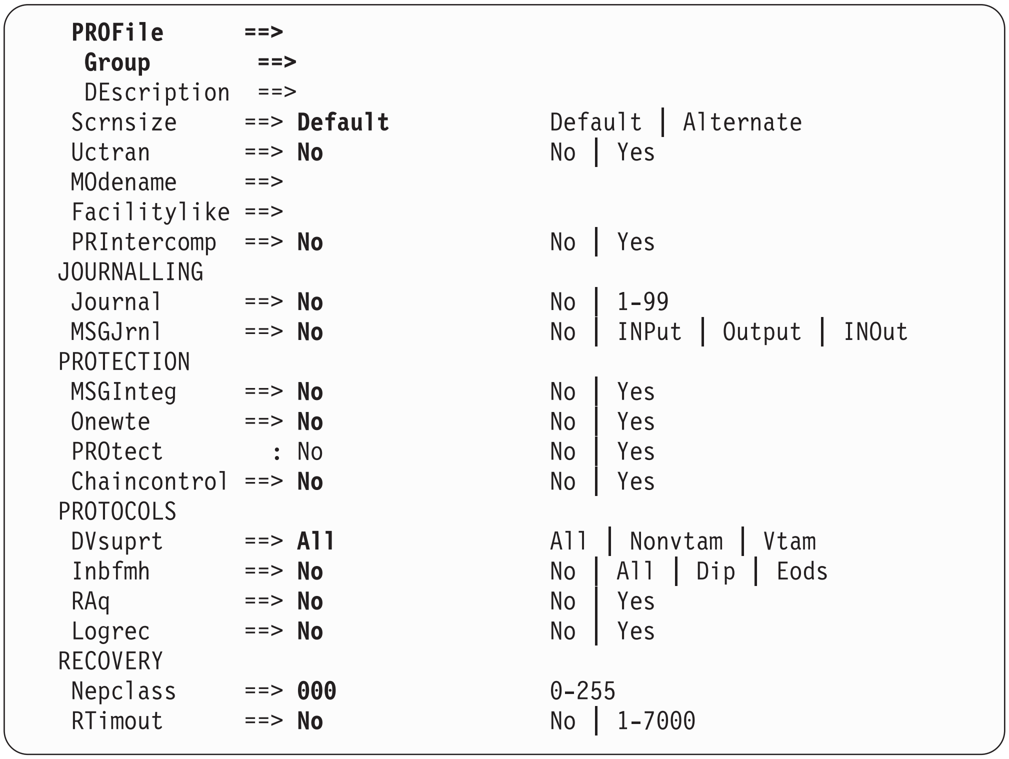
MODENAME
A profile is associated with the communication between a transaction and an LUTYPE6.1 or APPC session to another system. For APPC sessions, you refer on the PROFILE definition to the MODENAME that is also named on the SESSIONS definitions. This MODENAME is the name of the mode set to which the sessions belong. See the *CICS Resource Definition Guide* manual.

When installed in CICS, the information from the PROFILE definition creates an entry in the profile table. This entry is later used by each transaction that references that PROFILE.

There are CICS-supplied PROFILE definitions suitable for most purposes. Each TRANSACTION definition names the PROFILE to be used. If you do not specify a PROFILE, the transaction uses the PROFILE supplied for using a terminal in a standard way.

With CICS intercommunication facilities (for example, function shipping), a PROFILE is needed for the communication between the transaction and the session. The attributes of the CICS-supplied profiles are shown in ″PROFILE definitions in group ISC″ in the *CICS Resource Definition Guide* manual. The *CICS Intercommunication Guide* gives further information about the CICS-supplied PROFILEs, and tells you about defining your own profiles.

Defining a PROFILE



```
 PROFile      ==>
  Group        ==>
  DEscription  ==>
 Scrnsize     ==> Default            Default | Alternate
 Uctran       ==> No                 No | Yes
 MOdename     ==>
 Facilitylike ==>
 PRIntercomp  ==> No                 No | Yes
JOURNALLING
 Journal      ==> No                 No | 1-99
 MSGJrnl      ==> No                 No | INPut | Output | INOut
PROTECTION
 MSGInteg     ==> No                 No | Yes
 Onewte       ==> No                 No | Yes
 PROtect        : No                 No | Yes
 Chaincontrol ==> No                 No | Yes
PROTOCOLS
 DVsuprt      ==> All                All | Nonvtam | Vtam
 Inbfmh       ==> No                 No | All | Dip | Eods
 RAq          ==> No                 No | Yes
 Logrec       ==> No                 No | Yes
RECOVERY
 Nepclass     ==> 000                0-255
 RTimout      ==> No                 No | 1-7000
```

Figure 14. The DEFINE panel for PROFILE

Options

CHAINCONTROL({NO|YES})
specifies whether the application program can control the outbound chaining of request units. If you specify CHAINCONTROL(YES), ONEWTE(YES) means one chain and not one terminal control output request.

DESCRIPTION(*text*)
You can provide a description of the resource you are defining in this field. The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, ensure that for each left parenthesis there is a matching right one. For each single apostrophe in the text, code two apostrophes.

DVSUPRT({ALL|NONVTAM|VTAM})
specifies the devices (terminals or logical units) that are to be supported. The access method used by a particular terminal or logical unit is specified in its associated TCTTE.

ALL The profile can be used with any terminal or logical unit.

NONVTAM
The profile can be used only with non-VTAM terminals.

VTAM The profile can be used only with logical units.

FACILITYLIKE(*name*)
This is an optional parameter that specifies the name of an existing (four

character) terminal resource definition to be used as a template for the bridge facility. It can be overridden by specifying FACILITYLIKE in the bridge exit.

There is no default value for this parameter.

If you are running in a CICS system started with the VTAM=NO system initialization (SIT) parameter, the resource definition specified by FACILITYLIKE must be defined as a remote terminal.

GROUP(*groupname*)
Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see *CICS Resource Definition Guide*.

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

INBFMH({NO|ALL|DIP|EODS}) (SNA LUs only)
specifies, for profiles used with logical units, whether a function management header (FMH) received from a logical unit is to be passed to the application program.

ALL All FMHs (except APPC FMHs and LU6.1 ATTACH and SYNCPOINT FMHs that are processed by CICS) are passed to the application program. This value is required for function shipping transactions such as CSMI, transactions which use distributed transaction processing, and for distributed program link requests.

DIP The batch data interchange program (DFHDIP) is to process inbound FMHs. BMS issues a batch data interchange receive request if a BMS receive request has been issued, and a batch data interchange receive request is issued instead of a terminal control receive request.

EODS An FMH is passed to the application program only if it indicates end of data set (EODS).

NO The FMHs are discarded.

JOURNAL({NO|*number*)
specifies that you want automatic journaling of messages to take place, by giving the identifier of the journal.

NO No automatic journaling of messages is to take place.

number
The journal identification to be used for automatic journaling. This can be any number in the range 01 through 99. This number is appended to the letters DFHJ to give a journal identification of the form DFHJ*nn*.

LOGREC({NO|YES})
specifies whether the design of the application requires that each EXEC CICS RECEIVE request is to be satisfied by a logical record. This option allows existing 2770-and 2780-based application programs to be attached to a batch logical unit (for example, 3790 or 8100) without modification to the program.

MODENAME(*name*)
specifies the name that identifies a group of sessions for use on an APPC connection. The name can be up to eight characters in length, and must be the name of a VTAM LOGMODE entry defined to VTAM. It must not be the

reserved name SNASVCMG. If you omit the modename, it defaults to blanks. See the *CICS Intercommunication Guide* for more information about VTAM modenames.

If a transaction that specifies this profile has been started using an EXEC CICS START command, the MODENAME is used for allocation of the principal facility. If a transaction performs an EXEC CICS ALLOCATE command specifying this profile, the MODENAME is used for allocation of the alternate facility.

If you do not specify a MODENAME, CICS selects a session from any one of the mode sets that have been defined.

The CICS-supplied profile DFHCICSA is used, if PROFILE is not specified on an EXEC CICS ALLOCATE command. For function shipping, the profile DFHCICSF is always used. MODENAME is not specified on the definition for either of these profiles, but you can add a MODENAME if you make your own copy. You must then ensure that the mode sets using your MODENAME have been defined in the TERMINAL or SESSIONS definition for all the systems with which communication takes place using APPC.

If a MODENAME is specified and you wish to remove it, delete completely the value previously specified by pressing the ERASE EOF key.

MSGINTEG({NO|YES}) (SNA LUs only)
specifies whether a definite response is to be requested with an output request to a logical unit. You cannot specify YES for a pipeline transaction.

MSGJRNL({NO|INPUT|OUTPUT|INOUT})
specifies which messages are to be automatically journaled. If you specify a value other than NO, you must also supply a value for the JOURNAL attribute.

NO No message journaling is required.

INPUT Journaling is required for input messages.

OUTPUT
Journaling is to be performed for output messages.

INOUT
Journaling is to be performed for input and output messages.

NEPCLASS({0|*value*}) (VTAM only)
specifies the node error program transaction class. This value overrides the value specified on the TYPETERM and SESSION definitions.

0 This results in a link to the default node error program module for VTAM devices, or is the default value for non-VTAM devices.

value The transaction class for the (nondefault) node error program module. The value can be in the range 1 through 255. For programming information on the node error program, see the *CICS Customization Guide*.

ONEWTE({NO|YES})
specifies whether the transaction is permitted only one write operation or EXEC CICS SEND during its execution. YES has the effect of forcing the LAST option on the first write of the transaction. Any additional write requests are treated as errors, and the task is made ready for abnormal termination.

You must specify YES for a PIPELINE transaction.

PRINTERCOMP({NO|YES})
specifies the level of compatibility required for the generation of data streams to support the printer compatibility option for the BMS SEND TEXT command.

NO Each line of output starts with a blank character, so that the format is equivalent to that on a 3270 display where an attribute byte precedes each line.

YES No blank character is inserted, so that forms-feed characters included as the first character of your data are honored and the full width of the printer is available for your data.

If you use the BMS forms feed option, specify YES.

PROFILE(*name*)
specifies the name of this PROFILE definition. The name can be up to eight characters in length. The acceptable characters are: A-Z 0-9 $ @ # . / - _ % & ¢ ? ! : | " = ¬ , ; < and >. Do not use profile names beginning with DFH, because these characters are reserved for use by CICS.

Note: If you use a comma (,) in a name, you will be unable to use those commands such as

```
CEMT INQUIRE PROFILE(value1,value2)
```

where the comma serves as a list delimiter. See the *CICS Supplied Transactions* manual for information about using lists of resource identifiers.

A profile specifies the options that control the interaction between CICS and a terminal or logical unit. A profile name is specified on the transaction definition to indicate the set of options that control the communication between the transaction and its principal terminal. You can also specify a profile name on an EXEC CICS ALLOCATE command to indicate the options that control communication between the transaction and the allocated session.

CICS supplies a number of profile definitions that are suitable for most purposes. For guidance on the names of the definitions, see the *CICS Operations and Utilities Guide*. Further guidance is also given in the *CICS Intercommunication Guide*.

PROTECT({NO|YES}) (SNA LUs only)
This attribute is obsolete from CICS Transaction Server for VSE/ESA, but is retained for compatibility with earlier releases. If you already use PROTECT, you can still access it by using compatibility mode (see *CICS Resource Definition Guide* for information). See *CICS Resource Definition Guide* for a description of PROTECT.

RAQ({NO|YES}) (SNA terminals only)
specifies whether the ‘read ahead queuing’ option is required.

NO The transaction obeys SNA protocols and only SEND and RECEIVE when in the correct mode. If it does not follow the protocol, it may be abended with code ATCV.

YES The transaction may not obey SNA protocols, and CICS queues incoming data on temporary storage until the data is specifically requested by the transaction. RAQ(YES) is provided only for compatibility with transactions that support both bisynchronous devices and logical units, and its use is not recommended.

RTIMOUT({NO|*value*})
specifies the time-out value for the read time-out feature. The task that is timed

out receives an AKCT or AZCT abend. (Note that if a value is specified and you wish to let it default to NO, you must completely delete the value previously specified.)

RTIMOUT has no effect for MRO or basic (unmapped) APPC connections.

NO The read time-out feature is not required.

value This is an interval (MMSS for minutes and seconds) after which the task is terminated if no input has been received from the terminal. The maximum value that can be specified is 70 minutes. The value specified in this option is rounded up to units of 16.78 seconds. Thus, the minimum value (after rounding-up) is 16.78 seconds.

SCRNSIZE({DEFAULT|ALTERNATE})
specifies whether the DEFAULT or ALTERNATE buffer size for a 3270 display or printer is to be used. For further information on the choice of screen sizes and buffer sizes, refer to the ALTSCREEN and DEFSCREEN attributes on the TYPETERM definition.

The SCRNSIZE value is ignored if the TYPETERM definition has ALTSCREEN(0,0) and DEFSCREEN(0,0). That is, the screen size is assumed from the related TERMMODEL attribute in the TYPETERM definition; the page size is taken from PAGESIZE, and the ALTPAGE value is ignored. The 3270 erase write (EW) command is inserted for output requests with the ERASE option.

ALTERNATE
If the TYPETERM definition has nonzero ALTSCREEN, the alternate screen size mode is applied, using the erase write alternate (EWA) command. That is, whenever a terminal output request with the ERASE option is issued, the 3270 EWA command is inserted in the data stream. The ALTSCREEN value is assumed as the screen size, and BMS uses the value in ALTPAGE as the page size.

SCRNSIZE(ALTERNATE) may be used for all CICS service transactions (for example, CSMT).

DEFAULT
If the TYPETERM definition has nonzero ALTSCREEN or nonzero DEFSCREEN, the default screen size mode is applied, using the erase write (EW) command. That is, whenever the terminal issues a terminal output request with the ERASE option, the 3270 EW command is inserted in the data stream. The screen size specified in the DEFSCREEN attribute is assumed, and BMS uses the value specified in the PAGESIZE attribute as the page size.

Note: Both DEFAULT and ALTERNATE can be overridden by the DEFAULT and ALTERNATE options on the SEND MAP, SEND TEXT, and SEND CONTROL commands. See the *CICS Application Programming Reference* manual for programming information about these commands.

UCTRAN({NO)|YES}) (VTAM only)
specifies whether terminal input is to be translated to uppercase before passing to programs for the transaction using this profile.

You can also request translation to uppercase at the terminal level on the associated TYPETERM definition (see *CICS Resource Definition Guide*) but be aware of the following points:

- A TYPETERM UCTRAN(YES) definition overrides a PROFILE UCTRAN(NO) definition. So, if you specify TYPETERM UCTRAN(YES), a PROFILE UCTRAN(NO) has no effect.
- A PROFILE UCTRAN(YES) definition overrides a TYPETERM UCTRAN(NO) definition.
- Specifying TYPETERM UCTRAN(TRANID) causes the tranid to be translated to uppercase so that CICS can locate the transaction definition. All other input received by the application is translated according to what is specified for PROFILE UCTRAN.
- UCTRAN(YES) on a profile definition does not cause translation of the input data until an EXEC CICS RECEIVE or CONVERSE is executed. This means that if the transaction is routed through a dynamic routing program, for example DFHDYP, the copy of the input data passed to the routing program is unaffected by the UCTRAN option of the PROFILE definition.

Note: In a transaction routing environment where your VTAM terminals have a remote definition on the AOR, and the AOR has a different UCTRAN value from the TOR, the TOR value of UCTRANST (as specified in an EXEC CICS SET TERMINAL command) overrides that on the AOR.

Table 3 shows which portion of the terminal input is translated (transaction id and/or data) according to the setting of the UCTRAN on the PROFILE and TYPETERM resource definitions.

Table 3. The effect of UCTRAN attributes on tranid and data translation

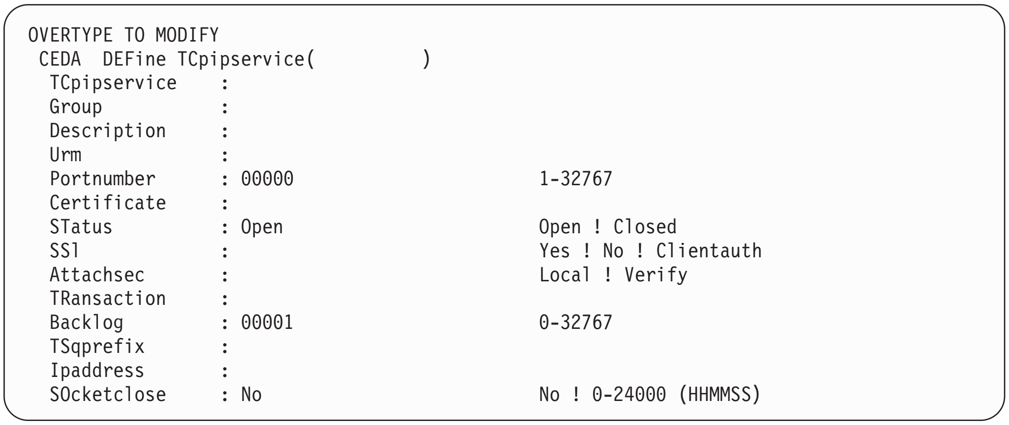
Profile (PROFILE)	**Terminal (TYPETERM)**		
	UCTRAN (YES)	**UCTRAN (NO)**	**UCTRAN (TRANID)**
UCTRAN (YES)	Tranid: Yes Data: yes	Tranid: No Data: Yes	Tranid: Yes Data: Yes
UCTRAN (NO)	Tranid: Yes Data: Yes	Tranid: No Data: No	Tranid: Yes Data: No

Chapter 16. CEDA DEFINE TCPIPSERVICE

Use this resource definition to define which TCP/IP services are to use CICS internal sockets support. The internal CICS service that can be defined is the CICS Web Interface.

The TCPIPSERVICE definition allows you to manage these internal CICS interfaces, with CICS listening on multiple ports, with different flavors of CICS Web support on different ports.

Defining a TCPIPSERVICE



```
OVERTYPE TO MODIFY
 CEDA  DEFine TCpipservice(          )
  TCpipservice    :
  Group           :
  Description     :
  Urm             :
  Portnumber      : 00000                      1-32767
  Certificate     :
  STatus          : Open                       Open ! Closed
  SSl             :                            Yes ! No ! Clientauth
  Attachsec       :                            Local ! Verify
  TRansaction     :
  Backlog         : 00001                      0-32767
  TSqprefix       :
  Ipaddress       :
  SOcketclose     : No                         No ! 0-24000 (HHMMSS)
```

Figure 15. The DEFINE panel for the TCPIPSERVICE resource definition

Examples of TCPIPSERVICE resource definitions are given in:

Figure 1 on page 77

Figure 11 on page 94

Options

ATTACHSEC

specifies the ECI security requirements of the CICS client. Possible values are:

VERIFY

The CICS client must provide a userid and password to the VSE/ESA host. These are checked before the CICS client is allowed to communicate with the VSE/ESA host.

LOCAL

The CICS client is not required to provide a userid and password to the VSE/ESA host.

You can define some CICS clients to have LOCAL security, and others to have VERIFY by using *two* TCPIPSERVICE definitions that:

- Have different security requirements.
- Listen on different ports.

You can then set up each CICS client (using the client's CICSCLI.INI file) to communicate with the port supporting a specific security type.

BACKLOG(*number***)**
specifies the number of TCP/IP connections for this service which are queued in TCP/IP before TCP/IP starts to reject incoming client requests.

CERTIFICATE(*certificate-label***)**
The TCP/IP service uses one of the certificates in the VSE Keyring Library as its server certificate. Use the CERTIFICATE option to specify a particular certificate (where *certificate_label* is the name that you assigned to the certificate). If you do not specify CERTIFICATE, CICS uses the default server certificate in the VSE Keyring Library.

DESCRIPTION(*text***)**
You can provide a description of the resource you are defining in this field. The DESCRIPTION text can be up to 58 characters in length. There are no restrictions on the characters that you may use. However, if you use parentheses, ensure that for each left parenthesis there is a matching right one. For each single apostrophe in the text, code two apostrophes.

GROUP(*groupname***)**
Every resource definition must have a GROUP name. The resource definition becomes a member of the group and is installed in the CICS system when the group is installed. For more information about groups, see *CICS Resource Definition Guide*.

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

IPADDRESS
specifies the dotted decimal IP address on which this TCPIPSERVICE will listen for incoming connections. It must be of the form nnn.nnn.nnn.nnn where nnn is 0 through 255. Possible values are:

blank When no IPaddress is specified, the default is *INADDR_ANY*

value The TCPIPSERVICE accepts connections on this particular address. If the address specified is not known to TCP/IP on the VSE/ESA host, the TCPIPSERVICE will not open. If you enter a specific address here, this definition may not be valid for CICS servers running on other VSE/ESA partitions, and you may not be able to share the definition with those servers.

INADDR_ANY
The TCPIPSERVICE listens on any of the addresses known to TCP/IP for the VSE/ESA host. It is possible to have multiple IP addresses defined for a host. Specifying INADDR_ANY also allows for the TCPIPSERVICE definition to be shared among CICS servers.

PORTNUMBER(*number***)**
specifies, in the range 1 through 65535, the decimal number of the TCP/IP port on which CICS is to listen for incoming client requests. For SSL support, this is the port number on which the SSL service is provided. For ECI support, this is the port number to be used by CICS when listening for ECI requests (default 1435).

SOCKETCLOSE(NO|*0-240000***)**
specifies if, and for how long, CICS should wait before closing the socket, after issuing a receive for incoming data on that socket.

No The socket is left open until data is received. While the socket is open it is unavailable to other tasks, and its associated CICS task is suspended indefinitely.

0-240000
The period of time (in HHMMSS format) after which CICS is to close the socket. Specifying 000000 closes the socket immediately if data is available

After the TCPIPSERVICE is installed, this value cannot be modified using CEMT; you must set the TCPIPSERVICE out of service, then re-install the TCPIPSERVICE with the modified definition.

CICS ignores the SOCKETCLOSE attribute on the first SOCKET RECEIVE issued after a connection is made. When a connection is accepted by the socket listener, it attaches a task to handle to connection. The first thing this task does is to issue a SOCKET RECEIVE. This first RECEIVE waits indefinitely for data, whatever you have specified on SOCKETCLOSE. This prevents a socket connection being closed as soon as it is initiated, even if no data is immediately available, and so prevents connection reset error at the client.

SSL(NO|YES|CLIENTAUTH)
specifies the level of SSL to be used:

NO No SSL support.

YES SSL support is to be activated. Clients connecting to the specified port number must use the SSL protocol to connect with CICS (that is, they must specify `https` rather than `http` as the protocol in the URL used to access the service).

CLIENTAUTH
The CICS client, as well as the server, must have a certificate. The client certificate is received by CICS during the SSL handshake, and can be used to determine the userid under which the CICS transaction can be executed.

If you specify SSL(CLIENTAUTH), this will not force the client to supply a certificate. Your client's Web browser program might not support client certificates, or the client might not have a certificate installed.

STATUS({OPEN|CLOSED})
Indicates the initial status of the service after installation. Set it to OPEN if CICS is to begin listening for this service after installation. Set to CLOSE if CICS is not to listen on behalf of this service after installation.

TCPIPSERVICE(*name*)
specifies the 8-character name of this service.

TRANSACTION(*name*)
specifies the 4-character ID of the CICS transaction attached to process new requests received for this service. For a Web TCPIPSERVICE definition, specify CWXN or an alias of CWXN. For ECI support, specify CIEP (which is required together with its associated program DFHIEP).

TSQPREFIX
specifies the 6-character prefix of the temporary storage queue used to store inbound data and Web documents created by applications.

URM(*name*)
specifies the name of the user-replaceable module to be invoked by this service. Implementation of this depends on the service being defined; this can be the name of any CICS program.

Part 7. CICS Supplied Transactions Changes

Chapter 17. CEBR—temporary storage browse

Use the CEBR transaction to browse temporary storage queues and to delete them.

You can also use CEBR to transfer a transient data queue to temporary storage in order to look at its contents, and to recreate the transient data queue when you have finished. The CEBR commands that perform these transfers allow you to create and delete transient data queues as well.

Remember that:

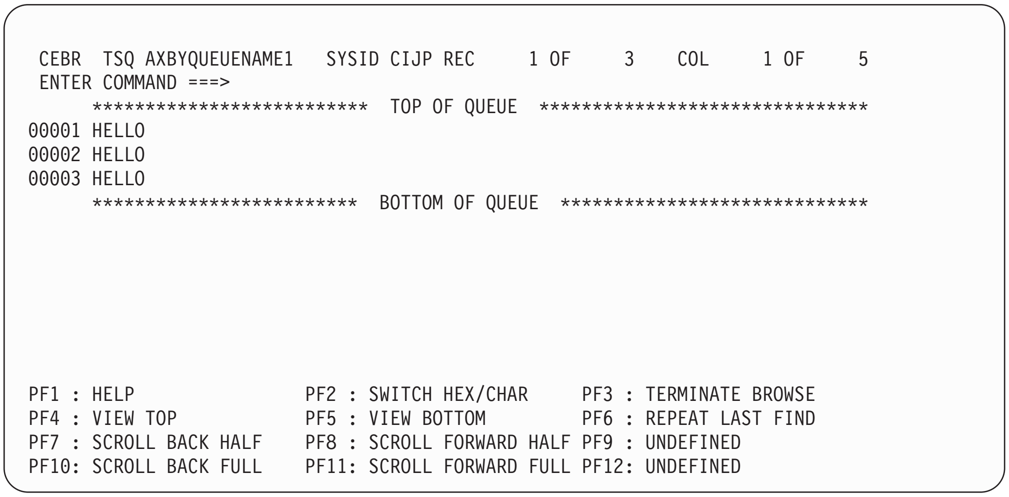
- browsing of Temporary Storage queues retrieves the next record, following whichever record has most recently been retrieved by *ANY active task.*

  This is can lead to confusion if for example an EXEC CICS READQ NEXT and a CEBR transaction attempt to work with the same Temporary Storage queue at the same time.
- transient data queue reads are destructive. If you read a transient data queue that is being used on an active system by applications, this is likely to cause problems.

You start the CEBR transaction by entering the transaction identifier CEBR, followed by the name of the queue you want to browse. For example, to display the temporary storage queue named CEBRS209, you enter:

```
CEBR CEBRS209
```

CICS responds with a display of the queue, a sample of which is shown in Figure 16:

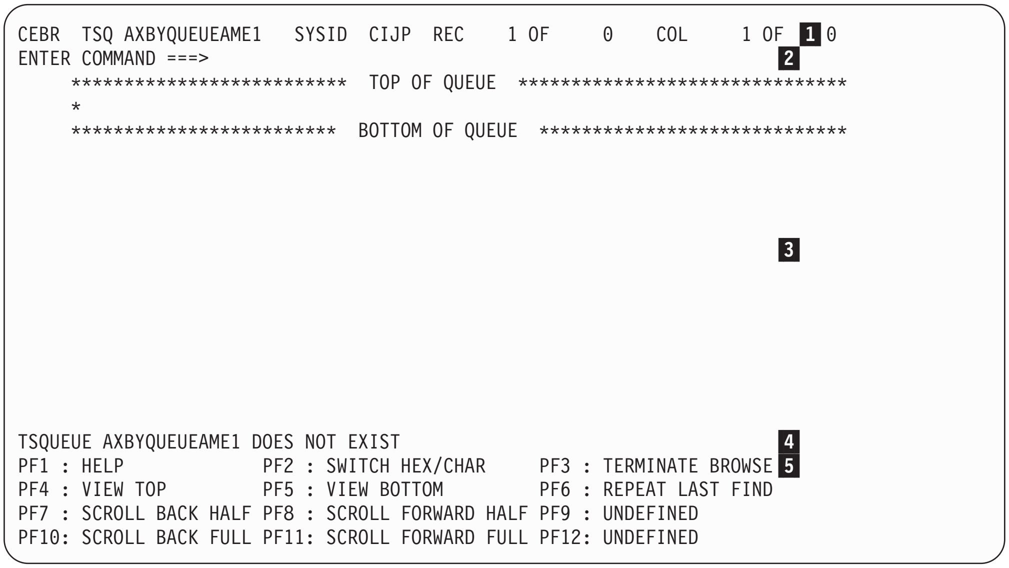


```
 CEBR  TSQ AXBYQUEUENAME1   SYSID CIJP REC     1 OF     3    COL     1 OF     5
 ENTER COMMAND ===>
      **************************  TOP OF QUEUE  *******************************
00001 HELLO
00002 HELLO
00003 HELLO
      *************************  BOTTOM OF QUEUE  *****************************







PF1 : HELP               PF2 : SWITCH HEX/CHAR     PF3 : TERMINATE BROWSE
PF4 : VIEW TOP           PF5 : VIEW BOTTOM         PF6 : REPEAT LAST FIND
PF7 : SCROLL BACK HALF   PF8 : SCROLL FORWARD HALF PF9 : UNDEFINED
PF10: SCROLL BACK FULL   PF11: SCROLL FORWARD FULL PF12: UNDEFINED
```

Figure 16. Typical CEBR screen displaying temporary storage queue contents

Alternatively, you can start the CEBR transaction from CEDF. You do this by pressing PF5 from the initial CEDF screen (see *CICS Supplied Transactions*) which takes you to the working-storage screen, and then you press PF2 from that screen to browse temporary storage (that is, you invoke CEBR). CEBR responds by displaying the temporary storage queue whose name consists of the four letters ‘CEBR’ followed by the four letters of your terminal identifier. (CICS uses this same default queue name if you invoke CEBR directly and do not supply a queue name.) The result of invoking CEBR without a queue name or from an EDF session at terminal S21A is shown in Figure 17.

If you enter CEBR from CEDF, you will return to the EDF panel when you press PF3 from the CEBR screen.



```
CEBR  TSQ AXBYQUEUEAME1   SYSID  CIJP  REC    1 OF     0    COL     1 OF  0
ENTER COMMAND ===>
     **************************  TOP OF QUEUE  *******************************
     *
     *************************  BOTTOM OF QUEUE  *****************************















TSQUEUE AXBYQUEUEAME1 DOES NOT EXIST
PF1 : HELP             PF2 : SWITCH HEX/CHAR     PF3 : TERMINATE BROWSE
PF4 : VIEW TOP         PF5 : VIEW BOTTOM         PF6 : REPEAT LAST FIND
PF7 : SCROLL BACK HALF PF8 : SCROLL FORWARD HALF PF9 : UNDEFINED
PF10: SCROLL BACK FULL PF11: SCROLL FORWARD FULL PF12: UNDEFINED
```

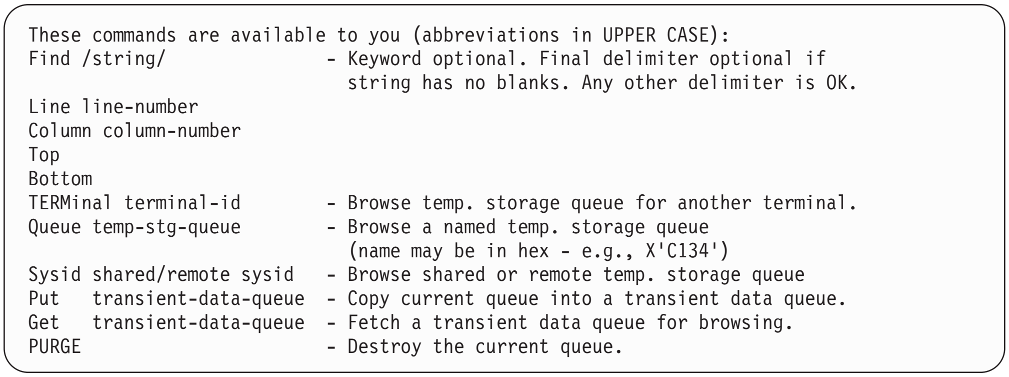
Note: **1** Header **2** Command line **3** Body **4** Message line **5** Menu of options

Figure 17. Typical CEBR display of default temporary storage queue

For information about the CEBR temporary storage browse transaction and guidance on using it, see the *CICS Application Programming Guide*.

The HELP panel

If you press the help key (PF1), the following panel is displayed:



```
These commands are available to you (abbreviations in UPPER CASE):
Find /string/              - Keyword optional. Final delimiter optional if
                             string has no blanks. Any other delimiter is OK.
Line line-number
Column column-number
Top
Bottom
TERMinal terminal-id       - Browse temp. storage queue for another terminal.
Queue temp-stg-queue       - Browse a named temp. storage queue
                             (name may be in hex - e.g., X'C134')
Sysid shared/remote sysid  - Browse shared or remote temp. storage queue
Put   transient-data-queue - Copy current queue into a transient data queue.
Get   transient-data-queue - Fetch a transient data queue for browsing.
PURGE                      - Destroy the current queue.
```

Chapter 18. CEDX — the execution diagnostic facility

The CICS execution diagnostic facility (EDF) provides two transactions that you can use for testing application programs. These transactions—CEDF and CEDX—enable you to test application programs interactively without having to supply special program testing procedures.

CEDF
Use CEDF to invoke EDF for testing application programs that are associated with user transactions initiated from a terminal.

CEDX
Use CEDX to invoke EDF for testing application programs that are associated with non-terminal transactions.

CEDF transaction

The CEDF transaction is fully documented in the *CICS Supplied Transactions* manual and is not further described here.

For information on how to use the CEDF transaction, see the *CICS Application Programming Guide*.

You must ensure that the EDF resource definitions are installed. These resource definitions are provided in the IBM-supplied group, DFHEDF. For information about installing the resource definitions, see the *CICS Resource Definition Guide*.

CEDX transaction

Use CEDX to monitor and debug non-terminal transactions. The transaction you specify for debugging can be:

- Initiated without an associated terminal; for example, by an EXEC CICS START command, or by a transient data queue trigger-level.
- Initiated at a terminal, which can be either the EDF terminal or a different terminal.

CICS intercepts the transaction specified on the CEDX *tranid* command, and displays the EDF diagnostic panels at the terminal at which the EDF command is issued.

CEDX provides the same function and diagnostic display panels as CEDF, and the same basic rules for CEDF also apply to CEDX.

Command syntax

CEDX

```
                      ┌─,ON──┐
►►──CEDX──tranid──────┤      ├──────────────────────────────────────────►◄
                      └─,OFF─┘
```

Command options

OFF
specifies that the EDF screen is to be switched OFF. If you specify OFF you must enter the preceding comma, as shown in the following example:

```
CEDX TRNA,OFF
```

ON
specifies that the EDF screen is to be switched ON. The default is ON. If you specify ON you must enter the preceding comma, as shown in the following example:

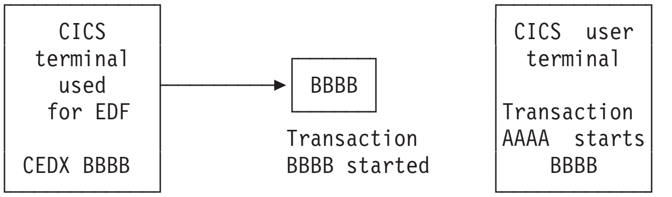
```
CEDX TRNB,ON
```

tranid
specifies the identifier of a transaction that you want to test using the execution diagnostic facility. The diagnostic panels are displayed on the terminal at which you issue the CEDX *tranid* command.

The transaction you specify on the CEDX command must run in the local CICS region.

CEDX cannot interrupt transactions that are already running when you issue the CEDX command. It affects only a transaction that starts *after* you issue the command.

The following diagram illustrates the use of CEDX to invoke EDF for a transaction initiated by an EXEC CICS START command:



The CEDX command is issued at this terminal, **before** BBBB is started, to invoke EDF services to intercept and debug the specified transaction.

AAAA issues EXEC CICS START command to start BBBB, without a terminal.

Chapter 19. CEMT INQUIRE/SET commands

Most CEMT requests either inquire about (INQUIRE), or change (SET) the status of one or more named instances of a resource (such as a terminal), a particular subgroup of resources (such as a class of terminals), or all resources of a particular type (such as all terminals).

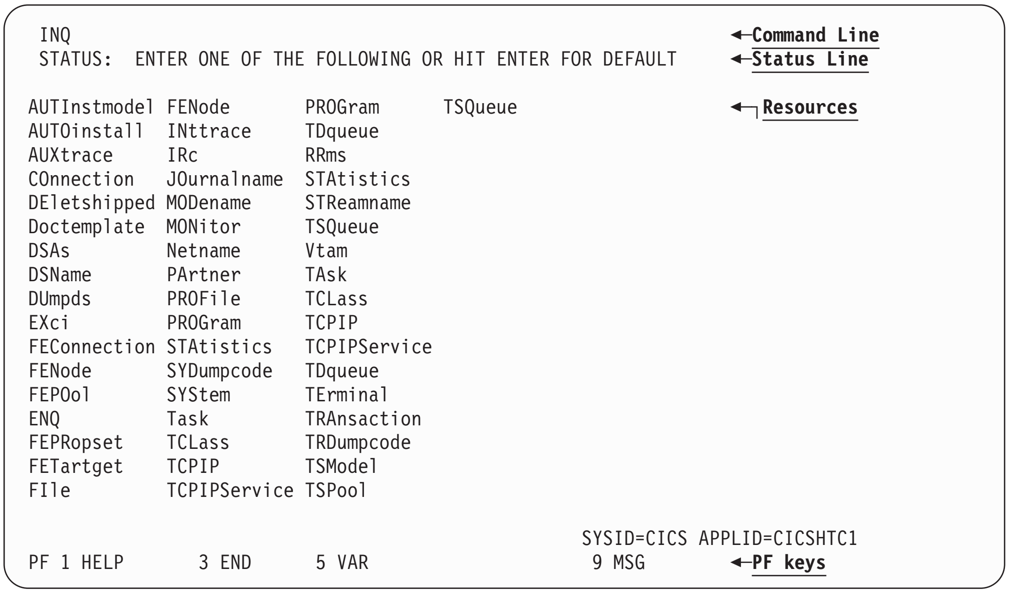
The INQUIRE command causes the status of the specified resources to be displayed. The SET command makes the changes that you specify, and displays the new status of the resources. No changes are made if there are syntax errors in the SET command.

If, for example, you want to inquire about or set a resource, enter INQUIRE (or a suitable abbreviation) on the command line. The keywords you can use with CEMT INQUIRE are described in the rest of this section.

If you want to perform those functions that are not concerned with resource status, enter PERFORM (or a suitable abbreviation) on the command line. The keywords you can use with PERFORM are described in *CICS Supplied Transactions*.

And finally, if you want to change the attributes of a resource, enter SET (or a suitable abbreviation) on the command line. The keywords you can use with SET are described in *CICS Supplied Transactions*.

If you enter INQUIRE, you get the following display:



```
 INQ
 STATUS:  ENTER ONE OF THE FOLLOWING OR HIT ENTER FOR DEFAULT

AUTInstmodel FENode       PROGram      TSQueue
AUTOinstall  INttrace     TDqueue
AUXtrace     IRc          RRms
COnnection   JOurnalname  STAtistics
DEletshipped MODename     STReamname
Doctemplate  MONitor      TSQueue
DSAs         Netname      Vtam
DSName       PArtner      TAsk
DUmpds       PROFile      TCLass
EXci         PROGram      TCPIP
FEConnection STAtistics   TCPIPService
FENode       SYDumpcode   TDqueue
FEPOol       SYStem       TErminal
ENQ          Task         TRAnsaction
FEPRopset    TCLass       TRDumpcode
FETartget    TCPIP        TSModel
FIle         TCPIPService TSPool

                                                  SYSID=CICS APPLID=CICSHTC1
PF 1 HELP       3 END      5 VAR                   9 MSG
```

Figure 18. Sample of the screen following either the INQUIRE or the SET command

You can inquire about any of the displayed options by typing its keyword after INQUIRE on the command line. For example,

```
INQUIRE PROGRAM
```

gives you the status of all programs, and for each program gives its attributes. Full details are given in *CICS Supplied Transactions*.

CEMT INQUIRE DOCTEMPLATE

Function

Retrieves information about a DOCTEMPLATE

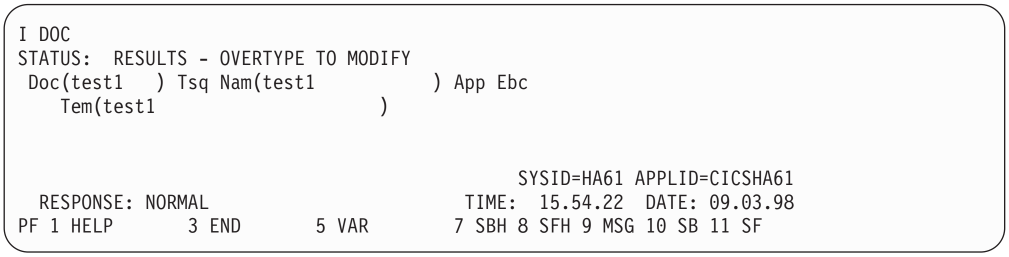
Description

INQUIRE DOCTEMPLATE returns information about any currently installed document template names.

Input

Press the Clear key to clear the screen, and type `CEMT INQUIRE DOCTEMPLATE` (the minimum abbreviation is `CEMT I DO`). You get a display that lists the names and status of any document templates. Note that you cannot change any of the displayed information.

