CICS® Transaction Server for VSE/ESA™

Operations and Utilities Guide

Release 1

SC33-1654-09

CICS® Transaction Server for VSE/ESA™

Operations and Utilities Guide

Release 1

SC33-1654-09

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 225.

Tenth Edition (September 2005)

This edition applies to Release 1 of CICS Transaction Server for VSE/ESA, program number 5648-054, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

The CICS for VSE/ESA Version 2.3 edition remains applicable and current for users of CICS for VSE/ESA Version 2.3.

This book is based on the *System Definition and Operations Guide*.

Order publications through your IBM representative or the IBM branch office serving your locality.

At the back of this publication is a page entitled “Sending your comments to IBM”. If you want to make any comments, please use one of the methods described there.

Contents

Preface

What this book is about

This book describes how to operate CICS regions in a Virtual Storage Extended/Enterprise System Architecture (VSE/ESA) environment, particularly when using the extended recovery facility (XRF), or multiregion operation (MRO). It also contains guidance about how to use the CICS batch utility programs.

Who is this book for?

This book is for system programmers responsible for:

- Controlling the operation of CICS regions
- Planning the use of the supporting utility programs.

What you need to know to understand this book

It is assumed that you have experience of the VSE/ESA operating system, that you are familiar with CICS, and with VSE job control language (JCL) and JCL procedures.

Book structure

Part 1, Operating CICS regions ... page 1 to page 63
Describes how to start up, run, and shut down CICS regions.

Part 2, The CICS utility programs ... page 65 to page 202
Describes how to run the CICS utility programs, and supplies sample jobs for each utility.

Appendix A, “Operating procedures” on page 205
Describes sample operating procedures that you can adapt for your CICS environment.

How to use this book

The parts and chapters in this book are self-contained. You should use an individual part or chapter where it contains information about the particular task you are engaged in. For example, see Part 2 if you need information about running one of the CICS utility programs.

Notes on terminology

The terms listed in Table 1 are commonly used in the CICS Transaction Server for VSE/ESA Release 1 library. See the *CICS Glossary* for a comprehensive definition of terminology.

Table 1 (Page 1 of 2). Commonly used words and abbreviations in CICS Transaction Server for VSE/ESA Release 1

Term	**Definition (and abbreviation if appropriate)**
$(the dollar symbol)	In the character sets and programming examples given in this book, the dollar symbol ($) is used as a national currency symbol and is assumed to be assigned the EBCDIC code point X'5B'. In some countries a different currency symbol, for example the pound symbol (£), or the yen symbol (¥), is assigned the same EBCDIC code point. In these countries, the appropriate currency symbol should be used instead of the dollar symbol.
BSM	BSM is used to indicate the basic security management supplied as part of the VSE/ESA product. It is RACROUTE-compliant, and provides the following functions: • Signon security • Transaction attach security
C	The C programming language
CICSplex	A CICSplex consists of two or more regions that are linked using CICS intercommunication facilities. Typically, a CICSplex has at least one terminal-owning region (TOR), more than one application-owning region (AOR), and may have one or more regions that own the resources accessed by the AORs
CICS Data Management Facility	The new CICS Transaction Server for VSE/ESA Release 1 facility to which all statistics and monitoring data is written, generally referred to as “DMF”
CICS/VSE	The CICS product running under the VSE/ESA operating system, frequently referred to as simply “CICS”
COBOL	The COBOL programming language
DB2 for VSE/ESA	Database 2 for VSE/ESA which was previously known as “SQL/DS”.

Table 1 (Page 2 of 2). Commonly used words and abbreviations in CICS Transaction Server for VSE/ESA Release 1

Term	Definition (and abbreviation if appropriate)
ESM	ESM is used to indicate a RACROUTE-compliant external security manager that supports some or all of the following functions: • Signon security • Transaction attach security • Resource security • Command security • Non-terminal security • Surrogate user security • MRO/ISC security (MRO, LU6.1 or LU6.2) • FEPI security.
FOR (file-owning region)—also known as a DOR (data-owning region)	A CICS region whose primary purpose is to manage VSAM and DAM files, and VSAM data tables, through function provided by the CICS file control program.
IBM C for VSE/ESA	The Language Environment version of the C programming language compiler. Generally referred to as “C/VSE”.
IBM COBOL for VSE/ESA	The Language Environment version of the COBOL programming language compiler. Generally referred to as “COBOL/VSE”.
IBM PL/I for VSE/ESA	The Language Environment version of the PL/I programming language compiler. Generally referred to as “PL/I VSE”.
IBM Language Environment for VSE/ESA	The common runtime interface for all LE-conforming languages. Generally referred to as “LE/VSE”.
PL/I	The PL/I programming language
VSE/POWER	Priority Output Writers Execution processors and input Readers. The VSE/ESA spooling subsystem which is exploited by the report controller.
VSE/ESA System Authorization Facility	The new VSE facility which enables the new security mechanisms in CICS TS for VSE/ESA R1, generally referred to as “SAF”
VSE/ESA Central Functions component	The new name for the VSE Advanced Function (AF) component
VSE/VTAM	“VTAM”

Part 1. Operating CICS regions

This part is about operating CICS® regions. It begins with an overview of operating CICS before going into more detail about specific aspects of operating CICS.

Table 2. Operating CICS regions road map

If you want ...	Refer to...
An overview of CICS operations	Chapter 1, "Overview of CICS operations" on page 3
To start up a CICS region	Chapter 2, "Starting up CICS regions" on page 17
To operate CICS in an MRO environment	Chapter 3, "Operating CICS in a multiregion environment" on page 23
To operate CICS in an XRF environment	Chapter 4, "Operating CICS in an XRF environment" on page 27
To operate CICS from a console device	Chapter 5, "Operating CICS from a console device" on page 59
To shut CICS down	Chapter 6, "How to shut down CICS" on page 61

Chapter 1. Overview of CICS operations

This chapter gives an overview of CICS operations.

When operating CICS, you should have clearly defined operating procedures for your CICS environment. These procedures should provide information about how CICS should be operated in your CICS environment, and record actions taken while operating CICS. Some example procedures and forms for use when operating CICS are provided in Appendix A, “Operating procedures” on page 205. These are for guidance only. You should modify the procedures and forms to suit your CICS environment.

Starting up CICS

When you start up CICS, you start a process called **CICS system initialization**. This process must finish before you run any transactions.

CICS system initialization involves many activities, some of which are:

- Obtaining the storage required for CICS execution from the private area in the CICS address space, above and below the 16MB line
- Setting up CICS system parameters for the run, as specified by the system initialization parameters
- Loading and initializing the CICS domains[1] according to the option specified on the START system initialization parameter.
- Loading the CICS nucleus with the required CICS modules
- Installing CICS resource definitions by:
  - Loading the groups of resources specified by the GRPLIST system initialization parameter from the CSD
  - Loading the control tables specified by various system initialization parameters
- If XRF=YES is specified, signing on to the CICS availability manager (CAVM) to check that it is possible to continue initialization and perform the role requested, that is, as an active or alternate CICS region
- Opening the data sets necessary for initialization, including any needed for backout if the previous run of your CICS region was not shut down normally (except for START=STANDBY, when most data sets are not opened until after takeover)
- Opening SAM sequential devices as required in the terminal control table (TCT).

If you are operating CICS with CICS recovery options, backout procedures may be used to restore recoverable resources to a logically consistent state. Backout occurs at a subsequent re-start if you start CICS in one of the following ways:

[1] A domain is a functionally isolated part of CICS that can communicate with the rest of CICS by means of strictly-defined standard interfaces.

- With START=AUTO and CICS detects that the previous shutdown was immediate or uncontrolled
- With START=STANDBY and XRF=YES, and a takeover occurs.

For background information about backout, and recovery and restart, see the *CICS Recovery and Restart Guide*.

The types of CICS startup

CICS startup can be any of the types shown in Table 3.

Table 3. The different types of CICS startup

Type	Effect
Initial	CICS starts with no reference to any system activity in the CICS global catalog from a previous run of CICS. For more information, see “CICS actions on an initial start” on page 5.
Cold	CICS initializes with limited reference to any system activity recorded in the CICS catalogs from a previous run of CICS. For more information, see “CICS actions on a cold start” on page 5.
Warm	CICS initializes, after a normal shutdown, restoring CICS to the status it was in at the last normal CICS shutdown, except for some facilities that it initializes as for a cold start. CICS always restores the trace (TR) domain according to the system initialization parameters, and can restore other facilities depending on the setting on the COLD option of their associated system initialization parameters. For more information, see “CICS actions on a warm start” on page 6.
Emergency	CICS initializes, after an abnormal shutdown, restoring recoverable resources to their committed states. For more information, see “CICS actions on an emergency restart” on page 8.

When CICS is started up, the type of startup (and therefore the actions it takes) depends primarily on the following:

- The value of the START system initialization parameter
- Information in the CICS catalogs from previous runs of CICS

The values of other system initialization parameters also influence the actions taken on CICS startup.

For information about the types of startup, the roles of the CICS catalogs, and the effect of the START system initialization parameter, see the *CICS System Definition Guide*.

Note: You cannot request a warm or emergency restart explicitly. When selecting the type of start (using the CICS system initialization parameter, START), the choices are COLD or AUTO. AUTO can result in either a warm or an emergency restart; CICS itself determines which to use depending on information in the CICS catalogs at the time startup takes place.

CICS actions on an initial start

The CICS global and local catalogs are initialized and all information in them is lost. It should rarely be necessary to perform an initial start; examples of times when an initial start is necessary are:

- When bringing up a new CICS system for the first time.
- After a serious software failure, when the global catalog or the system log have been corrupted.

CICS actions on a cold start

In a cold start, initialization of CICS occurs with limited reference to any system activity recorded in the CICS catalogs. No system log or warm keypoint information is used from any previous run of CICS. Dump table entries from a previous run are also deleted in a cold start.

In a cold start:

- For journals without the automatic archiving option, the status of journaling is restored from information recorded in the CICS catalog. For journals with the PAUSE option, journal status is preserved. For journals with the NOTREADY status, PAUSE messages are reissued.
- For journals with the automatic archiving option, the journal status (including the sequence number) is retrieved from the journal archive control data set (DFHJACD). For those journal data sets that have the NOTREADY status, an archive job is submitted.
- TERMINAL definitions are purged from the recovery file and from the CICS global catalog.
- Existing TYPETERM and MODEL terminal definitions are purged from the CICS global catalog.
- PROGRAM definitions are purged from the recovery file and from the CICS global catalog.
- TRANSACTION and PROFILE definitions are purged from the CICS global CICS global catalog.
- File control records are purged from the CICS global catalog.
- Resource definition information is obtained as follows:
  - Tables specified by system initialization parameters, such as DCT=xx, are obtained from the program library.
  - Information in the groups in any of the lists named by the GRPLIST system initialization parameter for *this* initialization is taken from the CICS system definition (CSD) file and merged with information from the program library.
  - Information in groups that have been defined or added to group lists is taken from the CSD.

CICS actions on a warm start

A warm start restores certain elements of the CICS components that can be warm started to the status that was recorded in the warm keypoint of the previous normal shutdown.

A partial warm start is similar to a complete warm start, except that some selected CICS facilities are cold-started, as specified in the system initialization parameters. Information is obtained for those facilities from the warm keypoint only if they are not specified to be cold started.

In a warm start:

- Resource definition information is obtained as follows:
  - Tables specified by system initialization parameters, such as DCT=xx, are obtained from the program library. Information contained in the warm keypoint of the previous run is used to update the information from the program library.
  - Information in the groups in the lists named by the GRPLIST system initialization parameter for *this* initialization is ignored.
  - Information in the groups in the lists named by the GRPLIST system initialization parameter for the *previous* initialization is obtained from the warm keypoint and the CICS global catalog.
  - Information in groups that have been installed since the last cold start is obtained from the warm keypoint and the global catalog.
  - Information in groups that have been defined or added to group lists is taken from the CSD.
  - Information about any autoinstalled terminal that has an automatic-initiate descriptor (AID) outstanding is retrieved from the global catalog.
- Selected system variables are restored from the warm keypoint, including:
  - Region exit time interval value
  - System runaway time interval value
  - Maximum number of tasks
  - High-water mark number of the unit of recovery descriptor
- The following intrapartition transient data information is restored:
  - All data held on intrapartition destinations
  - The following DCT information for each destination, from the warm keypoint:
    - Record count in the destination
    - GET and PUT pointer values
    - Trigger-level values
  - AIDs for tasks
  - The intrapartition transient data bit map.
- The following FCT information is restored to what it was at the time of the warm shutdown, using information from the global catalog:
  - The ENABLED/DISABLED/UNENABLED status
  - The SERVREQ options (UPDATE, DELETE, and so on)
  - Any alterations made to the DSNAME

- Files defined as initially OPEN are opened irrespective of their other attributes. If the file state *recovered* during initialization is ENABLED or UNENABLED, the file becomes OPEN, ENABLED after the OPEN. If the file state recovered is DISABLED, the file becomes OPEN, DISABLED.
- Installed transaction and profile definitions are obtained from:
  - The groups specified in the lists named by the GRPLIST system initialization parameter at the previous cold start
  - The groups that have been installed since the previous cold or emergency start

  The following attributes of the installed transactions and profiles are restored from the warm keypoint:

  - ENABLED/DISABLED status
  - Transaction priority
- Installed program and mapset definitions are obtained from these sources:
  - The groups specified in the lists specified on the GRPLIST system initialization parameter at the previous cold start
  - The groups that have been installed since the previous cold start or emergency restart
  - The changes (such as SVA-eligibility) made by CEMT or EXEC CICS SET PROGRAM commands in the previous run.

  The ENABLED/DISABLED status of each installed program and mapset is restored from the warm keypoint. Directory information is obtained for each program and mapset during CICS initialization.
- The following TCT information is restored from the warm keypoint information:
  - Processing status (transaction, transceive, input, or receive)
  - Service status (INSERVICE or OUTSERVICE)
  - Extended attributes supported (color, programmed symbols, and so on)
  - Partition support
  - Magnetic-stripe-reader support
  - Outboard formatting support
  - Coded graphic character set identifiers
  - APL/TEXT keyboard

  If any outstanding work was scheduled for an autoinstalled terminal at the previous warm shutdown, the terminal entry is recovered. (Terminal entries for autoinstalled terminals with no work outstanding are deleted at shutdown.)
- The following auxiliary temporary storage information is restored from the warm keypoint:
  - All data in the auxiliary temporary storage queues
  - The temporary storage use map
- Interval control elements (ICEs) for outstanding START TRANSID commands are restored from the warm keypoint.
- The BMS logical messages that were created by the functions listed below but have not yet been viewed by the terminal operator are restored:
  - Message switching transaction (CMSG)
  - ROUTE command

  - SEND MAP ACCUM and SEND TEXT ACCUM commands, except for those messages terminated by SEND PAGE without specifying RELEASE or RETAIN®. In those cases, the message might already have been viewed by the operator, but can be viewed again following the warm start.

- All unit of recovery descriptors (APPC log name, APPC resynchronization, and external resource manager) are restored from the warm keypoint, together with any associated deferred work elements (DWEs).
- The STORECLOCK value is restored from the warm keypoint.
- The intervals at which statistics were collected and status and the logical end-of-day time are restored from the global catalog.
- The monitoring status, class status and monitoring control table suffix are restored from the CICS global catalog.
- For journals with the automatic archiving option, the journal status, including the sequence number, is retrieved from the JACD (journal archive control data set), by means of a call to the journal archive program (DFHJAP) from DFHJCRP. For those journal data sets that have the NOTREADY status, an archive job is submitted.
- If you are using disk journaling, the status of the disk journals is also saved in the CICS 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 system initialization parameter to cause the status in the CICS global catalog to be ignored. During CICS startup, the status of all journal data sets is set to “ready for use”.
- Transaction and system dump table options are held in the CICS global catalog and reapplied at a warm start.

CICS actions on an emergency restart

A CICS system that operates on resources, such as files, that have been defined by the installation to be *recoverable*, records changes to those resources in the CICS system log. If the CICS system fails, the system log at the time of failure should typically contain records of changes made by tasks that have not completed (‘in-flight’ tasks) and by others that have completed.

Following an abnormal termination, emergency restart collects all of the log records pertaining to in-flight tasks, and uses the information in the log records to return recoverable resources to their **committed** states—that is:

- Changes to recoverable resources made by logical units of work (LUWs) that were interrupted are backed out.
- Messages associated with message-protected tasks are preserved.
- Dynamically added resource definitions that were committed during execution of the CEDA INSTALL command are preserved.
- CICS takes an activity keypoint, so that there is at least one activity keypoint on the new system log data set. It shows that there are no in-flight tasks, and delimits the backward scan of the system log on a subsequent emergency restart, in case no other activity keypoint is written during this CICS run. When the activity keypoint has been taken, emergency restart processing is complete.

CICS-VTAM actions after an emergency restart

When LU-LU sessions are re-established after an emergency restart (and subsequent processing), CICS participates in a resynchronization protocol with logical units to discover if any messages, in either direction, were lost when CICS was terminated.

The logical units for which resynchronization is required are marked in the TCTTEs. Resynchronization is *not* attempted in the following cases:

- If the terminal was acquired by a master terminal operation specifying COLDACQ.
- If the terminal was acquired with the EXEC CICS SET TERMINAL ACQSTATUS(COLDACQ) command.
- If the session is a pipeline session.
- If the TCTTE is marked to cold start the session by the TCT assembly process. This is done for terminals such as 3270 terminals that do not support the set and test sequence number (STSN) command.

Note: If the previous session abended, the use of COLDACQ overrides CICS integrity control. This could lead to data integrity problems. Also, you should check the CSMT log for an activity keypoint after the restart of a session following a CICS failure. If there is no activity keypoint, you should issue COLDACQ again after the next emergency restart.

For each logical unit that *does* require resynchronization, CICS issues an STSN command that notifies the logical unit of the sequence numbers known to CICS—that is, those numbers that backout processing placed in the TCTTE. The logical unit can compare these sequence numbers with those that it has logged for itself, and can thus determine if any messages were lost.

- If an *input* message was lost, the logical unit should retransmit it to CICS.
- If an *output* message was lost, CICS retransmits the message from the resend slot and, in so doing, deletes the resend slot.

Note: The message remains in the resend slot if CICS does *not* retransmit it. This occurs if the resynchronization process shows that the output message was not lost, or if the logical unit does not support the STSN command; the 3270 is in this category.

CICS startup and the VTAM session

In a VTAM network, the session between CICS and VTAM is started automatically if VTAM is started before CICS. If VTAM® is not active when you start CICS, you receive the following messages:

```
F NET,USERVAR,ID=generic-applid,VALUE=specific-applid
+DFHSI1589D 'applid' VTAM is not currently active.
+DFHSI1572 'applid'  Unable to OPEN VTAM ACB - RC=xxxxxxxx, ACB CODE=yy.
```

Although the MODIFY NET, USERVAR command is only significant when you are running CICS with XRF, the USERVAR message occurs for both XRF=YES and XRF=NO CICS systems. If you receive messages DFHSI1589D and DFHSI1572, and if the CICS region is not initializing as an alternate CICS region, you can start the CICS-VTAM session manually when VTAM is eventually started, by means of the CEMT SET VTAM OPEN command from a supported VSE console or a non-VTAM terminal.

If VTAM is active, but CICS still cannot open the VTAM ACB because VTAM does not recognize the CICS APPLID, you receive the following messages:

```
F NET,USERVAR,ID=generic-applid,VALUE=specific-applid
+DFHSI1592I 'applid' CICS applid not (yet) active to VTAM.
+DFHSI1572  'applid' Unable to OPEN VTAM ACB - RC=00000008, ACB CODE=5A.
```

This may be caused by an error in the value of APPLID operand, in which case you must correct the error and restart CICS. For information about other causes and actions, see "VTAM codes" in *VSE/ESA Messages and Codes, Volume 2*.

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.

Controlling CICS operation

While CICS is running, you can control its operation by changing CICS system definitions and by deleting and installing resource definitions.

Note: You cannot change CICS system definition values set by *some* system initialization parameters during CICS startup. To change such values, you must specify the new values on system initialization parameters, and restart CICS with those changed system initialization parameters.

CICS supplies a number of transactions that you can use to control CICS and its resources while it is running. It also supplies a variety of utility programs, some of which you can use to help with system management.

Controlling CICS with CICSPlex SM

If you are running your CICS regions in a CICSplex, you can use CICSPlex System Manager/ESA functions to control the operation of CICS; for example, you may:

- Change CICS system attributes.
- Inquire and set CICS resources.
- Define CICS resources from a single point of control.
- Monitor system availability.
- Reset CICS' date and time to match those of the operating system.
- Rebuild security profiles for CICS.
- Write CICS statistics to a DMF data set.
- Add, remove, or reset CICS system and transaction dump codes.

For information, see the *CICSPlex System Manager Concepts and Planning* manual.

CICS-supplied transactions

CICS supplies a number of transactions that you can use to control CICS and its resources while it is running. CICS-supplied transactions have identification codes that start with the letter C and are four characters long.

The most significant transactions for CICS operation are CEMT, CEST, and CEDA. The following sections outline these three transactions. For information about these and other CICS transactions, see the *CICS-Supplied Transactions* manual.

The master terminal transaction, CEMT

You can use the CEMT transaction to view the values of CICS system definitions and to change such definitions while CICS is running. You can also use CEMT to manage databases, in particular for the dynamic allocation and deallocation of data sets.

With CEMT, you can:

- Control the number of tasks, or the number of certain types of task, running at any given time.
- Purge tasks from the system.
- Enable or disable transactions.
- Enable or disable files; for example, to allow controlled access to them by application programs.
- Start or stop tracing (you can also use CETR for this), monitoring, or statistics activities.
- Switch dump data sets when one is full.
- Open and close interregion communication connections.
- Install newly link-edited copies of application programs.

To view the values of CICS system definitions, use the CEMT INQUIRE command.

To change the values of system definitions, or to change CICS operation, use the CEMT SET, CEMT PERFORM, or CEMT DISCARD command.

Note: CEMT is a powerful tool and its use can significantly affect your system, and its users. Therefore, you should give the transaction adequate security protection in a production CICS region.

The supervisor terminal transaction, CEST

CEST provides a subset of the CEMT function. The CEST INQUIRE and SET commands enable you to inquire on and alter some of the system values of control units, lines, netnames, tasks and terminals.

The resource definition online transaction, CEDA

You can use the CEDA transaction to:

- View resource definitions
- Change existing resource definitions
- Create new resource definitions

on the CSD that your CICS region is using. You can also use the CEDA transaction to install resource definitions into a running CICS region.

Similarly, you can use the CEDB transaction to view, change, or create resource definitions, and can use the CEDC transaction to view resource definitions.

For information about the CEDA, CEDB, and CEDC transactions, see the *CICS Resource Definition Guide*.

CICS-supplied utility programs

CICS supplies a number of utility programs to help you manage your system. These utility programs are described in Part 2, “The CICS utility programs” on page 65.

Shutting down CICS

This section describes the three types of CICS system shutdown (normal, immediate, or uncontrolled) and the events that cause them. For information about CICS shutdown, see the *CICS Recovery and Restart Guide*.

- In normal shutdown, CICS performs a controlled sequence of operations that leave the system in a well-defined state. Existing tasks are allowed to finish.

  The following events can cause normal shutdown of CICS:

  – The CEMT PERFORM SHUTDOWN transaction
  – The EXEC CICS PERFORM SHUTDOWN command

- In immediate shutdown, CICS remains in overall control, but it does a minimum of processing so the system can be terminated rapidly. Existing tasks are not allowed to finish, and could abend.

  The following events can cause immediate shutdown of CICS:

  – The CEMT PERFORM SHUTDOWN IMMEDIATE transaction
  – The EXEC CICS PERFORM SHUTDOWN IMMEDIATE command
  – A CICS system abend
  – A program check

- In uncontrolled shutdown, CICS is not given the chance to do any processing once the event causing it to terminate has occurred.

  The following events can cause uncontrolled shutdown of CICS:

  – Power failure
  – Machine check
  – Operating system failure

After a normal shutdown, it is possible to warm start CICS. After an immediate or an uncontrolled shutdown, an emergency restart or a cold start must be performed.

CICS XRF systems

When an XRF active CICS region is terminated abnormally, the alternate CICS region normally completes initialization and takes over. However, the alternate CICS region also terminates if the user has initiated a normal shutdown of the active CICS region and has *not* specified that takeover is to take place.

Normal shutdown (PERFORM SHUTDOWN)

Normal shutdown is initiated by the master terminal operator or by an application program, and is accomplished in two stages. (In comparison, immediate shutdown is accomplished by termination processing.)

First stage of normal shutdown

During the first stage of CICS normal shutdown, all terminals are active and all CICS facilities are available. The following actions take place concurrently:

- Message DFHTM1715 is issued to the console and the master terminal user to inform the operator that CICS is terminating.
- The CICS local and global catalogs are closed.
- Tasks that already exist complete. (Long-running tasks, such as conversational tasks, must end before system termination can complete.)
- Tasks to be automatically initiated run, if they can start before the second stage.
- Any user-written programs listed in the first part of the shutdown program list table (PLT) are run sequentially.
- The Front End Programming Interface (FEPI) is requested to shut down.
- The terminal that initiated the shutdown, if any, is detached. This allows the operator to start any further tasks that might be required, or to purge any tasks.

  A new task started as a result of terminal input is allowed to start only if it has been defined as SHUTDOWN(ENABLED) in its TRANSACTION resource definition or if the transaction identifier is listed in the current transaction list table (XLT). The XLT list of transactions restricts the tasks that can be started by terminals and allows the system to shut down in a controlled manner. The current XLT is the one specified by the XLT=xx system initialization parameter, which may be overridden by the XLT option of the CEMT or EXEC CICS PERFORM SHUTDOWN command.

  Certain CICS-supplied transactions are, however, allowed to start whether or not their code is listed in the XLT. These transactions are CEMT, CESF, CLS1, CLS2, CSAC, CSTE, and CSNE.

Note: You should not change the SHUTDOWN(ENABLED) attribute of the resource definitions for these transactions, otherwise CICS may not shut down successfully.

- A request is issued to quiesce all interregion communication (IRC) activity.
- Terminal control is requested to ignore all further input.
- If this is a non-XRF system, CLSDST requests are issued for all VTAM terminals.
- The termination task waits for all terminal activity to cease, before entering the second stage of shutdown.

The first shutdown stage is complete when the last of the programs specified in the first part of the shutdown PLT has run and all user tasks are complete.

Second stage of normal shutdown

During the second stage of shutdown, terminals are not active, and no new tasks are allowed to start. The following processing takes place:

1. User-written programs listed in the second part of the shutdown PLT (if any) are executed sequentially. These programs cannot communicate with terminals, or make any request that would cause a new task to start.
2. If DL/I is active, DL/I termination is started.
3. All currently open CICS files are now closed.
4. Language Environment® for VSE/ESA™ is called to perform partition termination.
5. Journal control is shutdown.
6. The transient data CI buffer and the temporary storage buffers are flushed.
7. CICS writes the following information to the CICS global catalog:
   - A warm keypoint. This contains information that is used to restore the operating environment during a subsequent warm start.
   - A warm-start-possible indicator. This status applies on the next initialization of CICS if START=AUTO is specified.

   The CICS global catalog domain cannot complete its processing until this information has been recorded, as the global catalog must remain open until it is done.
8. Transient data is terminated.
9. A dump is taken, if one is required.
10. If TAKEOVER was specified on the command to shut down an XRF CICS region, a “signoff abnormal” request is made from the CICS availability manager (CAVM).
11. The following message is issued:

    DFHKE1799 *applid* **TERMINATION OF CICS IS COMPLETE**
12. CICS completes some internal processing, then returns control to VSE.

Immediate shutdown (PERFORM SHUTDOWN IMMEDIATE)

During immediate shutdown of CICS, possibly requested by the master terminal operator or an application program, processing is different from a normal shutdown in the following important ways:

- User tasks are not guaranteed to complete.
- None of the programs listed in the shutdown PLT are run.
- CICS does *not* write a warm keypoint or a warm-start-possible indicator to the global catalog.
- CICS does not close files defined to CICS file control.

To preserve data integrity, the next initialization of CICS **must** be an emergency restart. If the next initialization of CICS specifies START=AUTO, there will be an emergency restart.

The processing involved in immediate shutdown is described as CICS system termination processing. (In comparison, normal shutdown involves processing.)

Unlike processing, controls are not exercised to ensure that resources and services remain available as long as they are needed. One consequence of this is that transaction and CICS system abends can occur during immediate shutdown. Thus, if a task tries to use a resource that has already been terminated, the task abends. Then dynamic transaction backout is invoked, and that might also fail because it could also try to use a resource that has been terminated.

In addition, if CICS system termination processing is delayed significantly, tasks in the system waiting for input from terminals that are no longer available are likely to extend beyond the period for deadlock timeout specified in the DTIMOUT option of the TRANSACTION definition and will abend.

First stage of immediate shutdown

During the first stage of an immediate shutdown, the following processes take place:

1. The system termination task drives the collection of termination statistics.
2. If there is a terminal associated with the event that caused the immediate shutdown, a message is sent to inform the operator that CICS is terminating.
3. If the shutdown request has arrived by transaction routing, the associated terminal is freed.
4. Terminal input is no longer accepted.
5. The Front End Programming Interface (FEPI) is requested to shut down immediately.

Second stage of immediate shutdown

During the second stage of an immediate shutdown, the following processing takes place:

1. DL/I is terminated.
2. Journal control is shut down.
3. Transient data is terminated.

4. A dump is taken, if requested.
5. Interregion sessions are terminated.
6. If CICS is signed on to the CICS availability manager (CAVM), a “signoff abnormal” request is made from CAVM.
7. The CICS local and global catalogs are left to be closed by the operating system.
8. The following message is issued:

   `DFHKE1799` *`applid`* **`TERMINATION OF CICS IS COMPLETE`**

9. CICS completes some internal processing, then returns control to VSE.

Uncontrolled shutdown

An uncontrolled shutdown of CICS can be caused by a power failure, a machine check, or an operating system failure.

In each case, CICS cannot perform any shutdown processing. In particular, CICS does *not* write a warm keypoint or a warm-start-possible indicator to the global catalog.

To preserve data integrity, the next initialization of CICS **must** be an emergency restart. If the next initialization of CICS specifies START=AUTO, there is an emergency restart.

Chapter 2. Starting up CICS regions

- How to specify system initialization parameters before starting CICS, in "Specifying system initialization parameters before startup"
- How to start CICS as a batch job, in "Starting CICS as a batch job" on page 18
- How to override the system initialization parameters, in "Overriding system initialization parameters during startup" on page 20
- The console messages which CICS issues during startup, in "System console messages for CICS startup" on page 21

For an overview of CICS startup, see "Starting up CICS" on page 3.

This chapter assumes that any customization of CICS, the generation of any additional support required, and all the necessary CICS system definitions have already been carried out.

For information about defining CICS systems, see the *CICS System Definition Guide*, which describes the system initialization parameters in detail, and tells you how to create a CICS startup job.

You determine how CICS starts up and the facilities and resources that it can use, by specifying values for system initialization parameters to be used by the CICS startup job stream. You would normally specify the system initialization parameters that CICS is to use before you start CICS. (See "Specifying system initialization parameters before startup.") However, once you have started CICS initialization, you can override the system initialization parameters specified before startup; for example, to enable a specific facility for that run of CICS. (See "Overriding system initialization parameters during startup" on page 20.)

For details about which system initialization parameters can be overwritten, see the *CICS System Definition Guide*.

Specifying system initialization parameters before startup

You normally specify the system initialization parameters that CICS is to use in the following ways, before starting CICS:

1. In the system initialization table, loaded from a library of the LIBDEF SEARCH chain for the CICS startup job
2. In the PARM parameter of the EXEC DFHSIP statement of the CICS startup job
3. In the SYSIPT data stream defined in the startup job (but only if SYSIPT is coded in the PARM parameter)

The system initialization parameters are processed in the preceding order, with later system initialization parameter values overriding those specified earlier.

In particular, you can specify a new value for the START system initialization parameter, which can have any of the following values:

START=AUTO
If you specify the START=AUTO system initialization parameter, CICS determines whether to perform a cold, warm, or emergency start by inspecting the control record in the global catalog,

START=AUTO should be the normal mode of operation, with the choice of start being made by CICS automatically.

START=COLD
If you specify the START=COLD system initialization parameter, CICS performs a cold start, but with some reference to the previous run of a CICS system using the same global catalog.

START=STANDBY
CICS starts up as an XRF alternate CICS region, by beginning an emergency restart then suspending until it needs to perform a takeover.

Note: You must also specify the XRF=YES system initialization parameter.

START=LOGTERM
CICS restarts only to put an end of file label on the system log, and then terminates ***before*** doing any backout processing.

Note: This option is intended for use only when you are running CICS with XRF=NO.

For example, if your CICS startup job specifies:

```
// EXEC DFHSIP,SIZE=DFHSIP,                                        *
               PARM='SYSIDNT=HTH1,SIT=6$,DCT=2$,SYSIPT,CN'         *
               DSPACE=2M,OS390
/*
```

CICS uses system initialization parameters from the following sources, with later system initialization parameters overriding earlier ones:

1. The system initialization table, DFHSIT6$, from the LIBDEF search chain for the CICS job.
2. Any input supplied in the SYSIPT data stream.
3. The system console.

Note: OS390 must be the last parameter on the EXEC card.