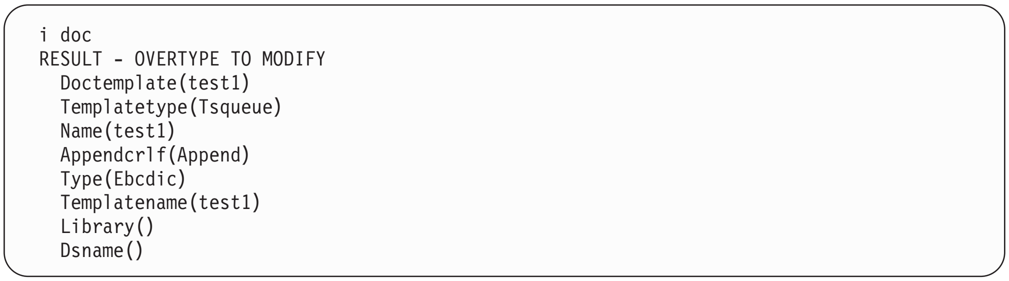
Sample screen



```
I DOC
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Doc(test1   ) Tsq Nam(test1           ) App Ebc
    Tem(test1                     )



                                             SYSID=HA61 APPLID=CICSHA61
  RESPONSE: NORMAL                        TIME:  15.54.22  DATE: 09.03.98
PF 1 HELP       3 END       5 VAR        7 SBH 8 SFH 9 MSG 10 SB 11 SF
```

Figure 19. CEMT INQUIRE DOCTEMPLATE screen

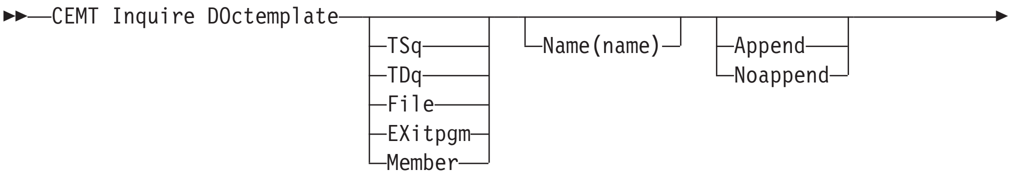
If you place the cursor against a specific entry in the list and press ENTER, CICS displays an expanded format.



```
  i doc
  RESULT - OVERTYPE TO MODIFY
    Doctemplate(test1)
    Templatetype(Tsqueue)
    Name(test1)
    Appendcrlf(Append)
    Type(Ebcdic)
    Templatename(test1)
    Library()
    Dsname()
```

Figure 20. The expanded display of an individual entry

CEMT INQUIRE DOCTEMPLATE



```
►──┬────────┬──┬────────────────────┬──┬────────────────────┬──┬─────────┬──────►◄
   ├─Ebcdic─┤  └─Templatename(name)─┘  └─Templatetype(name)─┘  ├─Library─┤
   └─Binary─┘                                                  └─Dsname──┘
```

parameters;

APPENDCRLF
returns whether CICS is to delete trailing blanks from and append carriage-return line-feed to each logical record of the template.

LIBRARY
returns the file name of the library containing the template. The library applies only to a template of type Library. If a membername is supplied without a value for LIBRARY, the default value DFHHTML is used.

DOCTEMPLATE
returns the name of this DOCTEMPLATE definition.

DSNAME
returns the file-id of the library containing the template. The Dsname applies only to a template of type LIBRARY.

NAME
returns the name of the location defined in TEMPLATETYPE.

TEMPLATENAME
returns the extended template name by which the doctemplate is known outside the resource definition function.

TEMPLATETYPE
returns the type of resource whose name is returned in NAME.

EXITPGM
an exit program.

FILE a CICS file name for a data set.

MEMBER
a name of the member in the library described in LIBRARY.

PROGRAM
a name of a program.

TDQUEUE
a name of a TD queue.

TSQUEUE
a name of a TS queue.

TYPE(BINARY|EBCDIC)
returns the format of the template contents.

BINARY
When the template is loaded from the template library, no parsing of the template's contents is done.

EBCDIC
When the template is loaded from the template library, the contents are parsed as EBCDIC text.

CEMT INQUIRE/SET TASK

Function

Retrieve information about a user task.

Description

INQUIRE TASK returns information about user tasks. Only information about user tasks can be displayed or changed; information about CICS-generated system tasks or subtasks cannot be displayed or changed. System tasks are those tasks started (and used internally) by CICS, and not as a result of a user transaction.

Input

Press the Clear key to clear the screen. There are two ways of commencing this transaction:

- Type `CEMT INQUIRE TASK` (the minimum abbreviation is `CEMT I TA`). You get a display that lists the current status.
- Type `CEMT INQUIRE TASK` (`CEMT I TA`) followed by as many of the other attributes as are necessary to limit the range of information that you require. So, for example, if you enter `cemt i ta i`, the resulting display will show you the details of only those tasks for which the data is not shared with other tasks (isolated).

To change various attributes, you can:

- Overtype your changes on the INQUIRE screen after tabbing to the appropriate field (see ″Overtyping a display″ in the CICS Supplied Transaction manual).
- Use the CEMT SET TASK command.

CEMT INQUIRE TASK

```
                           ┌─ALl──────────────┐
►►──CEMT Inquire TAsk──────┼──────────────────┼──┬─────────────────┬──────────────────────────►
                           ├─(value)──────────┤  └─TRanid(value)───┘
                           └─TClass(value)────┘

►──┬──────────────────┬──┬───────────────┬──┬───────┬──┬───────────────────┬──────────────────►
   └─FAcility(value)──┘  ├─Running───────┤  ├─TAsk──┤  └─PRiority(value)───┘
                         ├─DIspatchable──┤  ├─TErm──┤
                         └─SUspended─────┘  └─DEst──┘

►──┬───────────────────┬──┬───────────────┬──┬──────────────┬──┬───────────────┬──────────────►
   └─STartcode(value)──┘  └─USerid(value)─┘  └─HTYpe(value)─┘  └─HValue(value)─┘

►──┬───────────────┬──┬───────────────┬──┬───────────────────┬───────────────────────────────►◄
   └─HTIme(value)──┘  └─BRidge(value)─┘  └─IDentifier(value)─┘
```

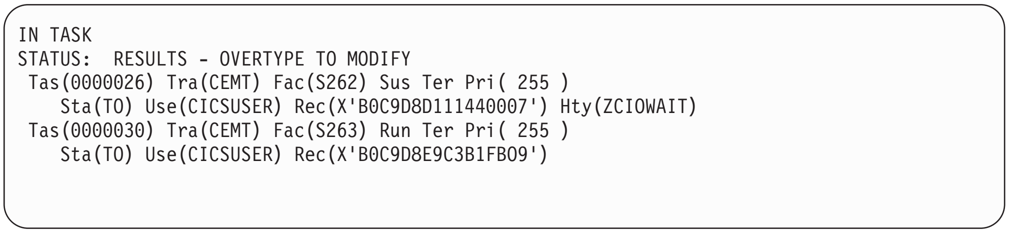
(*value***)**
  is the CICS-generated task number, in the range 1–99999.

ALl
  is the default.

TClass**(***value***)**
  is the 8-character transaction class name to which the transaction belongs.

You cannot specify a list of identifiers, nor can you use the symbols * and + to specify a family of tasks.

Sample screen

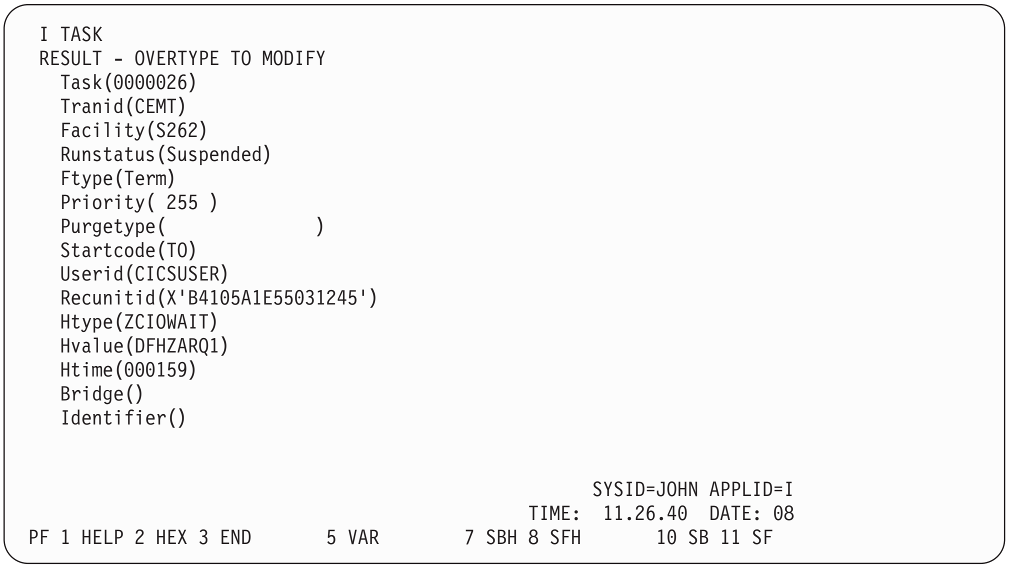


```
IN TASK
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Tas(0000026) Tra(CEMT) Fac(S262) Sus Ter Pri( 255 )
    Sta(TO) Use(CICSUSER) Rec(X'B0C9D8D111440007') Hty(ZCIOWAIT)
 Tas(0000030) Tra(CEMT) Fac(S263) Run Ter Pri( 255 )
    Sta(TO) Use(CICSUSER) Rec(X'B0C9D8E9C3B1FBO9')
```

Figure 21. CEMT INQUIRE TASK screen

Note: There are blank fields on the screen where a value does not apply or is ‘negative’; that is, the value begins with ‘No’. To modify these fields, locate them by tabbing (they appear in the same sequence as in the expanded format), and overtype with input valid for that field. You may find it more convenient to use the expanded format when setting one of these values, (negating is easier because you are overtyping a displayed field with its negative causing the result to become non-displayed).

If you place the cursor against a specific entry in the list and press ENTER, CICS displays an expanded format as shown in Figure 22.



```
  I TASK
  RESULT - OVERTYPE TO MODIFY
    Task(0000026)
    Tranid(CEMT)
    Facility(S262)
    Runstatus(Suspended)
    Ftype(Term)
    Priority( 255 )
    Purgetype(             )
    Startcode(TO)
    Userid(CICSUSER)
    Recunitid(X'B4105A1E55031245')
    Htype(ZCIOWAIT)
    Hvalue(DFHZARQ1)
    Htime(000159)
    Bridge()
    Identifier()


                                                      SYSID=JOHN APPLID=I
                                                TIME:  11.26.40  DATE: 08
 PF 1 HELP 2 HEX 3 END       5 VAR        7 SBH 8 SFH       10 SB 11 SF
```

Figure 22. The expanded display of an individual entry

parameters;

BRidge(*value*)
Returns the 4-character name of the bridge monitor transaction if the current

task is running in a 3270 bridge environment, and was started by a bridge monitor transaction with a START BREXIT TRANSID command. Otherwise, blanks are returned.

FAcility**(***value***)**
displays a 4-character string identifying the name of the terminal or queue that initiated the task. If no FACILITY value is displayed, the task was started without a facility.

Ftype
displays the type of facility that initiated this task. The values are:

TAsk The task was initiated from another task.

TErm The task was initiated from a terminal.

DEst The task was initiated by a destination trigger level as defined in the destination control table (DCT).

HTIme**(***value***)**
displays the time (in seconds) that the task has been in the current suspended state.

HTYpe**(***value***)**
displays the reason why the task is suspended. A null value indicates that there is no hold-up, except for the necessity of reaching the head of the queue.

HValue**(***value***)**
displays a 16-character resource name, such as a file name, or a value such as a TCLASS value.

For information on the values that can appear in the HTYPE and HVALUE options, and how they can be used as an aid in problem determination, see the “resource type” and “resource name” details in the *CICS Problem Determination Guide*.

IDentifier**(***value***)**
returns a 48-character field containing user data provided by the bridge exit, if the task was initiated in the 3270 bridge environment, or blanks, otherwise. This field is intended to assist in online problem resolution.

For example, it could contain the MQ™ correlator for the MQ bridge, or a Web token.

PRiority**(***value***)**
displays the priority of the task, in the range 0–255 where 255 is the highest priority.

Note: You can reset this value by overtyping it with a different value.

Purgetype **(***input only field***)**
specifies whether a task is to be purged or forcepurged. The values are:

Purge The task is to be terminated. Termination occurs only when system and data integrity can be maintained.

Forcepurge
The task is to be terminated immediately. System integrity is not guaranteed. In some extreme cases, for example if a task is forcepurged during backout processing, CICS terminates abnormally. If you want to terminate a task but do not want to terminate CICS, you should use PURGE instead of FORCEPURGE.

In some BTAM situations, further user action is required to complete the purging of the task. If a task is in a terminal read on a BTAM nonlocal terminal and that task is canceled, it can require input from another terminal on the same line before the cancel operation completes.

Recunitid**(***value***)**
displays a unique identifier of the current unit of recovery. The identifier takes the form of an eight byte clock value and is displayed in hexadecimal format.

Runstatus
displays the status of this task. The values are:

Running
The task is running.

DIspatchable
The task is dispatchable.

SUspended
The task is suspended.

STartcode**(***value***)**
displays how this task was started. The values are:

D	A distributed program link (DPL) request. The program cannot issue I/O requests against its principal facility or any syncpoint requests.
DS	A distributed program link (DPL) request, as for code D, with the exception that the program can issue syncpoint requests.
QD	A transient data trigger level was reached.
S	Start command (no data)
SD	Start command (with data)
TO	The operator typed a transaction code at the terminal.
TP	The transaction was started by presetting the transaction ID for the terminal.
U	User-attached task.

TAsk**(***value***)**
indicates that this panel relates to a TASK inquiry and displays a CICS-generated task number in the range 1–99999.

TRanid**(***value***)**
displays a 4-character string identifying the transaction name associated with the task.

USerid**(***value***)**
displays the user currently associated with the task.

CEMT INQUIRE/SET TCPIP

Function

Inquire about CICS internal TCP/IP support status.

Description

INQUIRE TCPIP returns information about the current status of CICS internal TCP/IP support.

Input

Press the Clear key to clear the screen. There are two ways of commencing this transaction:

- Type `CEMT INQUIRE TCPIP` (the minimum abbreviation is `CEMT I TCPIP`). You get a display that lists the current status.
- Type `CEMT INQUIRE TCPIP` (`CEMT I TCPIP`) followed by as many of the other attribute settings that you wish to view.

To change various attributes, you can:

- Overtype your changes on the INQUIRE screen after tabbing to the appropriate field (see *CICS Supplied Transactions*).
- Use the CEMT SET TCPIP command.

CEMT INQUIRE TCPIP

```
►►──INQUIRE TCPIP──┬─────────────────────┬──────────────────────────────────►◄
                   └─OPENSTATUS(value)───┘
```

OPENSTATUS(value)
displays the status of CICS internal TCP/IP support. The values are:

OPEN CICS internal sockets support is open.

CLOSED
CICS internal sockets support has not yet been activated, or has been terminated.

CLOSING
CICS internal sockets support is in the process of closing.

IMMCLOSING
CICS internal sockets support is in the process of immediate termination.

CEMT INQUIRE/SET TCPIPSERVICE

Function

Retrieve information about TCP/IP ports on which CICS internal TCP/IP support is currently listening on behalf of other CICS services.

Description

INQUIRE TCPIPSERVICE returns information about the state of a service using CICS internal TCP/IP support.

Input

Press the Clear key to clear the screen. There are two ways of commencing this transaction:

- Type `CEMT INQUIRE TCPIPSERVICE` (the minimum abbreviation is `CEMT I TCPIPS`). You get a display that lists the current status.
- Type `CEMT INQUIRE TCPIPSERVICE` (`CEMT I TCPIPS`) followed by as many of the other attribute settings that you wish to view.

To change various attributes, you can:

- Overtype your changes on the INQUIRE screen after tabbing to the appropriate field (see *CICS Supplied Transactions*).
- Use the CEMT SET TCPIPSERVICE command.

CEMT INQUIRE TCPIPSERVICE

```
►►──INQUIRE TCPIPSERVICE(value)──┬──────────────────┬──┬────────────────┬──────────►
                                 ├─Attachsec(value)─┤  └─Backlog(value)─┘
                                 ├─VERIFY───────────┤
                                 └─LOCAL────────────┘

►──┬────────────────────┬──┬────────────────────┬──┬───────────────────┬─────────►
   └─Certificate(value)─┘  └─CLOsetimeout(value)─┘  └─Connection(value)─┘

►──┬──────────────────┬──┬───────────────────┬──┬─────────────┬───────────────────►
   └─IPaddress(value)─┘  ├─OPENSTATUS(value)─┤  └─Port(value)─┘
                         ├─OPEN──────────────┤
                         ├─CLOSING───────────┤
                         ├─CLOSED────────────┤
                         └─IMMCLOSING────────┘

►──┬────────────────────┬──┬────────────┬──┬────────────────────┬────────────────►
   ├─SOcketclose(value)─┤  ├─SSL(value)─┤  └─TCPIPService(value)─┘
   ├─Wait───────────────┤  ├─NO─────────┤
   └─TImeout────────────┘  ├─YES────────┤
                           └─CLIENTAUTH─┘

►──┬──────────────┬──┬─────────────────┬──┬──────────┬───────────────────────────►◄
   └─TRansid(value)─┘  └─TSqprefix(value)─┘  └─URM(value)─┘
```

Attachsec
returns the ECI security requirements of a CICS client. Attachsec values are:

VERIFY
The CICS client must provide a userid and password to the VSE/ESA host. These are checked before the CICS client is allowed to communicate with the VSE/ESA host.

LOCAL
The CICS client is not required to provide a userid and password to the VSE/ESA host.

Backlog
Change the maximum number of requests which can be queued in TCP/IP waiting to be processed by the service.

Certificate
returns the label of a server certificate stored in the VSE Keyring Library. If your own certificate has not been specified, the label of the default server certificate in the VSE Keyring Library is returned.

Closetimeout
returns a fullword value containing the number of seconds specified for the timeout period. This can be 0 through 86400 (24 hours).

Connection
The number of current sockets connections for this service

Ipaddress
returns an IP address. If you specify an IP address in the TCPIPSERVICE definition, that address is returned, otherwise the default IP address is returned. If there is more than one IP address on an OS/390 system, only the default is returned.

Port
returns the number of the port on which CICS is listening on behalf of this service.

Socketclose
returns a CVDA value indicating whether a timeout value is in effect for the TCPIPSERVICE. CVDA values are:

WAIT
NO was specified on the definition. Socket receives will wait for data indefinitely.

TIMEOUT
A value was specified for the SOCKETCLOSE parameter on the definition.

SSL
returns the level of SSL that has been activated. SSL values are:

NO
SSL is not activated.

YES
SSL support is activated. Clients connecting to the specified port number use the SSL protocol to connect with CICS (that is, they must specify `https` rather than `http` as the protocol in the URL used to access the service).

CLIENTAUTH
The client, as well as the server, must have a certificate. The client certificate is received by CICS during the SSL handshake, and can be used either to determine the userid under which the CICS transaction can be executed.

TCPIPService
8-character name identifying this service.

OPENSTATUS(cvda)
returns a CVDA value indicating the state of the service. CVDA values are:

OPEN CICS internal sockets support is open.

CLOSED
CICS internal sockets support has not yet been activated, or has been terminated.

CLOSING
CICS internal sockets support is in the process of closing.

IMMCLOSING
CICS internal sockets support is in the process of immediate termination.

Transid
4-character transaction id used on the attach for the task started to process a new request. For a Web TCPIPSERVICE definition, CWXN or an alias of CWXN will be returned. For ECI, CIEP will be returned.

TSQprefix
returns the 6–character prefix used when the service generates TS queue names.

URM
8 character name of Service URM invoked by attached task.

CEMT INQUIRE/SET TRANSACTION

Function

Retrieve information about transactions.

Description

INQUIRE TRANSACTION returns information about transaction definitions.

Input

Press the Clear key to clear the screen. There are two ways of commencing this transaction:

- Type `CEMT INQUIRE TRANSACTION` (the minimum abbreviation is `CEMT I TRANS`). You get a display that lists the current status.
- Type `CEMT INQUIRE TRANSACTION` (`CEMT I TRANS`) followed by as many of the other attributes as are necessary to limit the range of information that you require. So, for example, if you enter `cemt i trans en pu`, the resulting display will show you the details of only those transactions that are enabled and system-purgeable.

To change various attributes, you can:

- Overtype your changes on the INQUIRE screen after tabbing to the appropriate field (see *CICS Supplied Transactions*).
- Use the CEMT SET TRANSACTION command.

CEMT INQUIRE TRANSACTION

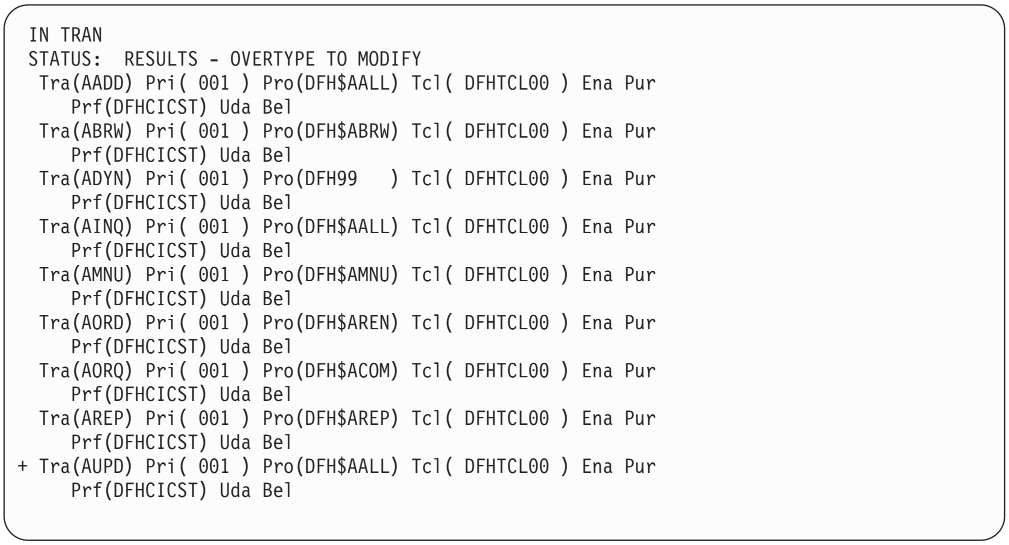
```
                              ┌─ALl─────────────┐
►►──CEMT Inquire TRAnsaction──┼─────────────────┼──┬──────────────────┬──►
                              ├─(value)─────────┤  └─PRIority(value)──┘
                              └─CLass(value)────┘

►──┬─────────────────┬──┬───────────────┬──┬──────────┬──┬──────────────┬──►
   └─PROgram(value)──┘  └─TClass(value)─┘  ├─Enabled──┤  ├─PUrgeable────┤
                                           └─Disabled─┘  └─NOTPurgeable─┘

►──┬────────────────┬──┬──────────┬──┬───────┬──┬───────────────┬──►
   └─PRFile(value)──┘  ├─CDatakey─┤  ├─ANy───┤  └─TRprof(value)─┘
                       └─Udatakey─┘  └─BElow─┘

►──┬────────────────┬──┬─────────────────────┬──────────────────────►◄
   └─BRexit(value)──┘  └─FAcilitylike(value)─┘
```

(*value***)**
is a 1–4 character transaction identifier. Only transactions that have been defined in the CICS system definition (CSD) file and installed on the running CICS system are accessible through CEMT.

ALl
is the default.

CLass**(***value***)**
is the 2-character suffix of a transaction list table (XLT).

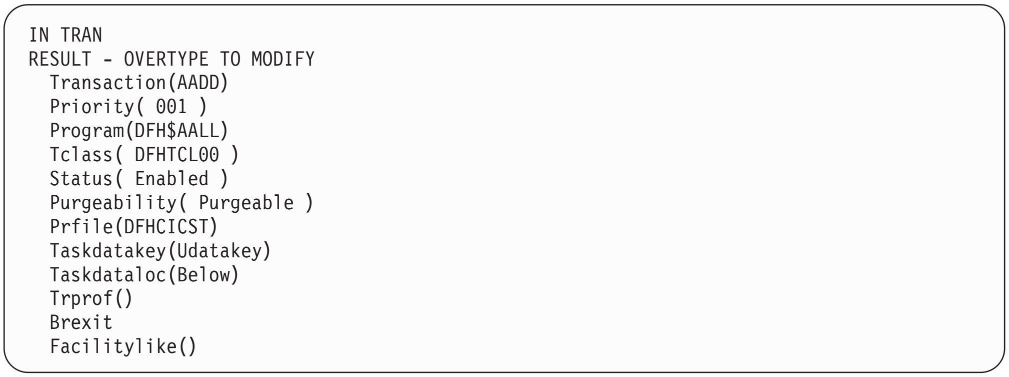
Sample screen



```
 IN TRAN
 STATUS:  RESULTS - OVERTYPE TO MODIFY
  Tra(AADD) Pri( 001 ) Pro(DFH$AALL) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
  Tra(ABRW) Pri( 001 ) Pro(DFH$ABRW) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
  Tra(ADYN) Pri( 001 ) Pro(DFH99   ) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
  Tra(AINQ) Pri( 001 ) Pro(DFH$AALL) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
  Tra(AMNU) Pri( 001 ) Pro(DFH$AMNU) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
  Tra(AORD) Pri( 001 ) Pro(DFH$AREN) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
  Tra(AORQ) Pri( 001 ) Pro(DFH$ACOM) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
  Tra(AREP) Pri( 001 ) Pro(DFH$AREP) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
+ Tra(AUPD) Pri( 001 ) Pro(DFH$AALL) Tcl( DFHTCL00 ) Ena Pur
     Prf(DFHCICST) Uda Bel
```

Figure 23. CEMT INQUIRE TRANSACTION screen

If you place the cursor against a specific entry in the list and press ENTER, CICS displays an expanded format as shown in Figure 24.



```
 IN TRAN
 RESULT - OVERTYPE TO MODIFY
   Transaction(AADD)
   Priority( 001 )
   Program(DFH$AALL)
   Tclass( DFHTCL00 )
   Status( Enabled )
   Purgeability( Purgeable )
   Prfile(DFHCICST)
   Taskdatakey(Udatakey)
   Taskdataloc(Below)
   Trprof()
   Brexit
   Facilitylike()
```

Figure 24. The expanded display of an individual entry

parameters;

BRexit**(***value***)**

returns the 8-character name of the bridge exit defined by the BREXIT parameter of the named transaction resource definition.

If BREXIT is not defined, blanks are returned.

FAcilitylike**(***value***)**

returns the 4-character name of the terminal defined by the FACILITYLIKE parameter of the PROFILE associated with the named transaction resource definition.

PRFile**(***value***)**
displays the name of the profile definition that defines additional options associated with this transaction.

PRIority**(***value***)**
displays a value indicating the priority of a transaction relative to other transactions. When a transaction is running as a CICS task, the priority of a task is the sum of the transaction priority, the terminal priority, and the operator priority.

Note: You can reset this value by overtyping it with a different value.
The value is in the range 0–255, where 255 is the highest priority.

PROgram**(***value***)**
displays an 8-character string identifying the name of the first program to be executed when this transaction is started.

Purgeability
displays whether the transaction is purgeable in system stall conditions. The values are:

PUrgeable
The transaction is system-purgeable. This value relates to the SPURGE parameter on the transaction resource definition and indicates that CICS can purge the transaction in a deadlock time-out situation. See the *CICS Resource Definition Guide* for information about the SPURGE and DTIMEOUT parameters on a transaction resource definition.

NOTPurgeable
The transaction cannot be purged.

Note: You can reset this value by overtyping it with a different value.

Status
displays whether the transaction is available for use. The values are:

Enabled
The transaction is available for use.

Disabled
The transaction is not available for use.

Note: If a transaction is disabled, this does not prevent a START command that names this transaction from being shipped to a remote region. When a task is attached for the requested transaction, CICS checks that the transaction is enabled in the remote region.

Note: You can reset this value by overtyping it with a different value.

Taskdatakey
displays the storage key in which CICS obtains all storage for use by the transaction. This includes the task life-time storage—the transaction work area (TWA) and the EXEC interface block (EIB)—and the storage that CICS obtains on behalf of programs that run under the transaction.

The values are:

CDatakey
CICS obtains storage for the transaction from CICS-key storage.

Application programs that execute in CICS key have read-write access to this storage, but user-key programs have read-only access.

Udatakey
CICS obtains storage for the transaction from user-key storage. Application programs that execute in any key have read-write access to this storage.

See the description of the TASKDATAKEY parameter on the transaction resource definition in the *CICS Resource Definition Guide*.

Taskdataloc
displays whether certain CICS control blocks (including EIB and TWA) for a transaction are acquired above or below the 16MB line. The values are:

ANy The transaction accepts task-related data anywhere.

BElow
The transaction requires any task-related data (TWA and EIB plus any internal control blocks) to be located below the 16MB line.

TClass**(***value***)**
displays an 8-character string identifying the name of the transaction class to which the transaction belongs. If the transaction does not belong to a class, DFHTCL00 is returned.

Note: You can reset this value by overtyping it with a different value.

To remove a transaction from its TCLASS, set this field to DFHTCL00. An added or changed TCLASS **must** be one that has already been defined.

TRAnsaction**(***value***)**
indicates that this panel relates to a TRANSACTION inquiry and displays a 4-character transaction identifier. Only transactions that have been defined in the CICS system definition (CSD) file and installed on the running CICS system are accessible through CEMT.

TRprof**(***value***)**
displays the name of the transaction routing profile that defines additional options associated with this transaction if it is defined as a remote transaction.

CEMT INQUIRE/SET TSQUEUE

Function

Retrieve information about temporary storage queues.

Description

The INQUIRE TSQUEUE command returns information about temporary storage queues (TS queues). The INQUIRE TSQUEUE command operates on all the temporary storage queues that exist in the CICS region, including those created internally by CICS for use by CICS itself (for example, queues used by BMS). You can identify the temporary storage queues created by CICS for its own use by queue names that begin with the following character strings:

**	BMS paging
$$	BMS route
X'fa' to X'ff'	CICS
CEBR	Default CEBR queue name
DF	CICS
DFHM	Message cache for message-protected tasks
DFxxxx	CICS REQIDS (where x is hexadecimal)

Input

Press the Clear key to clear the screen. There are two ways of commencing this transaction:

- Type `CEMT INQUIRE TSQUEUE` (the minimum abbreviation is `CEMT I TSQ`). You get a display that lists the current status.
- Type `CEMT INQUIRE TSQUEUE` (`CEMT I TSQ`) followed by as many of the other attributes as are necessary to limit the range of information that you require. So, for example, if you enter `cemt i tsq main`, the resulting display will show you the details of only those temporary storage queues that are resident in main storage.

To change various attributes, overtype your changes on the INQUIRE screen after tabbing to the appropriate field (see *CICS Supplied Transactions*).

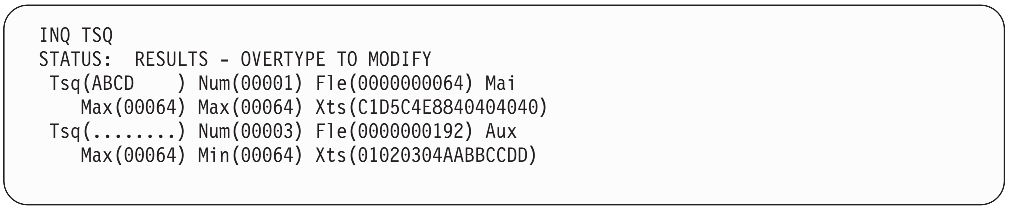
CEMT INQUIRE TSQUEUE

```
                                ┌─All─────┐
►►──CEMT Inquire TSQueue────────┼─────────┼──┬─────────────────┬──┬─────────────────┬──►
                                └─(value)─┘  └─Numitems(value)─┘  └─FLength(value)──┘

►──┬─────────────┬──┬────────────────────┬──┬───────────────────┬──┬─────────────┬──►◄
   ├─MAIn────────┤  └─MAXitemlen(value)──┘  └─MInitemlen(value)─┘  └─Xtsqueue()──┘
   └─AUxiliary───┘
```

All
    is the default.

(*value***)**
    is the name of the temporary storage queue for which information is requested.

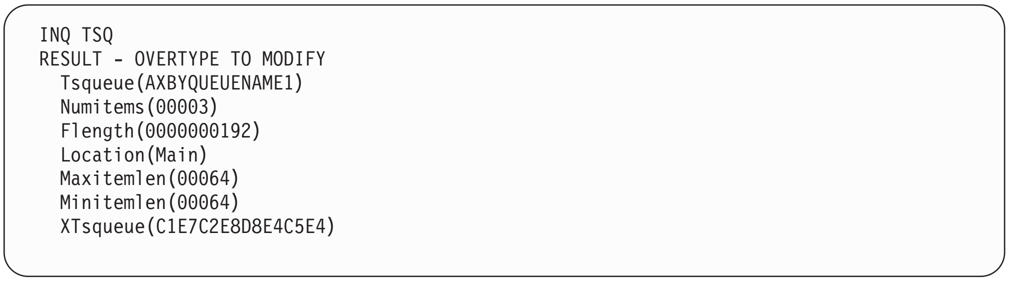
Sample screen



```
INQ TSQ
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Tsq(ABCD    ) Num(00001) Fle(0000000064) Mai
    Max(00064) Max(00064) Xts(C1D5C4E8840404040)
 Tsq(........) Num(00003) Fle(0000000192) Aux
    Max(00064) Min(00064) Xts(01020304AABBCCDD)
```

Figure 25. CEMT INQUIRE TSQUEUE screen

If you place the cursor against a specific entry in the list and press ENTER, CICS displays an expanded format as shown in Figure 26.



```
INQ TSQ
RESULT - OVERTYPE TO MODIFY
  Tsqueue(AXBYQUEUENAME1)
  Numitems(00003)
  Flength(0000000192)
  Location(Main)
  Maxitemlen(00064)
  Minitemlen(00064)
  XTsqueue(C1E7C2E8D8E4C5E4)
```

Figure 26. The expanded display of an individual entry

If you place the cursor against a specific entry in the list and type a ‘B’ (upper or lower case), CICS invokes the CEBR transaction to browse the contents of this TS queue, (see Chapter 17, “CEBR—temporary storage browse” on page 125 for details).

parameters;

Flength**(***value***)**
displays the total length in bytes of all the items in the temporary storage queue. For information about how CICS calculates the length of items, see the CICS System Programming Reference.

LOcation**(***value***)**
displays where the temporary storage queue resides. The values are:

AUxiliary
The temporary storage queue is held on the CICS temporary storage VSAM data set DFHTEMP, or in temporary storage pools in the coupling facility.

MAIn
The temporary storage queue is held in main storage.

MAXitemlen**(***value***)**
displays the length in bytes of the largest item in the temporary storage queue. For information about how CICS calculates the length of items, see the CICS System Programming Reference.

MInitemlen**(***value***)**
displays the length in bytes of the smallest item in the temporary storage queue. For information about how CICS calculates the length of items, see the CICS System Programming Reference.

Numitems**(***value***)**
displays the number of items in the temporary storage queue.

TSqueue**(***value***)**
indicates that this panel relates to a TSQUEUE inquiry and displays the 16-character name of a temporary storage queue.

Note: Nondisplayable characters appear as periods. You can use PF2 on the expanded panel to see the value in hexadecimal.

Part 8. CICS System Initialization changes

Chapter 20. CICS system initialization parameters

This chapter describes the CICS system initialization process; describes the DFHSIT macro and its parameters, and tells you how to supply initialization parameters to CICS.

Specifying system initialization parameters

The primary method of providing system initialization parameters is with a system initialization table (SIT). The parameters of the SIT, which you assemble as a load table, supply the system initialization program with most of the information necessary to initialize the system to suit your unique environment. You can generate more than one SIT, and (at the time of system initialization) select the one that is appropriate to your needs.

You can also specify other system initialization parameters, which cannot be coded in the SIT. You specify which SIT you want, and other system initialization parameters (with a few exceptions), using any of the following methods:

1. On the PARM parameter of the EXEC DFHSIP statement
2. In the SYSIPT data set defined in the startup job stream
3. Through the system operator's console

You can also use these methods to override most of the system initialization parameters assembled in the SIT.

The information defined by system initialization parameters can be grouped into three categories:

1. Information used to initialize and control CICS system functions (for example, information such as the dynamic storage area limits and the region exit time interval)
2. Module suffixes used to load the user-specified version of the CICS modules (for example, DFHDBPxx) and tables (for example, DFHJCTxx)
3. Special information used to control the initialization process

The syntax of the various DFHSIT macro parameters is listed in Table 4 on page 150. Except for those parameters marked "**SIT macro only**", all the system initialization parameters can be provided at run time, although there are restrictions in some cases. The restrictions are explained at the end of the description of the system initialization parameter to which they apply. See the CHKSTRM parameter in the CICS System Definition Guide for an example of such a restriction.

There are some other CICS system initialization parameters (and options of the parameters in Table 4 on page 150) that you cannot define in the DFHSIT macro. (See 179.) The parameters that you cannot define in the DFHSIT macro are shown in "Parameters that you cannot code in the DFHSIT macro" on page 159.

System initialization keywords grouped by function

For a list of all the system initialization keywords grouped by their functional area, see CICS Resource Definition Guide. This ensures that you do not miss coding an important parameter relating to a particular CICS function. For details of how to code a parameter, you still have to refer to the parameter descriptions that are listed alphabetically in this chapter.

Migration considerations

If you have existing system initialization tables, you must modify them. Remove all obsolete parameters, and specify the required values for new or changed parameters if you want to run with other than the defaults. When you have made the necessary changes, reassemble the tables using the CICS Transaction Server for VSE/ESA macro libraries.

If you have system initialization parameters defined in CICS start-up procedures, you must modify these also.

To avoid generating specific system initialization tables for each CICS system, a simple solution is to let CICS load the default, unsuffixed table (DFHSIT) at start-up, and supply the system initialization parameters for each CICS system in a SYSIPT data set. For more information about the source of the default system initialization table, see “DFHSIT, the default system initialization table” on page 153.