In the above example, if you specify DCT=2$ in the SYSIPT data stream, and then enter DCT=3$ at the system console, CICS loads DFHDCT3$.

Starting CICS as a batch job

To start CICS as a batch job, submit the job through the reader. The job can remain on the RDR queue as DISP=L. It can be released:

- Manually
- By an operator
- Automatically as part of the VSE IPL.

Figure 1 shows some sample JCL you can use to start CICS as a batch job.

```
* $$ JOB JMN=DFHCICS1,CLASS=x
// JOB DFHCICS1
// OPTION LOG,DUMP
// OPTION SYSDUMP,SYSDUMPC
// DLBL IJSYSUC,'CICS410',,VSAM
// EXEC IDCAMS,SIZE=600K
   VERIFY DATASET(CICS410.DFHCSD)
   VERIFY DATASET(CICS410.DFHTEMP)
   VERIFY DATASET(CICS410.DFHNTRA)
   VERIFY DATASET(CICS410.DFHLCD)
   VERIFY DATASET(CICS410.DFHGCD)
   VERIFY DATASET(CICS410.DFHRSD)
/*
// DLBL IJSYSUC,'CICS410',,VSAM
// ASSGN SYS020,SYSLST                                             1
// DLBL DFHRSD,'CICS410.DFHRSD',,VSAM
// DLBL DFHLCD,'CICS410.DFHLCD',,VSAM
// DLBL DFHGCD,'CICS410.DFHGCD',,VSAM
// DLBL DFHCSD,'CICS410.DFHCSD',,VSAM
// DLBL DFHTEMP,'CICS410.DFHTEMP',,VSAM
// DLBL DFHNTRA,'CICS410.DFHNTRA',,VSAM
// DLBL FILEA,'CICS410.FILEA',,VSAM
// DLBL DFHDMPA,'CICS410.DFHDMPA',0,SD
// DLBL DFHDMPB,'CICS410.DFHDMPB',0,SD
// DLBL DFHAUXT,'CICS410.DFHAUXT',0,SD
// DLBL DFHBUXT,'CICS410.DFHBUXT',0,SD
*****************************************************************
* **      LIBDEF for CICS for Transaction Server for VSE/ESA ** *
*****************************************************************
// LIBDEF *,SEARCH=(user.library,                                      X
                 PRD2.SCEECICS,PRD2.SCEEBASE,                          X
                 PRD2.CICSR,PRD2.DBASE,PRD2.PROD)                  2
// EXEC DFHSIP,SIZE=DFHSIP,PARM='SI',OS390
SIT=$$,EDSALIM=24M,DSALIM=4M,APPLID=IYBBCICS
DCT=2$,JCT=NO,TCT=NO,MXT=20
SRT=1$,FCT=NO,SEC=NO,TRTABSZ=4000
ISC=YES,IRCSTRT=NO
GRPLIST=(LISTV41,LELIST,DFHLIST)
GMTEXT='CICS for Transaction Server SYSTEM - APPLID IYBBCICS'
PGAIEXIT=DFHPGADX
PGAIPGM=ACTIVE
PGAICTLG=NONE
START=COLD
/*
// EXEC LISTLOG
/&
* $$ EOJ
```

Figure 1. Sample job to start CICS as a batch job

Note: 1 SYS020 can be assigned to SYSLST to direct CXRF messages to the printer. If SYS020 is not assigned to a printer, you must supply a DLBL for DFHCXRF.

2 The LIBDEF search chain must include the sublibrary containing Journal Archive skeleton JCL. If you want to use Automatic Journal Archive, see Chapter 7, “Journal utility programs” on page 67.

Overriding system initialization parameters during startup

When you have started the initialization of CICS, you may want to override system initialization parameters specified in the SIT, PARM parameter, and SYSIPT data stream of the CICS startup job. You can do this by specifying new values for system initialization parameters at the system console.

Note: You can specify system initialization parameters at the system console only if the CONSOLE keyword was specified in either the PARM parameter or in the SYSIPT data stream.

If you specify the CONSOLE (or CN) keyword in the PARM statement of the EXEC DFHSIP statement or SYSIPT data stream of your CICS startup job, CICS prompts you to enter system initialization parameters at the system console.

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 stream. CICS accepts system initialization parameters from the console until you terminate the input with .END.

Through the console, you can specify the SIT system initialization parameter only as the first parameter when prompted by message DFHPA1921, at which point CICS tries to load the specified SIT. If you try to specify the SIT system initialization 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 the PARMERR=INTERACT system initialization parameter, and CICS detects a parameter error, either in the keyword or in the value you have assigned to it, CICS prompts you to correct the error with message DFHPA1912 or DFHPA1915:

```
DFHPA1912 'applid' SIT OVERRIDE 'keyword' IS NOT RECOGNIZED.
                   SPECIFY CORRECT SIT OVERRIDE.
DFHPA1915 'applid' INVALID DATA HAS BEEN DETECTED FOR SIT OVERRIDE 'keyword'.
                   RESPECIFY THE OVERRIDE.
```

CICS prompts you to enter corrections to any errors it finds in the PARM parameter or the SYSIPT data stream **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 stream, it is the APPLID coded in the SIT that CICS displays in messages DFHPA1912 and DFHPA1915.

System console messages for CICS startup

This section gives some typical message sequences that are displayed when starting up CICS.

```
Y3  0047  DFHPA1101   IYBBCICS DFHSIT    IS BEING LOADED.
Y3  0047  DFHPA1108   IYBBCICS DFHSIT    HAS BEEN LOADED.  (GENERATED TEXT AT : MM/DD= 06/26  HH:MM= 16:47).
Y3  0047  DFHPA1100   IYBBCICS OVERRIDE PARAMETERS FROM JCL EXEC STATEMENT : SI
Y3  0047  DFHPA1102   IYBBCICS OVERRIDE PARAMETERS FROM SYSIPT:
Y3  0047  DFHPA1927   IYBBCICS SIT=$$,EDSALIM=20M,DSALIM=4M,APPLID=IYBBCICS
Y3  0047  DFHPA1927   IYBBCICS SRT=1$,FCT=NO,SEC=NO,TRTABSZ=4000
Y3  0047  DFHPA1927   IYBBCICS ISC=YES,IRCSTRT=NO
Y3  0047  DFHPA1927   IYBBCICS GRPLIST=(DFHLIST,LISTV41,LELIST)
Y3  0047  DFHPA1927   IYBBCICS GMTEXT='CICS TRANSACTION SERVER SYSTEM - APPLID IYBBCICS
Y3  0047  DFHPA1927   IYBBCICS PGAIEXIT=DFHPGADX
Y3  0047  DFHPA1927   IYBBCICS PGAIPGM=ACTIVE
Y3  0047  DFHPA1927   IYBBCICS PGAICTLG=NONE
Y3  0047  DFHPA1927   IYBBCICS START=COLD
Y3  0047  DFHPA1103   IYBBCICS END OF FILE ON SYSIPT.
Y3  0047  DFHTR0103   TRACE TABLE SIZE IS 4000K
Y3  0047  DFHSM0122I  IYBBCICS LIMIT OF DSA STORAGE BELOW 16MB IS 4,096K.
Y3  0047  DFHSM0123I  IYBBCICS LIMIT OF DSA STORAGE ABOVE 16MB IS 20M.
Y3  0047  DFHSM0113I  IYBBCICS STORAGE PROTECTION IS NOT ACTIVE.
Y3  0094  DFHDM0101I  IYBBCICS CICS IS INITIALIZING.
Y3  0095  DFHXS1100I  IYBBCICS SECURITY INITIALIZATION HAS STARTED.
Y3  0095  DFHSI1500   IYBBCICS CICS STARTUP IS IN PROGRESS FOR CICS TRANSACTION SERVER VERSION 1.1.0
Y3  0095  DFHXS1102I  IYBBCICS SECURITY IS INACTIVE.
Y3  0095  DFHSI1501I  IYBBCICS LOADING CICS NUCLEUS.
Y3  0095  DFHDU0304I  IYBBCICS TRANSACTION DUMP DATA SET DFHDMPA  OPENED.
Y3  0095  DFHXS1101I  IYBBCICS SECURITY INITIALIZATION HAS ENDED.
Y3  0095  DFHMN0105I  IYBBCICS USING DEFAULT MONITORING CONTROL TABLE.
Y3  0095  DFHMN0110I  IYBBCICS CICS MONITORING IS INACTIVE.
Y3  0095  DFHSI1502I  IYBBCICS CICS STARTUP IS COLD.
Y3  0095  DFHSI1503I  IYBBCICS TERMINAL DATA SETS ARE BEING OPENED.
Y3  0095  DFHCP0101I  IYBBCICS CPI INITIALIZATION HAS STARTED.
Y3  0095  DFHPR0104I  IYBBCICS PARTNER RESOURCE MANAGER INITIALIZATION HAS STARTED.
Y3  0095  DFHTD0100I  IYBBCICS TRANSIENT DATA INITIALIZATION HAS STARTED.
Y3  0095  DFHFC0100I  IYBBCICS FILE CONTROL INITIALIZATION HAS STARTED.
Y3  0095  DFHAI0101I  IYBBCICS AITM INITIALIZATION HAS STARTED.
Y3  0095  DFHFC0101I  IYBBCICS FILE CONTROL INITIALIZATION HAS ENDED.
Y3  0095  DFHTD0101I  IYBBCICS TRANSIENT DATA INITIALIZATION HAS ENDED.
Y3  0095  DFHCP0102I  IYBBCICS CPI INITIALIZATION HAS ENDED.
Y3  0095  DFHPR0105I  IYBBCICS PARTNER RESOURCE MANAGER INITIALIZATION HAS ENDED.
Y3  0095  DFHAI0102I  IYBBCICS AITM INITIALIZATION HAS ENDED.
Y3  0095  DFHSI1511I  IYBBCICS INSTALLING GROUP LIST DFHLIST.
Y3  0095  DFHSI1511I  IYBBCICS INSTALLING GROUP LIST LISTV41.
Y3  0095  DFHSI1511I  IYBBCICS INSTALLING GROUP LIST LELIST.
Y3  0095  DFHAP1203I  IYBBCICS LANGUAGE ENVIRONMENT FOR VSE/ESA IS BEING INITIALIZED.
Y3  0095  DFHSI1517   IYBBCICS CONTROL IS BEING GIVEN TO CICS.
```

Figure 2. Console messages for startup of CICS Transaction Server for VSE/ESA Release 1

Chapter 3. Operating CICS in a multiregion environment

This chapter outlines the operation of CICS regions in a multiregion environment using multiregion operation (MRO) or intersystem communication (ISC) to communicate between CICS regions. It describes how to:

- Enable multiregion operation, in “Enabling MRO” on page 24
- Open communications between CICS regions, in “Opening interregion communication (IRC)” on page 24
- Define connections between CICS regions, in “Defining MRO connections” on page 24
- Add new multiregion connections without stopping CICS, in “Adding new MRO connections while CICS is running” on page 24
- Change existing multiregion connections without stopping CICS, in “Changing MRO connections while CICS is running” on page 25
- Close communications between CICS regions, in “Closing interregion communication (IRC)” on page 25

It assumes that you have already defined your multisystem environment, and are familiar with the concepts of CICS intercommunication facilities, as described in the *CICS Intercommunication Guide*.

To provide the optimum transaction processing environment for your online business applications, you may choose to spread the workload of your business applications across several CICS regions. You can exploit the CICS intercommunication facilities to communicate between the CICS regions, and from CICS to other types of systems (such as IMS™).

The CICS intercommunication facilities that you can use are:

- **Multiregion operation (MRO)**. This is communication between two or more CICS regions running in the same VSE image using CICS internal facilities and protocols. The CICS MRO interregion communication facility is entirely independent of the SNA access method.
- **Intersystem communication (ISC)**. This is communication between two or more systems in the same host, or in different hosts, through an SNA access method such as VTAM.

  Using ISC, the hosts can be different operating systems, and the communicating systems do not both have to be CICS. For example, a CICS Transaction Server region running in a VSE/ESA™ image can communicate with a CICS Transaction Server region running under OS/390®.

Using CICS intercommunication facilities, CICS functions can be separated into individual regions, the different types of CICS region being classified as resource managers. These CICS resource managers can reside in one or more VSE images. CICS regions in the same VSE image can use MRO, which is the preferred method for performance reasons.

CICS regions segregated into resource-manager regions are generally known by the name of the principal resource they “own”. The names used in this book are summarized in “Notes on terminology” on page viii.

The *CICS Intercommunication Guide* contains information about CICS intercommunication, and how to implement both intersystem communication and multiregion operation. This chapter assumes that you are familiar with the concepts of CICS intercommunication facilities, as described in the *CICS Intercommunication Guide*.

Enabling MRO

To be able to use CICS MRO, you must first install support for MRO, as described in the *CICS Intercommunication Guide*. In particular, when starting up your CICS regions, you must specify the ISC=YES system initialization parameter, to include the CICS programs required for MRO into your CICS regions.

Opening interregion communication (IRC)

Before a CICS region can use MRO to communicate with other CICS regions, each participating region must open interregion communication (IRC). You can do this in either of the following ways:

- Specify the IRCSTRT=YES system initialization parameter when you start the CICS region.
- Use the CEMT SET IRC OPEN command while CICS is running.

Defining MRO connections

MRO connections for each CICS region are defined in the CICS system definition (CSD) file, by the RDO CONNECTION and SESSIONS resource definitions. The definitions needed for a CICS region to communicate with other CICS regions are normally installed when you startup the CICS region.

Adding new MRO connections while CICS is running

While your CICS regions are running, you can add new connections to a CICS region by using the CEDA INSTALL command on that CICS region, to install the group containing the associated RDO CONNECTION and SESSIONS resource definitions. For example:

```
CEDA INSTALL GROUP(CONNHT#1)
```

installs the resource group, CONNHT#1, containing the CONNECTION and SESSIONS resource definitions for the TOR, CICSHTH1.

Notes:

1. You ***do not*** need to close interregion communication to be able to install new MRO connections.
2. CICS commits the installation of connection definitions at the group level only if all the connections in the group are installed successfully. If the install of any connection fails, CICS backs out the installation of all connections in the group. Therefore, when adding new connections to a CICS region while IRC is open, you must:
   - Ensure that the new connections are in a group of their own.
   - Install the group.

Changing MRO connections while CICS is running

While your CICS regions are running, you can change existing RDO CONNECTION and SESSIONS definitions, by using the CEDA INSTALL command. However, you must first close IRC by using the CEMT SET IRC CLOSED command.

Closing interregion communication (IRC)

You can close interregion communication (IRC) by using the CEMT SET IRC CLOSED command while CICS is running.

Note: Before IRC can close, all pipes (sessions) of the external call interface must have been closed by the batch programs. For information about the external CICS interface, see the *CICS External CICS Interface* manual.

Chapter 4. Operating CICS in an XRF environment

This chapter describes the operation of a CICS system running with the CICS extended recovery facility, XRF (that is, XRF=YES).

You must also read the other chapters about operations; this chapter does not replace the earlier chapters—it adds ***extra*** information about XRF.

Information can be found under the following headings:

For further guidance information about XRF, see the *CICS XRF Guide*.

Introduction

XRF provides a high level of availability to its end-users by employing a degree of automated takeover from an active CICS system to an alternate CICS system, which can then effect a rapid emergency restart.

The level of automatic takeover is outlined in this chapter, but there are situations in which the alternate CICS cannot gather enough information to determine whether it is safe to complete the takeover.

For example, if you are running CICS in a 2-CPC XRF configuration, and the CPC on which the active CICS is running fails, the alternate CICS cannot receive reliable information about the state of the active CICS. In this situation, the operator must help the alternate CICS.

You might also define some systems in such a way that the operator must intervene to allow the alternate system to take over.

These operating scenarios are described in this chapter. If you are using XRF, VSE POWER is required.

It is important that the operator has clearly defined procedures to follow in as many problem situations as you can define for your installation. XRF is designed to promote more rapid recovery, and you can only achieve that goal if your operators can react correctly and quickly when a fully automatic takeover is not possible.

Using the CEMT and CEBT transactions with XRF

For further guidance information about the CEMT and CEBT commands, see the *CICS-Supplied Transactions* manual. There is also some guidance information about CEBT in the *CICS XRF Guide*.

The master terminal transaction (CEMT)

The CEMT transaction is used to control the active CICS system, but not the alternate CICS, which is controlled by the CEBT transaction. See Table 4 and Table 5, for a summary of the effects of a CEMT command on CICS running with XRF=YES.

The command syntax for shutdown is:

```
CEMT PERFORM SHUTDOWN [IMMEDIATE|TAKEOVER] [DUMP]
```

When the CEMT PERFORM SHUTDOWN command is issued without additional options, both the active CICS and the alternate CICS which was tracking it (if any) shut down normally. Adding the TAKEOVER or IMMEDIATE option causes the alternate system to attempt to take over instead of terminating, **unless its takeover option is defined as COMMAND**.