The DFHSIT macro parameters

Table 4. The DFHSIT macro parameters

<table>
<tr><td></td><td>DFHSIT</td><td>[TYPE={<b>CSECT</b>|DSECT}]
[,ADI={<b>30</b>|number}]
[,AIEXIT={<b>DFHZATDX</b>|DFHZATDY|name}]
[,AILDELAY={<b>0</b>|hhmmss}]
[,AIQMAX={<b>100</b>|number}]
[,AIRDELAY={<b>700</b>|hhmmss}]
[,AKPFREQ={<b>200</b>|number}]
[,APPLID=({<b>DBDCCICS</b>|name1}[,name2])]
[,AUTCONN={<b>0</b>|hhmmss}]
[,AUXTR={<b>OFF</b>|ON}]
[,AUXTRSW={<b>NO</b>|ALL|NEXT}]
[,BMS=({MINIMUM|STANDARD|<b>FULL</b>}[,COLD][
        [,{<b>UNALIGN</b>|ALIGN}][,{<b>DDS</b>|NODDS}])]
[,CLSDSTP={<b>NOTIFY</b>|NONOTIFY}]
[,CLT=xx]
[,CMDPROT={<b>YES</b>|NO}]
[,CMDSEC={<b>ASIS</b>|ALWAYS}
[,CONFDATA={<b>SHOW</b>|HIDETC}]
[,CONFTXT={<b>NO</b>|YES}]
[,CSDACC={<b>READWRITE</b>|READONLY}]
[,CSDBKUP={<b>STATIC</b>|DYNAMIC}]
[,CSDBUFND=number]
[,CSDBUFNI=number]
[,CSDFRLOG=number]
[,CSDJID={<b>NO</b>|number}]
[,CSDLSRNO={<b>1</b>|number|NONE|NO}]
[,CSDRECOV={<b>NONE</b>|ALL|BACKOUTONLY}]
[,CSDSTRNO={<b>2</b>|number}]
[,CWAKEY={<b>USER</b>|CICS}]
[,DATFORM={<b>MMDDYY</b>|DDMMYY|YYMMDD}]
[,DBP={1$|2$|xx|YES}] (Must specify in the SIT macro)
[,DBUFSZ=({<b>500</b>|number})]
[,DCT=({YES|xx|<b>NO</b>})]
[,DFLTUSER={<b>CICSUSER</b>|userid}]
[,DIP={<b>NO</b>|YES}]
[,DISMACP={<b>YES</b>|NO}]</td></tr>
</table>

Table 4. The DFHSIT macro parameters (continued)

<table>
<tr><td></td><td></td><td>[{,DLI|DL1}=({<b>NO</b>|YES|REMOTE},[COLD])]
[,DLIOER=({<b>ABEND</b>|CONTINUE})]
[,DOCCODEPAGE={<b>037</b>|codepage}]
[,DSALIM={<b>5M</b>|number}]
[,DSHIPIDL={<b>020000</b>|hhmmss}]
[,DSHIPINT={<b>120000</b>|hhmmss}]
[,DTRPGM={<b>DFHDYP</b>|program-name}]
[,DTRTRAN={<b>CRTX</b>|name|NO}]
[,DUMP={ <b>YES</b>|NO}]
[,DUMPDS={<b>AUTO</b>|A|B}]
[,DUMPSW={<b>NO</b>|NEXT}]
[,EDSALIM={<b>20M</b>|number}]
[,ENCRYPTION={WEAK|<b>NORMAL</b>|STRONG}]
[,EODI={<b>E0</b>|xx}]
[,ESMEXITS={<b>NOINSTLN</b>|INSTLN}] <b><i>(SIT macro only)</i></b>
[,FCT={<b>YES</b>|xx|NO}]
[,FEPI={<b>NO</b>|YES}]
[,FLDSEP={<b>‘   ’</b>|‘xxxx’}]
[,FLDSTRT={<b>‘ ’</b>|‘x’}]
[,FSSTAFF={YES|<b>NO</b>}]
[,FTIMEOUT={<b>30</b>|nn}]
[,GMTEXT={<b>‘WELCOME TO CICS’</b>|‘text’}]
[,GMTRAN={<b>CSGM</b>|CESN|name}]
[,GNTRAN={<b>CESF</b>|transaction-id}]
[,GRPLIST={<b>DFHLIST</b>|name|(name[,name2][,name3][,name4])}]
[,ICP=COLD]
[,ICV={<b>1000</b>|number}]
[,ICVR={<b>5000</b>|number}]
[,ICVTSD={<b>500</b>|number}]
[,INITPARM=(pgmname_1=‘parmstring_1’
        [, ... ,pgmname_n=‘parmstring_n’])]
[,INTTR={<b>ON</b>|OFF}]
[,IRCSTRT={<b>NO</b>|YES}]
[,ISC={<b>NO</b>|YES}]
[,JCT={<b>YES</b>|xx|NO}]
[,KEYFILE=name]
[,LEVSE={<b>YES</b>|NO}]
[,LGNMSG={<b>NO</b>|YES}]
[,MCT={<b>NO</b>|YES|xx}]
[,MN={<b>OFF</b>|ON}]
[,MNCONV={<b>NO</b>|YES}]
[,MNEXC={<b>OFF</b>|ON}]
[,MNFREQ={<b>0</b>|hhmmss}]
[,MNPER={<b>OFF</b>|ON}]
[,MNSYNC={<b>NO</b>|YES}]
[,MNTIME={<b>GMT</b>|LOCAL}]
[,MROBTCH={<b>1</b>|number}]
[,MROFSE={ <b>NO</b>|YES}]
[,MROLRM={<b>NO</b>|YES}]
[,MSGCASE={<b>MIXED</b>|UPPER}]
[,MSGLVL={<b>1</b>|0}]
[,MXT={<b>5</b>|number}]
[,NATLANG=(<b>E</b>,x,y,z,...)]
[,OPERTIM={<b>120</b>|number}]</td></tr>
</table>

Table 4. The DFHSIT macro parameters (continued)

<table>
<tr><td></td><td></td><td>[,PARMERR={<b>INTERACT</b>|IGNORE|ABEND}]
[,PDI={<b>30</b>|number}]
[,PGAICTLG={<b>MODIFY</b>|NONE|ALL}]
[,PGAIEXIT={<b>DFHPGADX</b>|name}]
[,PGAIPGM={<b>INACTIVE</b>|ACTIVE}]
[,PGCHAIN=character(s)]
[,PGCOPY=character(s)]
[,PGPURGE=character(s)]
[,PGRET=character(s)]
[,PLTPI={<b>NO</b>|xx|YES}]
[,PLTPISEC={<b>NONE</b>|CMDSEC|RESSEC|ALL}]
[,PLTPIUSR=userid]
[,PLTSD={<b>NO</b>|xx|YES}]
[,PRGDLAY={<b>0</b>|hhmm}]
[,PRINT={<b>NO</b>|YES|PA1|PA2|PA3}]
[,PRTYAGE={<b>32768</b>|number}]
[,PSDINT={<b>0</b>|hhmmss}]
[,PVDELAY={<b>30</b>|number}]
[,RAMAX={<b>256</b>|number}]
[,RAPOOL={ <b>50</b>|value1|(value1,value2)}]
[,RENTPGM={<b>PROTECT</b>|NOPROTECT}]
[,RESP={<b>FME</b>|RRN}]
[,RESSEC={<b>ASIS</b>|ALWAYS}
[,RMTRAN=({<b>CSGM</b>|name1}
[,{<b>CSGM</b>|name2}])]
[,RUWAPOOL={<b>NO</b>|YES}]
[,SEC={<b>YES</b>|NO}]
[,SECPRFX={<b>NO</b>|YES}]
[,SKRxxxx=‘page-retrieval-command’]
[,SNSCOPE={<b>NONE</b>|CICS|VSEIMAGE}]
[,SPCTR={(<b>1,2</b>|1[,2][,3])|ALL|OFF}]
[,SPOOL={<b>NO</b>|YES}]
[,SRT={<b>YES</b>|xx|NO}]
[,SSLDELAY={<b>600</b>|number}]
[,START={<b>AUTO</b>|COLD|STANDBY}]
[,STARTER={<b>NO</b>|YES}] <b><i>(SIT macro only)</i></b>
[,STATRCD={<b>OFF</b>|ON}]
[,STGPROT={<b>NO</b>|YES}]
[,STGRCVY={<b>NO</b>|YES}]
[,STNTR={<b>1</b>|(1[,2][,3])|ALL|OFF}]
[,SUFFIX=xx] <b><i>(SIT macro only)</i></b>
[,SVA={<b>NO</b>|YES[}]
[,SYDUMAX={<b>999</b>|number}]
[,SYSIDNT={<b>CICS</b>|name}]
[,SYSTR={<b>ON</b>|OFF}]
[,TAKEOVR={<b>MANUAL</b>|AUTO|COMMAND}]
[,TBEXITS=([nm1][,nm2][,nm3]
[,nm4][,nm5])]
[,TCP={<b>YES</b>|NO}]
[,TCPIP={<b>NO</b>|YES}]
[,TCSACTN={<b>NONE</b>|UNBIND]FORCE}]
[,TCSWAIT={<b>4</b>|number|NO|NONE|0}]
[,TCT={<b>YES</b>|xx|NO}]
[,TCTUAKEY={<b>USER</b>|CICS}]</td></tr>
</table>

Table 4. The DFHSIT macro parameters (continued)

<table>
<tr><td></td><td></td><td>[,TCTUALOC={**BELOW**|ANY}]
[,TD=({**3**|number1}[,{**3**|number2}])]
[,TRAP={**OFF**|ON}]
[,TRDUMAX={**999**|number}]
[,TRTABSZ={**16**|number}]
[,TRTRANSZ={**16**|number}]
[,TRTRANTY={**TRAN**|ALL}]
[,TS=([COLD][,{0|**3**|value-1}][,{**3**|value-2}])]
[,TSMSGSET={**4**|number}]
[,TST={**NO**|YES|xx}]
[,USERTR={**ON**|OFF}]
[,USRDELAY={**30**|number}]
[,VTAM={**YES**|NO}]
[,VTPREFIX={****|character}]
[,WEBDELAY={**5**|time_out,**60**|keep_time}]
[,WRKAREA={**512**|number}]
[,XAPPC={**NO**|YES}]
[,XCMD={**NO**|name|YES}]
[,XDCT={**NO**|name|YES}]
[,XFCT={**NO**|name|YES}]
[,XJCT={**NO**|name|YES}]
[,XLT={**NO**|xx|YES}]
[,XPCT={**NO**|name|YES}]
[,XPPT={**NO**|name|YES}]
[,XPSB={**NO**|name|YES}]
[,XRF={**NO**|YES}]
[,XRFSOFF={**NOFORCE**|FORCE}]
[,XRFSTME={**5**|number}]
[,XRFTODI={**30**|number}]
[,XSWITCH=([{**0**|1–254}][{,**????????**|progname}][{,**A**|B}])]
[,XTRAN={**YES**|name|NO}]
[,XTST={**NO**|name|YES}]
[,XUSER={**YES**|NO}]

You must terminate your macro parameters with the following END statement.</td></tr>
<tr><td></td><td>END</td><td>DFHSITBA</td></tr>
</table>

DFHSIT, the default system initialization table

The macro source statements used to assemble the default system initialization table are given in Figure 27 on page 154.

The VSE/ESA default SIT members (DFHSITSP.PHASE and DFHSITSP.A) are located in IJSYSRS.SYSLIB. Their source (DFHSIT$$) is located in PRD1.BASE.

```
*                                                                                *
* SIT parameters (in alphabetical order)                                         *
*                                                                                *
SIT      TITLE 'DFHSIT - CICS DEFAULT SYSTEM INITIALIZATION TABLE'
         DFHSIT TYPE=CSECT,                                                      *
               ADI=30,                     XRF(B) - Alternate delay interval     *
               AIEXIT=DFHZATDX,            Autoinstall user program name         *
               AILDELAY=0,                 Delete delay period for AI terminals  *
               AIQMAX=100,                 Maximum no. of terminals queued for AI*
               AIRDELAY=700,               Restart delay period for AI terminals *
               AKPFREQ=200,                Activity keypoint frequency           *
               APPLID=DBDCCICS,            VTAM APPL identifier                  *
               AUTCONN=0,                  Autoconnect delay                     *
               AUXTR=OFF,                  Auxiliary trace option                *
               AUXTRSW=NO,                 Auxiliary trace autoswitch facility   *
               BMS=(FULL,,UNALIGN,DDS),    Basic Mapping Support options         *
               CLSDSTP=NOTIFY,             Notification for ISSUE PASS command   *
               CLT=,                       The command list table option/suffix  *
               CMDPROT=YES,                Exec storage command checking         *
               CMDSEC=ASIS,                API command security checking         *
               CONFDATA=SHOW,              Show confidential data in dump/trace  *
               CONFTXT=NO,                 Don't prevent VTAM tracing user data  *
               CSDACC=READWRITE,           CSD access                            *
               CSDBUFND=,                  Number of data buffers for the CSD    *
               CSDBUFNI=,                  Number of index buffers for the CSD   *
               CSDFRLOG=NO,                Journal id. for CSD forward recovery  *
               CSDJID=NO,                  Journal id. for CSD auto. journaling  *
               CSDLSRNO=1,                 The VSAM LSR pool number for the CSD  *
               CSDRECOV=NONE,              CSD recoverable file option           *
               CSDSTRNO=2,                 CSD Number of strings                 *
               CWAKEY=USER,                CWA storage key                       *
               DATFORM=MMDDYY,             CSA date format                       *
               DBP=1$,                     REquired version of DBP with DLI=NO   *
               DBUFSZ=500,                 Dynamic backout buffer size           *
               DCT=YES,                    Dest. control table option/suffix     *
               DFLTUSER=CICSUSER,          Default user                          *
               DIP=NO,                     Batch data interchange program        *
               DISMACP=YES,                Disable macro programs                *
               DLI=NO,                     DL/I option                           *
               DOCCODEPAGE=037             Default host code page                *
               DSALIM=5M,                  Upper limit of DSA below 16Mb line    *
               DSHIPIDL=020000,            Delete shipped idle time              *
               DSHIPINT=120000,            Delete shipped interval               *
               DTRPGM=DFHDYP,              Dynamic routing program               *
               DTRTRAN=CRTX,               Default dynamic tran routing transid  *
               DUMP=YES,                   Dump option                           *
               DUMPDS=AUTO,                CICS dump data set opening option     *
               DUMPSW=NO,                  Dump data set autoswitch option       *
               EDSALIM=20M,                Upper limit of DSA above 16MB line    *
               ENCRYPTION=NORMAL,          SSL Encryption                        *
               EODI=E0,                    End-of-data indicator for seq. devices*
               ESMEXITS=NOINSTLN,          External security manager exits       *
               FCT=YES,                    File control table option/suffix      *
               FEPI=NO,                    Front-End Programming Interface       *
               FLDSEP='    ',              End-of-field separator characters     *
               FLDSTRT=' ',                Field start character for builtin fn  *
               FSSTAFF=NO,                 Function-shipped START affinity option*
               GMTEXT='WELCOME TO CICS',   Good-morning message text             *
               GMTRAN=CSGM,                Initial transaction                   *
               GNTRAN=NO,                  Signoff transaction                   *
```

Figure 27. DFHSIT, the pregenerated CICS default system initialization table 1/4

```
GRPLIST=DFHLIST,            List name of CSD groups for startup   *
ICP=,                       Interval control pgm. start option    *
ICV=1000,                   Region exit interval (milliseconds)   *
ICVR=5000,                  Runaway task interval (milliseconds)  *
ICVTSD=500,                 Terminal scan delay interval (  "  )  *
INITPARM=,                  Initialization parms for programs     *
INTTR=ON,                   CICS internal trace option            *
IRCSTRT=NO,                 Interregion communication start       *
ISC=NO,                     Intersystem communication option      *
JCT=NO,                     No journalling                        *
KEYFILE=CRYPTO.KEYRING,     Key residence for SSL                 *
LEVSE=YES,                  Support for LE                        *
LGNMSG=NO,                  Extract VTAM logon data               *
MCT=NO,                     Monitoring cntl.table option/suffix   *
MN=OFF,                     CICS monitoring option                *
MNCONV=NO,                  Monitoring converse recording option  *
MNEXC=OFF,                  Monitoring exception class option     *
MNFREQ=0,                   Monitoring frequency period           *
MNPER=OFF,                  Monitoring performance class option   *
MNSYNC=NO,                  Monitoring syncpoint recording option *
MNTIME=GMT,                 Monitoring timestamp (GMT/LOCAL)      *
MROBTCH=1,                  Number of MRO requests to batch       *
MROFSE=NO,                  Extend lifetime of Long-running mirror*
MROLRM=NO,                  Long-running mirror task option       *
MSGCASE=MIXED,              CICS messages in mixed case           *
MSGLVL=1,                   System console MSG level option       *
MXT=5,                      Maximum number of tasks in CICS       *
NATLANG=E,                  List of national languages            *
OPERTIM=120,                Write to operator timeout (seconds)   *
PARMERR=INTERACT,           System init. parameter errors option  *
PDI=30,                     Primary delay interval - XRF active   *
PGAICTLG=MODIFY,            PG autoinstall catalog state          *
PGAIEXIT=DFHPGADX,          PG autoinstall exit program           *
PGAIPGM=INACTIVE,           PG autoinstall state                  *
PGCHAIN=,                   BMS CHAIN command                     *
PGCOPY=,                    BMS COPY command                      *
PGPURGE=,                   BMS PURGE command                     *
PGRET=,                     BMS RETURN command                    *
PLTPI=NO,                   Program list table PI option/suffix   *
PLTPISEC=NONE,              No PLT security checks on PI programs *
PLTPIUSR=,                  PLT PI userid = CICS region userid    *
PLTSD=NO,                   Program list table SD option/suffix   *
PRGDLAY=0,                  BMS purge delay interval              *
PRINT=NO,                   Print key option                      *
PRTYAGE=32768,              Dispatcher priority ageing value      *
PSDINT=0,                   Persistent session delay interval     *
PVDELAY=30,                 Timeout value for LUIT table          *
RAMAX=256,                  Max. I/O area for RECEIVE ANY         *
RAPOOL=50,                  Max. RECEIVE ANY request parm.lists   *
RENTPGM=PROTECT,            Reentrant program write protection    *
RESP=FME,                   Logical unit response type            *
RESSEC=ASIS,                Resource security check               *
RMTRAN=CSGM,                XRF alternate recovery transaction    *
RST=NO,                     Recovery service table (XRF-DBCTL)    *
RUWAPOOL=NO,                Allocating storage pool for LE        *
SEC=YES,                    External security manager option      *
SECPRFX=NO,                 Security prefix                       *
SKRPA1=,                    SKR PA1 PAGE RETRIEVAL CMD            *
SKRPA2=,                    SKR PA2 PAGE RETRIEVAL CMD            *
SKRPA3=,                    SKR PA3 PAGE RETRIEVAL CMD            *
SKRPF1=,                    SKR PF1 PAGE RETRIEVAL CMD            *
SKRPF2=,                    SKR PF2 PAGE RETRIEVAL CMD            *
```

Figure 28. DFHSIT, the pregenerated CICS default system initialization table 2/4

```
SKRPF3=,                    SKR PF3 PAGE RETRIEVAL CMD            *
SKRPF4=,                    SKR PF4 PAGE RETRIEVAL CMD            *
SKRPF5=,                    SKR PF5 PAGE RETRIEVAL CMD            *
SKRPF6=,                    SKR PF6 PAGE RETRIEVAL CMD            *
SKRPF7=,                    SKR PF7 PAGE RETRIEVAL CMD            *
SKRPF8=,                    SKR PF8 PAGE RETRIEVAL CMD            *
SKRPF9=,                    SKR PF9 PAGE RETRIEVAL CMD            *
SKRPF10=,                   SKR PF10 PAGE RETRIEVAL CMD           *
SKRPF11=,                   SKR PF11 PAGE RETRIEVAL CMD           *
SKRPF12=,                   SKR PF12 PAGE RETRIEVAL CMD           *
SKRPF13=,                   SKR PF13 PAGE RETRIEVAL CMD           *
SKRPF14=,                   SKR PF14 PAGE RETRIEVAL CMD           *
SKRPF15=,                   SKR PF15 PAGE RETRIEVAL CMD           *
SKRPF16=,                   SKR PF16 PAGE RETRIEVAL CMD           *
SKRPF17=,                   SKR PF17 PAGE RETRIEVAL CMD           *
SKRPF18=,                   SKR PF18 PAGE RETRIEVAL CMD           *
SKRPF19=,                   SKR PF19 PAGE RETRIEVAL CMD           *
SKRPF20=,                   SKR PF20 PAGE RETRIEVAL CMD           *
SKRPF21=,                   SKR PF21 PAGE RETRIEVAL CMD           *
SKRPF22=,                   SKR PF22 PAGE RETRIEVAL CMD           *
SKRPF23=,                   SKR PF23 PAGE RETRIEVAL CMD           *
SKRPF24=,                   SKR PF24 PAGE RETRIEVAL CMD           *
SKRPF25=,                   SKR PF25 PAGE RETRIEVAL CMD           *
SKRPF26=,                   SKR PF26 PAGE RETRIEVAL CMD           *
SKRPF27=,                   SKR PF27 PAGE RETRIEVAL CMD           *
SKRPF28=,                   SKR PF28 PAGE RETRIEVAL CMD           *
SKRPF29=,                   SKR PF29 PAGE RETRIEVAL CMD           *
SKRPF30=,                   SKR PF30 PAGE RETRIEVAL CMD           *
SKRPF31=,                   SKR PF31 PAGE RETRIEVAL CMD           *
SKRPF32=,                   SKR PF32 PAGE RETRIEVAL CMD           *
SKRPF33=,                   SKR PF33 PAGE RETRIEVAL CMD           *
SKRPF34=,                   SKR PF34 PAGE RETRIEVAL CMD           *
SKRPF35=,                   SKR PF35 PAGE RETRIEVAL CMD           *
SKRPF36=,                   SKR PF36 PAGE RETRIEVAL CMD           *
SNSCOPE=NONE,               Multiple CICS sessions per userid     *
SPCTR=(1,2),                Level(s) of special tracing required  *
SPOOL=NO,                   System spooling interface option      *
SSLDELAY=600,               Delay for SSL connection              *
SRT=YES,                    System recovery table option/suffix   *
START=AUTO,                 CICS system initialization option     *
STARTER=YES,                Starter ($ and #) suffixes option     *
STATRCD=OFF,                statistics recording status           *
STGPROT=NO,                 Storage protection facility           *
STGRCVY=NO,                 Storage recovery option               *
STNTR=1,                    Level of standard tracing required    *
SUFFIX=$$,                  Suffix of this SIT                    *
SVA=NO,                     Use SVA option for CICS/user modules  *
SYDUMAX=999,                No of SYSDUMPS to be taken            *
SYSIDNT=CICS,               Local system identifier               *
SYSTR=ON,                   Master system trace flag              *
TAKEOVR=MANUAL,             XRF alternate takeover option         *
TBEXITS=,                   Backout exit programs                 *
TCP=YES,                    Terminal control program option/suffix*
TCPIP=NO,                   TCP/IP support                        *
TCSACTN=NONE,               TC Shutdown action                    *
TCSWAIT=4,                  TC Shutdown wait                      *
TCT=YES,                    Terminal control table option/suffix  *
TCTUAKEY=USER,              TCT user area storage key             *
TCTUALOC=BELOW,             TCT user area below 16MB              *
TD=(3,3),                   Transient data buffers and strings    *
TRAP=OFF,                   F.E. global trap exit option          *
TRDUMAX=999,                No of TRANDUMPS to be taken           *
TRTABSZ=16,                 Internal trace table size in 1K bytes *
```

Figure 29. DFHSIT, the pregenerated CICS default system initialization table 3/4

```
      TRTRANSZ=40,                Transaction dump trace table size     *
      TRTRANTY=TRAN,              Transaction dump trace option         *
      TS=(,3,3),                  Temporary storage buffers and strings *
      TSMSGSET=4,                 # of entries for pointers to TS MSGset*
      TST=NO,                     Temporary storage table option/suffix *
      USERTR=ON,                  Master user trace flag                *
      USRDELAY=30,                Timeout value for User Dir. Entries   *
      VTAM=YES,                   VTAM access method option             *
      VTPREFIX=\,                 Client virtual terminal prefix        *
      WEBDELAY=(5,60),            Web timer values                      *
      WRKAREA=512,                Common work area (CWA) size in bytes  *
      WRKAREA=512,                Common work area (CWA) size in bytes  *
      XAPPC=NO,                   RACF class APPCLU required            *
      XCMD=NO,                    SPI use default name for RACF check   *
      XDCT=NO,                    DCT use default name for RACF check   *
      XFCT=NO,                    FCT use default name for RACF check   *
      XJCT=NO,                    JCT use default name for RACF check   *
      XLT=NO,                     Transaction list table option/suffix  *
      XPCT=NO,                    PCT use default name for RACF check   *
      XPPT=NO,                    PPT use default name for RACF check   *
      XPSB=NO,                    PSB use default name for RACF check   *
      XRF=NO,                     Extended recovery feature (XRF) option*
      XRFSOFF=NOFORCE,            XRF -  Re-sign on after takeover      *
      XRFSTME=5,                  XRF -  sign off timeout value         *
      XRFTODI=30,                 XRF takeover delay interval           *
      XSWITCH=(0,????????,A),     Pgrmable terminal switching unit      *
      XTRAN=YES,                  Transid use default name, RACF check  *
      XTST=NO,                    TST use default name for RACF check   *
      XUSER=NO,                   Surrogate user checking to be done
END   DFHSITBA
```

Figure 30. DFHSIT, the pregenerated CICS default system initialization table 4/4

Creating a system initialization table

Decide which parameters you want to use. List them, together with the appropriate setting, in a file named DFHSIT*xx*. The suffix *xx* can be any two characters that you want (apart from the characters reserved solely for use by CICS), and identifies the file as your own SIT. If you do not specify a two-character suffix, CICS uses the default system initialization table, DFHSIT1$.

See CICS Resource Definition Guide which summarizes the groups of system initialization parameters needed for different CICS functions.

If you specify more than one value for a system initialization parameter, separate the operand values by commas and enclose them all in parentheses. If you do not specify a parameter as a system initialization override, the system uses the value supplied in whichever SIT you specify.

End your system initialization table with an

```
END DFHSITBA
```

statement.

“The system initialization parameter descriptions” on page 176 describes all the system initialization parameters on an individual basis.

There are some parameters that cannot be coded in the DFHSIT macro. These are summarized in “Parameters that you cannot code in the DFHSIT macro” on page 159.

Coding more than one system initialization table

You can code several SITs, each reflecting different parameter combinations and settings, as long as you remember to give each DFHSIT*xx* a different two-character suffix. CICS uses only one of these tables during initialization. Choose the SIT that best suits your needs at the time, and code its suffix on the SIT system initialization parameter.

Assemble your chosen SIT and link-edit it into one of the sublibraries defined in the LIBDEF statement for your CICS startup job stream.

CICS-supplied system initialization tables

The VSE/ESA sublibrary, PRD1.BASE, contains the sample system initialization tables, DFHSIT6$ and DFHSIT$$. DFHSIT6$ contains all the SIT values needed for a standard CICS system. DFHSIT$$ contains all the default values for the system initialization parameters.

System initialization tables supplied with VSE/ESA

VSE/ESA supplies two sample system initialization tables. DFHSITSP is provided for a standard CICS system, and DFHSITC2 is provided for a second predefined CICS system. There are listings for both these tables in the *VSE/ESA Planning* manual.

Assembling the system initialization table

When you have defined the parameters you need, add your system initialization table to your working CICS system by assembling the DFHSIT macro. The macro assembly process produces the linkage-editor control statements needed to link-edit your table into your CICS (or private) sublibrary.

When you have assembled your SIT, make it known to CICS using one of the following methods:

- Including your sublibrary in the LIBDEF statement of the CICS startup job stream.
- Coding the suffix of your system initialization table on the SIT system initialization override parameter.
- Including the suffix of chosen system initialization table on the PARM parameter of the EXEC DFHSIP statement. Figure 31 on page 164 gives an example of using the PARM parameter to specify a SIT with the suffix 1&#163;.

The samples DFHSITSP and DFHSITC2 are located in ICCF Library 59.

For information about assembling and link-editing CICS control tables, see CICS Resource Definition Guide.

Examples of the syntax notation for describing CICS macros are given in the CICS Resource Definition Guide.

Assembler errors from duplicated or undefined keywords

If you duplicate a keyword when using VSE/ESA 2.4 and the High Level Assembler, the following error messages are produced:

```
ASMA018S *** ERROR *** DUPLICATE KEYWORD IN MACRO CALL;
                       LAST VALUE IS USED - DFHSI/keyword
```

If you use an undefined keyword, the following error message is produced:

```
ASMA017W ** WARNING ** UNDEFINED KEYWORD PARAMETER; DEFAULT TO POSITIONAL,
INCLUDING KEYWORD - DFHSI/keyword
```

However, be aware that because of work space limitations, there is a limit to the number of undefined keyword errors that the assembler can generate. This means that if your SIT contains more undefined keywords than the assembler can generate messages for, the excess errors will not be flagged until a second or even later assembly and, as you correct the currently flagged errors, other (previously unflagged) errors may appear during re-assembly.

Parameters that you cannot code in the DFHSIT macro

There are some CICS system initialization parameters that you cannot define in the DFHSIT macro. These are:

- CDSASZE={0K|number}
- CHKSTRM={CURRENT|NONE}
- CHKSTSK={ALL|CURRENT|NONE}
- ECDSASZE={0K|number}
- ERDSASZE={0K|number}
- ESDSASZE={0K|number}
- EUDSASZE={0K|number}
- JSTATUS=RESET
- NEWSIT={YES|NO}
- PRVMOD={name|(name,name...name)}
- RDSASZE={0K|number}
- SDSASZE={0K|number}
- SIT=xx
- SPCTRxx={(1[,2][,3])|ALL|OFF}
- START=LOGTERM
- START=(option,ALL)
- STNTRxx={(1[,2][,3])|ALL|OFF}
- UDSASZE={0K|number}

These parameters can only be supplied at CICS startup time using any of the following methods:

1. The PARM parameter of the EXEC DFHSIP statement
2. The SYSIPT data set defined in the startup job stream
3. Through the system operator’s console.

For information about coding system initialization parameters in PARM, SYSIPT or at the console, see “System initialization control keywords” on page 161.

Overriding SIT parameters at system startup

There may be times when you need to change some of the parameters specified in your SIT, but don’t want to have to modify your SIT and reassemble it; or you may need to supply parameters that cannot be coded in the DFHSIT macro (as listed in “Parameters that you cannot code in the DFHSIT macro” on page 159). To do this you supply system initialization parameters as **overrides** using the:

- PARM parameter on the EXEC DFHSIP statement
- SYSIPT data set
- Operator console.

You can use one, or a combination of the above methods. However, parameter manager domain processes these three sources in the following strict sequence:

1. The PARM parameter
2. The SYSIPT data set (but only if SYSIPT is coded in the PARM parameter
3. The console (but only if CONSOLE is coded in either the PARM parameter or in the SYSIPT data set.

Note: If you supply duplicate system initialization parameters, either through the same or a different medium, CICS takes the **last** one that it reads. For example, if you specify DCT=1&#163; in the PARM parameter, DCT=2&#163; in the SYSIPT data set, and finally enter DCT=3&#163; through the console, CICS loads DFHDCT3&#163;.

You complete the override process with the &#163;END statement.

During startup, CICS uses your chosen system initialization table and any system initialization parameter overrides you have specified to determine the CICS functions you require.

The CICS parameter manager domain

In addition to loading the system initialization table at the start of initialization, and reading any other parameters from PARM, SYSIPT, or the system console, the parameter manager domain is responsible for the management of the SIT. With the exception of the application domain (AP) which uses the SIT directly, the parameter manager domain passes system initialization parameters to the other CICS domains on request. The domain initialization process is as follows:

Query the type of startup
With the exception of the trace domain, each domain asks the parameter manager for the type of startup, cold or warm. (For this purpose, the parameter manager domain treats an emergency restart as a warm start.)

Startup is cold
If startup is cold, domains do not read their domain records from the catalogs. Instead, they request system initialization parameters from the parameter manager domain. Because it is a cold start, the parameter manager domain returns all system initialization parameters from the SIT, modified by any overrides.

Startup is warm
If startup is warm, domains try to perform a warm start by reading their domain records from the catalogs:

- If they succeed in reading their status records, domains perform a warm start. Where applicable, they also request system initialization parameters from the parameter manager domain. Because it is a warm start, the parameter manager domain returns only those system initialization parameters supplied as overrides via PARM, SYSIPT, or the system console.
- If they fail for any reason to read their status records, domains perform a cold start. They do this either by requesting ***all*** system initialization parameters from the parameter manager domain, or by using system default values if the domain does not have any system initialization parameters.

NEWSIT or new suffix
Although a START=AUTO may resolve to a warm start, parameter manager enforces most system initialization parameters if:

- You specify NEWSIT=YES as a system initialization parameter in PARM, SYSIPT, or through the console.

- You specify a different SIT suffix from the previous run of CICS. Parameter manager saves the suffix from each run in the global catalog, and, if it detects a new suffix, it forces the NEWSIT=YES option.

For details of the parameters that are ignored when you specify NEWSIT=YES, see the NEWSIT parameter description on page 209.

Note: The trace domain is an exception to the above rules in that it always cold starts. Trace does not save its status at CICS shutdown like the other domains, and regardless of the type of startup, it requests ***all*** of its system initialization parameters from the parameter manager domain.

System initialization control keywords

There are three special control keywords that you can use at start up to control how CICS reads in system initialization parameters. These are:

- SYSIN (abbreviated to SI)
- CONSOLE (abbreviated to CN)
- .END

The SYSIN (SI) keyword

Code the SYSIN keyword to tell CICS to read system initialization parameters from the SYSIPT data set. You can code SYSIN (or SI) only on the PARM parameter of the EXEC DFHSIP statement. The keyword can appear once only and must be at the end of the PARM parameter. CICS does not read the SYSIPT data stream until it has finished scanning all of the PARM parameter, or until it reaches an &#163;END keyword before the end of the PARM parameter. An example of coding the SYSIN keyword is:

```
// EXEC DFHSIP,SIZE=DFHSIP,PARM='SI',OS390
```

The CONSOLE (CN) keyword

Use the CONSOLE (CN) keyword to tell CICS to read initialization parameters from the console. CICS prompts you with message DFHPA1104 when it is ready to read parameters from the console.

You can code the CONSOLE (or CN) keyword in the PARM parameter of the EXEC DFHSIP statement, or in the SYSIPT data stream. The keyword can appear anywhere in the PARM parameter or SYSIPT data stream, but you must code it in one place only.

If you code CONSOLE on the PARM parameter and the PARM parameter also contains the SYSIN keyword CICS does not begin reading parameters from the console until it has finished reading and processing the SYSIPT data stream. Similarly, if you code CONSOLE in the SYSIPT data stream, CICS does not begin reading parameters from the console until it has finished reading and processing the SYSIPT data stream.

In general, after a given CICS system has become the production system, you should perform startup with as little operator intervention as possible (that is, use PARM or SYSIPT, and ***not*** the console). Better still, use the SIT only, without any overrides. An example of coding the CONSOLE keyword is as follows:

```
// EXEC DFHSIP,SIZE=DFHSIP,PARM='CN',OS390
```

The .END keyword

The meaning of the .END keyword varies, depending on its context.

Note that $END is also recognized for compatibility with earlier versions of CICS.

If you are using the PARM parameter, the use of the .END keyword is optional. If you omit it, CICS assumes it to be at the end of the PARM parameter. If you code .END in the PARM parameter it can have one of two meanings:

1. If you also code one, or both, of the other control keywords (CONSOLE and/or SYSIN) .END denotes the end of the PARM parameter only. For example:
   ```
   // EXEC DFHSIP,SIZE=DFHSIP,PARM='SI,CN,SIT=6$,.END',OS390
   ```
2. If you code .END as the only control keyword in the PARM parameter, it denotes the end of all system initialization parameters, and CICS begins the initialization process. For example:
   ```
   // EXEC DFHSIP,SIZE=DFHSIP,PARM='SIT=6$,.END',OS390
   ```

If .END is not the last entry in the PARM parameter, CICS truncates the PARM parameter and the parameters following the .END keyword are ignored.

If you are using the SYSIPT data stream, the use of the .END keyword is optional. If you omit it, CICS assumes it to be at the end of SYSIPT. If you code .END in the SYSIPT data stream its meaning depends on your use of the CONSOLE keyword, as follows:

- If you code the CONSOLE control keyword in the PARM parameter or in the SYSIPT data stream, .END denotes the end of the SYSIPT data stream only.
- If you do not code the CONSOLE control keyword in the PARM parameter or in the SYSIPT data stream, .END denotes the end of all CICS system initialization parameters, and CICS begins the initialization process.

If you code .END, and it is not the last entry in the SYSIPT data stream, or not at the end of a SYSIPT record, CICS initialization parameters following the .END are ignored. To avoid accidental loss of initialization parameters, ensure that the .END keyword is on the last record in the SYSIPT data stream, and that it is the only entry on that line. (However, if you want to remove some system initialization parameters from a particular run of CICS, you could position them after the .END statement just for that run.)

The following example shows the use of .END in a SYSIPT data set:

```
// EXEC DFHSIP,SIZE=DFHSIP,PARM='SI',OS390
* CICS system initialization parameters
SIT=6$,START=COLD,
AUXTR=OFF,CSDLSRNO=15
   ⋮
.END
/*
```

If you are using the CONSOLE keyword, the meaning of .END through the console depends on whether you are entering new parameters or entering corrections. The two meanings are as follows:

1. If you are keying new parameters in response to message DFHPA1104, .END terminates parameter reading, and CICS starts initialization according to the SIT it has loaded, but modified by any system initialization parameters you have supplied. Until you enter the .END control keyword, CICS continues to prompt you for system initialization parameters.

2. If you have coded PARMERR=INTERACT, and CICS detects a parameter error, either in the keyword or in the value that you have assigned to it, CICS prompts you to correct the error with message DFHPA1912 or DFHPA1915. If you enter the correct keyword or value, initialization resumes with CICS continuing to process either the PARM parameter, the SYSIPT data set, or prompting for more system initialization parameters through the console, depending on where CICS detected the error. If you cannot correct the error, but want CICS to continue with the initialization process, you can enter .END to end the error correction phase.

Processing the PARM parameter

If you omit the PARM parameter from the EXEC DFHSIP statement CICS assumes that there are no SIT override or other initialization parameters, and attempts to load an unsuffixed version of DFHSIT. As a general rule, this is unlikely to be your intention, so the PARM parameter should at least specify the suffix of your system initialization table, using the SIT system initialization parameter.

Alternatively, you can code the special SYSIN keyword as the only PARM parameter, and supply the suffix of your SIT and the other system initialization parameters from the SYSIPT data set.

CICS scans the PARM string looking for a SIT= parameter, any of the special control keywords, or any system initialization parameters, and proceeds as follows:

- If CICS finds a SIT= parameter but no SYSIN keyword, CICS tries to load the SIT as soon as it has finished scanning the PARM parameter. Processing any CICS system initialization parameters that are also present in the PARM parameter takes place only after the SIT has been loaded.
- If CICS finds a SIT= parameter **and** a SYSIN keyword, CICS does not try to load the SIT until it has also finished scanning the SYSIPT data set. In this case, loading the SIT is deferred because there can be other SIT= parameters coded in the SYSIPT data set that override the one in the PARM parameter.

  Processing any system initialization parameters that are also present in the PARM parameter takes place only after the SIT has been loaded.

Rules for coding PARM parameters

The rules for coding the PARM parameter on an EXEC job control statement are described fully in the *VSE/ESA System Control Statements* manual.

Briefly, the maximum number of characters you can code on each PARM parameter is 100, excluding the opening and closing apostrophes. All CICS system initialization parameters must be separated by a comma, and the separating commas are included in the 100 character limit. You can code the PARM parameter up to three times on one EXEC statement. The syntax rules above apply to each PARM parameter separately. Because of this limiting factor, you might prefer to limit the use of the PARM parameter to specify the SYSIN control keyword only.

Coding the PARM parameter over two lines

If you need to continue your PARM parameter on a second line, indicate the continuation with a nonblank character in column 72, and continue in column 16 on the next line. Do not leave any blank columns between your SIT overrides and the continuation character in column 72; your overrides must be entered up to, and including, column 71.

Figure 31 gives an example of the PARM parameter coded over two lines.

```
0----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
 // EXEC DFHSIP,SIZE=DFHSIP,PARM='SIT=1$,DCT=1B,GRPLIST=DFHLIST,TCT=5$,T*
                S=(,0),SPOOL=YES,START=AUTO',OS390
```

Figure 31. Example of a PARM parameter coded over two lines

Processing the SYSIPT data set

CICS scans the SYSIPT data set looking for a SIT= parameter and any of the special keywords, as well as system initialization parameters.

If CICS finds a SIT= parameter in the SYSIPT data set, it tries to load that SIT, overriding any that was specified in the PARM parameter. If CICS does not find a SIT= parameter in the SYSIPT data set, it tries to load any SIT specified in the PARM parameter.

However, if after scanning the PARM parameter and the SYSIPT data set CICS has not found a SIT= parameter, CICS does one of the following:

1. If you specified CONSOLE in the PARM parameter or in the SYSIPT data set, CICS prompts you with the following message to enter the SIT suffix as the first parameter through the console:

   ```
   DFHPA1921  DBDCCICS  PLEASE SPECIFY THE REQUIRED SIT SUFFIX, OR
                         SPECIFY 'NONE'(UNSUFFIXED).
   ```
2. If you did not specify CONSOLE, CICS automatically tries to load an unsuffixed SIT module (DFHSIT). If this load fails, CICS issues message DFHPA1106, requesting a SIT suffix in reply to the message.

Note: CICS does not process any system initialization parameters that are coded in the PARM parameter and the SYSIPT data set until after the SIT has been loaded.

Rules for coding parameters in the SYSIPT data set

There are a few simple rules to observe when coding CICS system initialization parameters in the SYSIPT data set. These are:

- Use a comma to separate parameters that are on the same line. The use of a comma at the end of a SYSIPT record is optional.
- You can use an asterisk in column 1 to code comments, or to remove temporarily an initialization parameter from a particular execution of CICS.
- You can also add comments after the parameters on a SYSIPT line, but they must be preceded by at least one blank character.
- The SYSIPT data set is an 80-byte file. Everything that appears in positions 1 through 80 is treated by CICS as input data.
- You can continue, on another line in SYSIPT, parameters that have multiple operands if you make the split immediately after a comma. CICS concatenates the operands, omitting the intervening blanks.
- As a general rule, you cannot split an individual operand between lines. However, in the case of the GMTEXT parameter, you can enter the operand over more than one line up to the maximum of 246 characters. The format of this parameter is:

  ```
  GMTEXT='User''s text'
  ```

You can use apostrophes to punctuate message text, provided that you code *two* successive apostrophes to represent a single apostrophe (as shown in the example above). The apostrophes delimiting the text are mandatory.

- You must take care when coding parameters that use apostrophes, parentheses, or commas as delimiters, because failure to include the correct delimiters is likely to cause unpredictable results.

Processing the console entries

Generally, CICS does not begin to read from the console until it has loaded the SIT and processed any initialization parameters that are coded in the PARM parameter and the SYSIPT data set. CICS accepts system initialization parameters from the console until you terminate the input with .END.

You can specify a SIT= parameter only as the first parameter through the console when prompted by message DFHPA1921, at which point CICS tries to load the specified SIT. If you try to specify a SIT= parameter after CICS has loaded the SIT it is rejected as an error.

Rules for coding parameters at the console

When it is ready to read parameters from the console, CICS displays the following message (where nn is the reply ID):

```
nn DFHPA1104 applid - SPECIFY ALTERNATIVE SIT PARAMETERS, IF ANY,
                      AND THEN TYPE '.END'.
```

You can enter as many initialization parameters as you can get on one line of the console, but you must use a comma to separate parameters. CICS continues to prompt for system initialization parameters with displays of message DFHPA1105 until you terminate console input by entering the .END control keyword.

Entering corrections to parameters at the console

If you have coded PARMERR=INTERACT, and CICS detects a parameter error, either in the keyword or in the value you have assigned to it, CICS prompts you to correct the error with message DFHPA1912 or DFHPA1915:

```
DFHPA1912 applid  SIT OVERRIDE 'keyword' IS NOT RECOGNIZED.
                  SPECIFY CORRECT SIT OVERRIDE.

DFHPA1915 applid INVALID DATA HAS BEEN DETECTED FOR SIT OVERRIDE
                 'keyword'.  RESPECIFY THE OVERRIDE.
```

CICS prompts you to enter corrections to any errors it find in the PARM parameter or the SYSIPT data set ***after*** it has loaded the SIT, and as each error is detected. This means that if there is an APPLID parameter ***following*** the parameter that is in error, either in the PARM parameter or in the SYSIPT data set, it is the APPLID coded in the SIT that CICS displays in messages DFHPA1912 and DFHPA1915.

Notes on CICS resource table and module keywords

Table 5 on page 166 shows the system initialization keywords for those CICS resources that:

- Have a suffix option
- Result in a dummy program or table if you code resource=NO
- You can COLD start individually.

Table 5. Summary of resources with a suffix, a dummy load module, or a COLD option

DFHSIT keyword	Default 1	Suffix 2	Dummy 3	COLD start 4
BMS	FULL	-	-	COLD
CLT	-	xx	-	-
DBP	-	xx	-	-
DCT	YES	xx	-	COLD
DIP	NO	-	program	-
DL1 or DLI	NO	-	-	COLD
FCT	YES	xx	-	-
ICP	-	-	-	COLD
JCT	YES	xx	program	-
MCT	NO	xx	5	-
PLTPI	NO	xx	-	-
PLTSD	NO	xx	-	-
SRT	YES	xx	-	-
TCT	YES	xx	table	-
TS	-	-	-	COLD
TST	NO	xx	-	-
XLT	NO	xx	-	-

Notes for Table 5:

1 The **Default** column indicates the default value for the keyword in the DFHSIT macro.

If you code YES in the SIT for those keywords with the suffix option, an unsuffixed version of the table or program is loaded. For example, DCT=YES results in a table called DFHDCT being loaded.

You can also select an unsuffixed module or table at CICS startup by specifying ***keyword=,*** or ***keyword=YES***. For example, if you code:

```
DBP=, or DBP=YES
FCT=, or FCT=YES
```

blanks are appended to DFHDBP and DFHFCT respectively, and these unsuffixed names are used during initialization.