If the system initiallization parameter TAKEOVR=COMMAND is specified for the alternate system, that system remains in standby mode, even after the active CICS has signed off abnormally, until you issue a specific CEBT PERFORM TAKEOVER command through the alternate system's console.

In the event of an immediate shutdown, CICS terminates all active tasks and SNA sessions, and may abend if tasks are still running. If you specify the takeover option, CICS performs a normal shut down before signing off abnormally from the CICS availability manager (CAVM), and only then does the alternate CICS (if any) attempt the takeover. This means that the takeover option is, in general, slower than the immediate option, and may be more suited to those occasions when you want to do some kind of planned maintenance.

Table 4 and Table 5 summarize the effects of the CEMT command issued to the active CICS system.

Table 4. CEMT PERFORM SHUTDOWN – effect on active CICS

Activity	**Normal**	**Takeover option**	**Immediate option**
Tasks	Task completes	Task completes	Task stops immediately
SNA-session	Unbind	Unbind	Unbind

Table 5. CEMT PERFORM SHUTDOWN – effect on alternate CICS

Activity	**Normal**	**Takeover option**	**Immediate option**
Alternate CICS with TAKEOVR=AUTO or MANUAL	Shuts down	Takes over	Takes over
Alternate CICS with TAKEOVR=COMMAND	Shuts down	Remains in standby mode	Remains in standby mode

The master terminal transaction (CEBT)

The CEBT transaction is used to control an alternate CICS before takeover occurs. From an operating standpoint, CEBT differs from CEMT in that:

- The only way you can use CEBT is by issuing the MSG Fn command, through the system console
- Use of CEBT is not limited by signon or by a security key.

Remember that CEBT runs in an environment in which CICS resources have not been recovered—that is, in the alternate CICS, before takeover has occurred and emergency restart has been completed.

Starting up and synchronizing an XRF system

The active CICS system is started in much the same way as a CICS system with XRF=NO.

In a two-CPC environment, it is important to synchronize the time-of-day clocks as closely as you can (to within as few seconds as possible.) If the active clock is ahead of the alternate clock, there is a delay during takeover until the later clock has caught up.

There is no comparable problem if the active clock is behind that of the alternate system; it simply means that, in the event of a takeover, there is a gap (apparent time loss) in the time stamp of records written to data sets (such as CICS system log data sets).

Starting up an active CICS system

This section summarizes the main points in the startup of an active CICS system.

Submit a VSE job

You start an active CICS by submitting a VSE job.

The job control statements for starting a CICS system with XRF=YES are similar to those for XRF=NO, with the addition of the DLBL statements for the CAVM data sets. You also require the XRF-specific system initialization parameters, specified either in the SIT, the PARM statement, or the SYSIPT data stream.

When running a production CICS system with XRF=NO, you would normally specify all the system initialization parameters in the SIT. However, when you run CICS with XRF=YES and, as recommended, you start the active and alternate CICS systems using the same SIT, you have to specify some system initialization parameters in the PARM parameter or the SYSIPT data stream. For example, the active and the alternate systems require different START values, and also different specific applids.

For example:

```
Active CICS:   START=AUTO
                         APPLID=(DBDCCICS,DBDCCIC1)

Alternate CICS:START=STANDBY
                         APPLID=(DBDCCICS,DBDCCIC2)
```

The active signs on to the CAVM

CICS signs on to the CAVM as the active CICS, and CAVM replies when the signon is accepted, with the following messages:

```
DFHXG6400I specificid Sign on to the CAVM as active with
                                 generic APPLID genericid
DFHXG6401I specificid Sign on to the CAVM as active accepted
```

Although the active and alternate CICS systems each have their own CAVM, they are logically one entity, each communicating through the same pair of CAVM data sets.

Normal initialization continues

CICS continues normal initialization, after determining the type of startup from the START system initialization parameter and the control record in the CICS global catalog. If the SIT specifies START=AUTO, and the control record in the RSD shows that CICS previously shut down normally with a successful warm keypoint, the active carries out a warm start.

Notifies VTAM of specific applid for this CICS

The active system informs VTAM of the specific applid that VTAM should now associate with the CICS generic applid. It does this by issuing the VSE “MODIFY netname,USERVAR” command.

For further guidance information about this command, and the VTAM user variable that it sets, see the *CICS XRF Guide*.

The active CICS detects the alternate CICS signing on to the CAVM

The alternate CICS signs on to the CAVM; this is detected by the active CICS, and acknowledged by message DFHXG6403.

```
DFHXG6403I specificid Sign on to the CAVM as alternate detected.
```

The active system starts the catch-up phase

When initialization is completed, the active system enters the catch-up phase, passing information to the alternate system via the message data set.

Starting up an alternate CICS system

This section briefly lists the main points in the startup of an alternate CICS system.

Submit a VSE job

You start an alternate CICS by submitting a VSE job.

It does not matter if you start the alternate CICS before its associated active CICS, although normally you would ensure that the active system is successfully started before submitting the alternate job.

The alternate system signs on to the CAVM

The alternate system signs on to the CAVM, and notifies you when the signon is successful, with the following messages:

```
DFHXG6500I specificid Sign on to the CAVM as alternate with
                                 generic applid genericid
DFHXG6501I specificid Sign on to the CAVM as alternate accepted
```

The alternate CICS detects the active CICS

Having signed on to the CAVM, the alternate CICS detects (via the CAVM) that the active CICS is running. This is indicated on the console by message DFHXG6503:

```
DFHXG6503I specificid Sign on to the CAVM as active detected.
```

Starts terminal control catch-up and tracking

When initialization of the alternate CICS is suspended pending takeover, terminal control tracking and catch-up begins and the following messages are issued:

```
DFHTC1041I specificid  Terminal control tracking started.
DFHTC1044I specificid  Terminal control catch-up started.
DFHTC1045I specificid  Terminal control catch-up complete.
```

The first stage of terminal control tracking is the catch-up phase. The active system notifies the alternate system of the current state of all its VTAM terminals (via the CAVM). See the *CICS XRF Guide* for information about terminal control tracking.

The alternate system continues surveillance and tracking

The alternate system is now in standby mode. Because it is only partially initialized, it cannot process normal transactions. In standby mode, the alternate system:

- Monitors the active system for any sign of failure
- Reads the active system's CAVM messages, and uses the information contained in them to maintain its own TCT.

If you shut down the active CICS normally, the alternate CICS also terminates; otherwise, the final task it performs as an alternate CICS is to take over from the active CICS. Takeover is started by explicit command, or because of actual or apparent failure of the active CICS, subject to the setting of the TAKEOVR system initialization parameter.

Shutting down a CICS system

If you shut down the active CICS normally, any alternate CICS system that was tracking the active CICS also shuts down normally. If you shut down the alternate CICS the active CICS is aware of your shutting down the alternate CICS because the alternate CICS signs off from CAVM. However, the active CICS continues running normally, but without any standby. You can start another alternate CICS to accompany the active CICS whenever you need one.

Shutting down the active CICS normally

This section lists the main points in the shutdown of an active CICS system. (See the flowchart Figure 3 on page 33.)

1. **Issue the shutdown command**

   When you issue the shutdown command to the active CICS, it performs normal CICS shutdown processing, beginning with the following message sequence:

```
DFHDM0102I specificid CICS is quiescing
DFHTM1715I specificid CICS is being quiesced by userid
           username in transaction tranname at terminal
           termname
DFHZC2305I specificid  Termination of VTAM sessions beginning
DFHZC2316I specificid  VTAM ACB is closed
```

2. **Signoff from the CAVM**

   The active CICS does not sign off from the CAVM until all tasks have terminated, and the warm keypoint is successfully taken, as shown by the following sequence of console messages:

   ```
   DFHJC4510   specificid  All open journals now closed
   DFHWK0105I  specificid  Warm keypoint successful
   DFHXG6404I  specificid  Signing off normally from the CAVM
   DFHXG6405I  specificid  Sign off normal from the CAVM accepted
   ```

3. **Termination complete**

   The shutdown of the active CICS is complete when the following console message and VSE EOJ message, giving a zero completion code, are issued:

   ```
   DFHKE1799 specificid  Termination of CICS is complete
   EOJ jobname MAX. RETURN CODE = 0000
   ```

4. **The signoff is detected by the alternate CICS**

   When the active CICS signs off normally from the CAVM, the alternate CICS detects that the shutdown of the active CICS it was tracking is normal, and issues the following message to the console:

   ```
   DFHXG6507 specificid Sign off normal from the CAVM detected
   ```

5. **The alternate CICS initiates its own shutdown**

   As a result of noting the normal shutdown of the active CICS, the alternate CICS starts its own shutdown, issuing the following message:

   ```
   DFHXG6520 specificid  CICS shutdown initiated by CAVM event
   ```

6. **The alternate CICS signs off from the CAVM**

   After issuing the shutdown initiation message, the alternate CICS signs off from the CAVM with the following messages:

   ```
   DFHXG6404I specificid  Signing off normally from the CAVM
   DFHXG6405I specificid  Sign off normal from the CAVM accepted
   ```

7. **Termination of the alternate complete**

   You know that shutdown of the alternate CICS is complete when you receive the following message and the VSE EOJ message, giving a zero completion code:

   ```
   DFHKE1799 specificid  Termination of CICS is complete
   EOJ jobname MAX. RETURN CODE 0000
   ```

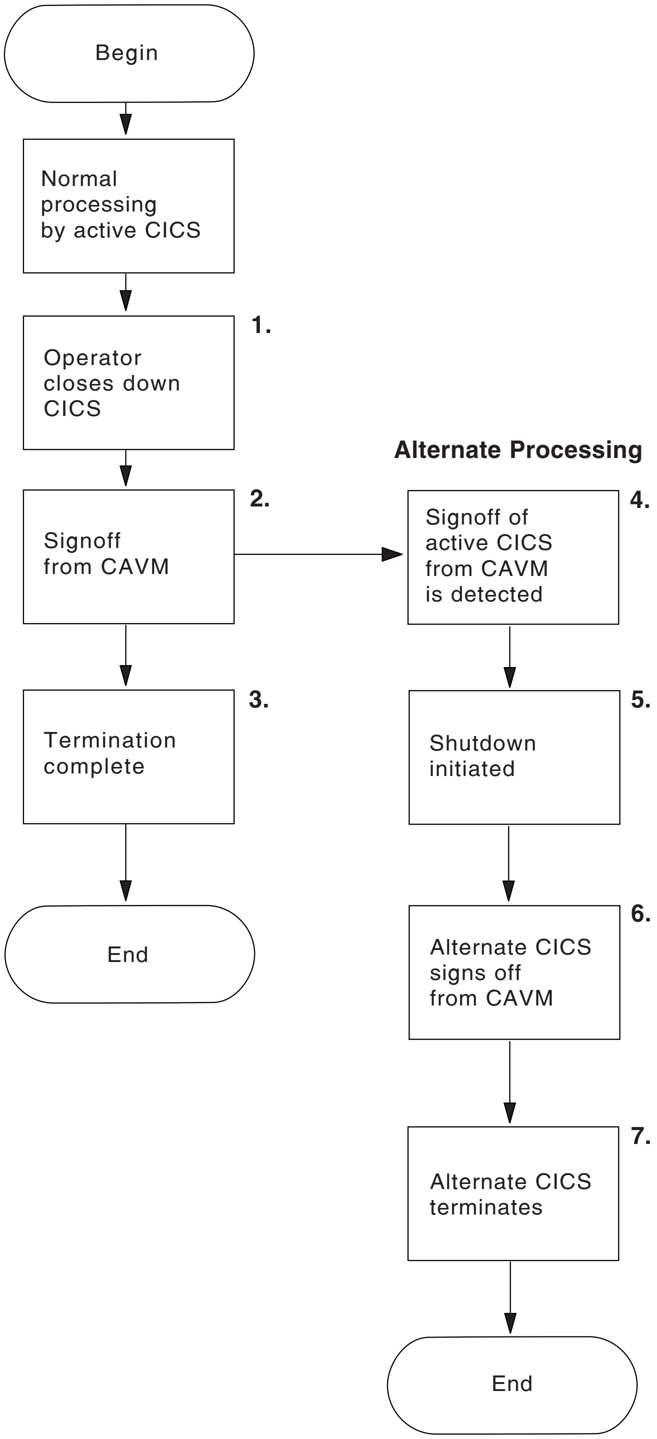

Begin
Normal processing by active CICS
1.
Operator closes down CICS
Alternate Processing
2.
Signoff from CAVM
4.
Signoff of active CICS from CAVM is detected
3.
Termination complete
5.
Shutdown initiated
End
6.
Alternate CICS signs off from CAVM
7.
Alternate CICS terminates
End

Figure 3. Shutting down the active CICS normally

Shutting down the alternate CICS system normally

This section briefly lists the main points in the shutdown of an alternate CICS system. (See the flowchart Figure 4 on page 35.)

1. **Issue the shutdown command**
   When you issue the CEBT PERFORM SHUTDOWN command to close an alternate CICS system, the alternate CICS starts to shut down, issuing the following messages:

   ```
   DFHXA6521I specificid  CICS shutdown initiated by CEBT event
   DFHDTM1715 specificid CICS is being quiesced by userid
        CICSUSER in transaction CSSY
   ```

   If you add the IMMEDIATE option to the CEBT transaction, the active CICS informs the operator with the following message:

   ```
   DFHXG6411I specificid  Sign off abnormal from the CAVM detected.
   ```

2. **Signoff from the CAVM**
   The alternate CICS signs off normally from the CAVM, issuing the following messages:

   ```
   DFHXG6404I specificid  Signing off normally from the CAVM
   DFHXG6405I specificid  Sign off normal from the CAVM accepted
   ```

3. **Termination complete**
   The successful shutdown of the alternate CICS is shown by the same VSE message as for the active CICS system.

4. **Active CICS informs operator of the alternate system’s signoff**
   When the alternate CICS signs off from the CAVM the sign-off is detected by the active CICS, which notifies the operator with the following message:

   ```
   DFHXG6407I specificid  Sign off normal from the CAVM detected
   ```

5. **The active CICS continues normal processing**
   The active CICS system continues normal processing, but without an alternate CICS as a standby system. However, the active CICS remains signed on to the CAVM, and you can start a new alternate CICS system at any time. This is detected by the active CICS, and surveillance and tracking resumes after the catch-up phase is completed.

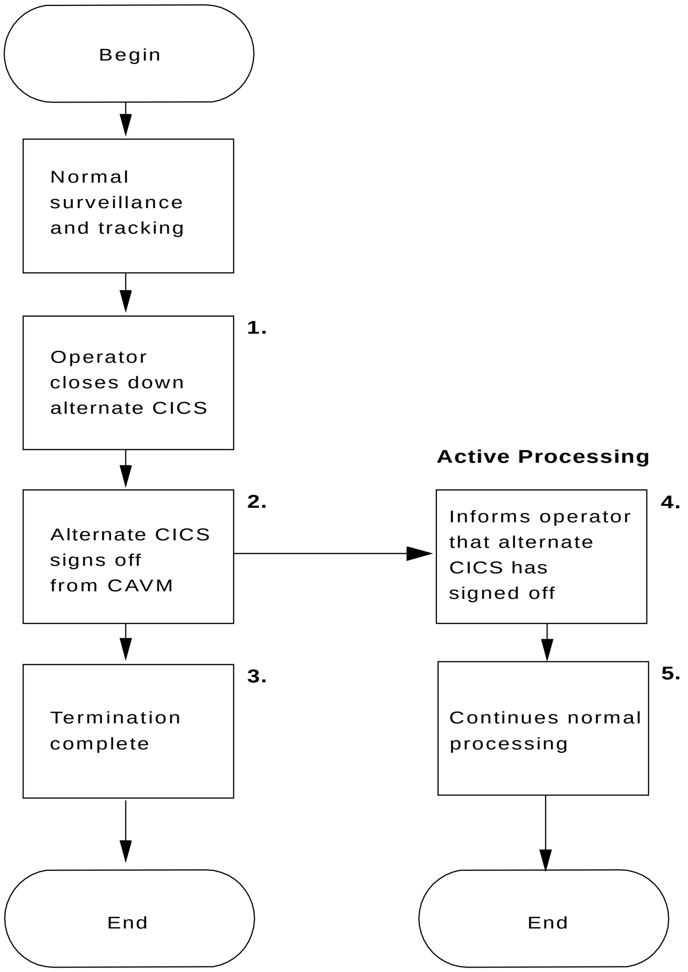

Begin
Normal surveillance and tracking
1.
Operator closes down alternate CICS
Active Processing
2.
Alternate CICS signs off from CAVM
4.
Informs operator that alternate CICS has signed off
3.
Termination complete
5.
Continues normal processing
End
End

Figure 4. Shutting down the alternate CICS normally

Takeover

This section provides an overview of takeover activity. It discusses:

- What events can cause a takeover
- When a takeover occurs
- What happens during a takeover
- Post-takeover processing

The alternate CICS monitors an active CICS, and takes no action as long as the surveillance signal is updated on time and no state change occurs.