The result of specifying ***keyword=,*** as a system initialization parameter through PARM, SYSIPT, or CONSOLE is not necessarily the same as in the DFHSIT macro. For example, TST=, (or omitted altogether) when coding the DFHSIT macro is taken to mean TST=NO, but TST=, through any of the other three methods is the same as TST=YES.

2 The **Suffix** column indicates whether you can code a suffix. (*xx* indicates that a suffix can be coded.)

The DBP keyword is mandatory; you must code either a suffix or YES for DBP. For more information about the DBP system initialization parameter, see page 189.

A suffix can be any 1 or 2 characters, but you must not use DY. You cannot use NO as a suffix.

If you code a suffix, a table or program with that suffix appended to the standard name is loaded. For example, DBP=2$ causes DFHDBP2$ dynamic backout program to be included in your CICS region.

When the suffix option is specified with other values, the two values must be enclosed within parentheses: for example, DCT=(xx,COLD).

3 The **Dummy** column indicates whether a dummy version of the program or table is loaded if you code NO. In some cases, coding NO for the operand associated with the **table** results in a dummy **program**. For more information about the effect of this option, see “Selecting versions of CICS programs and tables”.

4 The **COLD start** column indicates whether the resource can be forced to start COLD. (COLD indicates that the resource can be cold started individually).

If COLD is coded, it can be overridden only by coding START=(...,ALL) as a system initialization parameter. For more information about this option, see page 224.

For more information about CICS table and program selection, see “Selecting versions of CICS programs and tables”.

5 If you code MCT=NO, the CICS monitoring domain builds dynamically a default monitoring control table. This ensures that default monitoring control table entries are always available for use when monitoring is on and a monitoring class is active.

Selecting versions of CICS programs and tables

A CICS program is usually made up from a group of related CICS functional modules, one example of which is the terminal control program. For most CICS programs you can only have one version, which is supplied with CICS. However, for some CICS programs you can create more than one version; for example, with different service levels. To select a particular version of a program, you can include the load library containing that version in the CICS startup JCL. For the following programs, however, you can select from different versions, by specifying the version you require at system initialization:

1. The dynamic backout program (DBP).
2. The basic mapping support (BMS) program suite.

There are two ways of selecting the versions you need:

1. Using a suffix. Use this method for DBP.
2. Explicitly selecting the level of function needed. Use this method for BMS.

You can also specify that a program is ***not*** needed (see “Excluding unwanted programs” on page 168 for details).

You can use these methods ***only*** for the programs referred to in this section and in “Excluding unwanted programs” on page 168, by coding system initialization parameters.

Using a suffix to select the dynamic backout program

Suffixes are used to distinguish different versions of the dynamic backout program (DBP), a CICS management program. Two versions of DBP are supplied with CICS, both in pregenerated format. These two suffixed versions are:

Suffix	Description
1$	CICS local DL/I is not supported.
2$	CICS local DL/I is supported.

Using an explicit level of function to select programs

You use an explicit level of function to select the BMS suite of programs. When you specify your BMS requirement on the BMS system initialization parameter, you can select one of three versions. The BMS level of function is selected by the parameter options MINIMUM, STANDARD, or FULL, from which the system initialization program loads the set of programs you require.

Excluding unwanted programs

The three ways of excluding programs that are not required are by specifying one of programname=NO, tablename=NO, or function=NO.

Specifying programname=NO

If you code programname=NO in your system initialization table (for example, DIP=NO), or as a SIT override parameter, you exclude the named management program at CICS system initialization.

The programs that you can exclude by coding programname=NO are:
- Batch data interchange program (DIP)
- Terminal control program (TCP).

Note: In the case of DIP, you get a dummy version of the management program, which is supplied on the distribution tape with a suffix of **DY**.

Specifying tablename=NO for the program's control table

Not all of the CICS programs have a programname parameter in the SIT. The alternative method of excluding them is to code NO against the table name for these programs. This has the same effect as coding NO against a program name parameter, and the associated CICS program is excluded at system initialization, either by loading a dummy program, or by some other technique.

The tables that can be used in this way, and their associated management programs, are shown in Table 6.

Table 6. SIT control tables with a NO option

Control tables	***Associated management modules***
Journal control table (JCT)	Journal control program (JCP)
System recovery table (SRT)	System recovery program (SRP)

A dummy version of the management program (suffixed **DY**) is supplied for journal control. The dummy program is DFHJCPDY. because you don't need DFHJCP if you specify no journaling.)

The dummy TCT, DFHTCTDY

There is a special case where you can also specify tablename=NO, but this does not load a dummy terminal control program. You specify TCT=NO when you are using resource definition online, and all of your terminal resource definitions are in the CSD.

When you specify TCT=NO, CICS loads a dummy TCT named DFHTCTDY. If you specify TCT=NO, a generated table of this name must be available in a sublibrary of the LIBDEF PHASE, SEARCH chain for the CICS job when you start CICS. A pregenerated dummy table, and its source, are provided in the VSE/ESA sublibrary, PRD1.BASE.

The dummy TCT provides ***only*** the CICS and VTAM control blocks that you need if you are using VTAM terminals and using the CSD for storing terminal definitions. You define your VTAM terminals using the RDO transaction, CEDA, a user application program using EXEC CICS CREATE TERMINAL commands, or the DEFINE command of the DFHCSDUP utility program.

Specifying function=NO

If you code function=NO as a system initialization parameter (for example, XRF=NO), you exclude the management program associated with the named function at CICS system initialization.

You can exclude CICS DL/I support, intersystem communication (ISC), the 3270 print-request facility, the system spooling interface, or the extended recovery facility (XRF), in this way.

Classes of start and restart

The type of initialization that CICS performs is not only determined by the START parameter. The CICS local and global catalogs also play a major role in the initialization process, together with any system initialization parameters that you provide, either in the SIT or at run time by one of the three methods described in this chapter.

The global catalog

CICS uses the global catalog to save all resource definitions that are installed at CICS shutdown. These are:

- Programs
- Transactions and transaction profiles
- Transaction classes
- Terminals, including any which are autoinstalled
- Typeterms
- Connections and sessions
- BMS maps sets and partition sets
- Files.

The resource definitions that CICS saves at its shutdown may have been installed during a cold start (from a list of groups specified by a group list system initialization parameter), or during CICS execution (by CEDA INSTALL or EXEC CICS CREATE commands).

If you run CICS with START=AUTO, and a warm or emergency restart results, CICS restores all the installed resource definitions as they were at normal CICS shutdown, or at the time of system failure. The general rule is that you cannot alter installed resource definitions during a restart except by coding START=COLD.

The CICS domains also use the global catalog to save their domain status between runs. In some cases this information can be overridden during a restart by supplying system initialization parameters. For example, CICS monitoring uses the cataloged status at a restart, but modified by any system initialization parameters you provide. In other cases the domain information saved in the catalog is always used in a restart.

For example, CICS statistics interval time is always restored from the catalog in a warm or emergency restart, because the statistics domain does not have this as a system initialization parameter. To change this you must use CEMT or EXEC CICS commands after control is given to CICS. Alternatively, you can enforce system defaults by performing a cold start.

Note: If you need to reinitialize the CICS global catalog for any reason, you must also reinitialize the local catalog.

The local catalog

The CICS domains use the local catalog to save some of their information between CICS runs. If you delete and redefine the local catalog, you must:

- Initialize the CICS local catalog with an initial set of domain records.
- Use the CICS-supplied utility program, DFHSMUTL, to re-add records to enable the CICS self-tuning mechanism for storage manager domain subpools. For details of how to do this, see the *CICS Operations and Utilities Guide*.
- Delete and reinitialize the CICS global catalog.

For more information about initializing the local catalog, see the CICS System Definition Guide.

Some of the information that is saved in the local catalog can be overridden at CICS system initialization by system initialization parameters, such as CICS transaction dump data set status.

Note: If you need to reinitialize the local catalog for any reason, you must also reinitialize the global catalog.

The START system initialization parameter

You can influence the type of startup that CICS performs, by specifying the START system initialization parameter, as follows:

START=AUTO

If you code AUTO as the START operand, CICS determines, by inspecting the control record in the global catalog, which of the following three types of start to perform.

1. WARM

If the control record in the global catalog indicates that the previous run of CICS terminated normally with a successful warm keypoint, CICS performs a warm restart. The local catalog must also contain the information saved by the CICS domains during the previous execution for the warm restart to be successful. A warm start restores CICS to the state it was in at the previous shutdown.

If you are using disk journaling, the status of the disk journals is also saved in the global catalog. This information is used by CICS at startup to determine which journal data set is to be opened. You can use the JSTATUS=RESET startup parameter to cause the status in the global catalog to be ignored. During CICS startup, the status of all journal data sets is set to “ready for use”. For more information about using JSTATUS=RESET, see the CICS System Definition Guide and the description of the JSTATUS parameter on page 204.

You can modify a warm restart by coding the NEWSIT system initialization parameter. This has the effect of enforcing the system initialization parameters coded in the SIT, overriding any cataloged status from the previous CICS shutdown.

The exceptions to this are the system initialization parameters FCT, the CSDxxxxx group (for example CSDACC), and GRPLIST, which are always ignored in a warm restart, even if you specify NEWSIT=YES. Specifying NEWSIT=YES causes, in effect, a partial cold start.

2. COLD

If there is no control record in the CICS global catalog, CICS assumes that the catalog has been newly initialized, and forces a cold start. If you have recreated the global catalog for some reason, you must also reinitialize the local catalog.

3. EMERGENCY

If the control record in the global catalog indicates that the previous run of CICS terminated in an immediate or uncontrolled shutdown, CICS performs an emergency restart.

The emergency restart procedure uses the system log at the time of the failure to return recoverable resources to their committed states.

START=AUTO should be the normal mode of operation, with the choice of start being made by CICS automatically.

START=COLD

If you code COLD as the START operand, CICS initializes using the resource definitions specified by the system initialization parameters, ignoring any previously installed resource definitions saved in a warm keypoint in the global catalog. This includes all the groups of resources specified by the GRPLIST= system initialization parameter, and those resources specified in CICS control tables.

Be aware, however, that a start initiated by START=COLD is not entirely without reference to the previous run of a CICS system using the same global catalog.

For example, CICS checks the ‘global catalog’ for any dataset name block entries for VSAM data sets for which backout failures have occurred previously.

You can perform a fully cold start of CICS, without reference to any previous execution, only by reinitializing both CICS catalogs. Generally, do this only when you are starting your CICS system for the first time.

There may be times when it is necessary to restart CICS with START=COLD, irrespective of the type of system termination that has been recorded in the global catalog.

START=STANDBY

The STANDBY option is for use only with XRF=YES. START=STANDBY initializes an alternate CICS region. If you code START=STANDBY with XRF=NO, initialization fails with message DFHXA6530, and CICS terminates abnormally with a dump.

CICS initializes as an alternate CICS region by beginning to perform an emergency restart, which is then suspended until it needs to perform a takeover. During the period when initialization is suspended, the alternate CICS region is in standby mode and monitors the active CICS region. When it takes over, the alternate CICS region completes the emergency restart and becomes the active CICS region.

If you have specified a COLD start for other CICS resources, for example, DCT=(xx,COLD), they are cold started when the alternate CICS region (with START=STANDBY specified) takes over. This may cause CICS to lose data on an XRF takeover; for example, coding ICP=COLD results in outstanding STARTs being lost. You are recommended to code START=(STANDBY,ALL) to ensure a full emergency restart during takeover, unless you wish to specifically cold start individual resources.

For information about operating a CICS region with XRF, see the CICS Operations and Utilities Guide.

START=LOGTERM

The LOGTERM operand on the START parameter is a special restart option. It is for use only when the system log is defined on ***disk*** data sets, to cater for an abnormal termination of a CICS system that does not close the system log. The LOGTERM option causes a CICS restart to put only an end of file label on the log, and then terminate ***before*** doing any backout processing. You can use START=LOGTERM, for example, if you need to close the system log to perform offline file recovery, particularly in those cases when you know (or suspect) that emergency restart won't work.

LOGTERM is available only as a system initialization parameter at run time, and cannot be coded in the system initialization table. It is also intended for use only when you are running CICS with XRF=NO.

Table 7. Effect of the START= parameter in conjunction with the catalogs

START= parameter	State of the CICS catalogs	Result at restart
COLD	Local and global catalogs are both newly initialized.	CICS performs a fully cold start. All domains are initialized using system default values, modified by any system initialization parameters. All resources are installed as specified by system initialization parameters.
COLD	The global catalog contains a successful warm keypoint from previous run, and the local catalog contains information saved by the CICS domains.	CICS performs a cold restart, installing the resource definitions specified by system initialization parameters. The domains initialize according to system initialization parameters, or using system default values where there are no parameters (for example the statistics domain). CICS also checks the global catalog for any data set name block entries for VSAM data sets for which backout failures were recorded in the system log.

Table 7. Effect of the START= parameter in conjunction with the catalogs (continued)

START= parameter	State of the CICS catalogs	Result at restart
AUTO	Local and global are both newly initialized.	CICS enforces a fully cold start. All domains are initialized with system default values, modified by any system initialization parameters. All resources are installed as specified by system initialization parameters.
AUTO	The global catalog contains a successful warm keypoint from the previous run. The local catalog is newly initialized.	CICS begins to perform a warm restart, but fails during the initialization process because of absence of expected records in the local catalog. (***Do not reinitialize only one of the catalogs***.)
AUTO	The global catalog contains a successful warm keypoint from the previous run, and the local catalog contains information saved by the domains.	CICS performs a warm restart, restoring all of your CICS system except the trace domain to the same status it was in at CICS shutdown. (CICS trace domain does not save the status of the various trace options at CICS shutdown, and always uses the system initialization parameters.) Only those resources that have a COLD option on their system initialization parameter can be cold started (for example, the destination control table or auxiliary temporary storage).

Table 8 shows the effect of the various START options, combined with system initialization parameters where applicable, on the CICS trace, monitoring, statistics, and dump domains:

Table 8. Effect of the START= parameter on the CICS domains at initialization

Domain	State of the CICS catalogs	Result at startup if START=AUTO	Result at startup if START=COLD
Trace	Not relevant	Domain initializes according to the system initialization parameters.	Domain initializes according to the system initialization parameters.
Monitoring	The global catalog is newly initialized.	Domain initializes according to the system initialization parameters.	Domain initializes according to the system initialization parameters.
Monitoring	The global catalog contains status of monitoring at the previous CICS shutdown.	Domain uses monitoring status from the catalog, but modified by any system initialization override parameters.	Domain initializes according to the system initialization parameters.
Statistics	The global catalog is newly initialized.	Domain initializes according to CICS-defined system default values.	Domain initializes according to CICS-defined system default values.
Statistics	The global catalog contains status of statistics at CICS shutdown.	Domain uses statistics status from the catalog.	Domain initializes according to CICS-defined system default values.

Table 8. Effect of the START= parameter on the CICS domains at initialization (continued)

Domain	State of the CICS catalogs	Result at startup if START=AUTO	Result at startup if START=COLD
Dump	The global catalog is newly initialized.	Domain initializes the dump table according to CICS-defined system default values. Other dump attributes are set by system initialization parameters.	Domain initializes an empty dump table, and takes CICS-defined default action for all dump requests. Other dump attributes are set by system initialization parameters.
Dump	The global catalog contains dump status at CICS shutdown.	Domain reads the dump table and dump status from the catalog, but the dump status is modified by any system initialization parameters.	Domain initializes an empty dump table, and takes CICS-defined default action for all dump requests. Other dump attributes are set by system initialization parameters.

CICS startup and the VTAM session

In a VTAM network, the session between CICS and VTAM is started automatically if VTAM is started before CICS. `EXEC=IESWAITT` is a procedure that waits until VTAM is open. If VTAM is not active when you start CICS, you receive the following messages:

```
+DFHSI1589D 'applid' VTAM is not currently active.
+DFHSI1572 'applid'  Unable to OPEN VTAM ACB - RC=xxxxxxxx, ACB CODE=yy.
```

If you receive messages DFHSI1589D and DFHSI1572, and if the CICS region is not initializing as an alternate CICS region, you can start the CICS-VTAM session manually when VTAM is eventually started, by means of the CEMT SET VTAM OPEN command from any VSE console defined to CICS.

If VTAM is active, but CICS still cannot open the VTAM ACB because VTAM does not recognize the CICS APPLID, you receive the following messages:

```
+DFHSI1592I 'applid' CICS applid not (yet) active to VTAM.
+DFHSI1572  'applid' Unable to OPEN VTAM ACB - RC=00000008, ACB CODE=5A.
```

This may be caused by an error in the value of APPLID operand, in which case you must correct the error and restart CICS. For information about other causes and actions, see the CICS Messages and Codes manual.

Concurrent initialization of VTAM and XRF alternate CICS regions

An XRF alternate CICS region cannot initialize properly until it has successfully opened the VTAM ACB.

Because VTAM and the alternate CICS region may be initialized concurrently, it is possible that several tries may have to be made to open the VTAM ACB. If VTAM is not active, the following message is written to the system console every 15 seconds:

```
DFHSI1589D 'applid' VTAM is not currently active.
```

If VTAM is active, but CICS cannot open the VTAM ACB, the following messages are written to the system console:

```
+DFHSI1572 'applid' Unable to OPEN VTAM ACB - RC=xxxxxxxx, ACB CODE=yy.
DFHSI1590 'applid'  XRF alternate cannot proceed without VTAM.
```

CICS abends with a dump (abend code 1590).

End of CICS startup

Whichever type of startup is performed, when the message:

```
DFHSI1517 'applid' Control is being given to CICS.
```

is displayed on the operating system console, CICS is ready to process terminal requests (*applid* is the value of the specific APPLID system initialization parameter).

When the startup process is completed, users are able to enter transactions from any terminals that are connected to CICS. For information about the CICS-supplied transactions, see the CICS Supplied Transactions manual.

The system initialization parameter descriptions

Unless otherwise stated, all of the system initialization parameters described here can be defined to CICS by any of these four ways:

1. In a DFHSIT macro
2. In a PARM parameter on the DFHSIP statement
3. In the SYSIPT data set of the CICS startup job stream
4. Through the system console.

Default notation

Default values are **underscored**; for example, TYPE=**CSECT**. This notation applies to the SIT macro parameters only.

TYPE={CSECT|DSECT}
Indicates the type of SIT to be generated.

CSECT
A regular control section that is normally used.

DSECT
A dummy control section.

ADI={30|number}
Specifies, when you are running CICS with XRF, the alternate delay interval in seconds. The minimum delay that you can specify is 5 seconds. This is the time that must elapse between the (apparent) loss of the surveillance signal in the active CICS system, and any reaction by the alternate CICS system. The corresponding parameter for the active is PDI. ADI and PDI need not have the same value.

AIEXIT={DFHZATDX|DFHZATDY|name}
Specifies the name of the autoinstall user-replaceable program that you want CICS to use when autoinstalling local VTAM terminals, APPC connections, and remote terminals. Autoinstall is the process of installing resource definitions automatically, using VTAM logon or BIND data, model definitions, and an autoinstall program.

Important

You can specify only one user-replaceable program on the AIEXIT parameter. Which of the CICS-supplied programs (or customized versions thereof) that you choose depends on what combination of resources you need to autoinstall.

For background information about autoinstall, see the CICS Resource Definition Guide.

DFHZATDX
A CICS-supplied autoinstall user program. This is the default. It installs definitions for locally-attached VTAM terminals, remote shipped terminals, and remote shipped connections.

DFHZATDY
A CICS-supplied autoinstall user program. It installs definitions for both locally-attached VTAM terminals, local APPC connections, remote shipped terminals, and remote shipped connections.

name

The name of your own customized autoinstall program, which may be based on one of the supplied sample programs. For programming information about writing your own autoinstall program, see the CICS Customization Guide.

AILDELAY={0|hhmmss}

Specifies the delay period that elapses after a session between CICS and an autoinstalled terminal is ended, before the terminal entry is deleted. A session is ended when a terminal logs off or when a transaction disconnects a terminal from CICS.

hhmmss

Specify a 1-to 6-digit number. The default is 0, meaning the terminal entry is deleted as soon as the session is ended. If you leave out the leading zeros, they are supplied (for example, 123 becomes 000123, that is, 1 minute 23 seconds).

Note: The AILDELAY parameter does not apply to autoinstall of APPC connections, because they are not deleted.

AIQMAX={100|number}

Specifies the maximum number of VTAM terminals and APPC connections that can be queued concurrently for autoinstall.

number

A number in the range 0 through 999. The default is 100.

A zero value disables the autoinstall function.

You should specify a number that is large enough to allow for both APPC connnections and terminals.

Note: This value does not limit the total number of terminals that can be autoinstalled. If you have a large number of terminals autoinstalled, shutdown can fail due to the MXT system initialization parameter being reached or CICS becoming short on storage. For information about preventing this possible cause of shutdown failure, see the CICS Performance Guide.

AIRDELAY={700|hhmmss}

Specifies the delay period that elapses after an emergency restart before autoinstalled terminal entries that are not in session are deleted.

hhmmss

Specify a 1-to 6-digit number. If you leave out the leading zeros, they are supplied. The default is 700, meaning a delay of 7 minutes. A value of 0 means that autoinstalled terminal definitions are not written to the global catalog and therefore are not restored at an emergency restart. For guidance about the performance implications of setting different AIRDELAY values, see the CICS Performance Guide.

Note: The AIRDELAY parameter does not apply to autoinstall of APPC connections, because they are not cataloged.

XRF restriction

If you are running CICS with XRF, set the same value on the AIRDELAY parameter for both the active and the alternate CICS systems. It is particularly important, if you want autoinstall sessions to be reestablished after a takeover, that you avoid coding a zero on this parameter for either the active or the alternate CICS systems.

For background information, see the CICS XRF Guide.

AKPFREQ={200|number}

If AKPFREQ is a number other than zero, it specifies the number of consecutive blocks, written by DFHJCP to the system log data set, that triggers the activity keypoint function. The minimum number that should be coded is 200 (the default) and the maximum number is 65535. (The CICS region must support activity keypointing: that is, the CSKP transaction and DFHAKP program must be defined. For information about supporting activity keypointing, see the CICS Recovery and Restart Guide and the CICS Performance Guide.)

If AKPFREQ=0 is coded, no activity keypoints are taken and a subsequent emergency restart is not possible.

APPLID={DBDCCICS|applid}

The VTAM application identifier for this CICS system.

applid

This name, 1 through 8 characters, identifies the CICS system in the VTAM network. It must match the name field specified in the APPL statement of the VTAM VBUILD TYPE=APPL definition.

When you define this CICS system to another CICS system, in a CONNECTION definition, you specify the applid as the NETNAME.

If the CICS system uses XRF, the form of the APPLID parameter is:

APPLID=(generic_applid,specific_applid)

Specifies the generic and specific XRF applids for the CICS system. Both applids must be 1 through 8 characters.

generic_applid

This is the generic applid for both the active and the alternate CICS systems. Therefore, you must specify the same name for *generic_applid* on the APPLID system initialization parameter for both CICS systems. Because IRC uses *generic_applid* to identify the CICS systems, there can be no IRC connection for an alternate CICS system until takeover has occurred and the alternate CICS system becomes the active CICS system.

When you define this XRF pair to another CICS system, in a CONNECTION definition, you specify the generic applid as the NETNAME.

specific_applid

This identifies the CICS system in the VTAM network. It must match the label specified in the VTAM VBUILD TYPE=APPL definition. You must specify a different *specific_applid* on the APPLID system initialization parameter for the active and for the alternate CICS system. Also, *generic_applid* and *specific_applid* must be different.

The active and alternate CICS systems use the VTAM MODIFY USERVAR command to set a user application name variable, so end users do not need to know which CICS system is active at any instant. For background information about using this command, see the CICS XRF Guide.

AUTCONN={0|hhmmss}

Specify this to delay the reconnection of terminals after an XRF takeover, to allow time for manual switching. The delay is hh hours, mm minutes, ss seconds. The default value of zero means that there is no delay in the attempted reconnection.

The interval specified is the delay before the CXRE transaction runs. CXRE tries to reacquire terminals that were in session at the time of the takeover.

Note that the same delay interval applies to the connection of terminals with AUTOCONNECT(YES) specified in the RDO TYPETERM definition, at a warm or emergency restart, whether or not you have coded XRF=YES.

AUXTR={OFF|ON}

Indicates if the auxiliary trace destination is to be activated at system initialization. This parameter controls whether any of the three types of CICS trace entry are written to the auxiliary trace data set. The three types are: CICS system trace (see the SYSTR parameter), user trace (see the USERTR parameter), and exception trace entries (that are always made and are not controlled by a system initialization parameter).

OFF

Do not activate auxiliary trace.

ON

Activate auxiliary trace.

For details of internal tracing in main storage, see the INTTR parameter on page 204.

AUXTRSW={NO|ALL|NEXT}

Specifies whether you want the auxiliary trace autoswitch facility.

NO

Disables the autoswitch facility.

NEXT

Enables the autoswitch facility to switch to the next data set at end of file of the first data set used for auxiliary trace. Coding NEXT permits one switch only, and when the second data set is full, auxiliary trace is switched off.

ALL

Enable the autoswitch facility to switch to the inactive data set at every end of file. Coding ALL permits continuous switching between the two auxiliary trace data sets, DFHAUXT and DFHBUXT, and whenever a data set is full, it is closed and the other data set is opened.

BMS=({MINIMUM|STANDARD|FULL}[, COLD][,{UNALIGN|ALIGN}][, { DDS|NODDS}])

Specifies which version of basic mapping support you want to be included. The function included in each version of BMS is shown in Table 9 on page 180. The parameter BMS can be overridden during CICS initialization.

You need full or standard function BMS, if you are using XRF and have specified MESSAGE for RECOVNOTIFY on any of your RDO TYPETERM resource definitions.

MINIMUM
The minimum version of BMS is included.

STANDARD
The standard version of BMS is included.

FULL
The full version of BMS is included. This is the default in the SIT.

COLD
CICS deletes delayed messages from temporary storage, and destroys their interval control elements (ICEs).

UNALIGN
Specifies that all BMS maps assembled before CICS/DOS/VS Version 1 Release 6 are unaligned. Results are unpredictable if the stated alignment does not match the actual alignment.

ALIGN
Code this to indicate that all BMS maps assembled before CICS/DOS/VS Version 1 Release 6 are aligned.

DDS
BMS is to load suffixed versions of map sets and partition sets. BMS first tries to load a version that has the alternate suffix (if the transaction uses the alternate screen size). If the load fails, BMS tries to load a version that has the default map suffix. If this fails too, BMS tries to load the unsuffixed version. DDS, which stands for “device dependent suffixing”, is the default.

You need to use map suffixes only if the same transaction is to be run on terminals with different characteristics (in particular, different screen sizes). If you do not use suffixed versions of map sets and partition sets, CICS need not test for them.

NODDS
BMS is not to load suffixed versions of map sets and partition sets. Specifying NODDS avoids the search for suffixed versions, saving processor time.

Table 9. Versions of BMS

BMS version	Devices supported	Function provided
MINIMUM	All 3270 system display units and printers except SNA character string printers, which are defined as DEVICE(SCSPRINT) on the RDO TYPETERM definition or as TRMTYPE=SCSPRT in DFHTCT	SEND MAP command, RECEIVE MAP command, SEND CONTROL command. Default and alternate screens; extended attributes; map set suffixes; screen coordination with null maps; and block data
STANDARD	All devices are supported by BMS. These are listed in the CICS Application Programming Guide	All function of MINIMUM, ***plus*** outboard formats, partitions, controlling a magnetic slot reader, NLEOM mode for 3270 system printers, SEND TEXT command, and Subsystem LDC controls

Table 9. Versions of BMS (continued)

BMS version	Devices supported	Function provided
FULL	All devices supported by BMS. These are listed in the CICS Application Programming Guide	Same as STANDARD, ***plus*** terminal operator paging, cumulative mapping, page overflow, cumulative text processing, routing, message switching returning BMS-generated data stream to program before output.

CDSASZE={0K|number}

Specifies the size of the CDSA. The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 through 16777215 bytes in multiples of 262144 bytes (256KB). If the size specified is not a multiple of 256KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions

You can code the CDSASZE parameter in PARM, SYSIPT, or CONSOLE only.

CHKSTRM={CURRENT|NONE}

Activate, or deactivate, terminal storage-violation checking. The operands have the following meanings:

CURRENT

Code CURRENT to check for TIOA storage violations.

NONE

Code NONE to deactivate TIOA storage-violation checking.

You can also use the CICS-supplied transaction, CSFE, to switch terminal storage-violation checking on and off.

For information about checking for storage violations, see the CICS Problem Determination Guide.

Restrictions

You can code the CHKSTRM parameter in PARM, SYSIPT, or CONSOLE only.

CHKSTSK={ALL|CURRENT|NONE}

Activate, or deactivate, task storage-violation checking at startup. The operands have the following meanings:

ALL

Code ALL to check all storage areas on the transaction storage chains for all tasks.

CURRENT
Code CURRENT to check all storage areas on the transaction storage chain for the current task only.

NONE
Code NONE to deactivate task storage-violation checking.

You can also use the CICS-supplied transaction, CSFE, to switch task storage-violation checking on and off.

For information about checking for storage violations, see the CICS Problem Determination Guide.

Restrictions

You can code the CHKSTSK parameter in PARM, SYSIPT, or CONSOLE only.

CLSDSTP={NOTIFY|NONOTIFY}
Specifies the notification required for an EXEC CICS ISSUE PASS command. This parameter is applicable to both autoinstalled and non-autoinstalled terminals. You can use the notification in a user-written node error program to reestablish the CICS session when a VTAM CLSDST PASS request resulting from an EXEC CICS ISSUE PASS command fails. For more information about the EXEC CICS ISSUE PASS command, see the CICS Application Programming Reference manual.

NOTIFY
CICS requests notification from VTAM when the EXEC CICS ISSUE PASS command is executed.

NONOTIFY
CICS does not request notification from VTAM.

CLT=xx
Specify the suffix for the command list table (CLT), if this SIT is used by an alternate XRF system. The name of the table is DFHCLTxx.

For information about coding the macros for this table, see the CICS Resource Definition Guide.

CMDPROT={YES|NO}
Specifies whether to allow, or inhibit, CICS validation of start addresses of storage referenced as output parameters on EXEC CICS commands.

YES
If you specify YES, CICS validates the initial byte at the start of any storage that is referenced as an output parameter on EXEC CICS commands to ensure that the application program has write access to the storage. This ensures that CICS does not overwrite storage on behalf of the application program when the program itself cannot do so. If CICS detects that an application program has asked CICS to write into an area to which the application does not have addressability, CICS abends the task with an AEYD abend.

The level of protection against bad addresses depends on the level of storage protection in the CICS environment. The various levels of protection provided when you specify CMDPROT=YES are shown in Table 10 on page 183.

NO

If you specify NO, CICS does not perform any validation of addresses of the storage referenced by EXEC CICS commands. This means an application program could cause CICS to overwrite storage to which the application program itself does not have write access.

Table 10. Levels of protection provided by CICS validation of application-supplied addresses

Environment	Execution key of affected programs	Types of storage referenced by applications that cause AEYD abends
Read-only storage (RENTPGM=PROTECT)	CICS-key and user-key	CICS key 0 read-only storage (RDSA and ERDSA).
Subsystem storage protection (STGPROT=YES)	User-key	All CICS-key storage (CDSA and ECDSA)
Base CICS (all storage is CICS key storage) (RENTPGM=NOPROTECT; and STGPROT=NO	CICS-key and user-key	VSE storage only

CMDSEC={ASIS|ALWAYS}

Specifies whether or not you want CICS to honor the CMDSEC option specified on a transaction's resource definition.

ASIS

means that CICS honors the CMDSEC option defined in a transaction's resource definition. CICS calls its command security checking routine only when CMDSEC(YES) is specified in a RDO TRANSACTION resource definition.

ALWAYS

means that CICS overrides the CMDSEC option, and always calls its command security checking routine to issue the appropriate call to the System Authorization Facility (SAF) interface.

Notes:

1. Specify ALWAYS when you want to control the use of the SPI in all your transactions. Be aware that this might incur additional overhead. The additional overhead is caused by CICS issuing the command security calls on every eligible EXEC CICS command, which are *all* the system programming interface (SPI) commands.
2. If you specify ALWAYS, command checking applies to CICS-supplied transactions such as CESN and CESF. You must authorize all users of CICS-supplied transactions to use the internal CICS resources for the transactions, otherwise you will get unexpected results in CICS-supplied transactions.

Restrictions

You can code the CMDSEC parameter in the SIT, PARM, or SYSIPT only.

CONFDATA={SHOW|HIDETC}

Specifies whether CICS is to suppress (hide) user data that might otherwise appear in CICS trace entries or in dumps that contain the VTAM receive any input area (RAIA.). This option applies to initial input data received on a VTAM RECEIVE ANY operation, the initial input data received on an MRO link, and FEPI screens and RPLAREAs.

SHOW

Data suppression is not in effect. User data is traced regardless of the CONFDATA option specified in transaction resource definitions. This option overrides the CONFDATA option in RDO TRANSACTION resource definitions.

HIDETC

Specifies that you want CICS to 'hide' user data from CICS trace entries. It also indicates that VTAM RAIAs are to be suppressed from CICS dumps. The action actually taken by CICS is subject to the individual CONFDATA attribute on the RDO TRANSACTION resource definition (see Table 11 on page 185).

If you specify CONFDATA=HIDETC, CICS processes VTAM, MRO, and FEPI user data as follows:

- **VTAM**

  CICS clears the VTAM RAIA containing initial input as soon as it has been processed, and before the target transaction has been identified.

  The normal trace entries (FC90 and FC91) are created on completion of the RECEIVE ANY operation with the text "`SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT`" replacing all the user data except the first 4 bytes of normal data, or the first 8 bytes of function management headers (FMHs).

  CICS then identifies the target transaction for the data. If the RDO TRANSACTION resource definition specifies CONFDATA(NO), CICS traces the user data that it suppressed from the FC90 trace in the trace entry AP FC92. This trace entry is not created if the transaction is defined with CONFDATA(YES) on the RDO TRANSACTION resource definition.

- **MRO**

  CICS does not trace the initial input received on an MRO link.

  The normal trace entries (DD16, DD23, and DD25) are created with the text "`SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT`" replacing all the user data.

  CICS then identifies the target transaction for the data. If the transaction definition specifies CONFDATA(NO), CICS traces the user data that it suppressed from DD16 in the trace entry AP FC92. This special trace entry is not created if the transaction is defined with CONFDATA(YES).

- **FEPI**

  FEPI screens and RPL data areas (RPLAREAs) areas are suppressed from all FEPI trace points if CONFDATA(YES) is specified in the transaction resource definition. The user data in the FEPI trace points AP 1243, AP 1244, AP 145E, AP 145F, AP 1460, AP 1461, AP 1595, AP 1596, AP 1597, AP 1598, and AP 1599 is replaced with the message "`SUPPRESSED DUE TO CONFDATA=HIDETC IN SIT`". If the RDO TRANSACTION resource definition specifies CONFDATA(NO), the FEPI trace entries are created with the user data as normal.

Mirror transactions

The CICS-supplied mirror RDO TRANSACTION resource definitions are specified with CONFDATA(YES). This ensures that, when you specify CONFDATA=HIDETC as a system initialization parameter, CICS regions running mirror transactions suppress user data as described for VTAM and MRO data.

Modified data

By waiting until the transaction has been identified to determine the CONFDATA option, VTAM or MRO data may have been modified (for example, it may have been translated to upper case).

The interaction between the CONFDATA system initialization parameter and the CONFDATA attribute on the RDO TRANSACTION resource definition is shown in Table 11.

Table 11. Effect of CONFDATA system initialization and transaction definition parameters

CONFDATA on RDO TRANSACTION resource definition	CONFDATA system initialization parameter	
	SHOW	**HIDETC**
NO	Data not suppressed	VTAM RAIAs are cleared. Initial input of VTAM and MRO data is suppressed from the normal FC90, FC91, DD16, DD23, and DD25 trace entries. For FC90 and DD16 traces only, suppressed user data is traced separately in an FC92 trace entry. FEPI screens and RPLAREAs are traced as normal.
YES	Data not suppressed	VTAM RAIAs are cleared. All VTAM, MRO, and FEPI user data is suppressed from trace entries.

You cannot modify the CONFDATA option while CICS is running. You must restart CICS to make such a change.

Restrictions

You can code the CONFDATA parameter in the SIT, PARM, and SYSIPT only.

CONFTXT={NO|YES}

Specifies whether CICS is to prevent VTAM from tracing user data.

NO

CICS does not prevent VTAM from tracing user data.

YES

CICS prevents VTAM from tracing user data.

Restrictions

You can code the CONFTXT parameter in the SIT, PARM, and SYSIPT only.

CSDACC={READWRITE|READONLY}

Specifies the type of access to the CSD to be permitted to this CICS region. Note that this parameter is effective only when you start CICS with a START=COLD parameter. If you code START=AUTO, and CICS performs a warm or emergency restart, the file resource definitions for the CSD are recovered from the CICS global catalog. However, you can redefine the type of access permitted to the CSD dynamically with a CEMT SET FILE, or an EXEC CICS SET FILE, command.

READWRITE

Read/write access is allowed, permitting the full range of CEDA, CEDB, and CEDC functions to be used.

READONLY

Read access only is allowed, limiting the CEDA and CEDB transactions to only those functions that do not require write access.

CSDBUFND=number

Specifies the number of buffers to be used for CSD data. The minimum you should specify is the number of strings coded on the CSDSTRNO parameter plus 1, up to a maximum of 32768. Note that this parameter is used only if you have also coded CSDLSRNO=NONE; if you have coded CSDLSRNO=number, CSDBUFND is ignored.

If you specify a value for CSDBUFND that is less than the required minimum (the CSDSTRNO value plus 1), VSAM automatically changes the number of buffers to the number of strings plus 1 when CICS issues the OPEN macro for the CSD.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

CSDBUFNI=number

Specifies the number of buffers to be used for the CSD index. The minimum you should specify is the number of strings coded on the CSDSTRNO parameter, up to a maximum of 32768. This parameter is used only if you have also coded CSDLSRNO=NONE; if you have coded CSDLSRNO=number, CSDBUFNI is ignored.

If you specify a value for CSDBUFNI that is less than the required minimum (the CSDSTRNO value), VSAM automatically changes the number of buffers to the number of strings when CICS issues the OPEN macro for the CSD.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the CICS global catalog.

CSDFRLOG=number

Specifies a journal identifier to indicate the journal that you want to use for forward recovery of the CSD. This parameter is used only if CSDRECOV=ALL is specified, otherwise it is ignored. If you omit CSDFRLOG, but specify CSDRECOV=ALL, CSDFRLOG defaults to 1, which indicates that the CICS system log is to be used for forward recovery of the CSD.

The CSDRECOV, and CSDFRLOG system initialization parameters interact according to how they are specified. For information about their effects when the SIT is assembled and during CICS override processing, see theCICS System Definition Guide.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

number
The journal identification of the journal that is to be used for forward recovery. The number must be in the range 1 through 99. The number 1 indicates the CICS system log, and any other number refers to a user journal.

CSDJID={NO|number}
Specifies the journal identifier of the journal that you want CICS to use for automatic journaling of file requests against the CSD.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

NO
Code NO if you do ***not*** want automatic journaling for the CSD. This is the default.

number
A number in the range 1 through 99 to identify the journal that CICS is to use for automatic journaling for the CSD. The number 1 indicates the CICS system log, and any other number refers to another CICS journal.

The automatic journaling options enforced for the CSD when you code CSDJID=number are JNLADD=BEFORE and JNLUPDATE=YES. These options are sufficient to record enough information for a user-written forward recovery utility. No other automatic journaling options are available for the CSD. For information about the options JNLADD=BEFORE and JNLUPDATE=YES, see the CICS Resource Definition Guide.

CSDLSRNO={1|number|NONE|NO}
Specifies whether the CSD is to be associated with a local shared resource (LSR) pool.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog. However, you can redefine the LSR pool attribute for the CSD dynamically with an EXEC CICS SET FILE command.

1 The default LSR pool number is 1.

number
The number of the LSR pool the CSD is to be associated with. The number of the pool must be in the range 1 through 15.

NONE|NO
Code NONE (or NO) if the CSD is not to be associated with an LSR pool.

CSDRECOV={NONE|ALL|BACKOUTONLY}
Specifies whether the CSD is a recoverable file.

The CSDRECOV, and CSDFRLOG system initialization parameters interact according to how they are specified. For information about their effects when the SIT is assembled and during CICS override processing, see the CICS System Definition Guide.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog.

NONE
The default is that the CSD is not recoverable.

ALL
Code ALL to specify that you want both forward recovery and backout for the CSD. If you code ALL, also specify CSDFRLOG with the journal identification of the journal to be used for forward recovery of the CSD. If you do not code the CSDFRLOG parameter, CICS uses the system log for forward recovery.

Note: For backout purposes, CICS always uses the system log. (See the BACKOUTONLY option.)

BACKOUTONLY
Code BACKOUTONLY to limit CSD recovery to file backout only. If you specify backout for the CSD, CICS uses the system log to record before images for backout purposes.

CSDSTRNO={2|number}
Specifies the number of concurrent requests that can be processed against the CSD. When the number of requests reaches the STRNO value, CICS automatically queues any additional requests until one of the active requests terminates.

CICS requires two strings per CSD user, and you can increase the CSDSTRNO value, in multiples of two, to allow more than one concurrent CSD user.

See section “Multiple Users of the CSD Within a CICS Region” in the CICS System Definition Guide, before you code this parameter.

This parameter is effective only on a CICS cold start. On a warm or emergency restart, file resource definitions for the CSD are recovered from the global catalog. However, you can redefine the number of strings for the CSD dynamically with an EXEC CICS SET FILE command.

2 The minimum number of concurrent requests for the CSD is 2.

number
This number must be a multiple of 2, in the range 2 through 254.