What events can cause a takeover?

The events that cause takeover initiation, subject to the options specified in TAKEOVR system initialization parameter of the alternate CICS, are:

- An operator at a console issues an explicit CEBT PERFORM TAKEOVER command to an alternate CICS system. The CEBT PERFORM TAKEOVER command can also be issued during a takeover controlled by the command list table (CLT). See the *CICS XRF Guide* for detailed information about takeovers controlled by the CLT.
- An active system realizes that it is failing and signs off abnormally from the CAVM. The alternate CICS is governed by the takeover options defined by the system initialization parameter TAKEOVR, and acts as follows:
  - If **TAKEOVR=AUTO** or **MANUAL**, the alternate CICS begins takeover processing.
  - If **TAKEOVR=COMMAND**, the alternate CICS takes no action, and continues in standby mode, even though the active job has terminated abnormally. It can only perform a takeover when it receives an explicit CEBT command through the console. In this situation, you can choose to restart the active CICS in place, in which case the alternate CICS recognizes this and resumes tracking; or you can instruct the alternate CICS to take over by issuing a CEBT PERFORM TAKEOVER command.
- An active system might fail, but be unable to sign off abnormally. The disappearance of the surveillance signal indicates possible failure to the alternate system, and it acts as follows:
  - If **TAKEOVR=AUTO** or **MANUAL**, the alternate CICS begins takeover processing.
  - If **TAKEOVR=COMMAND**, the alternate CICS takes no action, and continues in standby mode. Again, you have the choice of restarting the active system in place, or instructing the alternate system to takeover by means of an explicit CEBT command through the console.

Figure 5 on page 37 sets out the causes of a takeover event, and Table 6 on page 37 summarizes the takeover events and the types of takeover that they can lead to.

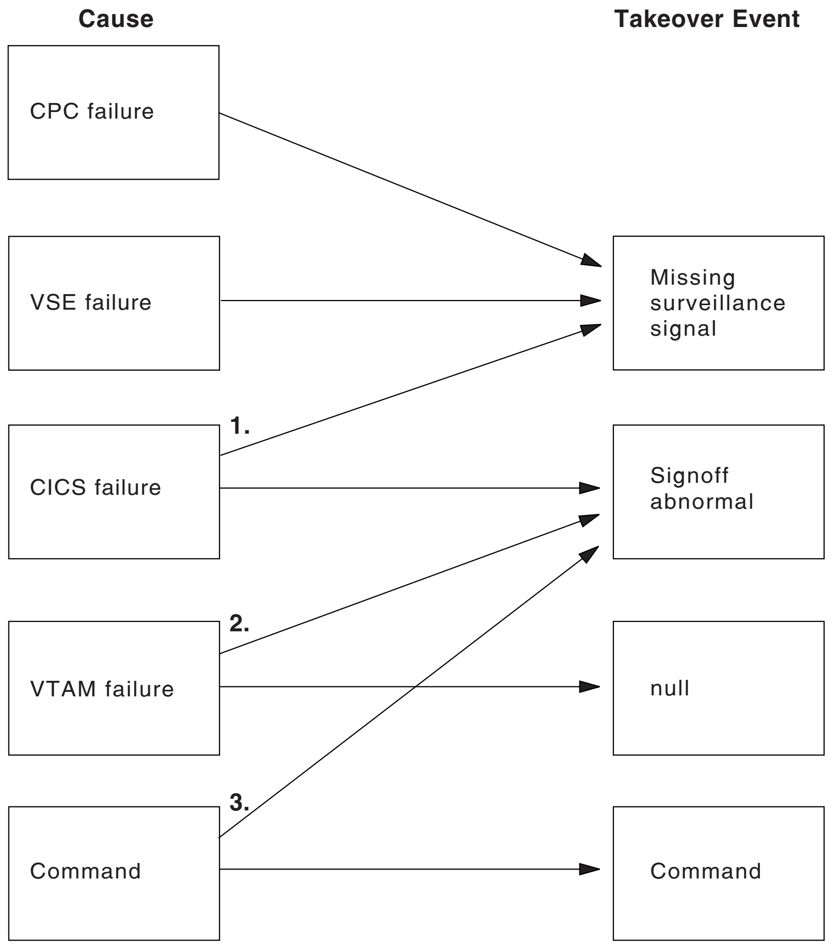


Figure 5. Causes of a takeover event

Notes:

1. For some CICS failures, CICS might be able to sign off abnormally from the CAVM. If it cannot sign off, the alternate system detects the missing surveillance signal.
2. A VTAM failure drives CICS user exit XXRSTAT. You code the exit to determine whether to cause a takeover by sign-off abnormal, or to allow CICS to continue normally while you restart VTAM. For further guidance information, see the *CICS Customization Guide*.
3. The commands have already been described. They are the CEBT PERFORM TAKEOVER command, issued to the alternate CICS system, and the CEMT PERFORM SHUTDOWN TAKEOVER|IMMEDIATE command, issued to the active CICS system. Both the immediate and takeover options on the CEMT command cause the active system to sign off abnormally.

Table 6 (Page 1 of 2). Types of takeover

Event	TAKEOVR= AUTO	TAKEOVR= MANUAL	TAKEOVR= COMMAND
Operator or program issues a CEBT command	Takeover	Takeover	Takeover
Signoff abnormal	Takeover	Takeover	No takeover

Table 6 (Page 2 of 2). Types of takeover

Event	TAKEOVR= AUTO	TAKEOVR= MANUAL	TAKEOVR= COMMAND
Active system's surveillance signal ceases	Takeover	Conditional takeover	No takeover
Operator issues a CEMT command	Takeover	Takeover	No takeover

When does a takeover occur?

As you can see from Table 6, the type of takeover that takes place depends upon the event and the takeover option defined in the alternate CICS system. The resultant action may be a takeover (which in one case is conditional), or no takeover at all.

If you specify TAKEOVR=COMMAND for an alternate CICS, events other than a specific CEBT takeover command do not initiate a takeover. Otherwise, a takeover can be initiated without involving the system operator. The events that lead to a takeover are as follows:

A CEBT PERFORM TAKEOVER command is issued

A CEBT PERFORM TAKEOVER command is used to control an alternate CICS system only. It can be issued through the VSE console, or issued by a program. For example, the command may be issued by another CICS region processing a command list table (CLT). When the alternate system receives this command, it starts to take over from the active CICS that it has been tracking, without reference to the operator, **regardless of the current takeover option**.

The active CICS signs off abnormally from the CAVM

If the active CICS signs off abnormally from the CAVM, for whatever reason, and the takeover option is **not** COMMAND, the alternate CICS initiates a takeover.

The alternate CICS detects that surveillance signals have ceased

If the alternate CICS detects that the active system's surveillance signals have ceased, it displays the following message:

```
DFHXG6516I - specificid : Apparent failure of active CICS detected.
```

After issuing message DFHXG6516I, the action taken by the alternate CICS depends upon its current takeover option. If the takeover option is **COMMAND**, the alternate system **does not** initiate a takeover. The actions resulting from the other takeover options are as follows:

TAKEOVR=AUTO: The alternate system initiates a takeover automatically, as soon as the alternate delay interval (ADI) has elapsed.

TAKEOVR=MANUAL: If the current takeover option is manual, the alternate system issues the following message:

```
DFHXG6518D specificid  APPARENT FAILURE OF ACTIVE CICS DETECTED.
                                 REPLY TAKEOVER OR IGNORE
```

After issuing message DFHXG6518D, the action taken by the alternate CICS is dependent on your response to the messages displayed on the console.

If you are able to repair the active CICS, perhaps by replying to an outstanding message, reply "IGNORE" to DFHXG6518I. If the active CICS resumes normally, and the alternate system detects the active system's surveillance signal, it issues the following message:

```
DFHXG6517I specificid  Recovery of active CICS detected
```

If you cannot repair the active CICS, reply "TAKEOVER", in which case the alternate system queues a takeover request, and displays the following message:

```
DFHXG6512I specificid  Takeover request passed to the CAVM
```

What happens following the takeover request is discussed in "Post-takeover processing" on page 41.

A CEMT PERFORM SHUT TAKEOVER (or IMMEDIATE) command is issued: A CEMT PERFORM SHUT TAKEOVER (or IMMEDIATE) command can be used to initiate a takeover by instructing the active CICS to shut down, and sign off abnormally from the CAVM. Note, a takeover only occurs if automatic or manual takeover options are defined for the alternate system.

What happens during a takeover

When the alternate CICS tries to take over from the active CICS, a takeover request is passed to the CAVM, and the following message is written to the system console:

```
DFHXG6512I specificid  Takeover request passed to the CAVM
```

In most cases the request is accepted, but it may be rejected for any of the following reasons:

- The active CICS system has already signed off normally.
- The active CICS is not the same active system that the alternate system has been tracking. The CAVM detects that it is a new active system, probably as a result of a restart in place. In this case, the alternate system is no longer able to continue its role, and a new alternate system should be started.
- The active and alternate systems are on different CPCs, and the alternate system has not been monitoring the active system's surveillance signals long enough to assess the difference between the time-of-day clocks.

If the CAVM accepts the request, the following message is written to the console:

```
DFHXG6513I specificid  Takeover request accepted by the CAVM
```

When this takeover request has been accepted, an attempt by another CICS to sign on to the CAVM as an active system is rejected. The alternate system next issues the MODIFY netname,USERVAR command to redefine the CICS application name.

During takeover, the alternate system uses two different mechanisms to try to force the termination of the active CICS job.

1. If the active CICS is still signed on to the CAVM, the alternate system uses the surveillance mechanism to try to pass a "takeover-requested" message to the active system, including a "dump" or "no dump" indicator. If the active system receives the message, it responds by issuing an abend, and eventually signs off abnormally from the CAVM.

2. If the active job is still being executed, the alternate system also issues a PCANCEL command (prefixed with PXMIT, the POWER routing command in a 2-CPC configuration). The PCANCEL command is issued in case the active CICS is unable to respond to the alternate system's request to take over. You must code AUTH=NOJOB in the PNODE definition for the applicable CPC.

   For guidance information about defining the PNODE tables, see the *VSE/POWER Application Programming* manual.

   Note that in some cases the PCANCEL command may take effect after the active CICS has already issued the abend, in which case CICS issues the following message:

   ```
   DFHSR0612 applid  Abend recovery has been reentered by same task.
   ```

The alternate CICS next starts to process the command list table (CLT).

Checking for termination of the active CICS

The alternate CICS asks VSE/POWER periodically about the status of the active CICS. If POWER replies that the job has terminated, the next phase (see "Completing the takeover" on page 41) can start immediately.

If POWER replies that the job is still being executed, the alternate CICS continues asking about the status until the time interval defined by the XRFTODI system initialization parameter expires, in which case the alternate system issues the following message:

```
DFHXA6561D specificid  WHEN ACTIVE JOB
           powerjobname,powerno ENDS REPLY 'JOB' OR WHEN CPC
           powersysid HAS FAILED REPLY 'CPC'
```

The alternate CICS also issues DFHXA6561D if POWER is not running, or does not respond.

When active and alternate CICS systems are running in different CPCs, POWER might continue to tell the alternate CICS that the active job is still running even though the active CPC or its VSE system has failed. In this case, the alternate CICS cannot complete its takeover without operator intervention.

Another possibility is that the active job is still running, and either never received the PCANCEL command, or received it but cannot terminate because a system error necessitating further action from the operator has occurred.

If the active system's CPC has not failed, you must take steps to ensure that the active job really has terminated before replying "JOB".

If the active system's CPC appears to have failed, you must:

1. Decide whether the best way to recover is to re-IPL.
2. If you decide to re-IPL, stop the processors of the failed CPC, then perform system reset.
3. When the reset is successfully completed, reply "CPC" to the alternate system's DFHXA6561D message.
4. If possible, take a stand-alone dump.
5. Re-IPL the active system's CPC.

If you reply “CPC”, an internal record is kept that the CPC, identified by the POWER SYSID, has failed. Other alternate CICS systems examine this record while they are performing a takeover, to try to avoid operator intervention.

Note: If you reply to message DFHXA6561D prematurely, data integrity may be lost.

The alternate CICS cannot complete takeover until you reply to message DFHXA6561D, unless either:

- The alternate system receives a late reply from POWER saying that the active job has terminated, and CICS issues the following message:

```
DFHXA6564I specificid  TERMINATION OF ACTIVE JOB powerjobname,
                                   powerno DETECTED
```

or

- A previous reply to another alternate system’s DFHXA6561D message has already confirmed CPC failure, and CICS issues the following message:

```
DFHXA6563I specificid  ACTIVE JOB powerjobname,powerno
              ENDED DUE TO FAILURE OF CPC powersysid
```

If either of these events occurs, a reply to DFHXA6561D is no longer required, and takeover continues.

Completing the takeover

When CAVM receives confirmation that the active CICS job has terminated, it notifies the alternate CICS that it may now assume the fully active role, and updates the CAVM control data set to this effect.

Emergency restart resumes, but in a 2-CPC environment it is delayed until the new active’s time-of-day clock has caught up.

Post-takeover processing

In a 2-CPC configuration, additional operator activity may be required as a direct consequence of a takeover. For example:

- Physical switching of terminals
- Switching of subsystems, such as DL/I

Three types of takeover

In general, a takeover is initiated by one of three events:

- A takeover command is issued, either by an operator or by another CICS region.
- The alternate system detects the absence of the active system’s surveillance signal.
- The alternate system detects that the active system has signed off abnormally.

See the *CICS XRF Guide* for more guidance information about the takeover process.

Takeover after a command

This section lists the main points of a takeover that follows from a takeover command.

Chart T-1

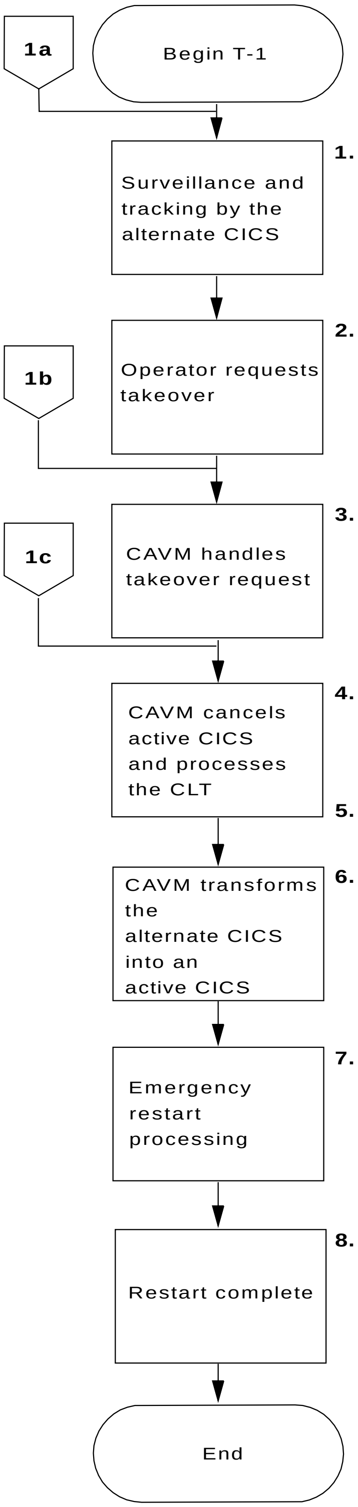

1a
Begin T-1
1.
Surveillance and tracking by the alternate CICS
2.
1b
Operator requests takeover
3.
1c
CAVM handles takeover request
4.
CAVM cancels active CICS and processes the CLT
5.
6.
CAVM transforms the alternate CICS into an active CICS
7.
Emergency restart processing
8.
Restart complete
End

Figure 6. Takeover requested by an operator-issued command

1. **Surveillance and tracking running normally**
   While the active CICS is running normally, surveillance and tracking by the alternate system continues. However, although the active system is running normally, you may need to perform a takeover for maintenance reasons. For example:

   - You may need to apply service to the running CICS system.
   - If the alternate system is running on a different CPC from the active system, you may need to perform a takeover to apply maintenance to the active system's VSE system.

2. **Issue the takeover command**
   You can initiate a takeover by either:

   - Issuing a CEBT PERFORM TAKEOVER command to the alternate CICS, or
   - Issuing a CEMT PERFORM SHUT TAKEOVER command to the active CICS.

   When you issue a takeover command, the alternate CICS attempts to start the takeover process. It does this regardless of the type of takeover defined by the TAKEOVR=AUTO|MANUAL|COMMAND system initialization parameter, which is overridden when you issue a takeover command.

3. **The CAVM handles the takeover request**
   The takeover request might not be accepted by the CAVM. For example, if:

   - The active system has already signed off normally from the CAVM, but the alternate system has not detected this yet.
   - The necessary time-of-day clock difference is not available (message DFHXC6634I) when the active and alternate CICS systems are running on different CPCs.