CWAKEY={USER|CICS}
Specifies the storage key for the common work area (CWA) if you are operating CICS with storage protection (STGPROT=YES). (You specify how much storage you want for the CWA on the WRKAREA parameter.) The permitted values are USER (the default), or CICS:

USER If you specify USER, or allow this parameter to default, CICS obtains storage for the CWA in user key. This allows a user program executing in any key to modify the CWA.

CICS If you specify CICS, CICS obtains storage for the CWA in CICS key. This means that only programs executing in CICS key can modify the CWA, and user-key programs have read-only access.

If CICS is running without storage protection, the CWAKEY parameter is ignored, and the CWA is always allocated from CICS-key storage.

DATFORM={MMDDYY|DDMMYY|YYMMDD}
Specifies the external date display standard that you want to use for CICS date displays. An appropriate indicator setting is made in the CSA. It is examined by CICS supplied system service programs that display a Gregorian date. CICS maintains the date in the form 0CYYDDD in the CSA (where C=0 for years

19xx, 1 for years 20xx, and so on; YY=year of century; and DDD=day of year), and converts it to the standard you specify for display.

MMDDYY
The date is in the form of month-day-year.

DDMMYY
The date is in the form of day-month-year.

YYMMDD
The date is in the form of year-month-day.

DBP={1$|2$|xx|YES}
Specifies which version of the dynamic transaction backout program is to be part of the system. This DBP parameter is mandatory, and has no default. (See page 168 for more information about coding this parameter.)

If you are using local DL/I support, specify DBP=2$. If you have generated a user-defined DFHDBP program for use with local DL/I support, specify either DBP=xx, where xx is the suffix of your program, or DBP=YES if you have defined an unsuffixed version of DFHDBP.

If you are not using local DL/I support, specify DBP=1$.

For background information about dynamic transaction backout, see the CICS Recovery and Restart Guide.

The dynamic transaction backout program needs a program resource definition entry in the CSD. This entry must be included in your GRPLIST list at CICS initialization. The CICS-supplied programs DFHDBP1$ and DFHDBP2$ have resource definition entries in the CSD group DFHBACK, which is included in the default group list DFHLIST. If you use your own dynamic backout program, you must create a RDO PROGRAM resource definition entry for it in the CSD, and include the entry in one of your GRPLIST lists at CICS initialization. You are recommended to code RESIDENT(YES) on your RDO PROGRAM definition. For information about the required program resource definition entry, see the CICS Resource Definition Guide.

Restrictions

You must code a DBP parameter in the system initialization table, although you can override that DBP parameter in PARM, SYSIPT, or CONSOLE.

DBUFSZ={500|number}
Specifies the maximum size of the dynamic log buffer (needed for dynamic transaction backout) for each transaction. CICS initially allocates for each buffer half the maximum size that you specify, and subsequently uses the value specified to calculate the actual size of the dynamic buffer. The value can be in the range 6 through 32000; the default is 500.

If the resultant dynamic buffer is too small, CICS spills the dynamic log data to a further area of main storage. (CICS allocates all dynamic log storage, including spilled buffers, above the 16MB line.)

For information on the dynamic log, and how you can tune this parameter, see *Dynamic transaction backout statistics* and *Dynamic log buffer size (DBUFSZ)* in the CICS Performance Guide.

DCT=({YES|xx|NO}[,COLD])
Specifies the destination control table suffix, for more information see page 165. For information about coding the macros for this table, see the CICS Resource Definition Guide.

DFLTUSER={CICSUSER|userid}
Specifies the external security manager (ESM) userid with the security attributes to be used for all terminal users who have not explicitly signed on (by the CESN transaction, the EXEC CICS SIGNON command, or by the preset security options of the TERMINAL resource definition).

The specified userid must be defined to your ESM if you are using external security (that is, you have coded SEC=YES).

The specified userid is signed on during CICS initialization. If it cannot be signed on, CICS fails to initialize.

Restrictions

You can code the DFLTUSER parameter in the SIT, PARM, or SYSIPT only.

DIP={NO|YES}
Code YES to include the batch data interchange program, DFHDIP, which supports the batch controller functions of the IBM 3790 Communication System and the IBM 3770 Data Communication System, for more information see page 165. (Support is provided for the transmit, print, message, user, and dump data sets of the 3790 system.)

DISMACP={YES|NO}
DISMACP=YES allows CICS to disable any transaction that terminates abnormally with an ASRD abend (caused by a user program invoking a CICS macro, or referencing the CSA or the TCA).

Note: DISMACP=YES has no effect if the ASRD abend is handled by an active abend exit.

{DLI|DL1}=({NO|xx|YES}[,COLD])
Specifies whether DL/I databases are to be accessed during this run of CICS.

This parameter selects only DL/I support that runs within the CICS address space.

NO
DL/I is not to be used.

xx
DL/I is used. xx is the 1- or 2-character suffix of the DL/I application control table (ACT) specified on the DLZACT TYPE=INITIAL macro, which results in the module DLZNUCxx.

YES
DL/I is to be used. You must also specify DBP=2&#163;.

COLD
If you code DLI=YES, you can specify the COLD option if you want to cold start the DL/I resources The default is the start option coded on the START parameter. If you specify COLD, DL/I databases are not backed out in the event of an emergency restart.

Note: On initialization of an XRF alternate system, the DLI COLD option is overridden so that DL/I is emergency restarted, and so DL/I databases are backed out.

DLIOER={ABEND|CONTINUE}

Code the DLIOER parameter to tell CICS what to do in the event of DL/I detecting an I/O error on a database.

ABEND

CICS abends. In an XRF configuration, the alternate system is canceled. Global user exit, XDBDERR, is not invoked.

CONTINUE

CICS continues to run. DL/I flags the database as unusable and allows the use of DL/I utilities to perform database recovery and to restart the database.

DOCCODEPAGE={037|codepage}

specifies the default host code page to be used by the document domain. The *codepage* is a field of up to 8 characters. If *codepage* value is not specified, the default *doccodepage* is set to *037*.

DSALIM={5M|number}

Specifies the upper limit of the total amount of storage within which CICS can allocate the individual dynamic storage areas (DSAs) that reside below the 16MB boundary.

5M

The default DSA limit is 5MB (5 242 880).

number

Specify *number* as an amount of storage in the range 2MB to 16MB (2 097 152 bytes to 16 777 216 bytes) in multiples of 262 144 bytes (256KB). If the size specified is not a multiple of 256KB, CICS rounds the value up to the next multiple.

You can specify *number* in bytes (for example, 4 194 304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

From the storage size that you specify on the DSALIM parameter, CICS allocates the following dynamic storage areas:

The user DSA (UDSA)

The user-key storage area for all user-key task-lifetime storage below the 16MB boundary.

The read-only DSA (RDSA)

The key-0 storage area for all reentrant programs and tables below the 16MB boundary.

The shared DSA (SDSA)

The user-key storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing EXEC CICS GETMAIN commands for storage below the 16MB boundary with the SHARED option.

The CICS DSA (CDSA)

The CICS-key storage area for all non-reentrant CICS-key RMODE(24) programs, all CICS-key task-lifetime storage below the 16MB boundary, and for CICS control blocks that reside below the 16MB boundary.

Note: CICS allocates the UDSA and the other DSAs below 16MB in multiples of 256KB. The maximum you can specify depends on a number of factors, such as how you have configured your VSE storage (which governs how much private storage remains below the line) and how much private storage you must leave free to satisfy VSE GETVIS requests for storage outside the DSAs.

For information about calculating the amount of storage to specify on the DSALIM parameter, see the CICS Performance Guide.

DSHIPIDL={020000|hhmmss}

Specifies the minimum time, in hours, minutes, and seconds, that an *inactive* shipped terminal definition must remain installed in this region. When the timeout delete mechanism is invoked, only those shipped definitions that have been inactive for longer than the specified time are deleted.

You can use this parameter in a transaction routing environment, on the application-owning and intermediate regions, to prevent terminal definitions having to be reshipped because they have been deleted prematurely.

The default minimum idle time is 2 hours.

hhmmss
: Specify a 1 to 6 digit number in the range 0 995959. Numbers that have fewer than six digits are padded with leading zeros.

DSHIPINT={120000|0|hhmmss}

Specifies the interval between invocations of the timeout delete mechanism. The timeout delete mechanism removes any shipped terminal definitions that have not been used for longer than the time specified by the DSHIPIDL parameter.

You can use this parameter in a transaction routing environment, on the application-owning and intermediate regions, to control:

- How often the timeout delete mechanism is invoked.
- The approximate time of day at which a mass delete operation is to take place, relative to CICS startup.

  Note: For more flexible control over when mass delete operations take place, you can use a CEMT SET DELETSHIPPED or EXEC CICS SET DELETSHIPPED command to reset the interval. (The revised interval starts *from the time the command is issued*, **not** from the time the remote delete mechanism was last invoked, nor from CICS startup.)

0
: The timeout delete mechanism is not invoked. You might set this value in a terminal-owning region, or if you are not using shipped definitions.

hhmmss
: Specify a 1 to 6 digit number in the range 1 995959. Numbers that have fewer than six digits are padded with leading zeros.

DTRTRAN={CRTX|name|NO}

This is the name of the transaction definition that you want CICS to use for dynamic transaction routing. This is intended primarily for use in a CICS terminal-owning region, although you can also use it in an application-owning region when you want to daisy-chain transaction routing requests. In a dynamic transaction routing environment, the transaction named on DTRTRAN must be installed in the CICS terminal-owning regions if you want to eliminate the need for resource definitions for individual transactions.

CRTX
This is the default dynamic transaction definition. It is the name of the CICS-supplied sample transaction resource definition provided in the CSD group DFHISC.

name
The name of your own dynamic transaction resource definition that you want CICS to use for dynamic transaction routing.

NO
The dynamic transaction routing program is not invoked when a transaction definition cannot be found.

For information about the CICS-supplied sample transaction resource definition, CRTX, and about defining you own dynamic transaction routing definition, see the CICS Resource Definition Guide.

DTRPGM={DFHDYP|program-name}
Specifies the name of the dynamic transaction routing program you want to use for routing transactions that are defined with the DYNAMIC attribute. DFHDYP, the default, is the name of the CICS-supplied version.

DUMP={YES|NO}
Specifies whether the CICS dump domain is to take SDUMPs.

YES
SDUMPs are produced, unless suppressed by the options specified in the CICS system dump table.

NO
SDUMPs are suppressed.

Note: This does not prevent the CICS kernel from taking SDUMPs.

For more information about SDUMPs, see the CICS System Definition Guide.

DUMPDS={AUTO|A|B}
Specifies the transaction dump data set that is to be opened during CICS initialization.

AUTO
For all types of start, CICS opens the transaction dump data set that was ***not*** in use when the previous CICS run terminated. This information is obtained from the CICS local catalog (DFHLCD).

If you specify AUTO, or let it default, code DLBL statements for both of the transaction dump data sets, DFHDMPA and DFHDMPB, in your CICS startup job stream.

A CICS opens transaction dump data set DFHDMPA.

B CICS opens transaction dump data set DFHDMPB.

DUMPSW={NO|NEXT}
Specifies whether you want CICS to switch automatically to the next dump data set when the first is full.

NO
Disables the CICS autoswitch facility. If the transaction dump data set opened during initialization becomes full, CICS issues a console message

to notify the operator. If you want to switch to the alternate data set, you must do so manually using the CEMT or EXEC CICS SET DUMPDS SWITCH command.

NEXT

Enables the autoswitch facility to switch to the next data set at end of file of the data set opened during initialization. Coding NEXT permits one switch only. If you want to switch to the alternate data set again, you must do so manually using CEMT or EXEC CICS SET DUMPDS SWITCH command. If you specify NEXT, code DLBL statements for both of the transaction dump data sets, DFHDMPA and DFHDMPB, in your CICS startup job stream.

For more information about transaction dump data sets, see the CICS System Definition Guide

ECDSASZE={0K|number}

Specifies the size of the ECDSA. The default size is 0 indicating that the DSA size can be changed dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1MB). If the size specified is not a multiple of 1MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

EDSALIM={20M|number}

Specifies the upper limit of the total amount of storage within which CICS can allocate the individual extended dynamic storage areas (EDSAs) that reside above the 16MB boundary.

20M

The default EDSA limit is 20MB (20 971 520 bytes).

number

Specify *number* as a value in the range 10MB to 2047MB, in multiples of 1MB. If the size specified is not a multiple of 1MB, CICS rounds the value up to the next multiple.

You can specify *number* in bytes (for example, 33 554 432), or as a whole number of kilobytes (for example, 32 768K), or a whole number of megabytes (for example, 32M).

The maximum value allowed depends on a number of factors, such as:

- The size of the CICS partition
- How much storage you require for the CICS internal trace table.
- How much private storage you must leave free to satisfy VSE GETVIS requests for storage above the 16MB boundary outside the DSAs.

From the storage value that you specify on the EDSALIM parameter, CICS allocates the following extended dynamic storage areas:

The extended user DSA (EUDSA)

The user-key storage area for all user-key task-lifetime storage above the 16MB boundary.

The extended read-only DSA (ERDSA)
The key-0 storage area for all reentrant programs and tables above the 16MB boundary.

The extended shared DSA (ESDSA)
The user-key storage area for any non-reentrant user-key RMODE(ANY) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage above the 16MB boundary with the SHARED option.

The extended CICS DSA (ECDSA).
The CICS-key storage area for all non-reentrant CICS-key RMODE(ANY) programs, all CICS-key task-lifetime storage above the 16MB boundary, and CICS control blocks that reside above the 16MB boundary.

CICS allocates all the DSAs above the 16MB boundary in multiples of 1MB.

Note: For information about calculating the amount of storage to specify on the EDSALIM parameter, see the CICS Performance Guide. For example, the value that you specify must allow for all temporary storage MAIN requests, for which CICS uses the ECDSA.

ENCRYPTION={WEAK|NORMAL|STRONG}
Specifies the level of encryption you want to use for TCP/IP connections using the secure sockets layer. The parameter selects the list of ciphers that are negotiated with the client program to choose the SSL encryption technique, key size, and message authentication code (MAC). You can specify an option only if you have the underlying encryption support in the VSE/ESA operating system. Possible values are:

WEAK
Available with VSE/ESA V2R6 and later releases. It specifies the following list of ciphers:
- No encryption with an MD5 MAC
- No encryption with an SHA MAC.

NORMAL
This is the default value. NORMAL encryption is available with VSE/ESA V2R6 and later releases. It specifies the following list of ciphers:
- DES encryption with a 56-bit key and an SHA MAC
- No encryption with an MD5 MAC
- No encryption with an SHA MAC.

STRONG
This is available with VSE/ESA V2R6 and later releases. It specifies the following list of ciphers:
- Triple DES encryption with a 168-bit key and an SHA MAC
- DES encryption with a 56-bit key and an SHA MAC
- No encryption with an MD5 MAC
- No encryption with an SHA MAC.

STRONG encryption is available outside the United States to organizations that have been authorized by the United States government.

EODI={E0|xx}
Specifies the end-of-data indicator for input from sequential devices. The characters “xx” represent two hexadecimal digits in the range 01 through FF. The default value is X'E0', which represents the standard EBCDIC backslash symbol (\).

ERDSASZE={0K|number}

Specifies the size of the ERDSA. The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1MB). If the size specified is not a multiple of 1MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4MB).

ESDSASZE={0K|number}

Specifies the size of the ESDSA. The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1MB). If the size specified is not a multiple of 1MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

ESMEXITS={NOINSTLN|INSTLN}

Specifies whether installation data is to be passed via the RACROUTE interface to the external security manager (ESM) for use in exits written for the ESM.

NOINSTLN

Specifies that the INSTLN parameter is not used in RACROUTE macros.

INSTLN

Specifies that CICS-related and installation-supplied data is passed to the ESM using the INSTLN parameter of the RACROUTE macro. For programming information, including the format of the data passed, see the CICS Customization Guide. This data is intended for use in exits written for the ESM.

Restrictions

You can code the ESMEXITS parameter in the SIT only.

EUDSASZE={0K|number}

Specifies the size of the EUDSA. The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 1073741824 bytes in multiples of 1048576 bytes (1MB). If the size specified is not a multiple of 1MB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4194304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

FCT={YES|xx|NO}

Specifies the suffix of the file control table (FCT) to be used.

This parameter is effective only on a CICS cold start. CICS does not load an FCT on a warm or emergency restart, and all file resource definitions are recovered from the CICS global catalog.

For information about coding the macros for this table, see the CICS Resource Definition Guide.

You can use a mixture of macro definitions and RDO definitions for files in your CICS region. However, your FCT should not contain definitions for files that are also defined in an RDO group specified in the group list for the CICS startup, and the converse. In a cold start, CICS first loads the file definitions from the FCT, and adds any files that are defined in RDO groups later (using the groups specified in the GRPLIST parameter). If CICS tries to install a file definition from a group list for a file already defined by an entry from the FCT, the install may fail because of conflicting file attributes. You can avoid this potential conflict by ensuring that you do not have duplicate file entries in the FCT and in your group list.

FEPI={NO|YES}

Specifies whether or not you want to use the Front End Programming Interface feature (FEPI).

NO

Means that FEPI support is not required. You should specify NO on this parameter (or allow it to default) if you do not have the feature installed, or if you do not require FEPI support.

YES

Means you require FEPI support, and causes CICS to start the CSZI transaction.

This book does not contain any information about the installation process for the Front End Programming Interface feature. Installation information can be found in the CICS Front End Programming Interface User's Guide.

FLDSEP={' '|'xxxx'

Specifies 1 through 4 field-separator characters, each of which indicates end of field in the terminal input data. The default is four blanks.

The field separator allows you to use transaction identifications of less than four characters followed by one of the separator characters. When less than four characters are coded, the parameter is padded with blanks, so that the blank is then a field separator. None of the specified field separator characters should be part of a transaction identification; in particular, the use of alphabetic characters as field separators is not recommended.

The character specified in the FLDSEP parameter must not be the same as any character specified in the FLDSTRT parameter. This means that it is invalid to allow both parameters to take the default value.

> **Restrictions**
>
> If you specify FLDSEP in the SIT, the characters must be enclosed in single quotation marks.
>
> If you specify FLDSEP as a PARM, SYSIPT, or CONSOLE parameter, do ***not*** enclose the characters in quotation marks, and the characters you choose must not include an embedded blank, or any of these characters:
>
> `( ) ' = ,`

FLDSTRT={‘ ’|‘x’}

Specifies a single character to be the field-name-start character for free-form input for built-in functions. The default is a blank.

The character specified should not be part of a transaction identification; in particular, the use of alphabetic characters is not recommended.

The character specified in the FLDSTRT parameter must not be the same as any character specified in the FLDSEP parameter. This means that it is invalid to allow both parameters to take the default value.

> **Restrictions**
>
> If you specify FLDSTRT in the SIT, the parameter must be enclosed in single quotation marks.
>
> If you specify FLDSTRT as a PARM, SYSIPT, or CONSOLE parameter, do ***not*** enclose the character in quotation marks, and the character you choose must not be a blank or any of the following characters:
>
> `( ) ' = ,`

FSSTAFF={YES|NO}

specify this parameter in an application-owning region (AOR) to prevent transactions initiated by function-shipped EXEC CICS START requests being started against incorrect terminals.

You may need to code the function-shipped START affinity (FSSTAFF) parameter in an AOR if all of the following are true:

1. The AOR is connected to two or more terminal-owning regions (TORs) that use the same, or a similar, set of terminal identifiers.
2. One or more of the TORs issues EXEC CICS START requests for transactions in the AOR.
3. The START requests are associated with terminals.
4. You are using shippable terminals, rather than statically defining remote terminals in the AOR.

Consider the following scenario:

Terminal-owning region TOR1 issues an EXEC CICS START request for transaction TRAR, which is owned by region AOR1. It is to be run against terminal T001. Meanwhile, terminal T001 on region TOR2 has been transaction routing to AOR1; a definition of T001 has been shipped to AOR1 from TOR2. When the START request arrives at AOR1, it is shipped to TOR2, rather than TOR1, for transaction routing from terminal T001.

To prevent this situation, code `YES` on the FSSTAFF parameter in the AOR.

YES When a START request is received from a terminal-owning region, and a shipped definition for the terminal named on the request is already installed in the AOR, the request is always shipped back to a TOR, for routing, *across the link it was received on*, unless the request supplies a TOR_NETNAME and a remote terminal with the correct TOR_NETNAME is located.

If the TOR to which the START request is returned is **not** the one referenced in the installed remote terminal definition, a definition of the terminal is shipped to the AOR, and the autoinstall user program is called. Your autoinstall user program can then allocate an *alias* termid in the AOR, to avoid a conflict with the previously installed remote definition. For information about writing an autoinstall program to control the installation of shipped definitions, see the CICS Customization Guide.

NO When a START request is received from a terminal-owning region, and a shipped definition for the named terminal is already installed in the AOR, the request is shipped to the TOR referenced in the definition, for routing.

Notes:

1. FSSTAFF has no effect:
   - On statically-defined (hard-coded) remote terminal definitions in the AOR. If you use these, START requests are always shipped to the TORs referenced in the definitions.
   - On START requests issued in the local region. It affects only START requests shipped from other regions.
   - When coded on intermediate regions in a transaction-routing path. It is effective only when coded on an application-owning region.
2. If the AOR contains no remote definition of a terminal named on a shipped START request, the “terminal not known” global user exits, XICTENF and XALTENF, are called. For details of these exits, see the CICS Customization Guide.

GMTEXT={‘WELCOME TO CICS’|‘text’}

Specifies whether the default logon message text (WELCOME TO CICS) or your own message text is to be displayed on the screen by the CSGM (good morning) transaction when a terminal is logged on to CICS through VTAM, by the CESN transaction if used to sign on to CICS, or by your own transactions using the EXEC CICS INQUIRE SYSTEM GMMTEXT command.

You can use apostrophes to punctuate your message, in addition to using them as message delimiters. However, you must code *two* successive apostrophes to represent a single apostrophe in your text. For example,

```
GMTEXT='User''s logon message text.'
```

The whole message must still be enclosed by a pair of single delimiting apostrophes.

Your message text can be from 1 through 246 characters (bytes), and can extend over two lines by extending the text to column 80 on the first line, and continuing in column 1 of the second line. For example, the following might be used in the SYSIPT data set:

```
GMTEXT='An Information Development CICS Terminal-Owning Region (TOR) - CICSIDC.
This message is to show the use of continuation lines when creating a GMTEXT par
ameter in the SYSIPT data set'
```

The CSGM transaction displays this as follows (with the time appended to the end of message):

```
An Information Development CICS Terminal-Owning Region (TOR) - CICSIDC. This me
ssage is to show the use of continuation lines when creating a GMTEXT parameter
in the SYSIPT data set 09:56:14
```

The CESN transaction displays this as follows:

```
                Signon to CICS                                    APPLID CICSHTH1

An Information Development CICS Terminal-Owning Region (TOR) - CICSIDC.
This message is to show the use of continuation lines when creating a GMTEXT
parameter in the SYSIPT data set
```

For any transaction other than CESN that displays the text specified by this parameter, you must use a RDO TYPETERM with LOGONMSG(YES) for all terminals requiring the logon message. For information about using RDO TYPETERM, see the CICS Resource Definition Guide.

GMTRAN={CSGM|CESN|transaction-id}

Specifies the name of the transaction that is initiated when terminals are logged on to CICS by VTAM. Do not specify the name of a remote transaction. The transaction must be capable of being automatically initiated (ATI). The default is the transaction CSGM, that displays the text specified in the GMTEXT parameter. Alternatively, you can specify the CICS signon transaction, CESN, which also displays the text specified in the GMTEXT parameter. The GMTRAN parameter can be used with the LGNMSG parameter to retrieve VTAM logon data.

GNTRAN={CESF|transaction_id}

Specifies the transaction that you want CICS to invoke when a user’s terminal-timeout period expires.

CESF

The default, CESF, is the basic CICS signoff transaction without any options. This transaction attempts to sign off a terminal, subject to the SIGNOFF attribute of the RDO TYPETERM resource definition for that terminal.

transaction_id

The name of an alternative timeout transaction to signoff the user at the timed-out terminal. Specifying your own transaction allows you to specify functions in addition to, or instead of, signoff. For example, your own transaction could issue a prompt for the terminal user’s password, and allow the session to continue if the correct password is entered.

The transaction to be used must have been specially written to handle the GNTRAN commarea that is passed to it. Of the CICS-supplied transactions, only CESF has been written to handle the GNTRAN commarea. For more information about writing your own transactions for GNTRAN, see the CICS Customization Guide.

Note: When either the default CESF transaction, or your own transaction, attempts to sign off a terminal, the result is subject to the SIGNOFF attribute of the RDO TYPETERM resource definition for the terminal, as follows:

SIGNOFF
Effect

YES The terminal is signed off, but not logged off.

NO The terminal remains signed on and logged on.

LOGOFF
The terminal is both signed off and logged off.

Note: If GNTRAN fails to attach, and SIGNOFF(LOGOFF) has been specified, the terminal which has reached timeout will be signed off and logged off. GNTRAN will not run and will have no effect.

GRPLIST={DFHLIST|name|(name[,name2][,name3][,name4])}
Specifies the names (each 1 through 8 characters) of up to four lists of resource definition groups on the CICS system definition (CSD) file. The resource definitions in all the groups in the specified lists are loaded during initialization when CICS performs a cold start. If a warm or emergency start is performed, the resource definitions are derived from the CICS global catalog, and the GRPLIST parameter is ignored.

Each name can be either a real group list name or a generic group list name that incorporates global filename characters (+ and *). If you specify more than one group list (either by specifically coding two or more group list names or by coding a group list name with global filename characters), the later group lists are concatenated onto the first group list. Any duplicate resource definitions in later group lists override those in earlier group lists.

Use the CEDA command LOCK to protect the lists of resource groups specified on the GRPLIST parameter.

The default is DFHLIST, the CICS-supplied list that specifies the set of resource definitions needed by CICS. If you create your own group list, either add to it the groups specified in DFHLIST (omitting only those for CICS functions that you know you do not need) or specify the DFHLIST name on the GRPLIST parameter. Do not code GRPLIST=NO unless you have a group list named NO.

Notes:

1. Group lists specified by a generic group list name are concatenated in alphabetic then numeric order. For example, the generic list name CICSHT*, would concatenate the group lists CICSHT#1, CICSHTAP, CICSHTSD, and CICSHT3V in that order. If the order of concatenation is important (for example, to ensure that a particular resource definition overrides another), you should consider coding real group list names.
2. If a group list contains resource definitions that are needed by another group list, the prequisite group list must be installed first. For example, if list A has TYPETERM definitions needed for TERMINAL definitions in list B, list A must be installed first. This may mean that you have to specifically name the prerequisite group on the GRPLIST parameter.
3. Take care when using generic group list names because, if a group list on your CSD satisfies the generic name, it will be installed. This means that a group list can be installed more than once; for example, if you specify the real group list name and a generic group list name that it satisfies, or if you specify two generic group list names that the group list name satisfies.

4. To override one or more of the group lists specified on the GRPLIST system initialization parameter, you must specify all list names (both real and generic) that you want to use, even if you are not changing the names.

For example, if you want to use the four group lists CICSHT#1, CICSHTAP, CICSHT3V, and CICSHTSD, you could specify either of the following system initialization parameters:

```
GRPLIST=(CICSHT#1,CICSHTAP,CICSHT3V,CICSHTSD)
GRPLIST=(CICSHT*)
```

In the first example GRPLIST, the group lists are loaded in the order specified, and resource definitions installed from the CICSHTSD group list will override any duplicate definitions installed by the other groups.

In the second example GRPLIST, the group lists are loaded in the order CICSHT#1, CICSHTAP, CICSHTSD, then CICSHT3V, and resource definitions installed from the CICSHT3V group list will override any duplicate definitions installed by the other groups.

If your SIT contains the parameter:

```
GRPLIST=(CICSHT#1,CICSAP*,CICSHT3V,CICSHTSD)
```

and you want to replace the list CICSHT3V with the list ANOLST05, you should specify the override:

```
GRPLIST=(CICSHT#1,CICSAP*,ANOLST05,CICSHTSD)
```

In general, any required resource definitions should appear in *one of* the group lists specified on the GRPLIST system initialization parameter.

For information about resource definitions, groups, lists, and the CSD, see the CICS Resource Definition Guide.

ICP=COLD

Code this parameter if you want to cold start the interval control program, for more information see page 165.

ICV={1000|number}

Code this parameter with the region exit time interval in milliseconds. The ICV system initialization parameter specifies the maximum time in milliseconds that CICS releases control to the operating system when there are no transactions ready to resume processing. This time interval can be any integer in the range 100 through 3600000 milliseconds (specifying an interval up to 60 minutes). A typical range of operation might be 100 through 2000 milliseconds.

A low value interval can enable much of the CICS nucleus to be retained in dynamic storage, and not be paged-out at times of low terminal activity. This reduces the amount of dynamic storage paging necessary for CICS to process terminal transactions (thus representing a potential reduction in response time), sometimes at the expense of concurrent batch region throughput. Large networks with high terminal activity are inclined to run CICS without a need for this value, except to handle the occasional, but unpredictable, period of inactivity. These networks can usually function with a large interval (10000 to 3600000 milliseconds). Once a task has been initiated, its requests for terminal services and the completion of the services are recognized by the system and this maximum delay interval is overridden.

Small systems, or those with low terminal activity, are subject to paging introduced by other jobs running in competition with CICS. By specifying a low value interval, key portions of the CICS nucleus are referenced more frequently, thus reducing the probability of these pages being paged-out. However, the execution of the logic without performing productive work might be considered wasteful. The need to increase the probability of residency by frequent but unproductive referencing must be weighed against the overhead and response time degradation incurred by allowing the paging to occur. By increasing the interval size, less unproductive work is performed at the expense of performance if paging occurs during the periods of CICS activity. For information about the effect of ICV on performance, see the CICS Performance Guide.

Note: The region exit time interval process contains a mechanism to ensure that CICS does not constantly set and cancel timers (thus degrading performance) while attempting to meet its objectives for a low region exit time interval. This mechanism can cause CICS to release control to the operating system for up to 0.5 seconds when the interval has been set at less than 250; and up to 0.25 seconds more than the region exit time interval when the interval has been set greater than 250.

ICVR={5000|number}

Specifies the default runaway task time interval in milliseconds as a decimal number. You can code zero, or a number in the range 500 through 2 700 000, in multiples of 500. CICS rounds down values that are not multiples of 500. This is the RUNAWAY interval used by transactions defined with RUNAWAY(SYSTEM) (see the CICS Resource Definition Guide). CICS may purge a task if it has not given up control after the RUNAWAY interval for the transaction (or ICVR if the transaction definition specified RUNAWAY(SYSTEM)). If you code ICVR=0, runaway task control is inoperative for transactions specifying RUNAWAY(SYSTEM) in their RDO TRANSACTION definition (that is, tasks do not get purged if they appear to be looping). The ICVR value is independent of the ICV value, and can be less than the ICV value. Note that CICS runaway task detection is based upon task time, that is, the interval is decremented only when the task has control of the processor. For information about commands that reinitialize the ICVR value, see the CICS Problem Determination Guide.

ICVTSD={500|number}

The terminal scan delay facility determines how quickly CICS deals with some terminal I/O requests made by applications. The range is 0 through 5000 milliseconds, with a default of ICVTSD=500.

There is an overhead in dealing with such requests. By specifying a nonzero value, the overhead may be spread over several transactions. A value close to zero (for example 200) would be adequate.

INITPARM=(pgmname_1=‘parmstring_1’[, ... ,pgmname_n=‘parmstring_n’])

Code this parameter in order to pass parameters to application programs that use the EXEC CICS ASSIGN INITPARM command. For example, you can use INITPARM to pass parameters to PLTPI programs to be executed in the final stages of system initialization. The area giving access to the parameters is specified by the EXEC CICS ASSIGN INITPARM command. For programming information about the EXEC CICS ASSIGN INITPARM command, see the CICS Application Programming Reference manual.

pgmname

Code pgmname with the name of a program. This name must be 1 through 8 alphanumeric or national language characters.

parmstring
Code parmstring with the parameter string (up to 60 characters enclosed by single quotation marks) to be passed to the associated program. Any quotation marks imbedded in the string must be duplicated. For information on coding INITPARM in the SYSIPT data set, see “Rules for coding parameters at the console” on page 165.

You can specify up to 255 pgmname=‘parmstring’ sets.

INTTR={ON|OFF}
Code this parameter to specify whether the internal CICS trace destination is to be activated at system initialization.

This parameter controls whether any of the three types of CICS trace entry are written to the internal trace table. The three types are: CICS system trace (see the SYSTR parameter), user trace (see the USERTR parameter), and exception trace entries (which are always made and not controlled by a system initialization parameter).

ON Activate main storage trace.

OFF Do not activate main storage trace.

IRCSTRT={NO|YES}
Code this parameter to indicate whether IRC is started up at system initialization. If IRCSTRT=YES is not coded, IRC can be initialized by issuing a CEMT or EXEC CICS SET IRC OPEN command.

ISC={NO|YES}
Code YES to include the CICS programs required for interregion or intersystem communication.

JCT={YES|xx|NO}
Code this parameter with the suffix of the journal control table to be used, for more information see page 165. This indicates whether journaling and volume control are to be used. Note that you must have journaling if you code XRF=YES. If you code JCT=NO, a dummy journal control program (DFHJCPDY) is loaded.

For information about coding the macros for this table, see the CICS Resource Definition Guide.

JSTATUS=RESET
Code JSTATUS=RESET to reset the journal status of all journal data sets that are defined on disk (JTYPE=DISK1|DISK2), but are ***not*** specified with JOUROPT=AUTOARCH.

This parameter resets the journal status held in the global catalog to “ready for use”. Before using the JSTATUS option at startup, ensure that any disk journals that contain essential information are archived, either to tape or another disk data set.

Restrictions

You can code the JSTATUS parameter in PARM, SYSIPT, or CONSOLE only. It is not applicable if you are using the CICS automatic archiving facility.

KEYFILE=*name*
Specifies the name of the VSE Keyring Library used for storing keys and certificates. *Your TCP/IP and CICS regions must be authorized to read the VSE*

Keyring Library specified here. The IBM-supplied VSE Keyring Library has the name CRYPTO.KEYRING. Therefore the KEYFILE parameter would be coded as:

```
KEYFILE=CRYPTO.KEYRING
```

The *library* name (in the above example, `CRYPTO`) can have a maximum of 7 characters. For further details, refer to chapter ″Configuring Your VSE/ESA Host for SLL″ in the *VSE/ESA e-business Connectors, User’s Guide*, SC33-6719.

LEVSE={YES|NO}

Specifies whether you want to initialize the language environment under CICS Transaction Server for VSE/ESA. If you specify NO, you will not be able to execute and PL/I for VSE/ESA, COBOL for VSE/ESA or C for VSE/ESA programs during this execution of CICS.

LGNMSG={NO|YES}

Code this to indicate whether VTAM logon data is made available to an application program.

NO

VTAM logon data is not available to an application program.

YES

VTAM logon data is available to an application program. The data can be retrieved with an EXEC CICS EXTRACT LOGONMSG command. For programming information about this command, see the CICS Application Programming Reference manual.

You can use this parameter with the GMTRAN parameter to retrieve the VTAM logon data at the time a terminal is logged on to CICS by VTAM.

MCT={NO|YES|xx}

Code this parameter to specify the monitoring control table suffix, for more information see page 165.

If you specify MCT=NO, CICS monitoring builds dynamically a default MCT, ensuring that default monitoring control table entries are always available for use when monitoring is on and a monitoring class (or classes) is active.

For information about coding the macros for this table, see the CICS Resource Definition Guide.

MN={OFF|ON}

Code this parameter to indicate whether monitoring is to be switched on or off at initialization, and use the individual monitoring class parameters to control which monitoring classes are to be active. (See the MNEXC, and MNPER parameter descriptions.) The default status is that the CICS monitoring facility is ***off***. The monitoring status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Switch off monitoring.

ON Switch on monitoring. However, unless at least one individual class is active, no monitoring records are written. For details of the effect of monitoring status being on or off, in conjunction with the status of the various monitoring classes, see the following notes:

Note: If the monitoring status is ON, CICS accumulates monitoring data continuously. For the performance and exception monitoring classes, CICS writes the monitoring data for each class that is active to a CICS Data Management Facility (DMF) data set.

If the monitoring status is OFF, CICS does not accumulate or write any monitoring data, even if any of the monitoring classes are active.

You can change the monitoring status and the monitoring class settings at any time, as follows:

- During a warm restart by coding an MN system initialization parameter in PARM, SYSIPT, or through the system console.
- While CICS is running, by either of:
  - The CEMT SET MONITOR command
  - The EXEC CICS SET MONITOR command.

When you change the status of monitoring, the change takes effect immediately. If you change the monitoring status from OFF to ON, monitoring starts to accumulate data and write monitoring records to DMF for all tasks that start after the status change is made ***for all active monitoring classes***.

If the status is changed from ON to OFF, monitoring stops writing records immediately and does not accumulate monitoring data for any tasks that start after the status change is made.

The monitoring status operand can be manipulated independently of the class settings. This means that, even if the monitoring status is OFF, you can change the monitoring class settings and the changes take effect for all tasks that are started after the monitoring status is next set to ON.

For programming information about controlling CICS monitoring, see the CICS System Programming Reference manual.

MNCONV={NO|YES}

Specifies whether or not conversational tasks are to have separate performance class records produced for each pair of terminal control I/O requests.

Any clock (including user-defined) that is active at the time such a performance class record is produced is stopped immediately before the record is written. After the record is written, such a clock is reset to zero and restarted. Thus a clock whose activity spans more than one recording interval within the conversational task appears in multiple records, each showing part of the time, and the parts adding up to the total time the clock is active. The high-water-mark fields (which record maximum levels of storage used) are reset to their current values. All other fields are set to X'00', except for the key fields (transid, termid). The monitoring converse status is recorded in the CICS global catalog for use during warm and emergency restarts.

MNEXC={OFF|ON}

Code this parameter to indicate whether the monitoring exception class is to be made active during initialization. The monitoring exception class status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Set the exception monitoring class to “not active”.

ON Set the exception monitoring class to “active”.

For programming information about exception monitoring records, see the CICS Customization Guide.

MNFREQ={0|hhmmss}

Specifies the interval for which CICS automatically produces a transaction performance class record for any long-running transaction. The monitoring frequency value is recorded in the CICS global catalog for use during warm and emergency restarts.

0 means that no frequency monitoring is active.

hhmmss
is the interval for which monitoring produces automatically a transaction performance class record for any long-running transaction. Specify a 1 to 6 digit number in the range 001500–240000. Numbers that are fewer than six digits are padded with leading zeroes.

MNPER={OFF|ON}
Code this parameter to indicate whether the monitoring performance class is to be made active during CICS initialization. The monitoring performance class status is recorded in the CICS global catalog for use during warm and emergency restarts.

OFF Set the performance monitoring class to “not active”.

ON Set the performance monitoring class to “active”.

For programming information about performance monitoring records, see the CICS Customization Guide.

MNSYNC={NO|YES}
Specifies whether or not you want CICS to produce a transaction performance class record when a transaction takes an implicit or explicit syncpoint (unit-of-work). No action is taken for syncpoint rollbacks. The monitoring syncpoint status is recorded in the CICS global catalog for use during warm and emergency restarts.

MNTIME={GMT|LOCAL}
Specifies whether you want the time stamp fields in the performance class monitoring data to be returned to an application using the EXEC CICS COLLECT STATISTICS MONITOR(taskno) command in either GMT or local time. The monitoring time value is recorded in the CICS global catalog for use during warm and emergency restarts.

For programming information on the EXEC CICS COLLECT STATISTICS command, see the CICS System Programming Reference.

MROBTCH={1|number}
Code this parameter to specify the number of events that must occur before CICS is posted for dispatch due to the batching mechanism. The number can be in the range 1 through 255, and the default is 1.

Use this batching mechanism to spread the overhead of dispatching CICS over several tasks. If the value is greater than 1 and CICS is in a system wait, CICS is not posted for dispatch until the specified number of events has occurred. Events include MRO requests from connected systems or DASD I/O. For these events, CICS is dispatched as soon as one of the following occurs:

- The current batch fills up (the number of events equals MROBTCH).
- An ICV interval expires.

Therefore, ensure that the time interval you specify in the ICV parameter is low enough to prevent undue delay to the system.

If CICS is dispatched for another reason, the current batch is dealt with in that dispatch of CICS.

Note: During periods of low utilization, a value greater than 1 specified on the MROBTCH parameter may result in increased transaction response times. Transactions issuing file I/O requests may be delayed due to increased FCIOWAIT. For further guidance information about the effect of MROBTCH on performance, see the CICS Performance Guide.

MROFSE={NO|YES}
specifies whether you want to extend the lifetime of the long-running mirror to keep it allocated until the end of the task rather than after a user syncpoint for Function Shipping applications.

NO The lifetime of the MRO long-running mirror is not extended.

YES The mirror task remains available to the application until the end of the application's task. This extended long-running mirror saves the overhead of re-attaching the mirror task following a user syncpoint.

This parameter is ignored for DPL requests (that is a DPL causes the session to be freed at the next syncpoint even if is has been kept for a previous sequence of syncpoints).

It should be used with caution. For additional information, see the Long Running Mirror sections of the CICS Intercommunication Guide and the CICS Performance Guide.

MROLRM={NO|YES}
Code this parameter to specify whether you want to establish an MRO long-running mirror task.

NO
The MRO long-running mirror task is not required.

YES
The mirror transaction remains available to the application issuing the remote request. This long-running mirror saves the overhead of re-establishing communication with the mirror transaction if the application makes more function shipping requests in this unit of work.

For information about long-running mirror tasks, see the CICS Intercommunication Guide.

MSGCASE={MIXED|UPPER}
CICS messages handled by the CICS message domain are in mixed case. Code the MSGCASE parameter to indicate how you want the message domain to display these mixed case messages.

MIXED
This is the default in the SIT; all messages displayed by the CICS message domain remain in mixed case,

UPPER
The message domain displays all mixed case messages in uppercase only.

Note: Mixed case output is not displayed correctly on Katakana display terminals and printers. Uppercase English characters appear correctly as uppercase English characters, but lowercase appears as Katakana symbols. If you have any Katakana terminals connected to your CICS region, specify MSGCASE=UPPER.

MSGLVL={1|0}

Code this parameter with the message level that controls the generation of messages to the console.

1 All messages are to be printed.

0 Only critical errors or interactive messages are to be printed.

MXT={5|number}

Specifies the maximum number, in the range 1 through 999, of ***user*** tasks CICS allows to exist at any time. CICS queues requests for tasks above this number but does not action (attach) them until the number of tasks attached drops below the MXT limit.

Note: The MXT value does **not** include CICS system tasks.

NATLANG=(E,x,y,z,...)

Specify on this parameter the single-character codes for the languages to be supported in this CICS run. For more information see Figure 15 on page 119.

E Code E for English, which is the *system* default (that is, is provided even if you do not specifically code E).

x,y,z,...
Code the appropriate letters for the other supported languages that you require.

For the codes that you specify on this parameter, you must ensure that a DFHMET1x module (where x is the language code) is in a sublibrary in the LIBDEF PHASE,SEARCH chain for the CICS job.

English language support is provided, even if you do not specifically code E for English.

The first language code specifies the default language for those elements of CICS enabled to receive NLS messages, such as some destinations used for CICS messages, and the terminals or users not signed-on with an NLS code. The other language codes are provided to specify the language to be used for messages sent to terminals that are defined with the appropriate language support code.

For example, coding NATLANG=(G,C,K) has the same effect as coding NATLANG=(G,C,E,K); that is, in both cases the default NLS language is German (G), and the languages English, Chinese (C), and Kanji (K) are supported.

CICS console messages are available in English and German only.

NEWSIT={YES|NO}

Specify NEWSIT=YES to cause CICS to load the specified SIT, and enforce the use of all system initialization parameters, modified by any system initialization parameters provided via PARM, SYSIPT, or the system console, even in a warm start. Enforcing the use of system initialization parameters in this way overrides any parameters that may have been stored in a warm keypoint at shutdown.

However, there are some exceptions; the following system initialization parameters are always ignored in a warm start, even if they are supplied via PARM, SYSIPT, or the console:

CSDACC

CSDBUFND
CSDBUFNI
CSDFRLOG
CSDJID
CSDLSRNO
CSDRECOV
CSDSTRNO
FCT
GRPLIST

In a warm restart CICS uses the ***installed*** resource definitions saved in the CICS global catalog at warm shutdown, and therefore the CSD, FCT, and GRPLIST parameters are ignored. (At CICS startup, you can only modify installed resource definitions, including file control table entries, or change to a new FCT, by performing a cold start of CICS with START=COLD.)

For more information about the use of the NEWSIT parameter, see “Classes of start and restart” on page 169.

Restrictions

You can code the NEWSIT parameter in PARM, SYSIPT, or CONSOLE only.

OPERTIM={120|number}

Code this parameter with the write-to-operator timeout value, in the range 0 through 86400 seconds (24 hours). This is the maximum time (in seconds) that CICS waits for a reply before returning control to this transaction. For information about using the write-to-operator timeout value, see the CICS Application Programming Reference manual.

PARMERR={INTERACT|IGNORE|ABEND}

Code this parameter to specify what action you want to follow if CICS detects incorrect system initialization parameter overrides during initialization.

Note: When specified as an override, this parameter affects only subsequent system initialization parameter overrides. Errors in earlier system initialization parameter overrides are dealt with according to the PARMERR system initialization parameter value in the SIT.

INTERACT

Enables the operator to communicate with CICS via the console and correct parameter errors.

Note: INTERACT is overridden with IGNORE in the following cases:

- If errors are found in PARM or SYSIPT for system initialization parameter overrides that are not allowed to be entered from the console
- In certain circumstances, in response to invalid data when you have been trying to correct a previous invalid system initialization parameter keyword or value.

IGNORE

CICS ignores errors, and tries to complete initialization.

ABEND

CICS abends.

PDI={30|decimal-value}

Use this parameter in a SIT for an active CICS region. It specifies the XRF primary delay interval, in seconds. The minimum delay that you can specify is 5 seconds. This is the time that must elapse between the (apparent) loss of the surveillance signal in the alternate CICS region, and any reaction by the active CICS region. The corresponding parameter for the alternate CICS region is ADI. PDI and ADI need not have the same value.

PGAICTLG={MODIFY|NONE|ALL}

Specifies whether autoinstalled program definitions should be cataloged. While CICS is running, you can set whether autoinstalled programs should be cataloged dynamically, by using either the EXEC CICS SET SYSTEM or CEMT SET SYSTEM command.

MODIFY

Autoinstalled program definitions are cataloged only if the program definition is modified by a SET PROGRAM command subsequent to the autoinstall.

NONE

Autoinstalled program definitions are not cataloged. This gives a faster CICS restart (warm and emergency) compared with the MODIFY or ALL options, because CICS does not reinstall definitions from the CICS global catalog (DFHGCD). Definitions are autoinstalled on first reference.

ALL

Autoinstalled program definitions are written to the global catalog at the time of the autoinstall, and following any subsequent modification.

PGAIEXIT={DFHPGADX|name}

Specifies the name of the program autoinstall exit program. While CICS is running, you can set the name of the program autoinstall exit program dynamically, by using either the EXEC CICS SET SYSTEM or CEMT SET SYSTEM command.

PGAIPGM={INACTIVE|ACTIVE}

Specifies the state of the program autoinstall function at initialization. While CICS is running, you can set the status of program autoinstall dynamically, by using either the EXEC CICS SET SYSTEM or CEMT SET SYSTEM command.

INACTIVE

The program autoinstall function is disabled.

ACTIVE

The program autoinstall function is enabled.

PGCHAIN=character(s)

Code this parameter with the character string that is identified by terminal control as a BMS terminal page-chaining command. It can be 1 through 7 characters. For more information about the character string, see the notes under the PGRET parameter.

PGCOPY=character(s)

Code this parameter with the character string that is identified by terminal control as a BMS command to copy output from one terminal to another. It can be 1 through 7 characters. For more information about the character string, see the notes under the PGRET parameter.

PGPURGE=character(s)

Code this parameter with the character string that is identified by terminal control as a BMS terminal page-purge command. It can be 1 through 7 characters. For more information about the character string, see the notes under the PGRET parameter.

PGRET=character(s)

The character string that is recognized by terminal control as a BMS terminal page-retrieval command. It can be 1 through 7 characters.

Notes:

1. Each character string is unique with respect to the leading characters of every other transaction identification defined in the CSD. A command requested by a single character precludes the use of all other transaction identifications starting with this character.
2. In pseudoconversational mode, each character string is unique with respect to the leading characters of any terminal input message.
3. A field-separator or other suitable delimiter may be specified in each character string to separate this command code from the remainder of the paging command when entered by an operator. For example:

   ```
   PGCHAIN=X/
   PGCOPY=C/
   PGPURGE=T/
   PGRET=P/
   ```

   This reduces the risk of creating a nonunique command. (See Note 1.)

   Restrictions

   If you specify PGCHAIN, PGCOPY, PGPURGE, or PGRET in the SIT, the characters you choose must not include any of the following: ( ) '

   If you specify PGCHAIN, PGCOPY, PGPURGE, or PGRET as a PARM, SYSIPT, or console parameter, do not enclose the characters in quotation marks. The characters you choose must not include an embedded blank or any of the following: ( ) ' =

4. PGCHAIN, PGCOPY, PGPURGE, and PGRET are required only if full function BMS is being used. For information about the BMS page retrieval transaction CSPG, see the CICS Supplied Transactions manual.
5. CICS always processes a paging command entered by the operator before initiating a transaction in response to a macro request, that consists of either DFHBMS or DFHPC with the TRANSID operand coded.

PLTPI={NO|xx|YES}

Code this parameter with a program list table that contains a list of programs to be executed in the final stages of system initialization. For more information see page 165.

You can use the system initialization parameter INITPARM to pass parameters to those programs.

For information about coding the macros for this table, see the CICS Resource Definition Guide manual.

PLTPISEC={NONE|CMDSEC|RESSEC|ALL}

Specifies whether or not you want CICS to perform command security or resource security checking for PLT programs during CICS initialization. The PLT programs run under the authority of the userid specified on PLTPIUSR, which must be authorized to the appropriate resources defined by PLTPISEC.

NONE
Specifies that you do not want any security checking on on PLT initialization programs.

CMDSEC
Specifies that you want CICS to perform command security checking only.

RESSEC
Specifies that you want CICS to perform resource security checking only.

ALL
Specifies that you want CICS to perform both command and resource security checking.

Restrictions

You can code the PLTPISEC parameter in the SIT, PARM, or SYSIPT only.

PLTPIUSR=userid
Specifies the userid that CICS is to use for security checking for PLT programs that run during CICS initialization. All PLT programs run under the authority of the specified userid, which must be authorized to all the resources referenced by the programs, as defined by the PLTPISEC parameter.

PLT programs are run under the CICS internal transaction, CPLT. Before the CPLT transaction is attached, CICS performs a surrogate user check against the CICS region userid (the userid under which the CICS region is executing). This is to ensure that the CICS region is authorized as a surrogate for the userid specified on the PLTPIUSR parameter. This ensures that you cannot arbitrarily specify any PLT userid in any CICS region—each PLT userid must first be authorized to the appropriate CICS region.

If you do not specify the PLTPIUSR parameter, CICS runs PLTPI programs under the authority of the CICS region userid, in which case CICS does not perform a surrogate user check. However, the CICS region userid must be authorized to all the resources referenced by the PLT programs.

Restrictions

You can code the PLTPIUSR parameter in the SIT, PARM, or SYSIPT only.

PLTSD={NO|xx|YES}
Code this parameter with a program list table that contains a list of programs to be executed during system termination. For more information see page 165.

PRGDLAY={0|hhmm}
Code this parameter with the BMS purge delay time interval that is added to the specified delivery time to determine when a message is to be considered undeliverable and therefore purged. This time interval is coded in the form “hhmm” (where “hh” represents hours from 00 to 99 and “mm” represents minutes from 00 to 59). If PRGDLAY is not coded, or is given a zero value, a message remains eligible for delivery either until it is purged or until temporary storage is reinitialized.

Note: If you specify PRGDLAY as a SIT override, you must still specify a 4-character value (for example 0000).

The PRGDLAY facility requires the use of full function BMS. Note also that you must code a PRGDLAY value if you want the ERRTERM|ERRTERM(name) parameter on EXEC CICS ROUTE commands to be operative. For programming information about notification of undelivered messages, see the CICS Application Programming Reference manual.

The PRGDLAY value determines the interval between terminal page clean-up operations. A very low value causes the CSPQ transaction to be initiated continuously, and can have a detrimental effect on task-related resources.

A zero value stops CSPQ initiating terminal page clean-up. However, this can cause messages to stay in the system forever, resulting in performance problems with long AID queues or lack of temporary storage. The actual purge delay time interval specified is dependent on individual system requirements.

PRINT={NO|YES|PA1|PA2|PA3}
Code this parameter with the method of requesting printout of the contents of a 3270 screen.

NO
Screen copying is not required.

YES
Screen copying can be requested by terminal control print requests only.

PA1, PA2, or PA3
Screen copying can be requested by terminal control print request, or by using the PA (program attention) key specified.

The PA key specified by this parameter must not be specified by the TASKREQ option of the RDO TRANSACTION definition or be used for 3270 single keystroke retrieval.

When YES, PA1, PA2, or PA3 is specified, transaction CSPP is initiated which invokes program DFHP3270. The transaction and programs are defined in the CSD group DFHHARDC. In the case of 3270 and LUTYPE2 logical units, the resources defined in CSD group DFHVTAMP are required.

The 3270 print-request facility allows either the application program or the terminal operator to request a printout of data currently displayed on the 3270 display.

If CSPP is invoked to print the screen contents at an associated VTAM printer, the screen size of the printer is chosen according to the screen size defined in the profile for the transaction CSPP. The CICS-supplied definitions use the default screen size. Therefore, if you want DFHP3270 to use the alternate screen size of the printer, you must alter the screen size defined in the profile for the transaction CSPP. For information about defining profiles for transactions, see the *CICS Resource Definition Guide*.

For a VTAM 3270 display without the printer-adapter feature, the PRINT request prints the contents of the display on the first available 3270 printer specified by PRINTER and ALTPRINTER options of the RDO TERMINAL definition. For a printer to be considered available, it must be in service and not currently attached to a task. It is not necessary for the printer to be on the same control unit.

In an MRO environment, the printer must be owned by the same system as the VTAM 3270 display.

For the 3275 with the printer-adapter feature, the PRINT request prints the data currently in the 3275 display buffer on the 3284 Model 3 printer attached to the 3275.

The format of the print operation depends on the size of the display buffer. For a 40-character wide display, the print format is a 40-byte line, and for an 80-character wide display the format is an 80-byte line.

For the 3270 compatibility mode logical unit of the 3790 (if the logical unit has the printer-adapter feature specified), the PRINT request prints the contents of the display on the first printer available to the 3790. The allocation of the printer to be used is under the control of the 3790.

For 3274, 3276, and LUTYPE2 logical units with the printer-adapter feature, the PRINT request prints the contents of the display on the first printer available to the 3270 control unit. The printer to be allocated depends on the printer authorization matrix. For information, refer to the *3270 Information Display System Component Description* manual.

For the 3270 compatibility mode logical unit without the printer-adapter feature, see the preceding paragraph on VTAM 3270 displays without the printer-adapter feature.

PRTYAGE={32768|value}

Code this parameter with the number of milliseconds to be used in the priority aging algorithm for incrementing the priority of a task. The value can be in the range 0 through 65535, and 32768 is the default.

The priority aging factor is used to increase the effective priority of a task according to the amount of time it is held on a ready queue. The value represents the number of milliseconds that must elapse before the priority of a waiting task can be adjusted upwards by 1. For example, if you code PRTYAGE=3000, a task has its priority raised by 1 for every 3000 milliseconds it is held on the ready queue. Thus a high value for PRTYAGE results in a task being promoted very slowly up the priority increment range, and a low value enables a task to have its priority incremented quickly.

If you specify a value of 0, the priority aging algorithm is not used (task priorities are not modified by age) and tasks on the ready queue are handled according to the user assigned priority.

PRVMOD={name|(name,name...name)}

Code the PRVMOD parameter with the name of those modules that are not to be used from the VSE shared virtual area (SVA).

The operand is a list of 1- to 8-character module names. This enables you to use a private version of a CICS nucleus module in the CICS address space, and not a version that might be in the SVA. For more information about preventing modules from being used from the SVA, see the CICS System Definition Guide.

Restrictions

You can code the PRVMOD parameter in PARM, SYSIPT, or CONSOLE only.

PSDINT={0|hhmmss}

Specifies the persistent session delay interval. This delay interval specifies if, and for how long, VTAM is to hold sessions in a recovery-pending state if CICS fails. The value for hours can be in the range 0 through 23; the minutes and seconds in the range 00 through 59 inclusive.

This value can be overridden during CICS execution (and hence change the action taken by VTAM if CICS fails).

0 A zero value specifies that, if CICS fails, sessions are terminated. This is the default.

hhmmss

Specifies a persistent session delay interval from 1 second up to the maximum of 23 hours 59 minutes and 59 seconds. If CICS fails, VTAM holds sessions in recovery pending state for up to the interval specified on the PSDINT system initialization parameter.

Specify a 1-to-6 digit time in hours, minutes and seconds, up to the maximum time. If you specify less than six digits, CICS pads the value with leading zeros. Thus a value of 500 is taken as five minutes exactly.

The interval you specify must cover the time from when CICS fails to when the VTAM ACB is opened by CICS during the subsequent emergency restart.

VTAM holds all sessions in recovery pending state for up to the interval specified (unless they are unbound through path failure or VTAM operator action, or other-system action in the case of intelligent LUs). The PSDINT value used must take account of the types and numbers of sessions involved.

You must exercise care when specifying large PSDINT values because of the problems they may give in some environments, in particular:

- Dial-up sessions—real costs may be incurred
- LU6.2 sessions to other host systems—such systems may become stressed.

Notes:

1. When specifying a PSDINT value, you must consider the number and, more particularly, the nature of the sessions involved. If LU6.2 sessions to other host systems are retained in recovery pending state, the other host systems may experience excessive queuing delays. This point applies to LU6.1 sessions which are retained until restart (when they are unbound).
2. The PSDINT parameter is incompatible with the XRF=YES parameter. If XRF=YES is specified, the PSDINT parameter is ignored.

PVDELAY={30|number}

Code this with the persistent verification delay as a value in the range 0 through 10080 minutes (up to 7 days). PVDELAY defines how long entries can remain in the signed-on-from lists for those connections for which persistent verification is specified in a connection resource definition. If you specify PVDELAY=0, entries are deleted immediately after use.

For information about the use of PVDELAY, see the CICS Performance Guide.

RAMAX={256|value}

Code this parameter with the size in bytes of the I/O area allocated for each RECEIVE ANY issued by CICS, in the range 0 through 32767 bytes.

Notes:

1. If you are using APPC, do not code a value less than 256; otherwise, the results are unpredictable.
2. If you are using pipeline terminals, do not code a value that is less than the RUSIZE (from the CINIT), because pipelines cannot handle data longer than this.

For information about coding this parameter, see the CICS Performance Guide.

RAPOOL={50|decimal-value}

Code this parameter to determine the size of the CICS receive any pool. It is the number of fixed request parameter lists (RPLs), receive any control elements (RACEs), and receive any input areas (RAIAs) that are to be generated.

Decimal-value can be in the range 1 through 999, and has a default of 2.

If you omit the RAPOOL parameter altogether, RAPOOL=50 is assumed. CICS maintains n VTAM RECEIVE ANYs, where n is either the RAPOOL “number active” value, or the MXT value minus the number of active tasks, whichever is the smaller. For example:

```
If RAPOOL=2, MXT=50, active tasks = 45 then RECEIVE ANY = 2
If RAPOOL=10, MXT=50, active tasks = 45 then RECEIVE ANY = 5
If RAPOOL=10, MXT=50, active tasks = 35 then RECEIVE ANY = 10
```

The number of RECEIVE ANYs needed depends on the expected activity of the system, the average transaction lifetime, and the MAXTASK value specified. For information about coding this parameter, see the CICS Performance Guide.

RDSASZE={0K|number}

Code this parameter with the size of the RDSA. The default size is 0 indicating that the DSA size can be changed dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 16,777,215 bytes in multiples of 262,144 bytes (256KB). If the size specified is not a multiple of 256KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4,194,304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restriction

You can code the RDSASZE parameter in PARM, SYSIPT or CONSOLE only.

RENTPGM={PROTECT|NOPROTECT}

Code this parameter to specify whether you want CICS to allocate the read-only DSAs, RDSA and ERDSA, from read-only key-0 protected storage. The permitted values are PROTECT (the default), or NOPROTECT:

PROTECT

CICS obtains the storage for the read-only DSAs from key-0 protected storage.

NOPROTECT
CICS obtains the storage from CICS-key storage, effectively creating two more CICS DSAs (CDSA and ECDSA). This allows programs eligible for the read-only DSAs to be modified by programs that execute in CICS key.

You are recommended to specify RENTPGM=NOPROTECT for development regions only, and to specify RENTPGM=PROTECT for production CICS regions.

RESP={FME|RRN}
Code this parameter to specify the type of request that CICS terminal control receives from logical units.

FME Function management end is the default.

RRN Reached recovery node.

RESSEC={ASIS|ALWAYS}
Specifies whether or not you want CICS to honor the RESSEC option specified on a transaction's resource definition.

ASIS
means that CICS honors the RESSEC option defined in a transaction's resource definition. CICS calls its resource security checking routine only when RESSEC(YES) is specified in a transaction resource definition. This is normally a sufficient level of control, because often you will need only to control the ability to execute a transaction.

ALWAYS
means that CICS overrides the RESSEC option, and always calls its resource security checking routine to issue the appropriate call to the SAF interface.

Use this option only if you need to control or audit all accesses to CICS resources. You should be aware that using this option can significantly degrade performance.

Restrictions
You can code the RESSEC parameter in the SIT, PARM, or SYSIPT only.

RMTRAN=({CSGM|name1}[,{CSGM|name2}])
Code this parameter with the name of the transaction you want to initiate when a terminal session is recovered at system restart for a CICS region running with VTAM persistent sessions support.

If you do not specify a name on the RMTRAN parameter, CICS uses the value specified on the GMTRAN system initialization parameter; the CSGM transaction is the default CICS good morning transaction.

name1
This is the transaction that CICS initiates at terminals that do ***not*** remain signed-on after system restart (that is, they are still connected to CICS, but are signed off).

name2
This is the transaction that CICS initiates at terminals that remain signed-on after system restart. If you specify only name1, CICS uses the CSGM transaction as the default for name2.

RUWAPOOL={NO|YES}

specifies the option for allocating a storage pool the first time an LE-conforming program runs in a task.

NO CICS disables the option and provides no RUWA storage pool. Every EXEC CICS LINK to an LE-conforming application results in a GETMAIN for RUWA storage.

YES CICS creates a pool of storage the first time an LE-conforming program runs in a task. This provides an available storage pool that reduces the need to GETMAIN and FREEMAIN run-unit work areas (RUWAs) for every EXEC CICS LINK request.

SDSASZE={0K|number}

Code this parameter with the size of the SDSA. The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 16,777,215 bytes in multiples of 262,144 bytes (256KB). If the size specified is not a multiple of 256KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4,194,304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restriction

You can code the SDSASZE parameter in PARM, SYSIPT or CONSOLE only.

SEC={YES|NO}

Code this parameter to indicate what level of external security you want CICS to use.

YES

Code YES if you want to use full external security. CICS requires the appropriate level of authorization for the access intent: a minimum of READ permission for read intent, and a minimum of UPDATE permission for update intent.

Note: You must also ensure that the default userid (CICSUSER or another userid specified on the DFLTUSER system initialization parameter) has been defined to your external security manager (ESM).

If command security checking is defined for EXEC CICS SP-type commands, then specifying SEC=YES means that the appropriate level of authority is checked for; therefore:

- A check for READ authority is made for EXEC CICS INQUIRE and COLLECT commands
- A check for UPDATE authority is made for EXEC CICS SET, PERFORM, and DISCARD commands

  For the results of the interaction between the access intent of the user application, and the permission defined to your ESM, see Table 12 on page 220.
- A check for ALTER authority is made for EXEC CICS CREATE commands.

NO
Code NO if you do not want CICS to use an ESM. All users have access to all resources, whether determined by attempts to use them or by the QUERY SECURITY command. Users are not allowed to sign on or off.

Note: With MRO bind-time security, even if you specify SEC=NO, the CICS region userid is still sent to the secondary CICS region, and bind-time checking is still carried out in the secondary CICS region. For information about MRO bind-time security, see the CICS Security Guide.

Define whether to use your ESM for resource level checking by using the XDCT, XFCT, XJCT, XPCT, XPPT, XPSB, and XTST system initialization parameters. Define whether to use an ESM for transaction-attach security checking by using the XTRAN system initialization parameter. Define whether ESM session security can be used when establishing APPC sessions by using the XAPPC system initialization parameter.

For information on defining command security checking for CICS SP-type commands, and about CICS security in general, see the CICS Security Guide.

For programming information about the use of external security for CICS system commands, see the CICS System Programming Reference manual.

Table 12. Results of ESM authorization requests (with SEC=YES)

Access Permission defined to ESM for CICS user	Access intent in application		
	READ	UPDATE	ALTER
NONE	Refused	Refused	Refused
READ	Permitted	Refused	Refused
UPDATE	Permitted	Permitted	Refused
ALTER	Permitted	Permitted	Permitted

Restrictions
You can code the SEC parameter in the SIT, PARM, or SYSIPT only.

SECPRFX={NO|YES}
Specifies whether or not CICS prefixes the resource names in any authorization requests to the ESM with a prefix corresponding to the ESM userid for the CICS region. The prefix to be used (the userid for the CICS region) is obtained by the DFHIRP module.

NO
CICS does not prefix the resource names in any authorization requests to the ESM.

YES
The ESM userid is used as the prefix for CICS resources defined to the ESM. CICS prefixes the resource name in any authorization requests to the ESM with a prefix corresponding to the ESM userid of the CICS region, obtaining the userid as follows:

- If you start CICS as a job, the prefix corresponds to the USER operand coded on the //ID JOB statement of the CICS startup job stream.
- If you start a CICS job without an associated ESM userid, the prefix defaults to CICS.

For information about using the PREFIX option, see the CICS Security Guide.

Restrictions

You can code the SECPRFX parameter in the SIT, PARM, or SYSIPT only.

The SECPRFX parameter is effective only if you specify YES for the SEC system initialization parameter.

SIT=xx

Code this parameter to specify the suffix, if any, of the system initialization table that you want CICS to load at the start of initialization. If you omit this parameter, CICS loads the unsuffixed table, DFHSIT, which is pregenerated with all the default values. This default SIT (shown in “DFHSIT, the default system initialization table” on page 153) named DFHSIT&#163;&#163;, and its source, is in the VSE/ESA sublibrary, PRD1.BASE.

Restrictions

You can code the system initialization parameter anywhere in PARM or SYSIPT, or as the ***first*** parameter entry at the CONSOLE.

SKRxxxx=‘page-retrieval-command’

Code this parameter if a single-keystroke-retrieval operation is required. ‘xxxx’ specifies a key on the 3270 keyboard which, during a page retrieval session, is to be used to represent a page retrieval command. The valid keys are PA1 through PA3, and PF1 through PF36. Thus up to 39 keys can be specified in this way (each by a separate command).

The ‘page-retrieval-command’ value represents any valid page retrieval command, and must be enclosed in apostrophes. It is concatenated to the character string coded in the PGRET parameter. The combined length must not exceed 16 characters.

Note: If full function BMS is used, all PA keys and PF keys are interpreted for page retrieval commands, even if some of these keys are not defined.

SNSCOPE={NONE|CICS|VSEIMAGE|}

Specifies whether or not a userid can be signed on to CICS more than once, within the scope of:

- A single CICS region
- A single VSE image

NONE

Each userid can be used to sign on for any number of sessions on any CICS region. This is the compatibility option, providing the same signon scope as in releases of CICS before CICS Transaction Server for VSE/ESA Release 1.

CICS

Each userid can be signed on once only in the same CICS region. A signon request is rejected if the userid is already signed on to the same CICS region. However, the userid can be used to signon to another CICS region in the same, or another, VSE image.

VSEIMAGE
Each userid can be signed on once only, and to only one of the set of CICS regions in the same VSE image that also specify SNSCOPE=VSEIMAGE. A signon request is rejected if the user is already signed on to another CICS region in the same VSE image.

The signon scope (if specified) applies to all userids signing on by an explicit signon request (for example, by an EXEC CICS SIGNON command or the CESN transaction). SNSCOPE is restricted to users signing on at local terminals, or signing on after using the CRTE transaction to connect to another system.

Signon scope specified by SNSCOPE *does not* apply to:

- Non-terminal users.
- The CICS default userid, specified by the DFLTUSER system initialization parameter.
- Preset userids, specified in the USERID option of the RDO TERMINAL resource definition.
- Userids for remote users, received in attach headers.
- Userids for link security. For information about which userid is used for link security on a specific connection, see the CICS Security Guide.
- The userid specified on the PLTPIUSR system initialization parameter.
- The CICS region userid.

Restrictions

You can code the SNSCOPE parameter in the SIT, PARM, or SYSIPT only.

SPCTR={(1,2|1[,2][,3])|ALL|OFF}
Code this parameter to set the level of tracing for all CICS components used by a transaction, terminal, or both, selected for special tracing. If you want to set different tracing levels for an individual component of CICS, use the SPCTRxx system initialization parameter. You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels. For a list of all the available trace points and their level numbers, see the CICS User's Handbook. For information about the differences between special and standard CICS tracing, see the CICS Problem Determination Guide.

number
Code the level numbers for the level of special tracing you want for all CICS components. The options are: 1, (1,2), or (1,2,3). The default, (1,2), specifies special tracing for levels 1 and 2 for all CICS components.

ALL Enables the special tracing facility for all available levels.

OFF Disables the special tracing facility.

SPCTRxx={(1,2|1[,2][,3])|ALL|OFF}
Code this parameter to set the level of tracing for a particular CICS component used by a transaction, terminal, or both, selected for special tracing. You identify

the component by coding a value for xx in the keyword. You code one SPCTRxx keyword for each component you want to define selectively. For a CICS component being specially traced that does not have its trace level set by SPCTRxx, the trace level is that set by SPCTR (which, in turn, defaults to (1,2)). You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels. The CICS component codes that you can code for xx on the SPCTRxx keyword are shown in Table 13:

Table 13. CICS component names and abbreviations

Code	Component name	Code	Component name
AP	Application domain	BF	Built-in function
BM	Basic mapping support	BR	3270 Bridge
CP	Common programming interface	DC	Dump compatability layer
DD	Directory manager domain	DH	Document handler domain
DI	Batch data interchange	DM	Domain manager domain
DS	Dispatcher domain	DU	Dump domain
EI	Exec interface	FC	File control
GC	Global catalog domain	IC	Interval control
IS	Inter-system communication	JC	Journal control
KC	Task control	KE	Kernel
LC	Local catalog domain	LD	Loader domain
LM	Lock domain	ME	Message domain
MN	Monitoring domain	PA	Parameter domain
PC	Program control	PG	Program manager domain
RC	Report Controller	SC	Storage control
SM	Storage domain	SO	Sockets domain
SP	Sync point	ST	Statistics domain
SZ	Front end programming interface	TC	Terminal control
TD	Transient data	TI	Timer domain
TR	Trace domain	TS	Temporary storage
UE	User exit interface	US	User domain
WB	Web support	XM	Transaction manager domain
XS	Security manager domain		

Note: The component codes BF, BM, BR, CP, DC, DI, EI, FC, IC, IS, JC, KC, PC, RC, SC, SP, SZ, TC, TD, TS, and UE are sub-components of the AP domain. As such, the corresponding trace entries will be produced with a point ID of AP nnnn.

For details of using trace, see the CICS Trace Entries Handbook.

number
Code the level numbers for the level of special tracing you want for the CICS component indicated by xx. The options are: 1, (1,2), or (1,2,3).

ALL Code ALL to indicate that you want all the available levels of special CICS tracing switched on for the specified component.

OFF Code OFF to switch off all levels of special CICS tracing for the CICS component indicated by xx.

Restrictions

You can code the SPCTRxx parameter in PARM, SYSIPT, or CONSOLE only.

SPOOL=({NO|YES}[,{A|identifier}][,{A|class}{YES|NO}])

Specifies whether the system spooling interface is required. The identifier and class values are for the report controller.

Use the final YES|NO option to specify whether you want a formfeed (blank page) to precede the first report printed on a non-SCS printer started to the report controller.

The specified identifier is a single character used to identify the printer-owning CICS system to the cross-partition communication (XPCC) component in the device-driving system name (ddsname).

The specified class is the default output class to be used for reports that have no class specified.

NO

Specifies that the system spooling interface is not required.

YES

Specifies that the system spooling interface is required.

A Specifies that the name SYSCICSA is used to identify this CICS system.

identifier

Specifies a single character identifier, which is appended to SYSCICS to create a name used to identify this CICS system. This identifier should be unique for each CICS system running with the system spooling interface. If you do not provide a unique identifier, you might receive message DFHRC5301I (indicating that the POWER interface has failed to initialize).

A Class A is the default for the output class for reports and printers.

class

Specifies a single alpha-numeric character identifier (A to Z, 0-9 only), used as the default output class for reports and printers.

YES

Specifies a formfeed (blank page) is to precede the first report printed on a non-SCS printer started to the report controller.

NO

Specifies a formfeed (blank page) is not to precede the first report printed on a non-SCS printer started to the report controller. If you code NO, be aware that a report can overtype the last line of the previous output when the printer is started to the report controller.

SRT={YES|NO|xx}

Specifies the system recovery table suffix, for more information see page 165.

For information about coding the macros for this table, see the CICS System Definition Guide.

If SRT=NO is coded, the system recovery program (DFHSRP) does not attempt to recover from a program check or from an operating system abend. However, CICS issues VSE ESTAEX macros to intercept program checks to perform clean-up operations before CICS terminates. Therefore, an SRT must be provided if recovery from either program checks or abnormal terminations, or both, is required.

SSLDELAY=600|number

Specifies the length of time in seconds for which CICS retains session IDs for secure socket connections. Session IDs are tokens that represent a secure connection between a client and an SSL server. While the session ID is retained by CICS within the SSLDELAY period, CICS can continue to communicate with the client without the significant overhead of an SSL handshake. The value is a number of seconds in the range 0 through 86400. The default value is 600.

START=({AUTO|COLD|STANDBY|LOGTERM|}[,ALL])

Specifies the type of start for the system initialization program. The value specified for START, or the default of AUTO, becomes the default value for each resource.

AUTO

CICS performs either a warm or an emergency restart, according to the status of the control record written to the global catalog by the previous execution of CICS. If the global catalog is newly initialized, it does not contain a control record, and CICS forces a cold start for START=AUTO.

If you code START=AUTO, you must provide a restart data set for use in case CICS has to perform an emergency restart, and the system log from the previous execution of CICS must be available. You must also have coded an activity keypoint value (see the AKPFREQ parameter on page 177) on the previous execution of CICS for an emergency restart to be successful.

COLD

Specifies a cold start. The status of CICS resource definitions saved in the global catalog at the previous shutdown is ignored, and all resource definitions are reinstalled, either from the CSD or CICS control tables. However, some information saved in the global catalog and local catalog is preserved across cold starts. For information about how CICS uses information saved in the global catalog and local catalog, see the CICS Recovery and Restart Guide.

LOGTERM

LOGTERM is a special option for use as an alternative to a full emergency restart when the previous run terminated in an uncontrolled shutdown, and is available only if the CICS system log is defined on disk data sets. If you specify START=LOGTERM, CICS initializes up to the point where it writes an end of file on the system log. After it has written the end of file, CICS ends the emergency restart process and shuts down without doing any backout processing.

For background information about this restart process, see the CICS Recovery and Restart Guide.

Restriction

START=LOGTERM is applicable only if XRF=NO is coded. You can code START=LOGTERM in PARM, SYSIPT, or CONSOLE only.

STANDBY

Coding START=STANDBY, but only when you have also specified

XRF=YES, defines this CICS as the alternate CICS region in an XRF pair. In other words, you ***must*** specify START=STANDBY for the system that starts off as the alternate. (To start an active CICS region, specify AUTO or COLD, as you would without XRF.)

If you have specified a COLD start for other CICS resources, for example, DCT=(xx,COLD), they are cold started when the alternate CICS region (with START=STANDBY specified) takes over. This may cause CICS to lose data on an XRF takeover; for example, coding ICP=COLD results in outstanding STARTs being lost. You are recommended to code START=(STANDBY,ALL) to ensure a full emergency restart during takeover, unless you wish to specifically cold start individual resources.

(option,ALL)
Specifies that the ALL operand is a special option you can use on the START parameter when you supply it as a system initialization parameter at CICS startup; you cannot code it in the SIT. If you specify START=(AUTO,ALL), CICS initializes all resources according to the type of startup that it selects (warm, emergency, or cold). The ALL option overrides any individual settings in other system initialization parameters (for example, DCT=(xx,COLD)).

However, if you do not use the ALL option, you can individually cold start those resources that have a COLD operand. For details of resources that have a COLD option, see Table 5 on page 166.

Restrictions

You can code START=(option,ALL) in PARM, SYSIPT, or CONSOLE only.

For more information about the types of CICS startup, see “Classes of start and restart” on page 169.

STARTER={NO|YES}
Code YES to indicate that the generation of starter system modules (with &#163; and # suffixes) is permitted, and various MNOTES are to be suppressed. This parameter should only be used when service is being performed on starter system modules.

Restrictions

You can code the STARTER parameter in the SIT only.

STATRCD=OFF|ON
Specifies the interval statistics recording status at CICS initialization. This status is recorded in the CICS global catalog for use during warm and emergency restarts. Statistics collected are written to the DMF data set.

OFF Interval statistics are not collected (no action is taken at the end of an interval).

End-of-day, Unsolicited and requested statistics are written to DMF regardless of the STATRCD setting.

ON Interval statistics are collected.

On a cold start of a CICS region, interval statistics are recorded by default at three-hourly intervals. All intervals are timed using the end-of-day time (midnight is the default) as a base starting time (***not***

CICS startup time). This means that the default settings give collections at 00.00, 03.00, 06.00, 09.00, and so on, regardless of the time that you start CICS.

On a warm or emergency restart the statistics recording status is restored from the CICS global catalog.

You can change the statistics recording status at any time as follows:

- During a warm or emergency restart by coding the STATRCD system initialization parameter.
- While CICS is running by using the CEMT or EXEC CICS SET STATISTICS command.

Whatever the value of the STATRCD system initialization parameter, you can ask for requested statistics and requested reset statistics to be collected. You can get statistics "on demand" for all, or for specified, resource types by using the CEMT or EXEC CICS PERFORM STATISTICS command. The period covered for statistics requested in this way is from the last reset time (that is, from the beginning of the current interval or from when you last issued a CEMT or EXEC CICS statistics command specifying RESETNOW) up to the time that you issue the PERFORM STATISTICS command.

For information about using these CEMT commands, see CICS Supplied Transactions manual. For programming information about the EXEC CICS PERFORM commands, see the CICS System Programming Reference manual.

For information about the statistics utility program DFHSTUP, see CICS Operations and Utilities Guide.

STGPROT={NO|YES}

Specifies whether you want storage protection in the CICS region. The permitted values are NO (the default), or YES:

NO If you specify NO, or allow this parameter to default, CICS does not operate any storage protection, and runs in a single storage key as in earlier releases. See the CICS System Definition Guide for a summary of how STGPROT=NO affects the storage allocation for the dynamic storage areas.

YES If you specify YES, and if you have the required hardware and software, CICS operates with storage protection, and observes the storage keys and execution keys that you specify in various system and resource definitions. See the CICS System Definition Guide for a summary of how STGPROT=YES affects the storage allocation for the dynamic storage areas.

If you do not have the required hardware and software support, CICS issues an information message during initialization, and operates without storage protection.

STGRCVY={NO|YES}

Specifies whether CICS should try to recover from a storage violation.

NO CICS does not try to repair any storage violation that it detects.

YES CICS tries to repair any storage violation that it detects.

In both cases, CICS continues unless you have specified in the dump table that CICS should terminate.

In normal operation, CICS sets up four task-lifetime storage subpools for each task. Each element in the subpool starts and ends with a ‘check zone’ that includes the subpool name. At each freemain, and at end-of-task, CICS checks the check zones and abends the task if either has been overwritten.

Terminal input-output areas (TIOAs) have similar check zones, which are set up with identical values. At each freemain of a TIOA, CICS checks the check zones and abends the task if they are not identical.

If you specify the STGRCVY(YES) system initialization parameter, CICS resets the check zones correctly and the task continues running.

STNTR={1|(1[,2][,3])|ALL|OFF}
Specifies the level of standard tracing required for CICS as a whole. You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels.

Note: Before globally activating tracing levels 3 and ALL for the storage manager (SM) component, read the warning given in the description for the STNTRxx system initialization parameter.

number
Code the level number(s) for the level of standard tracing you want for all CICS components. The options are: 1, (1,2), or (1,2,3). The default, 1, specifies standard tracing for level 1 for all CICS components.

ALL Enables standard tracing for all levels.

OFF Disables standard tracing.

For information about the differences between special and standard CICS tracing, see the CICS Problem Determination Guide.

STNTRxx={(1,21[,2][,3])|ALL|OFF}
Specifies the level of standard tracing you require for a particular CICS component. You identify the component by coding a value for xx in the keyword. You code one STNTRxx keyword for each component you want to define selectively. For a CICS component being specially traced that does not have its trace level set by STNTRxx, the trace level is that set by STNTR (which, in turn, defaults to 1). You can select up to three levels of tracing, but some CICS components do not have trace points at all these levels.

The CICS component codes that you can code for xx on this STNTRxx keyword are shown in Table 13 on page 223.

number
Code the level number(s) for the level of standard tracing you want for the CICS component indicated by xx. The options are: 1, (1,2), or (1,2,3).

ALL Code ALL to indicate that you want all the available levels of standard tracing switched on for the specified component.

OFF Code OFF to switch off all levels of standard CICS tracing for the CICS component indicated by xx.

Attention!

If you select tracing levels 3 or ALL for the storage manager (SM) component, the performance of your CICS system will be degraded. This is because options 3 and ALL switch on levels of trace that are also used for field engineering purposes. See the CICS Problem Determination Guide for information about the effects of trace levels 3 and ALL.

Restrictions

You can code the STNTRxx parameter in PARM, SYSIPT, or CONSOLE only.

SUFFIX=xx

Specifies the last two characters of the name of this system initialization table.

The first 6 characters of the name of the SIT are fixed as DFHSIT. You can specify the last two characters of the name, using the SUFFIX parameter. Because the SIT does not have a TYPE=INITIAL macro statement like other CICS resource control tables, you specify its SUFFIX on the TYPE=CSECT macro statement.

The suffix allows you to have more than one version of the SIT. Any one or two characters (other than NO and DY) are valid. You select the version of the table to be loaded into the system during system initialization by coding SIT=xx, either in the PARM parameter or the SYSIPT data set. (You can, in some circumstances, specify the SIT using the system console, but this is not recommended.)

Restrictions

You can code the SUFFIX parameter in the SIT only.

SVA={NO|YES}

Specifies whether any CICS management modules can be used from the VSE shared virtual area (SVA).

NO

No CICS management modules are used from the SVA.

YES

CICS management modules installed in the SVA can be used from there, instead of being loaded into the CICS system.