   If the takeover cannot proceed, the alternate CICS issues the following message:

   ```
   DFHXG6514I specificid Takeover request rejected by the CAVM
   ```

   If your takeover command is rejected, you also receive a message from the CAVM stating why. If your takeover command is accepted, you receive message DFHXG6513 accepting the request.

4. **The CAVM cancels the active CICS**
   The alternate system uses two mechanisms to force an active CICS system to terminate:

   - It uses the surveillance system to pass a "takeover-requested" message to the active CICS, and
   - It issues a POWER CANCEL command (PCANCEL) or a POWER TRANSMIT command (PXMIT) to transmit the PCANCEL command to the POWER system, which is controlling the active system, to cancel the active job.

   One of these normally causes the active CICS to terminate abnormally.

   If VSE rejects the CAVM's CANCEL command, the alternate system issues the following message:

```
DFHXA6560A specificid TERMINATION COMMAND FAILED: command
```

If, for some reason, the active system does not respond to the CAVM's takeover request either, you have to cancel the active CICS job manually to allow takeover to proceed.

As soon as the CAVM accepts the alternate system's takeover request, the alternate system issues the VSE MODIFY netname,USERVAR command to VTAM to change the VTAM user variable to the new specific applid.

5. **The alternate system processes the command list table (CLT)**
   At this point, if the alternate system taking over is a master or coordinator region in a 2-CPC configuration, CEBT PERFORM TAKEOVER commands are sent to dependent regions. In addition, any write-to-operator (WTO) messages in the CLT are sent to the system console. Any other VSE commands that you include in the CLT are also processed. See the *CICS XRF Guide* for more information about the CLT.

6. **The alternate CICS becomes the new active CICS**
   When the active CICS has signed off abnormally or terminated, the CAVM stops alternate mode surveillance processing, and starts active mode surveillance processing. When CAVM detects that the active CICS job has also terminated, it notifies the alternate system that it may now assume the fully active role.

7. **Emergency restart resumed**
   The emergency restart of CICS was suspended during transient data initialization pending the takeover. Now that the alternate CICS has become the active CICS, emergency restart is resumed.

   In addition to normal emergency restart, the new active CICS reestablishes the sessions for VTAM terminals.

   If the new active CICS is running in a different CPC from the old active CICS, time-of-day clock synchronization takes place, and the following messages are issued:

   ```
   DFHXG6682I specificid  XRF clock synchronization started
   DFHXG6683I specificid  XRF clock synchronization ended
   ```

   In a 2-CPC environment, if the time-of-day clocks have not been synchronized closely enough, emergency restart processing may be delayed. (Note that messages DFHXG6682I and DFHXG6683I only occur in a 2-CPC environment.)

8. **Emergency restart complete**
   When the restart is complete, control is given to CICS and transaction processing can resume.

Takeover after missing surveillance signal

This section lists the main points of a takeover that follows from a missing surveillance signal.

Chart T-2

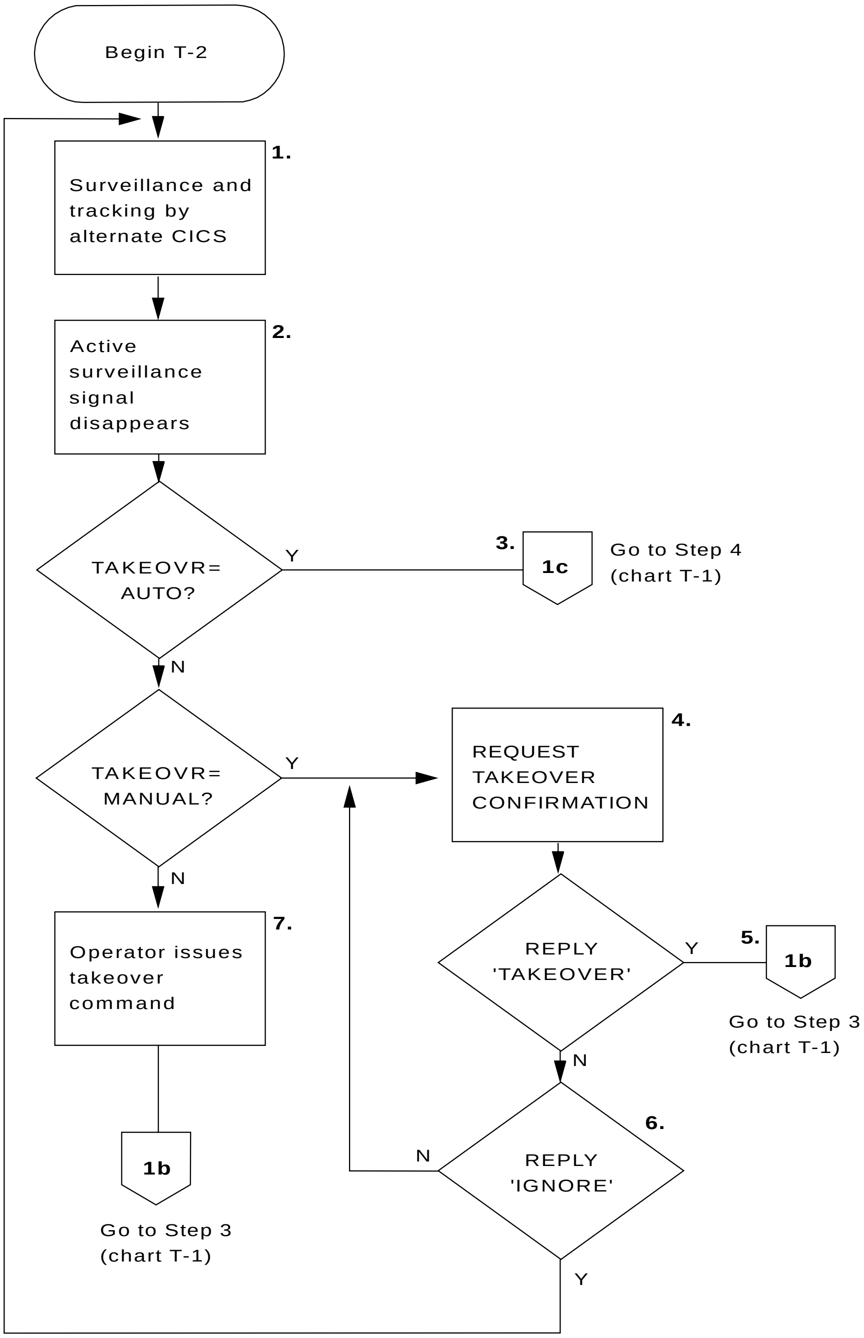


Figure 7. Takeover after missing surveillance signal

1. **Surveillance and tracking running normally**
   While the active system is running normally, surveillance and tracking by the alternate system continues. As long as the alternate system can detect the active system's surveillance signal, the alternate system remains in standby mode.

2. **The active system’s surveillance signal disappears**

   If the alternate system fails to detect the active system’s surveillance signal, the following message is issued at the console:

   ```
   DFHXG6516I specificid  Apparent failure of active CICS detected.
   ```

   The action that the alternate system takes next depends on the takeover option that you specified on the TAKEOVR system initialization parameter.

3. **Takeover option is AUTO**

   If you specified TAKEOVR=AUTO, then the CAVM automatically initiates a takeover. The process then continues as described in the takeover after a command—see Step 4 on page 43.

4. **Takeover option is MANUAL**

   If you specified TAKEOVR=MANUAL, then the CAVM issues a WTOR message requesting the operator to state what action to take:

   ```
   DFHXG6518D specificid APPARENT FAILURE OF ACTIVE CICS DETECTED.
                    REPLY "TAKEOVER" OR "IGNORE"
   ```

   It is then the responsibility of the operator to decide whether or not to permit the alternate system to perform the takeover. However, while the alternate system is waiting for your reply, it may detect another (overriding) event. For example:

   - If the surveillance signal resumes before you reply to message DFHXG6518D, then the following messages are sent to the system console:

     ```
     DFHXG6517I specificid  Recovery of active CICS detected.
     DFHXG6519I specificid  The reply to message DFHXG6518 is
                                         assumed to be IGNORE.
     ```

   - If the active system signs off normally before you reply, then the alternate system terminates normally after sending the following messages:

     ```
     DFHXG6507I specificid  Sign off normal from the CAVM detected.
     DFHXG6519I specificid  The reply to message DFHXG6518 is
                                         assumed to be IGNORE.
     ```

   - If the active system signs off abnormally before you reply, then the alternate system initiates a takeover after issuing the following messages:

     ```
     DFHXG6511I specificid  Sign off abnormal from the CAVM detected.
     DFHXG6519I specificid  The reply to message DFHXG6518 is
                                         assumed to be IGNORE.
     ```

5. **Operator reply is “TAKEOVER”**

   If you reply “TAKEOVER”, the alternate system proceeds to carry out a takeover in the same way as if you had issued a CEBT PERFORM TAKEOVER command. The process then continues as described in the takeover after a command—see Step 3 on page 43.

6. **Operator reply is “IGNORE”**

   If you reply “IGNORE”, the alternate system assumes either that the active CICS will recover from the apparent failure, or that it will be restarted. It continues running as an alternate CICS without an active partner, monitoring through the CAVM to see if an active system is started, or the old active system resumes.

7. **Takeover option is COMMAND**
   If you specified TAKEOVR=COMMAND, you must determine the cause of the problem on the active system, and, if it cannot be resolved, issue the takeover command. The process then continues as described in the takeover after a command—see Step 3 on page 43.

Takeover after abnormal signoff

This section lists the main points of a takeover that follows when the active system signs off abnormally from the CAVM.

Chart T-3

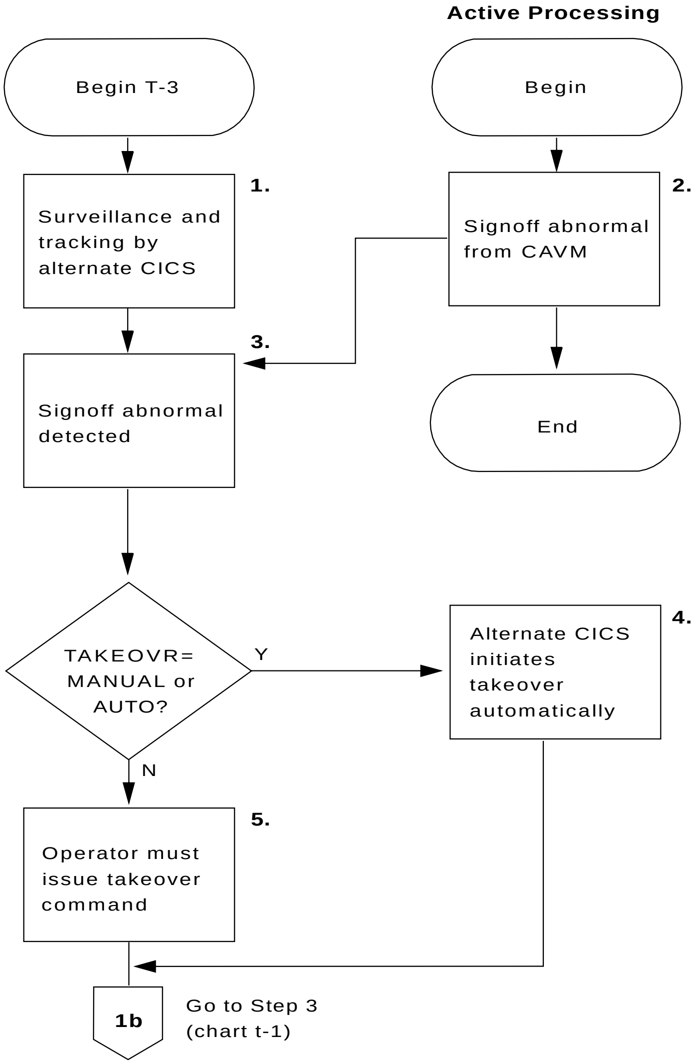

Active Processing
Begin T-3
Begin
1.
Surveillance and tracking by alternate CICS
2.
Signoff abnormal from CAVM
3.
Signoff abnormal detected
End
TAKEOVR= MANUAL or AUTO?
Y
4.
Alternate CICS initiates takeover automatically
N
5.
Operator must issue takeover command
1b
Go to Step 3 (chart t-1)

Figure 8. Takeover following abnormal signoff of the active from the CAVM

1. **Surveillance and tracking running normally**
   While the active system is running normally, surveillance and tracking by the alternate system continues, and the alternate system remains in standby mode.

2. **The active system signs off abnormally**
   The active CICS signs off abnormally with the following messages:

```
DFHXG6408I specificid  Signing off abnormally from the CAVM
DFHXG6409I specificid  Sign off abnormal from the CAVM accepted
```

3. **The active system's abnormal signoff is detected**
   The alternate detects the abnormal signoff and issues the following message:

   ```
   DFHXG6511I specificid  Sign off abnormal from the CAVM detected
   ```

   The action that the alternate system takes next depends on the takeover option that you specified on the TAKEOVR system initialization parameter.

4. **Takeover option is AUTO or MANUAL**
   If you specified TAKEOVR=AUTO, or TAKEOVR=MANUAL, the alternate system initiates a takeover, and issues the following message:

   ```
   DFHXG6512I specificid  Takeover request passed to the CAVM
   ```

   The process then continues as described in the takeover after a command—see Step 3 on page 43.

5. **Takeover option is COMMAND**
   If you specified TAKEOVR=COMMAND, you must determine the cause of the problem on the active system, and, if it cannot be resolved, issue the takeover command. The process then continues as described in the takeover after a command—see Step 3 on page 43.

Note: If a new active CICS signs on to the CAVM before the alternate system can complete the takeover from a failed active system, the alternate system signs off normally from the CAVM, and terminates normally. The alternate system terminates because it cannot be sure that its own resource definitions correspond with those of the new active CICS. This situation is indicated by the following CAVM message:

```
DFHXC6631I specificid  TAKEOVER REJECTED BECAUSE LATEST ACTIVE
                            INSTANCE NUMBER DOES NOT MATCH THAT SPECIFIED
```

When this happens, you must start a new alternate system to continue with XRF operation.

If the alternate system fails

The active CICS may detect that:

- The surveillance signal for the alternate system is overdue, according to the time-limit defined by the PDI system initialization parameter
- The alternate CICS has signed off abnormally.

If the alternate system's surveillance signal has stopped, the active CICS displays the following message:

```
DFHXG6416I specificid  APPARENT FAILURE OF ALTERNATE CICS DETECTED
```

If the alternate system has failed and signed off abnormally, the active CICS displays the following message:

```
DFHXG6411I specificid  Sign off abnormal from the CAVM detected
```

You can use the overseer to restart a failed alternate system. However, if you need to determine the nature of the failure, you might need to delay its restart.

Operating an XRF overseer

The *CICS Customization Guide* gives more information about the CICS overseer, and describes how to add your own code to the sample program provided, so that the overseer performs the functions that you want it to. This section looks at the operational aspects of the sample overseer program, DFH$AXRO.

You use the overseer in two different types of situation:

1. While the system is running normally, to monitor the active or alternate status of CICS systems running with XRF=YES
2. During failures, to restart failed CICS regions in place (probably failed dependent regions in 2-CPC operation)

The overseer sends messages to the system console. The operator can communicate with the overseer via console commands. Valid commands, which can be abbreviated to as little as the initial letter, are:

Display
Displays the current status of all CICS systems known to the overseer.

Restart
Enables and disables restart-in-place for all CICS systems known to the overseer.

Open
Asks the overseer to retry the open for CAVM data sets which failed to open previously.

Snap
Requests that a dump be taken of storage for a specified partition or the SVA. If the JCL used to execute the overseer includes the

```
// OPTION SYSDUMP
```

statement, an SDUMP is performed and the dump is placed in the dump library named in the

```
// LIBDEF DUMP,CATALOG ...
```

statement. If the

```
// OPTION SYSDUMP
```

statement is not included, the dump is sent to SYSLST.

End
Terminates the overseer.

? Requests the overseer to display, on the system console, a full list of available overseer commands with a description of their function.

The first two commands cover the major two functions covered by the overseer, namely display and restart in place.

Display command

You can monitor the status of a CICS system running with XRF by using the display command to provide a WTO over multiple lines (MLWTO) to show the last known state of each of the CICS systems defined to the overseer. The display shows a title line and underneath this a line for each active-alternate pair of CICS systems. Each display line includes:

- The generic applid.
- The job name of the active CICS.
- The specific applid of the active CICS.
- The POWER SYSID of the CPU on which the active system is being executed, or was last known to have been executed.
- The current, or last known, status of the active CICS. The status can be:

  ACT The active CICS is signed on to the CAVM and is running, or about to run, the active CICS workload.

  INCA An incipient active system, meaning the alternate CICS is taking over from the active CICS, and the alternate system is now partially active. The old active CICS has signed off abnormally from the CAVM, but has not yet terminated.