A list of the CICS modules that are read-only, and hence eligible for residence in the SVA, is given in Appendix B of the CICS System Definition Guide.

For details of the CICS system initialization parameter PRVMOD that you can use to override SVA=YES for selected modules, see page 215.

SYDUMAX={999|number}

Specifies the limit on the number of system dumps that may be taken per dump table entry. If this number is exceeded, subsequent system dumps for that particular entry will be suppressed.

number

Specifies a number in the range 0 through 999. The default, 999, enables an unlimited number of dumps to be taken.

SYSIDNT={CICS|name}

Specifies a 1- to 4-character name that is known only to your CICS region. If your CICS region also communicates with other CICS regions, the name you choose for this parameter to identify your local CICS region must not be the same name as an installed RDO CONNECTION resource definition for a remote region.

The value for SYSIDNT, whether specified in the SIT or as an override, can only be updated on a cold start. After a warm start or emergency restart, the value of SYSIDNT is that specified in the last cold start.

For information about the SYSIDNT of a local CICS region, see the CICS Intercommunication Guide.

SYSTR={ON|OFF}

Specifies the master system trace flag.

Code ON to obtain trace entries of CICS system activity. Entries are written to all the trace destinations that are active.

TAKEOVR={AUTO|MANUAL|COMMAND}

Code the TAKEOVR parameter in the SIT for an alternate XRF system. It specifies the action to be taken by the alternate CICS, following the (apparent) loss of the surveillance signal in the active CICS system. In doing this, it also specifies the level of operator involvement.

AUTO

No operator approval or intervention is needed for a takeover.

MANUAL

Specifies that the operator is asked to approve a take over if the alternate system cannot detect the surveillance signal of the active system.

Note that the alternate system does not ask the operator for approval if the active system signs off abnormally, nor is there an operator or program command for take over. In these cases, there is no doubt that the alternate system should take over, and manual involvement by the operator would be an unnecessary overhead in the take over process.

For example, you could code MANUAL to ensure manual take over of a master or coordinator region in MRO.

COMMAND

Take over only occurs when a CEBT PERFORM TAKEOVER command is received by the alternate system. It ensures, for example, that a dependent alternate CICS (in MRO) is activated only if it receives the command from the operator, or from a master (coordinator) region.

TBEXITS=([name1][,name2][,name3][,name4][,name5])

Code the names of your transaction backout exit programs. These exits are used for resource backout during emergency restart processing. For programming information about transaction backout exit programs, see the CICS Customization Guide. For background information about transaction backout, see the CICS Recovery and Restart Guide.

The order in which you code the names is critical. If you do not want to use all five exits, code commas in place of the ones you miss out. For example:

```
TBEXITS=(,,EXITF,EXITV)
```

name1
The name of your initialization/termination exit program.

name2
The name of your input exit program.

name3
The name of your file error exit program.

name4
The name of your open error exit program.

name5
The name of your DL/I error exit program.

If no transaction backout exit programs are required, you can do one of the following:

- Omit the whole parameter from the SIT
- Code TBEXITS=(,,,,) as a SIT parameter
- Code TBEXITS=(,,,,) as a SIT override.

TCP={YES|NO}

Code TCP=YES to include the pregenerated non-VTAM terminal control program, DFHTCP.

You must code TCP=YES if you intend using card reader/line printer (sequential) devices.

TCPIP={NO|YES}

Specifies whether CICS TCPIP services are to be activated at CICS startup. The default is NO, meaning that these services cannot be enabled. If TCPIP is set to YES, the HTTP service can process work. SSL support can only be used when TCPIP=YES.

Note: The TCPIP system initialization parameter affects only CICS internal TCP/IP Services defined by TCPIPSERVICE resource definitions. It has nothing to do with the TCP/IP Socket Interface for CICS feature of TCP/IP for VSE/ESA

TCSACTN={NONE|UNBIND|FORCE}

Specifies the required action that CICS terminal control should take if the terminal control shutdown wait threshold expires. For details of the wait threshold, see the TCSWAIT system initialization parameter. TCSACTN only takes effect when TCSWAIT is coded with a value in the range 1 through 99. This parameter only applies to VTAM terminals (including LU Type 6.2 single-session APPC terminals), not VTAM intersystem connections (LU Type 6.1 and LU Type 6.2 parallel connections). This is a global default action. On a terminal-by-terminal basis, you can code a DFHZNEP routine to override this action.

NONE
No action is taken. This can be overridden by DFHZNEP.

UNBIND
CICS terminal control attempts to force-close the session by issuing a VTAM CLSDST and sending an SNA UNBIND command to the hung terminal. This can be overridden by DFHZNEP.

FORCE
CICS terminal control attempts to forceclose the CICS VTAM ACB if there are any hung terminals. All CICS VTAM terminals and sessions are released and CICS normal shutdown continues.

TCSWAIT={4|number|NO|NONE|0}

Specifies the required CICS terminal control shutdown wait threshold. The wait threshold is the time, during shutdown, that CICS terminal control allows to pass before it considers terminal shutdown to be hung. If all VTAM sessions shutdown and close before the threshold expires then the CICS shutdown process moves on to its next stage, and the terminal control wait threshold then no longer applies. If, however, some of the VTAM session do not complete shutdown and close, then CICS takes special action with these sessions. For details of this special action see the description of the TCSACTN system initialization parameter. The wait threshold only applies to VTAM sessions; that is, VTAM terminals and VTAM intersystem connections. The wait time is specified as a number of minutes, in the range 1 through 99. As a special case, TCSWAIT=NO may be specified to indicate that terminal control shutdown is never to be considered hung, no matter how long the shutdown and close process takes. TCSWAIT=NONE and TCSWAIT=0 are alternative synonyms for TCSWAIT=NO, and all three have the same effect (internally they are held as the one value 0 (zero)).

TCT={YES|xx|NO}

Specifies which terminal control table, if any, is to be loaded, for more information see page 165.

For guidance about coding the macros for this table, see the CICS Resource Definition Guide

If you reassemble the TCT after starting CICS, any changes are applied when you next start CICS, even if it is a warm or emergency startup.

If you have VTAM-connected terminals only, you can code TCT=NO. If you do this, note that a dummy TCT, called DFHTCTDY, is loaded during system initialization. For more information about DFHTCTDY, see page 169. (If you code TCT=NO, you must specify a CSD group list in the GRPLIST parameter.)

TCTUAKEY={USER|CICS}

Specifies the storage key for the terminal control table user areas (TCTUAs) if you are operating CICS with storage protection (STGPROT=YES). The permitted values are USER (the default), or CICS:

USER If you specify USER, or allow this parameter to default, CICS obtains the amount of storage for TCTUAs in user key. This allows a user program executing in any key to modify the TCTUA.

CICS If you specify CICS, CICS obtains the amount of storage in CICS key. This means that only programs executing in CICS key can modify the TCTUA, and user-key programs have read-only access.

If CICS is running without storage protection, the TCTUAKEY parameter only designates which DSA (User or CICS) the storage comes from. The TCTUAs are accessed in CICS-key whether they are in the UDSA or CDSA.

See CICS System Definition Guide for more information about TCTUAs.

TCTUALOC={BELOW|ANY}

Specifies where terminal user areas (TCTUA) are to be stored.

BELOW

Specifies that the TCTUAs are stored below the 16MB line.

ANY

Specifies that the TCTUAs are stored anywhere in virtual storage. CICS stores TCTUAs above the 16MB line if possible.

For more information about TCTUAs, see the CICS System Definition Guide.

For details about defining terminals using RDO, see the CICS Resource Definition Guide.

TD=({3|decimal-value-1}[,{3|decimal-value-2}])
Specifies the number of VSAM buffers and strings to be used for intrapartition transient data (TD).

decimal-value-1
Specifies that the number of buffers to be allocated for the use of intrapartition transient data. The value must be in the range 1 through 32 767. The default value is 3.

CICS obtains, above the 16MB line, storage for the TD buffers in units of the page size (4KB). Because CICS optimizes the use of the storage obtained, TD may allocate more buffers than you specify, depending on the control interval (CI) size you have defined for the intrapartition data set.

For example, if the CI size is 1536, and you specify 3 buffers (the default number), CICS actually allocates 5 buffers. This is because 2 pages (8192 bytes) are required to obtain sufficient storage for three 1536-byte buffers, a total of only 4608 bytes, which would leave 3584 bytes of spare storage in the second page. In this case, CICS allocates another 2 buffers (3072 bytes) to minimize the amount of unused storage. In this way CICS makes use of storage that would otherwise be unavailable for any other purpose.

decimal-value-2
Specifies that the number of VSAM strings to be allocated for the use of intrapartition transient data. The value must be in the range 1 through 255, and must not exceed the value specified in decimal-value-1. The default value is 3.

For example, TD=(8,5) specifies 8 buffers and 5 strings.

The operands of the TD parameter are positional. You must code commas to indicate missing operands if others follow. For example, TD=(,2) specifies the number of strings and allows the number of buffers to default.

TRAP={OFF|ON}
Specifies whether the FE global trap exit is to be activated at system initialization. This exit is for diagnostic use under the guidance of service personnel. For background information about this exit, see the CICS Problem Determination Guide.

TRDUMAX={999|number}
Specifies the limit on the number of transaction dumps that may be taken per dump table entry. If this number is exceeded, subsequent transaction dumps for that particular entry will be suppressed.

number
Specifies a number in the range 0 through 999. The default, 999, enables an unlimited number of dumps to be taken.

TRTABSZ={16|number-of-kilobytes}
Specifies the size in kilobytes of the internal trace table. (1KB = 1024 bytes.) The CICS trace table is allocated in virtual storage above the 16MB line, and it is allocated ***before*** the extended CICS-key DSA (ECDSA) and the extended

user-key DSA (EUDSA). Ensure that there is sufficient virtual storage for the trace table, the ECDSA, and the EUDSA by defining a large enough VSE partition for your CICS job.

16 16KB is the default size of the trace table, and also the minimum size.

number
The number of kilobytes of storage to be allocated for the internal trace table, in the range 16KB through 1048576KB. Subpool 1 is used for the trace table storage, which exists for the duration of the CICS execution. The table is page aligned and occupies a whole number of pages. If the value specified is not a multiple of the page size (4KB), it is rounded up to the next multiple of 4KB.

Trace entries are of variable lengths, but the average length is approximately 100 bytes.

Note: To switch on internal tracing, use the INTTR parameter; for a description of INTTR, see page 204.

TRTRANSZ={40|number-of-kilobytes}
specifies the size in kilobytes of the transaction dump trace table. (1KB = 1024 bytes.)

When a transaction dump is taken, CICS performs a VSE GETMAIN for storage above the 16MB line for the transaction dump trace table.

40 40KB is the default size of the transaction dump trace table. The minimum size is 16KB.

number
The number of kilobytes of storage to be allocated for the transaction dump trace table, in the range 16KB through 1048576KB.

TRTRANTY={TRAN|ALL}
specifies which trace entries should be copied from the internal trace table to the transaction dump trace table.

TRAN Only the trace entries associated with the transaction that is abending will be copied to the transaction dump trace table.

ALL All of the trace entries from the internal trace table will be copied to the transaction dump trace table. If the internal trace table size is larger than the transaction dump trace table size, the transaction dump trace table could wrap. This results in only the most recent trace entries being written to the transaction dump trace table.

TS=([COLD][,{0|3|decimal-value-1}][,{3|decimal-value-2}])
Specifies:

- Whether or not you want to cold start temporary storage
- The number of VSAM buffers to be used for auxiliary temporary storage
- The number of VSAM strings to be used for auxiliary temporary storage.

COLD
Specifies that the type of start for the temporary storage program. If you do not want a cold start, code a comma before the second operand.

Note: IF ICP is warm-started (that is no ICP=COLD) and TS is cold-started, any ICEs and AIDs with data attached are lost.

0 No buffers are required; that is, only MAIN temporary storage is required.

decimal-value-1

Specifies that the number of buffers to be allocated for the use of auxiliary temporary storage. The value must be in the range 3 through 32 767.

CICS obtains, above the 16MB line, storage for the auxiliary temporary storage buffers in units of the page size (4KB). Because CICS optimizes the use of the storage obtained, TS may allocate more buffers than you specify, depending on the control interval (CI) size you have defined for the auxiliary temporary storage data set.

For example, if the CI size is 2048, and you specify 3 buffers (the default number), CICS actually allocates 4 buffers. This is because 2 pages (8192 bytes) are required to obtain sufficient storage for three 2048-byte buffers, a total of 6144 bytes, which would leave 2048 bytes of spare storage in the second page. In this case, CICS allocates another 2048-byte buffer to use all of the 8192 bytes obtained. In this way CICS makes use of storage that would otherwise be unavailable for any other purpose.

decimal-value-2

Specifies that the number of VSAM strings to be allocated for the use of auxiliary temporary storage. The value must be in the range 1 through 255, and must not exceed the value specified in decimal-value-1. The default value is 3.

For example, TS=(COLD,8,5) specifies 8 buffers and 5 strings.

The operands of the TS parameter are positional. You must code commas to indicate missing operands if others follow. For example, TS=(,8) specifies the number of buffers and allows the other operands to default.

TSMGSET={4|number}

Specifies that the number of entries for which dynamic storage is allocated for storing pointers to records put to a temporary storage message set. When the entries are used, space is acquired for the same number of entries as many times as required to accommodate the total number of records in the queue. The range is 4 through 100.

TST={NO|YES|xx}

Specifies the temporary storage table suffix, for more information see page 165.

For information about coding the macros for this table, see the CICS Resource Definition Guide

UDSASZE={0K|number}

Specifies the size of the UDSA. The default size is 0 indicating that the DSA size can change dynamically. A non-zero value indicates that the DSA size is fixed.

number

Specify number as an amount of storage in the range 0 to 16,777,215 bytes in multiples of 262,144 bytes (256KB). If the size specified is not a multiple of 256KB, CICS rounds the value up to the next multiple.

You can specify number in bytes (for example, 4,194,304), or as a whole number of kilobytes (for example, 4096K), or a whole number of megabytes (for example, 4M).

Restrictions

You can code the UDSASZE parameter in PARM, SYSIPT and CONSOLE only.

USERTR={ON|OFF}

To set the master user trace flag on or off. If the user trace flag is off, the user trace facility is disabled, and EXEC CICS ENTER TRACENUM commands receive an INVREQ condition if EXCEPTION is not specified. If the program does not handle this condition the transaction will abend with an abend code of AEIP.

For programming information about the user trace facility using EXEC CICS ENTER TRACENUM commands, see the CICS Application Programming Reference manual.

USRDELAY={30|number}

Specify the maximum time, in the range 0 through 10080 minutes (up to 7 days), that an eligible userid and its associated attributes are to be retained in the user table if the userid is unused. An entry in the user table for a userid that is retained during the delay period can be reused.

The userids eligible for reuse within the USRDELAY period are any that are:
- Received from remote systems
- Specified on SECURITYNAME in RDO CONNECTION definitions
- Specified on USERID in RDO SESSIONS definitions
- Specified on USERID on DFHDCT TYPE=INTRA definitions
- Specified to be used for non-terminal STARTed transactions.

Within the USRDELAY period, a userid in any one of these categories can be reused in one of the other categories, provided the request for reuse is qualified with the same qualifiers. If a userid is qualified by different group id, APPLID, or terminal id, a retained entry is not reused.

If a userid is unused for more than the USRDELAY limit, it is removed from the system, and the message DFHUS0200 is issued. You can suppress this message in an XMEOUT global user exit program. If you specify USRDELAY=0, all eligible userids are deleted immediately after use, and the message DFHUS0200 is not issued. Do not code USRDELAY=0 if this CICS region communicates with other CICS regions and:
- ATTACHSEC=IDENTIFY is specified on the CONNECTION definitions for the connections used
- The connections used carry high volumes of transaction routing or function shipping activity.

You should specify a value that gives the optimum level of performance for your CICS environment.

Note: If a value, other than 0, is specified for USRDELAY, the ability to change the user’s attributes or revoke the userid becomes more difficult because the userid and its attributes are retained in the region until the USRDELAY value has expired. For example, if you have specified USRDELAY=30 for a userid, but that userid continues to run transactions every 25 minutes, the USRDELAY value will never expire and any changes made to the userid will never come into effect.

When running a remote transaction, a userid remains signed-on to the remote CICS region (after the conversation associated with the first attach request is complete) until the delay specified by USRDELAY has elapsed since the last transaction associated with the attach request for the userid has completed. When this event occurs, the userid is removed from the remote CICS region.

VTAM={YES|NO}

Code VTAM=NO only if you do not use the VTAM access method. The default is VTAM=YES.

VTPREFIX={\|character}

Specifies that the first character to be used for the terminal identifiers (termids) of autoinstalled virtual terminals. Virtual terminals are used by the External Presentation Interface (EPI) and terminal emulator functions of the CICS Client products.

Termids generated by CICS for autoinstalled Client terminals consist of a 1-character prefix and a 3-character suffix. The default prefix is '\'. The suffix can have the values 'AAA' through '999'. That is, each character in the suffix can have the value 'A' through 'Z' or '0' through '9'. The first suffix generated by CICS has the value 'AAA'. This is followed by 'AAB', 'AAC', ... 'AAZ', 'AA0', 'AA1', and so on, up to '999'.

Each time a Client virtual terminal is autoinstalled, CICS generates a 3-character suffix that it has not recorded as being in use.

By specifying a prefix, you can ensure that the termids of Client terminals autoinstalled on this system are unique in your transaction routing network. This prevents the conflicts that could occur if two or more terminal-owning regions (TORs) ship definitions of Client virtual terminals to the same application-owning region (AOR).

If such a naming conflict does occur—that is, if a Client virtual terminal is shipped to an AOR on which a remote terminal of the same name is already installed—the autoinstall user program is invoked in the AOR. Your user program can resolve the conflict by allocating an alias terminal identifier to the shipped definition. (For details of writing an autoinstall user program to install shipped definitions, see the CICS Customization Guide). However, you can avoid potential naming conflicts by specifying a different prefix, reserved for virtual terminals, on each TOR on which Client virtual terminals are to be installed.

You must not use the characters + – * < > = { } or blank.

Notes:

1. The autoinstall user program is not called at install of Client terminals, so cannot be used to specify termids.
2. When specifying a prefix, ensure that termids generated by CICS for Client terminals do not conflict with those generated by your autoinstall user program for user terminals, or with the names of any other terminals or connections.
3. Client terminal definitions are not recovered after a restart. Immediately after a restart, no Client terminals are in use, so when CICS generates suffixes it begins again with 'AAA'. This means that CICS does **not** always generate the same termid for any given Client terminal. This in turn means that server applications should not assume that a particular CICS-generated termid always equates to a particular Client terminal.
4. Clients can override CICS-generated termids.

For further information about Client virtual terminals, see the *CICS Family: Communicating from CICS on System/390* manual.

WEBDELAY=(5|time_out,60|keep_time)
Specifies two Web delay periods:

1. A time-out period. The maximum time, in minutes, in the range 1-60, that a transaction started through the Web 3270 bridge interface, is allowed to remain in terminal wait state before it is automatically purged by CICS.
2. The terminal keep time. The time, in minutes, in the range 1-6000, during which state data is kept for a CICS Web 3270 bridge transaction, before CICS performs clean-up.

WRKAREA={512|number}
Specifies the number of bytes to be allocated to the common work area (CWA). This area, for use by your installation, is initially set to binary zeros, and is available to all programs. It is not used by CICS. The maximum size for the CWA is 3584 bytes.

XAPPC={NO|YES}
Specifies whether ESM security can be used when establishing APPC sessions.

NO
ESM session security cannot be used. Only the BINDPASSWORD (defined to CICS for an APPC connection) is checked.

YES
ESM session security can be used.

If you specify BINDSECURITY=YES for a particular APPC connection, a request to the ESM is issued to extract the security profile. If the profile exists, it is used to bind the session. If it does not exist, only the BINDPASSWORD (defined to CICS for the connection) is checked.

Note: If you specify XAPPC=YES, the external security manager that you use must support the APPCLU general resource class, otherwise CICS fails to initialize.

Restrictions

You can code the XAPPC parameter in the SIT, PARM, or SYSIPT only.

XCMD={NO|name|YES}
specifies whether you want CICS to perform command security checking, and, optionally, the ESM resource class name in which you have defined the command security profiles. If you specify YES, or an ESM resource class name, CICS calls the ESM to verify that the userid associated with a transaction is authorized to use a CICS command for the specified resource. Such checking is performed every time a transaction tries to use a COLLECT, CREATE, DISABLE, DISCARD, ENABLE, EXTRACT, INQUIRE, PERFORM, RESYNC, or SET command, or any of the FEPI commands, for a resource.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the CMDSEC(YES) option on the RDO TRANSACTION resource definition.

For information about preparing for and using security with CICS, see the CICS Security Guide.

NO
CICS does not perform any command security checks, allowing any user to use commands that would be subject to those checks.

name
CICS calls the ESM using the specified resource class name, prefixed by C or V, to verify that the userid associated with a transaction is authorized to use a CICS command for the specified resource. The resource class name is C*name* and the grouping class name is V*name*.

The resource class name specified must be 1 through 7 characters.

YES
CICS calls the ESM using the default class name of CICSCMD, prefixed by C or V, to check whether the userid associated with a transaction is authorized to use a CICS command for the specified resource. The resource class name is CCICSCMD and the grouping class name is VCICSCMD.

Restrictions

You can code the XCMD parameter in the SIT, PARM, or SYSIPT only.

XDCT={NO|name|YES}
Specifies whether you want CICS to perform transient data resource security checking. If you specify YES or an ESM resource class name, CICS calls the ESM to verify that the userid associated with a transaction is authorized to access the transient data destination. Such checking is performed every time a transaction tries to access a transient data destination.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the RDO TRANSACTION resource definition.

For information about preparing for and using security with CICS, see the CICS Security Guide.

NO
CICS does perform any transient data security checks, allowing any user to access any transient data destination.

name
CICS calls the ESM using the specified resource class name to check whether the userid associated with the transaction is authorized to access the specified destination. The resource class name is D*name* and the grouping class name is E*name*.

The resource class name specified must be 1 through 7 characters.

YES
CICS calls the ESM with the default CICS resource class name of CICSDCT, prefixed by D or E, to verify whether the userid associated with the transaction is authorized to access the specified destination.

The resource class name is DCICSDCT and the grouping class name is ECICSDCT.

Restrictions

You can code the XDCT parameter in the SIT, PARM, or SYSIPT only.

XFCT={NO|name|YES}

Specifies whether you want CICS to perform file resource security checking, and optionally specifies the ESM resource class name in which you have defined the file resource security profiles. If you specify YES, or an ESM resource class name, CICS calls the ESM to verify that the userid associated with a transaction is authorized to access file control-managed files. Such checking is performed every time a transaction tries to access a file managed by CICS file control.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the RDO TRANSACTION resource definitions.

For information about preparing for and using security with CICS, see the CICS Security Guide.

NO

CICS does not perform any file resource security checks, allowing any user to access any file.

name

CICS calls the ESM, using the specified resource class name, to verify that the userid associated with a transaction is authorized to access files referenced by the transaction. The resource class name is F*name* and the grouping class name is H*name*.

The resource class name specified must be 1 through 7 characters.

YES

CICS calls the ESM, using the default CICS resource class name of CICSFCT prefixed by F or H, to verify that the userid associated with a transaction is authorized to access files reference by the transaction. The resource class name is FCICSFCT and the grouping class name is HCICSFCT.

Restrictions

You can code the XFCT parameter in the SIT, PARM, or SYSIPT only.

XJCT={NO|name|YES}

Specifies whether you want CICS to perform journal resource security checking. If you specify YES, or an ESM resource class name, CICS calls the ESM to verify that the userid associated with a transaction is authorized to access the referenced journal. Such checking is performed every time a transaction tries to access a CICS journal.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the RDO TRANSACTION resource definitions.

For information about preparing for and using security with CICS, see the CICS Security Guide.

NO
CICS does not perform any journal resource security checks, allowing any user to access any CICS journal.

name
CICS calls the ESM, using the specified resource class name prefixed by J or K, to verify that the userid associated with a transaction is authorized to access CICS journals.

The resource class name specified must be 1 through 7 characters.

YES
CICS calls the ESM, using the default CICS resource class name of CICSJCT prefixed by a J or K, to check whether the userid associated with a transaction is authorized to access CICS journals referenced by the transaction. The resource class name is JCICSJCT and the grouping class name is KCICSJCT.

Restrictions

You can code the XJCT parameter in the SIT, PARM, or SYSIPT only.

XLT={NO|xx|YES}
Specifies a suffix for the transaction list table, for more information see page 165.

The XLT contains a list of transactions that can be attached during the first quiesce stage of system termination.

NO
Specifies that a transaction list table is not used.

xx Specifies that the transaction list table DFHXLTxx is used.

YES
Specifies that the default transaction list table, DFHXLT, is used.

For guidance information about coding the macros for this table, see the CICS Resource Definition Guide

XPCT={NO|name|YES}
Specifies whether you want CICS to perform started transaction resource security checking, and optionally specifies the name of the ESM resource class name in which you have defined the started task security profiles. If you specify YES, or an ESM resource class name, CICS calls the ESM to verify that the userid associated with a transaction is authorized to use started transactions and related EXEC CICS commands. Such checking is performed every time a transaction tries to use a started transaction or one of the EXEC CICS commands: COLLECT STATISTICS TRANSACTION, DISCARD TRANSACTION, INQUIRE TRANSACTION, or SET TRANSACTION.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the RDO TRANSACTION resource definitions.

For information about preparing for and using security with CICS, see the CICS Security Guide.

NO
CICS does not perform any started task resource security checks, allowing any user to use started transactions or related EXEC CICS commands.

name
CICS calls the ESM using the specified resource class name, to verify that the userid associated with a transaction is authorized to use started transactions or related EXEC CICS commands. The resource class name is ACICSPCT and the grouping class name is BCICSPCT.

The resource class name specified must be 1 through 7 characters.

YES
CICS calls the ESM, using the default CICS resource class name CICSPCT prefixed with A or B, to verify that the userid associated with a transaction is authorized to use started transactions or related EXEC CICS commands.

The resource class name is ACICSPCT and the grouping class name is BCICSPCT.

Restrictions

You can code the XPCT parameter in the SIT, PARM, or SYSIPT only.

XPPT={NO|name|YES}
Specifies that CICS is to perform application program resource security checks, and optionally specifies the ESM resource class name in which you have defined the program resource security profiles. Such checking is performed every time a transaction tries to invoke another program by using one of the CICS commands: LINK, LOAD, or XCTL.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the RDO TRANSACTION resource definitions.

For information about preparing for and using security with CICS, see the CICS Security Guide.

NO
CICS does not perform any application program authority checks, allowing any user to use LINK, LOAD, or XCTL commands to invoke other programs.

name
CICS calls the ESM, with the specified resource class name prefixed by M or N, to verify that the userid associated with a transaction is authorized to use LINK, LOAD, or XCTL commands to invoke other programs. The resource class name is M*name* and the grouping class name is N*name*.

The resource class name specified must be 1 through 7 characters.

YES
CICS calls the ESM, using the default resource class name prefixed by M or N, to verify that the userid associated with a transaction is authorized to use LINK, LOAD, or XCTL commands to invoke other programs. The resource class name is MCICSPPT and the grouping class name is NCICSPPT.

Restrictions

You can code the XPPT parameter in the SIT, PARM, or SYSIPT only.

XPSB={NO|name|YES}

Specifies whether you want CICS to perform program specification block (PSB) security checking, and optionally specifies the ESM resource class name in which you have defined the PSB security profiles. If you specify YES, or an ESM resource class name, CICS calls the ESM to check that the userid associated with a transaction is authorized to access PSBs (which describe databases and logical message destinations used by application programs). Such checking is performed every time a transaction tries to access a PSB.

The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the RDO TRANSACTION resource definitions.

For information about preparing for and using security with CICS, see the CICS Security Guide.

NO

CICS does not perform any PSB resource security checks, allowing any user to access any PSB.

name

CICS calls the ESM using the specified resource class name prefixed by P or Q, to verify that the userid associated with a transaction is authorized to access PSBs. The resource class name is P*name* and the grouping class name is Q*name*.

The resource class name specified must be 1 through 7 characters.

YES

CICS calls the ESM, using the default resource class name CICSPSB prefixed by P or Q, to verify that the userid associated with a transaction is authorized to access PSBs. The resource class name is PCICSPSB and the grouping class name is QCICSPSB.

Restrictions

You can code the XPSB parameter in the SIT, PARM, or SYSIPT only.

XRF={NO|YES}

You must specify YES if you want XRF support to be included in the CICS region. If the CICS region is started with the START=STANDBY system initialization parameter specified, the CICS region is the **alternate CICS region**. If the CICS region is started with the START=AUTO or START=COLD system initialization parameter specified, the CICS region is the **active CICS region**. The active CICS region signs on as such to the CICS availability manager. For background information about XRF, see the CICS XRF Guide.

Note: You must code JCT=xx|YES, if you are using XRF.

XRFSOFF={NOFORCE|FORCE}

Specifies whether all users signed-on to the active CICS region are to remain signed-on following a takeover. This parameter is only applicable if you also code XRF=YES as a system initialization parameter.

NOFORCE

Code NOFORCE to allow CICS to determine sign-off according to the option set in either of:

- The CICS segment of the ESM database
- The RDO TYPETERM resource definition for the user's terminal.

Note: For a terminal to remain signed-on after an XRF takeover, NOFORCE must be specified in the SIT, ESM database, and the terminal's RDO TYPETERM resource definition.

FORCE

Code FORCE if you want ***all*** terminal users to be signed off in the event of a takeover by an alternate CICS region, regardless of individual options set in the ESM database or in the terminals' RDO TYPETERM resource definition.

For information about the XRFSOFF option of the RDO TYPETERM resource definition, see the CICS Resource Definition Guide.

XRFSTME={5|decimal-value}

Specifies, in minutes, a time-out delay interval for users who are still signed-on when an XRF takeover occurs.

If you have specified NOFORCE in the ESM database, and in the terminals' RDO TYPETERM resource definition, and the takeover takes longer than the time specified in the XRFSTME, all users who are still signed-on after takeover are signed off.

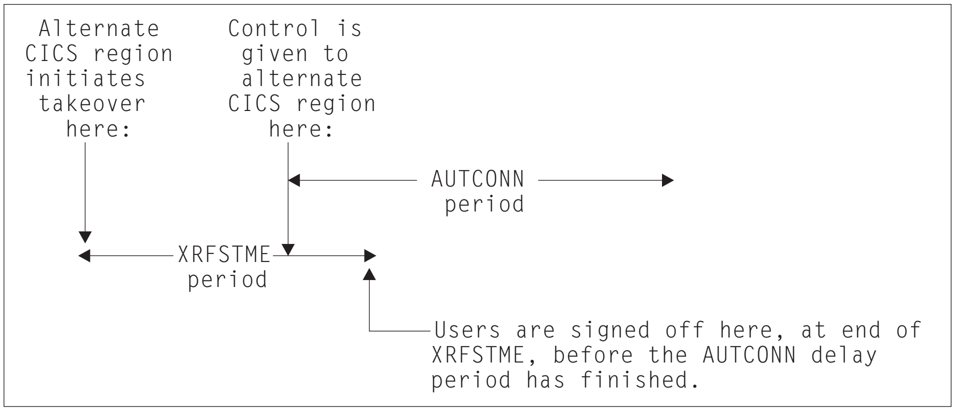
5 The default value is five minutes.

decimal-value

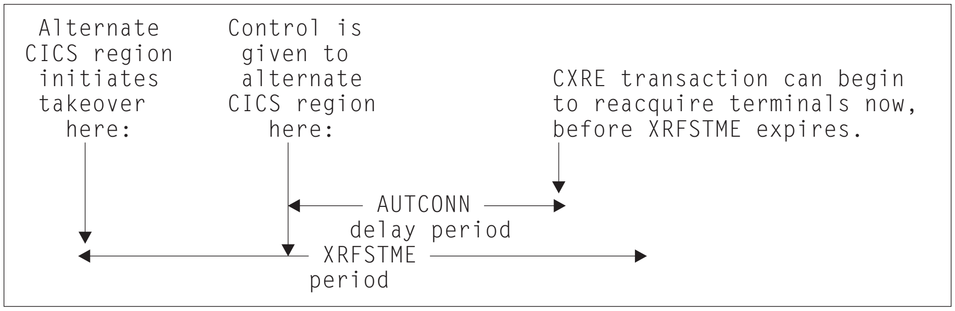
Code a value in the range 0 through 60 for the number of minutes CICS permits users to remain signed on during the takeover period. The takeover period is the time from when the takeover is initiated to the time at which CICS is ready to process user transactions. If the takeover takes longer than the specified period, all users signed-on at the time the takeover was initiated are signed-off.

A value of 0 specifies that there is no time-out delay, and terminals are signed off as soon as takeover commences, which means that XRFSTME=0 has the same effect as coding XRFSOFF=FORCE.

For non-XRF-capable terminals, take into account any AUTCONN delay period when setting the value for XRFSTME. (See the description of the AUTCONN parameter on page 178.) You may need to increase the XRFSTME value to allow for the delay to the start of the CXRE transaction imposed by the AUTCONN parameter; otherwise, terminals may be signed-off too early. For example:


Alternate CICS region initiates takeover here:
Control is given to alternate CICS region here:
AUTCONN period
XRFSTME period
Users are signed off here, at end of XRFSTME, before the AUTCONN delay period has finished.

You can avoid this situation by extending the XRFSTME period to exceed the AUTCONN period. For example:


Alternate CICS region initiates takeover here:
Control is given to alternate CICS region here:
CXRE transaction can begin to reacquire terminals now, before XRFSTME expires.
AUTCONN delay period
XRFSTME period

XRFTODI={30|number}

Use the XRFTODI parameter, in the SIT for an alternate XRF system, to specify, in seconds, the takeover delay interval. The minimum time value is 5 seconds. The alternate system has to ensure that the active system has been canceled before it can take over the resources owned by the active. Because the cancelation process is not fully automatic (for example, the primary CPC fails), the XRFTODI value specifies the interval before the operator becomes involved.

XSWITCH=([{0|1-254|NO}][,{????????|progname}][,{A|B}])

Specifies that the XSWITCH parameter defines a programmable terminal switching unit, which may be used with mid-range 2-CPC XRF systems instead of a communication controller. The program defined on the XSWITCH parameter instructs the unit to switch terminal lines to the active system’s CPC at start up, and to the alternate system’s CPC at take over.

0|1-254

Specifies that the logical unit to which the switch is assigned. 0 is the default.

????????|progname

Specifies that the user-written program that issues commands to the switching unit. The program is device-dependent, but has a standard interface that follows normal subroutine conventions.

A|B

Specifies that the CEC to which the terminal lines are to be directed. A is the active CPC and B is the alternate CPC,

XTRAN={YES|name|NO}

Specifies whether you want CICS to perform transaction-attach security checking, and optionally specifies the ESM resource class name in which you have defined the transaction security profiles. If you specify YES, or an ESM resource class name, CICS calls the ESM to verify that the userid associated with the transaction is permitted to run the transaction.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter and specified the RESSEC(YES) option on the RDO TRANSACTION resource definitions.

YES

CICS calls the ESM, using the default CICS resource class name of CICSTRN prefixed by T or G, to verify that the userid associated with the transaction is authorized to run the transaction. The resource class name is TCICSTRN and the grouping class name is GCICSTRN.

name

CICS calls the ESM, using the specified resource class name prefixed by T or G, to verify that the userid associated with the transaction is authorized to run the transaction. The resource class name is T*name* and the corresponding grouping class name is G*name*.

The name specified must be 1 through 7 characters.

NO

CICS does not perform any transaction-attach security checks, allowing any user to run any transaction.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter.

Restrictions

You can code the XTRAN parameter in the SIT, PARM, or SYSIPT only.

XTST={NO|name|YES}

specifies whether you want CICS to perform temporary storage security checking, and optionally specifies the ESM resource class name in which you have defined the temporary storage security profiles. If you specify YES or an ESM resource class name, CICS calls the ESM to verify that the userid associated with a temporary storage request is authorized to access the referenced temporary storage queue.

Note: The checking is performed only if you have specified YES for the SEC system initialization parameter, specified the RESSEC option on the RDO TRANSACTION resource definitions, and specified a DFHTYPE=SECURITY macro for the queue in the temporary storage table (TST).

NO

CICS does not perform any temporary storage security checks, allowing any user to access any temporary storage queue.

name

CICS calls the ESM, using the specified resource class name prefixed by S or U, to verify that the userid associated with a transaction is is authorized to access temporary storage queues.

The name specified must be 1 through 7 characters.

YES

CICS calls the ESM, using the default CICS resource class name of CICSTST prefixed by S or U, to verify that the userid associated with the transaction is authorized to access temporary storage queues referenced by the transaction. The resource class name is SCICSTST and the corresponding grouping class name is UCICSTST.

Restrictions

You can code the XTST parameter in the SIT, PARM, or SYSIPT only.

XUSER={NO|YES}

Specifies whether or not CICS is to perform surrogate user checks.

NO

Specifies that CICS is not to perform any surrogate user checking.

YES

specifies that CICS is to perform surrogate user checking in all those situations that permit such checks to be made (for example, on EXEC CICS START commands without an associated terminal). For information about the various circumstances in which CICS performs surrogate user checks, see the CICS Security Guide.

Restrictions

You can code the XUSER parameter in the SIT, PARM, or SYSIPT only.

Part 9. Trace entries

Chapter 21. Trace entries

Document handler AP trace points

Table 14. document handler

Point ID	Module	Lvl	Type	Data	
AP F920	DFHEIDH	AP 2	Entry	**1**	EIEI parameter list
AP F921	DFHEIDH	AP 2	Exit	**1**	EIEI parameter list
AP F922	DFHEIDH	Exc	Invalid format	**1**	EIEI parameter list
AP F923	DFHEIDH	Exc	Invalid function	**1**	EIEI parameter list
AP F924	DFHEIDH	Exc	Invalid API function	**1**	EIEI parameter list

Socket domain

Table 15. Socket domain trace points

Point ID	Module	Lvl	Type	Data	
SO 0101	DFHSODM	SO 1	Entry	**1**	DMDM parameter list
SO 0102	DFHSODM	SO 1	Exit	**1**	DMDM parameter list
SO 0103	DFHSODM	Exc	Recovery	**1**	DMDM parameter list
				2	Kernel error data
SO 0104	DFHSODM	Exc	Invalid format	**1**	DMDM parameter list
SO 0105	DFHSODM	Exc	Invalid function	**1**	DMDM parameter list
SO 0106	DFHSODM	Exc	Exclusive lock error	**1**	DMDM parameter list
				2	LMLM parameter list
SO 0107	DFHSODM	Exc	Exclusive unlock error	**1**	DMDM parameter list
				2	LMLM parameter list
SO 0108	DFHSODM	Exc	Storage error	**1**	DMDM parameter list
SO 0109	DFHSODM	Exc	Shared lock error	**1**	DMDM parameter list
				2	LMLM parameter list
SO 0110	DFHSODM	Exc	Shared unlock error	**1**	DMDM parameter list
				2	LMLM parameter list
SO 0111	DFHSODM	Exc	Recovery release lock error	**1**	DMDM parameter list
				2	LMLM parameter list
SO 0112	DFHSODM	Exc	Dispatcher error	**1**	DMDM parameter list
SO 0113	DFHSODM	Exc	Directory manager error	**1**	DMDM parameter list
SO 0201	DFHSOCK	SO 1	Entry	**1**	SOCK parameter list
				2	Send : sent data
SO 0201	DFHSOCK	SO 1	Entry	**1**	SOCK parameter list
SO 0201	DFHSOCK	SO 2	Entry	**1**	SOCK parameter list
				2	Send : sent data
SO 0202	DFHSOCK	SO 1	Exit	**1**	SOCK parameter list
SO 0202	DFHSOCK	SO 2	Exit	**1**	SOCK parameter list
				2	Receive : received data
SO 0203	DFHSOCK	Exc	Recovery	**1**	SOCK parameter list
				2	Kernel error data
SO 0204	DFHSOCK	Exc	Invalid format	**1**	SOCK parameter list
SO 0205	DFHSOCK	Exc	Invalid function	**1**	SOCK parameter list
SO 0206	DFHSOCK	Exc	Exclusive lock error	**1**	SOCK parameter list
				2	LMLM parameter list
SO 0207	DFHSOCK	Exc	Exclusive unlock error	**1**	SOCK parameter list
				2	LMLM parameter list
SO 0208	DFHSOCK	Exc	Getmain error	**1**	SMGF parameter list

Table 15. Socket domain trace points (continued)

Point ID	Module	Lvl	Type	Data	
SO 0209	DFHSOCK	SO 2	Before asyncio	**1**	bpx_interface
				2	LTE
				3	STE
				4	AioCb
SO 0210	DFHSOCK	SO 2	After asyncio	**1**	bpx_interface
				2	LTE
				3	STE
				4	AioCb
SO 0211	DFHSOCK	Exc	Asyncio error	**1**	bpx_interface
				2	LTE
				3	STE
				4	AioCb
SO 0212	DFHSOCK	SO 2	Before select	**1**	bpx_interface
				2	LTE
				3	Read_List
				4	Write_List
				5	Exception_List
				6	ECB
SO 0213	DFHSOCK	SO 2	After select	**1**	bpx_interface
				2	LTE
				3	Read_List
				4	Write_List
				5	Exception_List
				6	ECB
SO 0214	DFHSOCK	Exc	Select error	**1**	bpx_interface
				2	LTE
				3	Read_List
				4	Write_List
				5	Exception_List
				6	ECB
SO 0215	DFHSOCK	SO 2	Before socket	**1**	bpx_interface
				2	LTE
SO 0216	DFHSOCK	SO 2	After socket	**1**	bpx_interface
				2	LTE
SO 0217	DFHSOCK	Exc	Socket error	**1**	bpx_interface
				2	LTE
SO 0218	DFHSOCK	SO 2	Before bind	**1**	bpx_interface
				2	LTE
				3	Sockaddr
SO 0219	DFHSOCK	SO 2	After bind	**1**	bpx_interface
				2	LTE
				3	Sockaddr
SO 0220	DFHSOCK	Exc	Bind error	**1**	bpx_interface
				2	LTE
				3	Sockaddr
SO 0221	DFHSOCK	SO 2	Before listen	**1**	bpx_interface
				2	LTE
SO 0222	DFHSOCK	SO 2	After listen	**1**	bpx_interface
				2	LTE
SO 0223	DFHSOCK	Exc	Listen error	**1**	bpx_interface
				2	LTE
SO 0224	DFHSOCK	SO 2	Before accept	**1**	bpx_interface
				2	LTE
				3	STE
				4	Sockaddr

Table 15. Socket domain trace points (continued)

Point ID	Module	Lvl	Type	Data	
SO 0225	DFHSOCK	SO 2	After accept	**1**	bpx_interface
				2	LTE
				3	STE
				4	Sockaddr
SO 0226	DFHSOCK	Exc	Accept error	**1**	bpx_interface
				2	LTE
				3	STE
				4	Sockaddr
SO 0227	DFHSOCK	SO 2	Before getclientid	**1**	bpx_interface
				2	LTE
				3	Clientid
SO 0228	DFHSOCK	SO 2	After getclientid	**1**	bpx_interface
				2	LTE
				3	Clientid
SO 0229	DFHSOCK	Exc	Getclientid error	**1**	bpx_interface
				2	LTE
				3	Clientid
SO 0231	DFHSOCK	SO 2	Before takesocket	**1**	bpx_interface
				2	LTE
				3	STE
				4	Clientid
SO 0232	DFHSOCK	SO 2	After takesocket	**1**	bpx_interface
				2	LTE
				3	STE
				4	Clientid
SO 0233	DFHSOCK	Exc	Takesocket error	**1**	bpx_interface
				2	LTE
				3	STE
				4	Clientid
SO 0234	DFHSOCK	SO 2	Before givesocket	**1**	bpx_interface
				2	LTE
				3	STE
				4	Clientid
SO 0235	DFHSOCK	SO 2	After givesocket	**1**	bpx_interface
				2	LTE
				3	STE
				4	Clientid
SO 0236	DFHSOCK	Exc	Givesocket error	**1**	bpx_interface
				2	LTE
				3	STE
				4	Clientid
SO 0237	DFHSOCK	SO 2	Before close	**1**	bpx_interface
				2	LTE
				3	STE
SO 0238	DFHSOCK	SO 2	After close	**1**	bpx_interface
				2	LTE
				3	STE
SO 0239	DFHSOCK	Exc	Close error	**1**	bpx_interface
				2	LTE
				3	STE
SO 0240	DFHSOCK	SO 2	Before setsockopt	**1**	bpx_interface
				2	LTE
				3	option_data
SO 0241	DFHSOCK	SO 2	After setsockopt	**1**	bpx_interface
				2	LTE
				3	option_data

Table 15. Socket domain trace points (continued)

Point ID	Module	Lvl	Type	Data	
SO 0242	DFHSOCK	Exc	Setsockopt error	**1**	bpx_interface
				2	LTE
				3	option_data
SO 0243	DFHSOCK	Exc	Socket in use	**1**	SOCK parameter list
SO 0244	DFHSOCK	Exc	Attach error	**1**	STE
				2	LTE
SO 0245	DFHSOCK	SO 2	IO wait posted for termination	**1**	STE
SO 0246	DFHSOCK	Exc	Unknown session token	**1**	SOCK parameter list
SO 0247	DFHSOCK	Exc	IO error	**1**	SOCK parameter list
SO 0248	DFHSOCK	Exc	DSSR wait error	**1**	SOCK parameter list
SO 0249	DFHSOCK	SO 2	Immclose requested		
SO 0250	DFHSOCK	SO 2	Register requested		
SO 0251	DFHSOCK	SO 2	Deregister requested		
SO 0252	DFHSOCK	SO 2	Quiesce requested		
SO 0253	DFHSOCK	SO 2	Terminate requested		
SO 0254	DFHSOCK	Exc	Unknown post code	**1**	bpx_interface
				2	LTE
				3	ECB
SO 0255	DFHSOCK	SO 2	Before gethostname	**1**	bpx_interface
				2	Name
SO 0256	DFHSOCK	SO 2	After gethostname	**1**	bpx_interface
				2	Name
SO 0257	DFHSOCK	Exc	Gethostname error	**1**	bpx_interface
				2	Name
SO 0258	DFHSOCK	SO 2	TCP/IP inactive		
SO 0260	DFHSOCK	SO 2	Listen subroutine entry	**1**	SOCK parameter list
SO 0261	DFHSOCK	SO 2	Listen subroutine exit	**1**	SOCK parameter list
SO 0262	DFHSOCK	Exc	EIO received	**1**	bpx_interface
				2	LTE
				3	Read_List
				4	Write_List
				5	Exception List
				6	ECB
SO 0263	DFHSOCK	Exc	Accept EIO received	**1**	bpx_interface
				2	LTE
				3	STE
				4	Sockaddr
SO 0270	DFHSOCK	Exc	Shared lock error	**1**	SOCK parameter list
				2	LMLM parameter list
SO 0271	DFHSOCK	Exc	Shared unlock error	**1**	SOCK parameter list
				2	LMLM parameter list
SO 0272	DFHSOCK	Exc	Recovery release lock error	**1**	SOCK parameter list
				2	LMLM parameter list
SO 0273	DFHSOCK	Exc	Unknown KE error code	**1**	SOCK parameter list
SO 0274	DFHSOCK	Exc	Asyncio function error	**1**	bpx_interface
				2	LTE
				3	STE
				4	AioCb
SO 0275	DFHSOCK	Exc	Getmain for STE failure		
SO 0280	DFHSOCK	SO 2	Before iwmsrsrg	**1**	iwm_parameters
SO 0281	DFHSOCK	SO 2	After iwmsrsrg	**1**	iwm_parameters
SO 0282	DFHSOCK	Exc	Iwmsrsrg error	**1**	iwm_parameters
SO 0283	DFHSOCK	SO 2	Before iwmsrdrs	**1**	iwm_parameters
SO 0284	DFHSOCK	SO 2	After iwmsrdrs	**1**	iwm_parameters
SO 0285	DFHSOCK	Exc	Iwmsrdrs error	**1**	iwm_parameters

Table 15. Socket domain trace points (continued)

Point ID	Module	Lvl	Type	Data	
SO 0290	DFHSOCK	SO 1	Initialize timer entry		
SO 0291	DFHSOCK	SO 1	Initialize timer exit		
SO 0292	DFHSOCK	SO 1	Timer event		
SO 0293	DFHSOCK	SO 1	Monitor data put entry		
SO 0294	DFHSOCK	SO 1	Monitor data put exit		
SO 0295	DFHSOCK	SO 1	Cancel timer entry		
SO 0296	DFHSOCK	SO 1	Cancel timer exit		
SO 0297	DFHSOCK	SO 2	Connection count increment	**1**	Tcpipservice name
				2	New connection count
SO 0298	DFHSOCK	SO 2	Connection count decrement	**1**	Tcpipservice name
				2	New connection count
SO 0299	DFHSOCK	SO 2	Asyncio wakeup	**1**	LTE
				2	STE
				3	AioRv
				4	AioRc
				5	AioRsn
SO 029A	DFHSOCK	SO 2	Before sigprocmask	**1**	bpx_interface
				2	How
				3	New_signal_mask
				4	Old_signal_mask
SO 029B	DFHSOCK	SO 2	After sigprocmask	**1**	bpx_interface
				2	How
				3	New_signal_mask
				4	Old_signal_mask
SO 029C	DFHSOCK	Exc	Sigprocmask error	**1**	bpx_interface
				2	How
				3	New_signal_mask
				4	Old_signal_mask
SO 0301	DFHSORD	SO 1	Entry	**1**	SORD parameter list
SO 0302	DFHSORD	SO 1	Exit	**1**	SORD parameter list
SO 0303	DFHSORD	Exc	Recovery	**1**	SORD parameter list
				2	Kernel error data
SO 0304	DFHSORD	Exc	Invalid format	**1**	SORD parameter list
SO 0305	DFHSORD	Exc	Invalid function	**1**	SORD parameter list
SO 0306	DFHSORD	Exc	Lock error	**1**	SORD parameter list
				2	LMLM parameter list
SO 0307	DFHSORD	Exc	Unlock error	**1**	SORD parameter list
				2	LMLM parameter list
SO 0308	DFHSORD	Exc	Storage error	**1**	SMGF parameter list
SO 0401	DFHSOIS	SO 1	Entry	**1**	SOIS parameter list
SO 0402	DFHSOIS	SO 1	Exit	**1**	SOIS parameter list
SO 0403	DFHSOIS	Exc	Recovery	**1**	SOIS parameter list
				2	Kernel error data
SO 0404	DFHSOIS	Exc	Invalid format	**1**	SOIS parameter list
SO 0405	DFHSOIS	Exc	Invalid function	**1**	SOIS parameter list
SO 0406	DFHSOIS	Exc	Exclusive lock error	**1**	SOIS parameter list
				2	LMLM parameter list
SO 0407	DFHSOIS	Exc	Exclusive unlock error	**1**	SOIS parameter list
				2	LMLM parameter list
SO 0408	DFHSOIS	Exc	Shared lock error	**1**	SOIS parameter list
				2	LMLM parameter list
SO 0409	DFHSOIS	Exc	Shared unlock error	**1**	SOIS parameter list
				2	LMLM parameter list
SO 0410	DFHSOIS	Exc	Recovery unlock error	**1**	SOIS parameter list
				2	LMLM parameter list

Table 15. Socket domain trace points (continued)

Point ID	Module	Lvl	Type	Data	
SO 0411	DFHSOIS	Exc	Storage error	**1**	SMGF parameter list
SO 0412	DFHSOIS	Exc	Unknown KE error code		
SO 0413	DFHSOIS	Exc	CEEPIPI error	**1**	c_interface
SO 0414	DFHSOIS	Exc	Gethostbyaddr error	**1**	c_interface
SO 0415	DFHSOIS	Exc	Gethostid error	**1**	c_interface
SO 0416	DFHSOIS	SO 2	Before CEEPIPI	**1** **2** **3**	c_interface pipi_table init_sub_dp : runopts
SO 0417	DFHSOIS	SO 2	After CEEPIPI	**1** **2** **3**	c_interface pipi_table init_sub_dp : runopts
SO 0418	DFHSOIS	Exc	Dispatcher error	**1**	SOIS parameter list
SO 0419	DFHSOIS	Exc	Dub error	**1**	dub_results
SO 0420	DFHSOIS	SO 2	Before sigprocmask	**1** **2** **3** **3**	bpx_interface How New_signal_mask Old_signal_mask
SO 0421	DFHSOIS	SO 2	After sigprocmask	**1** **2** **3** **3**	bpx_interface How New_signal_mask Old_signal_mask
SO 0422	DFHSOIS	Exc	Sigprocmask error	**1** **2** **3** **3**	bpx_interface How New_signal_mask Old_signal_mask
SO 0501	DFHSOUE	SO 1	Entry	**1**	SOUE parameter list
SO 0502	DFHSOUE	SO 1	Exit	**1**	SOUE parameter list
SO 0503	DFHSOUE	Exc	Recovery	**1** **2**	SOUE parameter list Kernel error data
SO 0504	DFHSOUE	Exc	Invalid function	**1**	SOUE parameter list
SO 0505	DFHSOUE	Exc	Invalid format	**1**	SOUE parameter list
SO 0506	DFHSOUE	Exc	Lock error	**1** **2**	SOUE parameter list LMLM parameter list
SO 0507	DFHSOUE	Exc	Unlock error recovery	**1** **2**	SOUE parameter list LMLM parameter list
SO 0508	DFHSOUE	Exc	Release lock error	**1** **2**	SOUE parameter list LMLM parameter list
SO 0509	DFHSOUE	Exc	Recovery release lock error	**1** **2**	SOUE parameter list LMLM parameter list
SO 050A	DFHSOUE	Exc	Unknown KE error code		
SO 0601	DFHSOAD	SO 1	Entry	**1**	SOAD parameter list
SO 0602	DFHSOAD	SO 1	Exit	**1**	SOAD parameter list
SO 0603	DFHSOAD	Exc	Recovery	**1** **2**	SOAD parameter list Kernel error data
SO 0604	DFHSOAD	Exc	Invalid format	**1**	SOAD parameter list
SO 0605	DFHSOAD	Exc	Invalid function	**1**	SOAD parameter list
SO 0606	DFHSOAD	Exc	Unlock error recovery	**1** **2**	SOAD parameter list LMLM parameter list
SO 0607	DFHSOAD	Exc	Storage error	**1**	SMGF parameter list
SO 0701	DFHSOTB	SO 1	Entry	**1**	SOTB parameter list
SO 0702	DFHSOTB	SO 1	Exit	**1**	SOTB parameter list
SO 0703	DFHSOTB	Exc	Recovery	**1** **2**	SOTB parameter list Kernel error data
SO 0704	DFHSOTB	Exc	Invalid format	**1**	SOTB parameter list
SO 0705	DFHSOTB	Exc	Invalid function	**1**	SOTB parameter list

Table 15. Socket domain trace points (continued)

Point ID	Module	Lvl	Type	Data	
SO 0706	DFHSOTB	Exc	Unlock error recovery	**1**	SOTB parameter list
				2	LMLM parameter list
SO 0707	DFHSOTB	Exc	Storage error	**1**	SMGF parameter list
SO 0801	DFHSOSE	SO 1	Entry	**1**	SOSE parameter list
SO 0802	DFHSOSE	SO 1	Exit	**1**	SOSE parameter list
SO 0803	DFHSOSE	Exc	Recovery	**1**	SOSE parameter list
				2	Kernel error data
SO 0804	DFHSOSE	Exc	Invalid format	**1**	SOSE parameter list
SO 0805	DFHSOSE	Exc	Invalid function	**1**	SOSE parameter list
SO 0806	DFHSOSE	Exc	Lock error	**1**	SOSE parameter list
				2	LMLM parameter list
SO 0807	DFHSOSE	Exc	Unlock error	**1**	SOSE parameter list
				2	LMLM parameter list
SO 0808	DFHSOSE	Exc	Storage error	**1**	SMGF parameter list
SO 0809	DFHSOSE	SO 2	Before CEEPIPI	**1**	gsk_interface
SO 080A	DFHSOSE	SO 2	After CEEPIPI	**1**	gsk_interface
SO 080B	DFHSOSE	Exc	CEEPIPI error	**1**	gsk_interface
SO 080C	DFHSOSE	Exc	GSK error	**1**	gsk_interface
SO 080D	DFHSOSE	Exc	Getmain error		
SO 080E	DFHSOSE	Exc	Repository error		
SO 080F	DFHSOSE	Exc	Directory error	**1**	SOSE parameter list
SO 0901	DFHSOXM	SO 1	Entry	**1**	XMXM parameter list
SO 0902	DFHSOXM	SO 1	Exit	**1**	XMXM parameter list
SO 0903	DFHSOXM	Exc	Invalid function	**1**	XMXM Parameter list
SO 0904	DFHSOXM	Exc	Invalid format	**1**	XMXM parameter list
SO 0905	DFHSOXM	Exc	Recovery	**1**	XMXM parameter list
				2	Kernel error data
SO 0A01	DFHSOST	SO 1	Entry	**1**	STST parameter list
SO 0A02	DFHSOST	SO 1	Exit	**1**	STST parameter list
SO 0A03	DFHSOST	Exc	Recovery	**1**	STST parameter list
				2	Kernel error data
SO 0A04	DFHSOST	Exc	Invalid format	**1**	STST parameter list
SO 0A05	DFHSOST	Exc	Invalid function	**1**	STST parameter list
SO 0A06	DFHSOST	Exc	Exclusive lock error	**1**	STST parameter list
				2	LMLM parameter list
SO 0A07	DFHSOST	Exc	Exclusive unlock error	**1**	STST parameter list
				2	LMLM parameter list
SO 0A08	DFHSOST	Exc	Shared lock error	**1**	STST parameter list
				2	LMLM parameter list
SO 0A09	DFHSOST	Exc	Shared unlock error	**1**	STST parameter list
				2	LMLM parameter list
SO 0A10	DFHSOST	Exc	Recovery release lock error	**1**	STST parameter list
				2	LMLM parameter list
SO 0A11	DFHSOST	Exc	Getmain error	**1**	SMGF parameter list
SO 0A12	DFHSOST	Exc	Unknown kernel error code	**1**	STST parameter list
				2	Kernel error data
SO 0A13	DFHSOST	Exc	Invalid parameters	**1**	STST parameter list
SO 0B01	DFHSOTI	SO 1	Entry	**1**	SO anchor block
SO 0B02	DFHSOTI	SO 1	Exit	**1**	SO anchor block
SO 0B03	DFHSOTI	SO 1	Before posting listener		
SO 0B04	DFHSOTI	SO 1	After posting listener		
SO 0B05	DFHSOTI	Exc	Recovery entry	**1**	SO anchor block
				2	Kernel error data
SO 0B06	DFHSOTI	Exc	Recovery exit	**1**	SO anchor block
				2	Kernel error data

Web domain trace points

Table 16. Web domain trace points

Point ID	Module	Lvl	Type	Data	
WB 0100	DFHWBDM	WB 1	Entry	**1**	WBDM parameter list
WB 0101	DFHWBDM	WB 1	Exit	**1**	WBDM parameter list
WB 0102	DFHWBDM	Exc	Recovery	**1**	WBDM parameter list
WB 0103	DFHWBDM	Exc	Invalid format	**1**	DMDM parameter list
WB 0104	DFHWBDM	Exc	Invalid function	**1**	DMDM parameter list
WB 0105	DFHWBDM	Exc	Release lock error	**1** **2**	DMDM parameter list LMLM parameter list
WB 0107	DFHWBDM	Exc	No storage for anchor block	**1**	DMDM parameter list
WB 0108	DFHWBDM	Exc	Get parameters failed	**1**	DMDM parameter list
WB 0200	DFHWBGP	WB 1	Entry	**1**	WBGP parameter list
WB 0201	DFHWBGP	WB 1	Exit	**1**	WBGP parameter list
WB 0202	DFHWBGP	Exc	Recovery	**1**	WBGP parameter list
WB 0203	DFHWBGP	Exc	Invalid format	**1**	WBGP parameter list
WB 0204	DFHWBGP	Exc	Invalid function	**1**	WBGP parameter list
WB 0300	DFHWBAP	WB 1	Entry	**1** **2**	WBAP parameter list Client codepage
WB 0301	DFHWBAP	WB 1	Exit	**1** **2**	WBAP parameter list Client codepage
WB 0302	DFHWBAP	Exc	Recovery	**1**	WBAP parameter list
WB 0303	DFHWBAP	Exc	Invalid format	**1**	WBAP parameter list
WB 0304	DFHWBAP	Exc	Invalid function	**1**	WBAP parameter list
WB 0305	DFHWBAP	Exc	Unlock error recovery	**1**	WBAP parameter list
WB 0410	DFHWBRQ	Event	Inbound HTTP request	**1**	HTTP request headers
WB 0411	DFHWBSR	Exc	TSQ put header failed	**1**	WRB
WB 0412	DFHWBRQ	Exc	TS put request body failed	**1**	WRB
WB 0413	DFHWBSR	Exc	No repository token returned	**1**	WRB
WB 0414	DFHWBSR	Event	Before call to Analyzer	**1**	Analyzer parameter list
WB 0415	DFHWBSR	Event	After call to Analyzer	**1**	Analyzer parameter list
WB 0416	DFHWBSR	Data	HTTP request body	**1**	HTTP request body
WB 0500	DFHWBSR	WB 1	Entry	**1**	WBSR parameter list
WB 0501	DFHWBSR	WB 1	Exit	**1**	WBSR parameter list
WB 0502	DFHWBSR	Exc	Invalid format	**1**	WBSR parameter list
WB 0503	DFHWBSR	Exc	Invalid function	**1**	WBSR parameter list
WB 0600	DFHWBXM	WB 1	Entry	**1**	XMAC parameter list
WB 0601	DFHWBXM	WB 1	Exit	**1**	XMAC parameter list
WB 0602	DFHWBXM	Exc	Invalid format	**1**	XMAC parameter list
WB 0603	DFHWBXM	Exc	Invalid function	**1**	XMAC parameter list
WB 0604	DFHWBXM	Exc	Recovery	**1**	XMAC parameter list
WB 4109	DFHWBLT	Exc	Trigger partner failed	**1**	WBST parameter list
WB 4654	DFHWBST	Exc	Directory browse error	**1** **2**	WBST parameter list DDBR parameter list
WB FF60	DFHWBQM	WB 1	Entry	**1**	WBQM parameter list
WB FF61	DFHWBQM	WB 1	Exit	**1**	WBQM parameter list
WB FF62	DFHWBQM	Exc	Recovery	**1**	WBQM parameter list
WB FF63	DFHWBQM	Exc	Invalid format	**1**	WBAP parameter list
WB FF64	DFHWBQM	Exc	Invalid function	**1**	WBQM parameter list
WB FF65	DFHWBQM	Exc	Invalid parameter	**1**	WBQM parameter list
WB FF66	DFHWBQM	Exc	TS queue error	**1**	WBQM parameter list

Bridge facility management

Table 17. Bridge facility management trace points

Point ID	Module	Lvl	Type	Data	
AP 2140	DFHBRFM	BR 1	Entry	**1**	BRFM parameter list
AP 2141	DFHBRFM	BR 1	Exit	**1**	BRFM parameter list
AP 2142	DFHBRFM	Exc	Invalid format	**1**	BRFM parameter list
AP 2143	DFHBRFM	Exc	Invalid function	**1**	BRFM parameter list
AP 2144	DFHBRFM	Exc	Invalid token	**1**	BRFM parameter list
AP 2145	DFHBRFM	Exc	Invalid template	**1**	BRFM parameter list
AP 2146	DFHBRFM	Exc	Token not current	**1**	BRFM parameter list
AP 2147	DFHBRFM	Exc	GETMAIN failed	**1**	BRFM parameter list
AP 2148	DFHBRFM	Exc	No free name	**1**	BRFM parameter list
AP 2149	DFHBRFM	Exc	Name create failed	**1**	BRFM parameter list
AP 214A	DFHBRFM	Exc	Recovery entered	**1**	BRFM parameter list
				2	Kernel error data
AP 214B	DFHBRFM	BR 1	Kept BFB expired	**1**	BFB name
				2	BFB index
				3	BFB expiry time
AP 214C	DFHBRFM	BR 1	Kept BFB used	**1**	BFB name
				2	BFB index
				3	BFB expiry time
AP 214D	DFHBRFM	BR 1	BFB added to keep chain	**1**	BFB name
				2	BFB index
				3	BFB expiry time
AP 214E	DFHBRFM	Exc	SNUS signon failed	**1**	BRFM parameter list
AP 2160	DFHBRMS	BR 1	Entry	**1**	Exec parameter list
AP 2161	DFHBRMS	BR 1	Exit	**1**	Exec parameter list
AP 2162	DFHBRMS	Exc	Recovery	**1**	Exec parameter list
				2	Kernel error data
AP 2163	DFHBRTC	BR 1	Entry	**1**	Exec parameter list
AP 2164	DFHBRTC	BR 1	Exit	**1**	Exec parameter list
AP 2165	DFHBRTC	Exc	Recovery	**1**	Exec parameter list
				2	Kernel error data
AP 2166	DFHBRIC	BR 1	Entry	**1**	Exec parameter list
AP 2167	DFHBRIC	BR 1	Exit	**1**	Exec parameter list
AP 2168	DFHBRIC	Exc	Recovery	**1**	Exec parameter list
				2	Kernel error data
AP 2169	DFHBRxx	Exc	Invalid function	**1**	Exec parameter list
AP 216A	DFHBRxx	Exc	Invalid group	**1**	Exec parameter list
AP 216B	DFHBRxx	Exc	Invalid EXEC function	**1**	Exec parameter list
AP 216C	DFHBRSP	BR 1	Entry	**1**	Exec parameter list
AP 216D	DFHBRSP	BR 1	Exit	**1**	Exec parameter list
AP 216E	DFHBRSP	Exc	Recovery	**1**	Exec parameter list
				2	Kernel error data
AP 216F	DFHBRMS	BR 2	Call user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2170	DFHBRMS	BR 2	Return from user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2171	DFHBRTC	BR 2	Call user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2172	DFHBRTC	BR 2	Return from user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2173	DFHBRIC	BR 2	Call user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2174	DFHBRIC	BR 2	Return from user exit	**1**	BRXA transaction area
				2	BRXA command area

Table 17. Bridge facility management trace points (continued)

Point ID	Module	Lvl	Type	Data	
AP 2175	DFHBRSP	BR 2	Call user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2176	DFHBRSP	BR 2	Return from user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2177	DFHBRSP	Exc	Syncpoint change	**1**	BRSP parameter list
				2	BRXA transaction area
				3	BRXA command area
AP 2178	DFHBRMS	BR 2	Call formatter	**1**	BRXA transaction area
				2	BRXA command area
AP 2179	DFHBRMS	BR 2	Formatter return	**1**	BRXA transaction area
				2	BRXA command area
AP 217A	DFHBRTC	BR 2	Call formatter	**1**	BRXA transaction area
				2	BRXA command area
AP 217B	DFHBRTC	BR 2	Formatter return	**1**	BRXA transaction area
				2	BRXA command area
AP 217C	DFHBRIC	BR 2	Call formatter	**1**	BRXA transaction area
				2;	BRXA command area
AP 217D	DFHBRIC	BR 2	Call formatter	**1**	BRXA transaction area
				2	BRXA command area

Bridge facility management 2

Table 18. Bridge facility management trace points

Point ID	Module	Lvl	Type	Data	
AP 2800	DFHBRAT	BR 1	Entry	**1**	BRAT parameter list
AP 2801	DFHBRAT	BR 1	Exit	**1**	BRAT parameter list
AP 2802	DFHBRAT	Exc	Invalid format	**1**	BRAT parameter list
AP 2803	DFHBRAT	Exc	Invalid function	**1**	BRAT parameter list
AP 2804	DFHBRAT	Exc	Recovery	**1**	BRAT parameter list
				2	Kernel error data
AP 2805	DFHBRAT	Exc	Xmat sno	**1**	BRAT parameter list
AP 2806	DFHBRAT	Exc	Xmxd sno	**1**	BRAT parameter list
AP 2807	DFHBRAT	Exc	Xsrc sno	**1**	BRAT parameter list
AP 2808	DFHBRAT	Exc	Usad sno	**1**	BRAT parameter list
AP 2809	DFHBRAT	Exc	Brfm allocate failed	**1**	BRAT parameter list
AP 280A	DFHBRAT	Exc	Tm locate failed	**1**	BRAT parameter list
AP 2820	DFHBRIQ	BR 1	Entry	**1**	BRIQ parameter list
AP 2821	DFHBRIQ	BR 1	Exit	**1**	BRIQ parameter list
AP 2822	DFHBRIQ	Exc	Invalid format	**1**	BRIQ parameter list
AP 2823	DFHBRIQ	Exc	Invalid function	**1**	BRIQ parameter list
AP 2824	DFHBRIQ	Exc	Recovery	**1**	BRIQ parameter list
				2	Kernel error data
AP 2840	DFHBRRM	BR 1	Entry	**1**	RMRO parameter list
AP 2841	DFHBRRM	BR 1	Exit	**1**	RMRO parameter list
AP 2842	DFHBRRM	Exc	Recovery	**1**	RMRO parameter list
				2	Kernel error data
AP 2847	DFHBRRM	BR 2	Call user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2848	DFHBRRM	BR 2	Return from user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2860	DFHBRXM	BR 1	Entry	**1**	XMAC parameter list
AP 2861	DFHBRXM	BR 1	Exit	**1**	XMAC parameter list
AP 2862	DFHBRXM	Exc	Invalid format	**1**	XMAC parameter list
AP 2863	DFHBRXM	Exc	Invalid function	**1**	XMAC parameter list

Table 18. Bridge facility management trace points (continued)

Point ID	Module	Lvl	Type	Data	
AP 2864	DFHBRXM	Exc	Recovery	**1**	XMAC parameter list
				2	Kernel error data
AP 2865	DFHBRXM	Exc	Smgf sno	**1**	XMAC parameter list
AP 2866	DFHBRXM	Exc	Xmxd sno	**1**	XMAC parameter list
AP 2867	DFHBRXM	BR 2	Call user exit	**1**	BRXA transaction area
				2	BRXA command area
AP 2868	DFHBRXM	BR 2	Return from user exit	**1**	BRXA transaction area
				2	BRXA command area

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 to any country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact Laboratory Counsel, MP151, IBM United Kingdom Limited, Hursley Park, Winchester, Hampshire, England SO21 2JN.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, U.S.A.

Programming interface information

This book is intended to help you use the external interfaces provided by the CICS Transaction Server for VSE/ESA. This book documents General-use Programming Interface and Associated Guidance Information provided by CICS.

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

This book also documents Product-sensitive Programming Interface and Associated Guidance Information and Diagnosis, Modification or Tuning Information provided by CICS.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of CICS. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface and Associated Guidance Information is identified, where it occurs, by an introductory statement to a chapter or section.

Diagnosis, Modification, or Tuning Information is provided to help you diagnose problems in your CICS system.

Note: Do not use this Diagnosis, Modification, or Tuning Information as a programming interface.

Diagnosis, Modification, or Tuning Information is identified, where it occurs, by an introductory statement to a chapter or section.

Trademarks and service marks

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

BookManager
CICS/VSE
DFS
IMS
OS/2
RACF
SAA

CICS
DB2
IBM
Language Environment
OS/390
RT
VTAM

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

LINUX is a registered trademark of Linus Torvalds and others.

Other company products, and service names, may be trademarks or service marks of others.

Bibliography

CICS Transaction Server for VSE/ESA Release 1 Library

Evaluation and planning	
CICS Release Guide	GC33-1645
CICS Migration Guide	GC33-1646
CICS Report Controller Planning Guide	SC33-1941
CICS Enhancements Guide	GC34-5763
General	
CICS Master Index	SC33-1648
CICS Trace Entries Handbook	SC34-5556
CICS User's Handbook	SX33-6101
CICS Glossary (softcopy only)	GC33-1649
Administration	
CICS System Definition Guide	SC33-1651
CICS Customization Guide	SC33-1652
CICS Resource Definition Guide	SC33-1653
CICS Operations and Utilities Guide	SC33-1654
CICS Supplied Transactions	SC33-1655
Programming	
CICS Application Programming Guide	SC33-1657
CICS Application Programming Reference	SC33-1658
CICS 4.1 Sample Applications Guide	SC33-1713
CICS Application Migration Aid Guide	SC33-1943
CICS System Programming Reference	SC33-1659
CICS Distributed Transaction Programming Guide	SC33-1661
CICS Front End Programming Interface User's Guide	SC33-1662
REXX Guide	SC34-5764
CICS Family: Client/Server Programming	SC33-1435
Diagnosis	
CICS Problem Determination Guide	GC33-1663
CICS Messages and Codes	SC33-6799
CICS Diagnosis Reference	LY33-6085
CICS Data Areas	LY33-6086
CICS Supplementary Data Areas	LY33-6087
Communication	
CICS Internet Guide	SC34-5765
CICS Intercommunication Guide	SC33-1665
CICS Family: Interproduct Communication	SC33-0824
CICS Family: Communicating from CICS on System/390	SC33-1697
Special topics	
CICS Recovery and Restart Guide	SC33-1666
CICS Performance Guide	SC33-1667
CICS Shared Data Tables Guide	GC33-1668
CICS Security Guide	SC33-1942
CICS External Interfaces Guide	SC33-1669
CICS XRF Guide	SC33-1671
CICS Report Controller User's Guide	SC34-5688
CICS Clients	
CICS Clients: Administration	SC33-1792
CICS Universal Clients Version 3 for OS/2: Administration	SC34-5450
CICS Universal Clients Version 3 for Windows: Administration	SC34-5449

CICS Universal Clients Version 3 for AIX: Administration	SC34-5348
CICS Universal Clients Version 3 for Solaris: Administration	SC34-5451
CICS Family: OO programming in C++ for CICS Clients	SC33-1923
CICS Family: OO programming in BASIC for CICS Clients	SC33-1671
CICS Family: Client/Server Programming	SC33-1435
CICS Transaction Gateway Version 3: Administration	SC34-5448

Books from VSE/ESA 2.6 base program libraries

VSE/ESA Version 2 Release 6

Book title	Order number
Administration	SC33-6705
Diagnosis Tools	SC33-6614
Extended Addressability	SC33-6621
Guide for Solving Problems	SC33-6710
Guide to System Functions	SC33-6711
Installation	SC33-6704
Licensed Program Specification	GC33-6700
Messages and Codes Volume 1	SC33-6796
Messages and Codes Volume 2	SC33-6798
Messages and Codes Volume 3	SC33-6799
Networking Support	SC33-6708
Operation	SC33-6706
Planning	SC33-6703
Programming and Workstation Guide	SC33-6709
System Control Statements	SC33-6713
System Macro Reference	SC33-6716
System Macro User's Guide	SC33-6715
System Upgrade and Service	SC33-6702
System Utilities	SC33-6717
TCP/IP User's Guide	SC33-6601
Turbo Dispatcher Guide and Reference	SC33-6797
e-business Connectors User's Guide	SC33-6719

High-Level Assembler Language (HLASM)

Book title	Order number
General Information	GC26-8261
Installation and Customization Guide	SC26-8263
Language Reference	SC26-8265
Programmer's Guide	SC26-8264

Language Environment for VSE/ESA (LE/VSE)

Book title	Order number
C Run-Time Library Reference	SC33-6689
C Run-Time Programming Guide	SC33-6688
Concepts Guide	GC33-6680
Debug Tool for VSE/ESA Fact Sheet	GC26-8925
Debug Tool for VSE/ESA Installation and Customization Guide	SC26-8798
Debug Tool for VSE/ESA User’s Guide and Reference	SC26-8797
Debugging Guide and Run-Time Messages	SC33-6681
Diagnosis Guide	SC26-8060
Fact Sheet	GC33-6679
Installation and Customization Guide	SC33-6682
LE/VSE Enhancements	SC33-6778
Licensed Program Specification	GC33-6683
Programming Guide	SC33-6684
Programming Reference	SC33-6685
Run-Time Migration Guide	SC33-6687
Writing Interlanguage Communication Applications	SC33-6686

VSE/ICCF

Book title	Order number
Adminstration and Operations	SC33-6738
User’s Guide	SC33-6739

VSE/POWER

Book title	Order number
Administration and Operation	SC33-6733
Application Programming	SC33-6736
Networking Guide	SC33-6735
Remote Job Entry User’s Guide	SC33-6734

VSE/VSAM

Book title	Order number
Commands	SC33-6731
User’s Guide and Application Programming	SC33-6732

VTAM for VSE/ESA

Book title	Order number
Customization	LY43-0063
Diagnosis	LY43-0065
Data Areas	LY43-0104
Messages and Codes	SC31-6493
Migration Guide	GC31-8072
Network Implementation Guide	SC31-6494
Operation	SC31-6495
Overview	GC31-8114
Programming	SC31-6496
Programming for LU6.2	SC31-6497
Release Guide	GC31-8090
Resource Definition Reference	SC31-6498

Books from VSE/ESA 2.6 optional program libraries

C for VSE/ESA (C/VSE)

Book title	Order number
C Run-Time Library Reference	SC33-6689
C Run-Time Programming Guide	SC33-6688
Diagnosis Guide	GC09-2426
Installation and Customization Guide	GC09-2422
Language Reference	SC09-2425
Licensed Program Specification	GC09-2421
Migration Guide	SC09-2423
User's Guide	SC09-2424

COBOL for VSE/ESA (COBOL/VSE)

Book title	Order number
Debug Tool for VSE/ESA Fact Sheet	GC26-8925
Debug Tool for VSE/ESA Installation and Customization Guide	SC26-8798
Debug Tool for VSE/ESA User's Guide and Reference	SC26-8797
Diagnosis Guide	SC26-8528
General Information	GC26-8068
Installation and Customization Guide	SC26-8071
Language Reference	SC26-8073
Licensed Program Specifications	GC26-8069
Migration Guide	GC26-8070
Migrating VSE Applications To Advanced COBOL	GC26-8349
Programming Guide	SC26-8072

DB2 Server for VSE

Book title	Order number
Application Programming	SC09-2889
Data Restore Guide	SC09-2991
Database Administration	SC09-2888
Database Services Utility	SC09-2983
Interactive SQL Guide and Reference	SC09-2990
Operation	SC09-2986
Performance Tuning Handbook	GC09-2987
Quick Reference	SC09-2988
SQL Reference	SC09-2989
Messages and Codes	GC09-2985
System Administration	SC09-2981

DL/I VSE

Book title	Order number
Application and Database Design	SH24-5022
Application Programming: CALL and RQDLI Interface	SH12-5411
Application Programming: High-Level Programming Interface	SH24-5009
Database Administration	SH24-5011
Diagnostic Guide	SH24-5002
General Information	GH20-1246
Guide for New Users	SH24-5001
Interactive Resource Definition and Utilities	SH24-5029
Library Guide and Master Index	GH24-5008
Licensed Program Specifications	GH24-5031
Low-level Code and Continuity Check Feature	SH20-9046
Library Guide and Master Index	GH24-5008
Messages and Codes	SH12-5414
Recovery and Restart Guide	SH24-5030
Reference Summary: CALL Program Interface	SX24-5103
Reference Summary: System Programming	SX24-5104
Reference Summary: HLPI Interface	SX24-5120
Release Guide	SC33-6211

PL/I for VSE/ESA (PL/I VSE)

Book title	Order number
Compile Time Messages and Codes	SC26-8059

Book title	Order number
Debug Tool For VSE/ESA User's Guide and Reference	SC26-8797
Diagnosis Guide	SC26-8058
Installation and Customization Guide	SC26-8057
Language Reference	SC26-8054
Licensed Program Specifications	GC26-8055
Migration Guide	SC26-8056
Programming Guide	SC26-8053
Reference Summary	SX26-3836

Screen Definition Facility II (SDF II)

Book title	Order number
VSE Administrator's Guide	SH12-6311
VSE General Introduction	SH12-6315
VSE Primer for CICS/BMS Programs	SH12-6313
VSE Run-Time Services	SH12-6312

TCP/IP for VSE/ESA

Documentation for TCP/IP for VSE/ESA is available by ordering the ″TCP/IP for VSE/ESA PDF Product Library″ product kit, SK2T-1336. **Not** all the books are separately orderable as they are only available in PDF form. The following titles are available on the product kit at the time of going to press:

Book title	Order number
TCP/IP for VSE/ESA — IBM Program Setup and Supplementary Information	SC33-6601
TCP/IP for VSE/ESA Flyer	G221-9030
TCP/IP for VSE/ESA Performance Considerations	n/a
TCP/IP for VSE/ESA LPS	GC33-6594
The Native TCP/IP forVSE	GC33-6594
TCP/IP Tutorial and Technical Overview	GG24-3376
VSE/ESA as a Web Server	SG24-2040
TCP/IP for VSE 1.4 Installation Guide	n/a
TCP/IP for VSE 1.4 User's Guide	n/a
TCP/IP for VSE 1.4 Commands	n/a
TCP/IP for VSE 1.4 Programmer's Reference	n/a
TCP/IP for VSE 1.4 Messages and Codes	n/a
TCP/IP for VSE 1.4 Optional Products	n/a

Information on the World Wide Web

Information about the hypertext transfer protocol (HTTP) and the hypertext markup language (HTML) is on the World Wide Web. URLs are provided in this book with the caveat that their permanence cannot be guaranteed.

HTTP/1.0

CICS supports HTTP/1.0. Unpredictable results can occur if you use HTTP/1.1–specific headers. For HTTP/1.0 information, consult the following:

- *Overview of HTTP*

  `http://www.w3.org/hypertext/WWW/Protocols/Overview.html`
- *Hypertext Transfer Protocol (HTTP/1.0)*

  `http://ds.internic.net/rfc/rfc1945.txt`

The following references are to information about the ISO 8859-1 (Latin-1) character set:

- *ISO 8859-1 National Character Set FAQ*

  `http://aliga.cesca.es:1025/%7Ezopcgp01/manuals/ISO8859-1.faq`
- *ISO 8859-1:1987* (ordering information)

  `http://www.iso.ch/cate/d16338.html`
- *ISO 8859-1 (Latin-1) Characters List*

  `http://www.utoronto.ca/webdocs/HTMLdocs/NewHTML/iso_table.html`
- Table of Latin-1 character glyphs

  `http://www.w3.org/pub/WWW/MarkUp/Wilbur/latin1.gif`

HTML

CICS has no dependency on the level of HTML used. There are many reference sites available on the World Wib Web. The most useful starting place for HTML reference material is:

`http://www.w3.org/`

Determining if a publication is current

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

For CICS for VSE/ESA books, these softcopy updates appear regularly on the *Transaction Processing and Data Collection Kit* CD-ROM, SK2T-0730-xx and on the

VSE/ESA Collection Kit CD-ROM, SK2T-0060-xx. Each reissue of the collection kit is indicated by an updated order number suffix (the -xx part). For example, collection kit SK2T-0730-06 is more up-to-date than SK2T-0730-05. The collection kit is also clearly dated on the cover.

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

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

Index

Sending your comments to IBM

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Please limit your comments to the information in this book and the way in which the information is presented.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

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

You can send your comments to IBM in any of the following ways:

- By mail, to this address:

  Information Development Department (MP095)
  IBM United Kingdom Laboratories
  Hursley Park
  WINCHESTER,
  Hampshire
  SO21 2JN
  United Kingdom
- By fax:
  - From outside the U.K., after your international access code use 44–1962–870229
  - From within the U.K., use 01962–870229
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink™: HURSLEY(IDRCF)
  - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Program Number: 5648-054

Printed in U.S.A.

GC34-5763-03

Spine information:

IBM

CICS TS for VSE/ESA

Enhancements Guide

Release 1