  SOFA The active system has signed off abnormally from the CAVM.

  SOFN The active has signed off normally from the CAVM as a result of a normal shutdown.

  TKIP Takeover in progress. An alternate CICS is attempting a takeover of this active system. When the takeover is complete, the status is changed.

  UNKN Unknown, meaning that the overseer has no current information about the status, which was in an intermediate state when the display command was processed. Reissue the command to replace UNKN by a more meaningful status value.

  An 'X' following any of these states indicates that the job is being executed. No 'X' indicates that the job is not being executed.

- The job name of the alternate CICS.
- The specific applid of the alternate CICS.
- The POWER SYSID of the CPU on which the alternate system is being executed (or was last known to have been executed).
- The current, or last known, status of the alternate CICS. The status can be:

  BACK The alternate CICS signed on to the CAVM and is running normally.

  SOFA The alternate system has signed off abnormally from the CAVM.

  OLD The information displayed for the alternate refers to out-of-date information about the system that was the alternate until a recent takeover. That system is the current active, and the information displayed for the alternate is marked as OLD until a new alternate is signed on on and running normally.

  SOFN The alternate system has signed off normally from the CAVM, either as a result of a normal shutdown by the active system, or as a result of a CEBT PERFORM SHUT command issued to the alternate system.

TKOV A takeover has been initiated by this alternate system.

UNKN Unknown, meaning that the overseer has no current information about the status, which was in an intermediate state when the display command was processed. Reissue the command to replace UNKN by a more meaningful status value.

An ‘X’ following any of these states indicates that the job is being executed. No ‘X’ indicates the job is not being executed.

- The takeover option of the alternate system.

Restart-in-place command

The overseer sample program is designed to restart failed active and alternate CICS systems automatically under certain circumstances. It achieves this by sending a PRELEASE RDR command to POWER.

You use the restart-in-place (or R) command to enable or disable the ability of the overseer to attempt to restart a failed region. After initialization, restart is enabled. If you issue the ‘R’ command, you disable restart processing. If you issue it again, you reenable restart processing, and so on. Each time you issue the ‘R’ command, the overseer displays a message to inform you of the effect. Restart processing is enabled and disabled for all jobs known to the overseer.

In brief, the restart-in-place command is used when you would prefer to restart a region in the same CPC, instead of allowing an alternate system in a different CPC to take over. A dependent region in a 2-CPC configuration is a good example, but you might also prefer, if possible, to restart-in-place regions that you would normally expect to be taken over, like a VTAM terminal-owning region. For each region, you have to weigh up the possibility that, if successful, a restart in place could involve less interruption to service than a takeover to another CPC. Guidance information about how to define each region to the overseer, and about the circumstances in which it attempts to restart in place, are in the *CICS Customization Guide*.

How the overseer restarts failed regions in place

The job to be restarted must be placed on the POWER RDR QUEUE with a disposition of ‘L’. To restart the job, the overseer sends a POWER PRELEASE RDR command for the associated job name and number.

- To restart a failed active or alternate CICS job, DFH$AXRO issues the following command:

  ```
  R RDR,xxxxxxxx,nnnnnnnn
  ```

  where *xxxxxxxx* is the job name of the CICS JOB that is being restarted, and *nnnnnnnn* is its job number.

Preparing to run the CICS overseer

The CICS overseer comprises three modules, two of which are CICS-supplied, and a user-written module, which you generate. A CICS-supplied sample of a user-written overseer program (called DFH$AXRO), that you can customize and rename, is supplied in the VSE/ESA sublibrary PRD1.BASE.

The CICS overseer modules, which are also supplied in the production sublibrary, are:

1. DFHWOSA – the XRF overseer region initialization module
2. DFHWOSB – the XRF overseer region service routines

The CICS overseer modules provide a stable interface to the CAVM data sets, and to some VSE supervisor and POWER services that the overseer program requires. The overseer program invokes those services through a CICS-supplied group of macros called the DFHWOSM macros. More information about the DFHWOSM macros is given in the *CICS Customization Guide*.

When the region is initialized, DFHWOSA invokes either the CICS-supplied sample overseer, or a user-written version. You supply the name as a parameter on the EXEC statement to start the overseer.

The rest of this chapter describes how to create an overseer data set for use with the CICS-supplied sample program, DFH$AXRO, and discusses the job control statements you can use to run the sample overseer.

Using a DFHOSD data set to tell the overseer what to monitor

The sample overseer program is written to handle active-alternate pairs of CICS systems, and related system names. The number is limited by the amount of storage available in the overseer partition.

A “related system name” identifies those regions or systems that cannot be considered in isolation by the overseer. The most common example of this is an MRO environment, where the overseer needs to be able to identify related systems when deciding whether to restart a failed job in place. Those regions or systems that are identified with a common related system name must be executed on the same CPC.

The sample program determines which active-alternate pairs it is monitoring from a VSAM key-sequenced data set with a file name of DFHOSD, which contains a single entry for each active-alternate pair.

You create the DFHOSD data set and initialize it with information about active-alternate pairs before you use the overseer for the first time.

You also have to redefine the DFHOSD data set whenever you want to change the information that it holds. The sample job stream listed in Figure 9 on page 54 can be used to:

1. Delete and define the DFHOSD data set
2. Initialize the DFHOSD data set with information about sample active-alternate pairs.

The sample overseer program reads the DFHOSD records in key sequence and builds a table of entries, stopping when the first 20 entries have been read.

Each active-alternate pair is known by its generic applid on this data set. The records in the DFHOSD data set contain the information described in Table 7.

Table 7. The DFHOSD record structure

Field and start position	**Description**
Record key, starting in column 1	A 12-byte key, containing the 4-byte value “GNbb” followed by the 8-byte generic applid of the active-alternate pair. (The applid starts in column 5.)
Related system identifier, starting in column 14	An optional 8-byte name, to identify related systems (for example, to identify related MRO regions).
Control data set file name, starting in column 23	The 7-byte file name, as on the DLBL statement in the JCL to run the overseer, that identifies the CAVM control data set associated with this generic applid.
Message data set file name, starting in column 32	The 7-byte file name, as on the DLBL statement in the JCL to run the overseer, that identifies the CAVM message data set associated with this generic applid.
Restart indicator, starting in column 41	A 1-byte restart-in-place indicator, to show whether a region can be restarted in place. The only value that prevents an attempt to restart in place is the letter N.

The data structure (DSECT) of the DFHOSD data set entries is provided in member DFH$XRDS of the production sublibrary. You can use the job in Figure 9 on page 54 to create and initialize DFHOSD.

```
* $$ JOB JNM=DFHIVXR,CLASS=A,DISP=D
// JOB DEFOSD   DELETE / DEFINE OSD FILE
// EXEC IDCAMS,SIZE=AUTO
   DELETE (user.dfhosd)                -
          CLUSTER                      -
   PURGE                               -
   CATALOG (usercat)
/*
// OPTION LOG
// EXEC IDCAMS,SIZE=AUTO
     DEFINE CLUSTER (TRACKS(1,1)-
      NAME (user.dfhosd)    -
      SHAREOPTIONS(2) -
      VOLUMES (volid)) -
     DATA -
      (RECSZ(80,80) -
      CONTROLINTERVALSIZE(2048) -
      NAME (user.dfhosd.data)     -
      KEYS(12,0) -
      FREESPACE(10 10))-
     INDEX -
      (NAME(user.dfhosd.index))    -
      CATALOG(usercat)
/*
/&
// JOB INITOSD   INITIALIZE THE OSD FILE
*
*  COLUMN
* 1   5        14       23       32       41
* GN  GEN-APP  RELATEID DDNXCTL  DDNXMSG  RND
*
// DLBL CATA,'usercat',,VSAM
// DLBL DFHOSD,'user.dfhosd',,VSAM,CAT=CATA
// EXEC IDCAMS,SIZE=AUTO
      REPRO  INFILE(SYSIPT) -
             OUTFILE(DFHOSD)
GN  DBDCCICS          CTLDBDC  MSGDBDC         1
GN  DBDCMRO1 MROSYS   CTLMRO1  MSGMRO1  N      2
GN  DBDCMRO2 MROSYS   CTLMRO2  MSGMRO2
GN  DBDCMRO3 MROSYS   CTLMRO3  MSGMRO3
/*
/&
* $$ EOJ
```

Figure 9. Sample job to create and initialize the DFHOSD data set

Notes for Figure 9:

1 This entry is for an XRF system that is not associated with any other CICS region, and does not require a related system identifier.

2 The last three entries are for three related MRO regions which are identified by the related system name of MROSYS. One of these, indicated by the letter N, is not allowed to restart in place.

Job control statements to run the CICS overseer

The job shown in Figure 10 is an example of the job control statements you need to run the CICS overseer in an XRF environment. The accompanying notes explain some of the points you must observe when writing a job stream to run a CICS overseer. This job is supplied in the production sublibrary with the pregenerated system.

```
* $$ JOB JNM=ovrseer,CLASS=A,DISP=D
// JOB RUNOSR
* ***********************************************************
* This job is an example of the job control statements needed
* to run the CICS overseer in an XRF environment.
* ***********************************************************
// DLBL CICSUCT,'usercat',,VSAM
// DLBL CTLDBDC,'user.dfhxctl',,VSAM,CAT=CICSUCT           1
// DLBL MSGDBDC,'user.dfhxmsg',,VSAM,CAT=CICSUCT           1
*  WARNING - THE FOLLOWING 6 CARDS ARE ONLY AN EXAMPLE
*  AND SHOULD BE REMOVED OR CHANGED  BEFORE RUNNING THE JOB
// DLBL CTLMRO1,'user.mro1.dfhxctl',,VSAM,CAT=CICSUCT      1
// DLBL MSGMRO1,'user.mro1.dfhxmsg',,VSAM,CAT=CICSUCT      1
// DLBL CTLMRO2,'user.mro2.dfhxctl',,VSAM,CAT=CICSUCT      1
// DLBL MSGMRO2,'user.mro2.dfhxmsg',,VSAM,CAT=CICSUCT      1
// DLBL CTLMRO3,'user.mro3.dfhxctl',,VSAM,CAT=CICSUCT      1
// DLBL MSGMRO3,'user.mro3.dfhxmsg',,VSAM,CAT=CICSUCT      1
// DLBL DFHOSD,'user.dfhosd',,VSAM,CAT=CICSUCT             2
*  WARNING - THE OPTION SYSDUMP AND LIBDEF DUMP CARDS SHOULD BE
*  INCLUDED IF SNAP DUMPS ARE TO GO TO THE DUMP LIBRARY - IF NOT,
*  THEN THE SNAP DUMPS GO TO SYSLST.
// OPTION SYSDUMP                                           3
// LIBDEF DUMP,CATALOG=(dump.library.name)                  3
// LIBDEF PHASE,SEARCH=(library.name),TEMP
// EXEC DFHWOSA,SIZE=AUTO,PARM='IOPN=DFH$AXRO,CYTIM=nnnnn' 4
/*
/&
* $$ EOJ
```

Figure 10. Sample job to run the CICS overseer

Notes for Figure 10:

1 The file names used on the DLBL statements for the CAVM control and message data sets must be those defined in the DFHOSD data set for each generic applid. See the descriptions of the DFHOSD record fields shown in Table 7 on page 53.

Attention!

The control and message data sets identified by the DLBL statements in this job must all have been created before you attempt to run the overseer job, but not necessarily initialized by their respective CICS systems.

Note that, if the control and message data sets have not yet been initialized when you start the overseer, **you cannot receive any information about the CICS**

systems to which they refer. You can, however, retry the open for these data sets using the OPEN overseer command, when the data sets are initialized.

2 The DFHOSD data set provides the control information for the CICS sample overseer program, DFH$AXRO. If you choose the same method of providing control information for your own overseer, you must ensure that the data set is initialized before use. The sample job in Figure 9 on page 54 shows you how to create and initialize the DFHOSD data set for use by DFH$AXRO.

3 The OPTION SYSDUMP statement and the LIBDEF statement for the DUMP library are required if you want any SNAP requests to send the dumps to the dump library. An SDUMP is performed by the overseer for the requested partition or the SVA, and the dump is created in the SYSDUMP sublibrary.

If these statements are not included, the dump is output on SYSLIST. This is much slower than dumping to a dump sublibrary, and stops the overseer monitoring the XRF systems until the dump has completed. If you decide to use this option, it is advisable to include the POWER JECL statement

```
* $$ LST,RBS=n
```

to split the printout and not use all of the space in your POWER queue. Put the POWER JECL statement after the $$ JOB statement (line 1 in Figure 10 on page 55).

4 The full syntax of the PARM option on the EXEC statement for module DFHWOSA is as follows:

```
PARM='IOPN=iopgmname[,CYTIM={60|nnnnn}]'
```

The parameters are:

IOPN=iopgmname
: The installation's overseer program name. For the CICS-supplied sample program, it is DFH$AXRO, as shown in the sample job stream. If you write your own overseer program, code its name here.

CYTIM=[60|nnnnn]
: The time interval, or "cycle time" (CYTIM), allowed for a CAVM data set read to complete. The time that you specify is allowed for each attempt, and it must be in the range of 20 through 32 767 seconds. The default value is 60 seconds.

A runtime example

During its execution, the overseer program keeps the operator informed of its operations and of any errors by issuing messages to the VSE console. The first line of every message from the overseer begins with the characters `OVERSEER:`.

A list of these messages, with an explanation of each, is provided in the prolog of the source code listing of the sample overseer program. Figure 11 on page 57 is a console listing of the overseer runtime example.

```
 BG 000 // JOB OVERSEER  VSE OVERSEER JOB
 DATE 03/26/1998,CLOCK 15/31/39
 BG 000 OVERSEER: ENTERING INITIALIZATION
 BG 000 OVERSEER: DFHOSD DATASET OPENED
 BG 000 OVERSEER: DFHOSD DATASET CLOSED
 BG 000 OVERSEER: INITIALIZATION COMPLETE
 BG 000 ENTER COMMAND OR 'E' TO EXIT
 BG 000
 0 r
 BG 000 R
 BG 000 OVERSEER: RESTART IN PLACE IS NOW DISABLED
 BG 000 ENTER COMMAND OR 'E' TO EXIT
 BG 000
 0 r
 BG 000 R
 BG 000 OVERSEER: RESTART IN PLACE IS NOW ENABLED
 BG 000 ENTER COMMAND OR 'E' TO EXIT
 BG 000
 0 d
 BG 000 D
 BG 000 OVERSEER: CPU   1  DISPLAY
 GEN-APP  ACT-JOB  ACT-APP  APWR A-ST  BKP-JOB  BKP-APP  BPWR B-ST  TKV
 DBDCCICS DBCICSJ1 DBDCCIC1 PWR1 ACT X DBCICSJ2 DBDCCIC2 PWR2 SOFN  COM
 BG 000 ENTER COMMAND OR 'E' TO EXIT
 BG 000
 0 s,0
 BG 000 S,0
 BG 000 OVERSEER: SNAP DUMP REQUESTED FOR PARTITION BG.
 BG 000 OVERSEER: SNAP DUMP TAKEN
 BG 000 ENTER COMMAND OR 'E' TO EXIT
 BG 000

Job DBCICSJ1 fails and no alternate is available:

 BG 000  OVERSEER: JOB IS BEING RESTARTED
 R RDR,DBCICSJ1,12
  .
 0 e
 BG 000 E
 BG 000 OVERSEER: TERMINATION COMPLETE
 BG 000 EOJ OVRSEER
 DATE 03/26/1998,CLOCK 15/49/46,DURATION 00/18/06
```

Figure 11. The overseer runtime example

Chapter 5. Operating CICS from a console device

You can operate CICS from a **console device**[2]. In particular, you can use the console device for CICS master terminal (CEMT) functions, to control CICS terminals or to control several CICS regions in conjunction with multiregion operation. Normal operating-system use of the console device is not inhibited, and CICS supports multiple console devices where present.

This chapter describes how to use a console device to enter CICS commands.

Entering commands from a console device

You can invoke CICS transactions from a console device by using the VSE MSG command.

To enter a CICS command from a console device, enter:

```
MSG Fn, DATA=datastring
```

datastring is a string of data, starting with a CICS transaction identifier.

For example, to start transaction CEMT in partition F2 from the console, type:

```
Msg F2, DATA=CEMT PERFORM TAKEOVER
```

You can type more than one MSG command at a console; each is processed in order of entry.

For information about defining console devices to CICS, see the *CICS System Definition Guide*.

Using the CEDA transaction

The CEDA transaction can be used from a console device only to INSTALL resource definitions. The sample programs cannot be executed from a console device.

Using the CEBT transaction

To communicate with an alternate CICS system in an XRF environment, you must use the CEBT command, and issue it from a console device.

[2] A console device can be one of:

- The VSE system console
- A console managed by the Interactive Interface
- A console managed by the Distributed Workstation feature
- A console managed by the VM/VSE feature.

Chapter 6. How to shut down CICS

This chapter describes how to shutdown CICS. It outlines:

- Normal CICS shutdown, in “Shutting down CICS normally”
- Immediate CICS shutdown, in “Shutting down CICS immediately” on page 62
- Shutdown of XRF CICS regions, in “Shutting down XRF CICS regions” on page 62

and what output you should see at the console in each case.

For an overview of CICS shutdown, see “Shutting down CICS” on page 12.

Shutting down CICS using the CEMT PERFORM SHUTDOWN command

To shutdown CICS, you can issue the CEMT PERFORM SHUTDOWN command with appropriate options, depending on the type of shutdown that you want. You can specify any of the following shutdown options on the command, without affecting the type of shutdown performed:

Option **Effect**

DUMP CICS produces a dynamic storage dump after shutdown is complete.

PLT(xx) CICS runs programs in the PLT, DFHPLTxx, during shutdown.

XLT(xx) Only those transactions listed in the XLT, DFHXLTxx, can be started after the SHUTDOWN command, and before shutdown is complete.

You can use the CEMT PERFORM SHUTDOWN command at the master terminal, or at the system console.

For guidance on using the CEMT transaction, see the *CICS-Supplied Transactions* manual.

Shutting down CICS normally

Use the CEMT PERFORM SHUTDOWN command to shut down CICS normally.

A normal shutdown of CICS cannot complete until all pipes (sessions) in use for the external call interface (EXCI) have been closed.

What you should see

When you use the CEMT PERFORM SHUTDOWN command, CICS responds directly by issuing the following messages at the console:

```
DFHTM1715I applid CICS is being quiesced by userid
username in transaction CEMT at netname netname.

DFHDM0102I applid CICS is quiescing.
```

Message DFHTM1715 is also issued to the master terminal to inform the operator that CICS is terminating.

If the normal shutdown is successful, CICS issues message DFHKE1799 at the console:

```
DFHKE1799 applid TERMINATION OF CICS IS COMPLETE.
```

For example, the following sequence of messages was issued on a normal shutdown of the CICS TOR, CICSHTH1. In these messages, 'F2' is the partition identifier and '0065' is the task number:

```
MSG F2,DATA=CEMT PERFORM SHUTDOWN
F2 0065 +DFHTM1715 CICSITH1 CICS is being quiesced by userid
                     CICSUSER in transaction CEMT at netname IG2S2CA8.
F2 0065 +DFHDM0102I CICSHTH1 CICS is quiescing.
F2 0065 +DFHTM1782I CICSHTH1 All non-system tasks have been successfully
                     terminated.
F2 0065 +DFHZC2305I CICSHTH1 Termination of VTAM sessions beginning
F2 0065 +DFHZC2316 CICSHTH1 VTAM ACB is closed
F2 0065 +DFHJC4553 07/15/98 15:58:48 CICSHTH1 Archive job submitted for 01 A.
F2 0065 +DFHJC4510I CICSHTH1 All open journals now closed
F2 0065 +DFHWK0105I CICSHTH1 Warm keypoint successful.
F2 0002 +DFHDU0303I CICSHTH1 Transaction Dump Data set DFHDMPA closed.
F2 0002 +DFHKE1799 CICSHTH1  TERMINATION OF CICS IS COMPLETE.
```

Shutting down CICS immediately

You shut down CICS immediately by using the CEMT PERFORM SHUTDOWN IMMEDIATE command.

What you should see

When you use the CEMT PERFORM SHUTDOWN IMMEDIATE command, CICS responds directly by issuing message DFHTM1703 at the console. Message DFHTM1703 is also issued to the master terminal to inform the operator that CICS is terminating.

If CICS shutdown is successful, CICS issues message DFHKE1799 at the console:

```
DFHKE1799 applid TERMINATION OF CICS IS COMPLETE
```

For example, the following sequence of messages was issued on an immediate shutdown of the CICS TOR, CICSHTH1, that was executing in partition F2:

```
MSG F2,DATA=CEMT PERFORM SHUTDOWN IMMEDIATE
F2 0065 +DFHTM1703 CICSITH1 CICS Is being terminated by userid CICSUSER
                     in transaction CEMT at terminal SAMA
F2 0065 +DFHJC4553 07/15/98 15:58:48 CICSHTH1 Archive job submitted for 01 A.
F2 0065 +DFHJC4510I CICSHTH1 All open journals now closed
F2 0002 +DFHDU0303I CICSHTH1 Transaction Dump Data set DFHDMPA closed .
F2 0002 +DFHKE1799 CICSHTH1  TERMINATION OF CICS IS COMPLETE.
```

Shutting down XRF CICS regions

You can use the following commands at an XRF active region to shut it down:

- **CEMT PERFORM SHUTDOWN TAKEOVER**
  This shuts down the active region normally, but causes it to sign off abnormally from the CAVM, and causes the XRF alternate CICS region to takeover control.
- **CEMT PERFORM SHUTDOWN IMMEDIATE**
  This shuts down the active region immediately and causes the XRF alternate CICS region to takeover control.
- **CEMT PERFORM SHUTDOWN**

This shuts down both the active and alternate CICS regions.

You can use the following commands at an XRF alternate region to shut it down:

- **CEBT PERFORM SHUTDOWN**
  This shuts down the alternate region normally.
- **CEBT PERFORM SHUTDOWN IMMEDIATE**
  This shuts down the alternate region normally, but causes it to sign off abnormally from the CAVM.

For more information about shutting down XRF CICS regions, see Chapter 4, “Operating CICS in an XRF environment” on page 27.

Part 2. The CICS utility programs

This part of the book describes the CICS utility programs, and also provides some sample job streams that you can use to run them.

Table 8. CICS utility programs road map

If you want to know more about	Refer to...
DFHJUP, the journal select, print, and copy utility	Chapter 7, “Journal utility programs” on page 67.
DFHJACDU, the automatic journal archiving utility for checking and updating the archiving control data set	Chapter 7, “Journal utility programs” on page 67.
DFHDMF, the Data Management Facility for collecting monitoring and statistics data	Chapter 8, “Data Management Facility (DMF) Data Handler” on page 85.
DFHSTUP, the statistics formatting utility	Chapter 9, “Statistics utility program, DFHSTUP” on page 101.
DFHMNDUP, the monitoring dictionary record creation utility	Chapter 10, “Monitoring utility programs” on page 111.
DFH$MOLS, the sample monitoring formatting utility	Chapter 10, “Monitoring utility programs” on page 111.
DFHTU410, the trace formatting utility	Chapter 11, “Trace utility print programs” on page 129.
DFHDU410, the transaction dump formatting utility	Chapter 12, “Dump utility programs” on page 137.
DFHPD410, the SDUMP formatting program for use with Info/Analysis	Chapter 12, “Dump utility programs” on page 137.
DFHCSDUP, the system definition data set utility	Chapter 13, “System definition file utility program, DFHCSDUP” on page 151.
DFHMSCAN, the macro-level programs identification utility	Chapter 14, “Identify macro-level programs utility program, DFHMSCAN” on page 183.
DFHM$INDB, the sample resource manager in-doubt resolution utility	Chapter 15, “Resource manager in-doubt resolution program, DFH$INDB” on page 185.
DFH$IWUP, the sample in-doubt window resolution utility	Chapter 16, “In-doubt window resolution utility, DFH$IWUP” on page 189.
DFH$STED, the sample stagger end-of-day time utility	Chapter 17, “Stagger end-of-day time sample utility program, DFH$STED” on page 199.
DFHSMUTL, the local catalog storage manager domain subpool record manipulation utility	Chapter 18, “Local catalog storage program, DFHSMUTL” on page 201.

Chapter 7. Journal utility programs

This chapter describes two journal utility programs:

DFHJUP The journal utility program, for reading and processing data in CICS journal data sets.

DFHJACDU The journal archive control data set utility program, for checking and updating the journal archive control data set during automatic journal archiving operations. (For details of DFHJACDU, see “The automatic journal archiving utility” on page 80.)

CICS also provides two journal utility programs for formatting tape and disk journals before they are used and another utility program for writing an end-of-file mark on tape journals:

DFHJCJFP used to preformat each disk data set for CICS journals.

DFHFTAP used to preformat tapes for use as CICS journals.

DFHTEOF used to reposition and correctly close tape journals after an uncontrolled shutdown.

For information about these utility programs, see the *CICS System Definition Guide*.

The DFHJUP utility program

You can use the journal utility program (DFHJUP) to examine data from the CICS journal data sets. The utility can:

- Print or copy an entire journal data set.
- Print or copy selected journal records, from either a single or from multiple journal data sets, as specified by control statement input.
- Select and print journal records on the basis of their sequential position in the data set.
- Select and print journal records as determined by data contained within the records themselves, such as the contents of time, date, or identification fields.
- Allow EXIT routines to process any selected journal records.

These features are selected and controlled by a series of statements that allow you to define the input and output options, selection ranges, and various field and record selection criteria.

Input and output

DFHJUP uses SAM to process journal data that can reside on either tape or direct access storage devices. The record format must be variable in length, blocked or unblocked.

If a fixed length file is input to DFHJUP, the results are unpredictable, and it may either format the data incorrectly or report a permanent I/O error.

The copy operation without record selection is designed to produce one output block of data for each input block, that is, reblocking is not performed so the copied data set is therefore suitable for input to a forward recovery program.

When records are selected, the copy operation reblocks the data on the output data set.

The control information must be as 80-byte records in the SYSIPT data stream. These control statements are reproduced on the output print data set in the same format and sequence as they are processed. If DFHJUP finds any error conditions, error messages are produced following the statement to which they apply.

You can format and print output data on the SYSLST data set, or copy it to a specified data set unchanged, or both.

Data to be printed is formatted into 32-byte segments and displayed in both hexadecimal and EBCDIC forms, with the hexadecimal relative offset value preceding each segment.

The flow of control for the program passes through two stages:

1. **Control statement processing**, which constructs rules for testing and selecting records, and diagnoses control statement errors.
2. **Record selection and output processing**, where the input data is read, analyzed, and compared with the selection criteria to determine the applicability of the record for output.

During the first stage, the journal utility reads and examines the parameter statements, and constructs the required test or test series to create a test group. When control passes to the next stage of the program, this test group is then used to select records. In the second stage, the input data records are read, and any action is decided by the results of each test in the group. When the end of the input data is reached, either by an end-of-file condition, or by the indicated record count being satisfied, program control shifts back to the first stage, where the next group of tests is constructed.

JCL requirements

The journal utility program runs as a standard operating system job. You must define a JOB statement, an EXEC statement, and DLBL statements defining input and output. “Examples of using DFHJUP” on page 75 gives some sample jobs that illustrate the use of DFHJUP.

SYSLST
This statement defines the output data set that contains the formatted print records and control messages. It is specified with the POWER/VSE JCL statement *** $$ LST** card.

SYSIPT datastream
This file must be in 80-byte record format, and must be entered following the EXEC DFHJUP statement.

Input or data DLBL or TLBL
This statement defines the input data sets to be examined to produce the output data.

These data sets must be standard labeled files, either DASD or tape. They must be sequential data sets.

The default filename is DFHJRNL on logical unit SYS004.

An example of a DLBL statement is as follows:

```
// DLBL       DFHJRNL,'CICSLOG'
```

An example of a TLBL statement is as follows:

```
// TLBL       DFHJRNL,'CICSLOG'
```

Output or data DLBL or TLBL

This statement defines the optional output data sets to contain the selected records.

The default filename used is DFHJOUT on logical unit SYS005.

Utility control statements

You can use the control statements, CONTROL, OPTION, and END, to guide DFHJUP through the stages described on page 68.

Use the END statement as a delimiter to separate one group of tests (comprising one or more OPTION statements) from subsequent groups of tests on the next data set. When an END statement is encountered in the control input stream, the construction of record selection parameters ceases and the processing of input data records starts. Proper use of the END statement allows one execution of the utility program to perform a varied number of tests on one or more CICS journal data sets.

You can use the statement, * or COMMENTS, to provide titles or comments on the output listings. Use it to include any information you think is helpful to identify tests or data. It has no effect on the utility program.

Each full keyword has a corresponding abbreviated form that you may use.

You can continue keyword operands of the DFHJUP statements on the next record, up to a maximum of 9 records, provided you code a nonblank character in position 72, and continue the operands in column 16 of the next statement. If a statement is not a continuation record of the preceding statement, the character in column 72 of that preceding statement must be a blank.

CONTROL statement

The CONTROL statement (see Table 9 on page 70) is optional, and you can omit it if the default operand values are satisfactory. It defines the filenames to be used for the input and output data sets and the beginning and ending limits of the data set to be scanned. If you don’t specify this statement, DFHJUP defaults to reading the input file named in a DFHJRNL DLBL statement. The optional output data set defined on the DFHJOUT DLBL statement is opened only if you specify the OPTION COPY function in the current group of tests, and also code the COND=E parameter.

Table 9. The DFHJUP CONTROL statement

```
1          10        16
CONTROL    CNTL      [{SKIP|K}={0|number}]
                     [,{STOPAFT|H}={16777215|number|EOF|(number,E)}]
                     [,{DDNAME|D}={DFHJRNL|filename}]
                     [,{DDNOUT|O}={DFHJOUT|filename}]
                     [,{BLKSIZE|B}={27998|number}]
```

SKIP= or K= defines the first record tested. All prior records are ignored. If this keyword is not specified, a default value of zero is used and causes the first record on the input file to be tested.

- **number** must be specified in the range 0 through 99999999, and cannot have imbedded commas.

STOPAFT= or H= defines the last record to be tested. When this value has been reached by counting processed records, the current group of tests is terminated.

- **number** must be specified in the range 0 through 99999999, with no imbedded commas.

  If you do not specify this keyword, the default value of 16777215 is assumed. If you specify a value of zero, one record is processed.

- **EOF** denotes end-of-file condition; allows record processing beyond the stated maximum of 99999999 records.

- **E** causes records to be counted for test sequence termination only if they satisfy selection criteria. Otherwise, all records read (after the SKIP value) are counted.

DDNAME= or D= identifies the DLBL filename for the input data set for the current group of tests.

The default ddname of DFHJRNL is used if you do not code this keyword, and a DFHJRNL DLBL statement or a DFHJRNL TLBL statement must be included in your job stream. If you code this parameter to specify a different ddname, your job stream must include the corresponding DLBL statement or TLBL statement.

DDNOUT= or O= identifies the DLBL filename for the optional output data set for the current group of tests.

This keyword is used in conjunction with the OPTION COPY function, and you need only code this parameter if you want to use a ddname other than the default of DFHJOUT. Coding DDNOUT, or the presence of DFHJOUT in the DFHJUP job stream, does not cause this data set to be used. An output data set is used only if OPTION COPY is specified with COND=E.

BLKSIZE= or B= specifies the maximum block size of the DFHJOUT file on a COPY operation.

The default is 32761. If specified, it must be between 1 and 5 digits long and may not exceed 32761.

Specifying a BLKSIZE that is smaller than the longest logical record causes VSE to truncate data records without warning.

For efficiency on CKD dasd use:

3380 - 23476
3390 - 27998
9345 - 22982

OPTION statement

The OPTION statement (see Table 10) defines the test or series of tests to be performed upon the data of the candidate record to determine whether it is selected. Each OPTION statement constructs one set of tests. You can specify one or more OPTION statements, in any combination, to define more closely the selection criteria and output processing to be performed against each input record. If you omit all keyword operands (except for EXITR and DDNAME), all records processed by stage 2 of DFHJUP are either written to the SYSLST data set, or copied to the specified output data set.

You can execute one or more tests on each logical record by coding the appropriate number of OPTION statements, creating the logical OR function. You can analyze records with the logical AND function by using the multifield test capability of the COND operand and the appropriate OPTION statements, creating a test series. Use the operands COND=M and COND=E to denote the beginning and ending, respectively, of a series for multifield testing of a record.

Each OPTION statement has its own output processing defaults. If you use multiple OPTION statements to create a multifield test series, final output processing is determined by the OPTION statement and its associated keywords that are defined along with the COND=E keyword.

Table 10. The DFHJUP OPTION statement

```
1          10       16
OPTION     {PRINT|  [{OFFSET|O}={1|number}]
             COPY|
             NEGOF} [,{FLDTYP|T}={X|C}]
                    [,{VALUE|V}=string]
                    [,{FLDLEN|L}={1|number}]
                    [,{COND|C}={E|M|T{Y|N}|ET{Y|N}|MT{Y|N}}]
                    [,{EXITR|E}=name]
                    [,{DDNAME|D}={TRCPUNCH|ddname}]
                    [,{PRTSYS|P}={N|Y}]
```

Options

Each option has two distinct functions:

1. Determine the starting position for the OFFSET keyword
2. Determine the output processing to be performed.

If individual options are combined to form a multifield test, the use of OFFSET remains unchanged; however, output processing is determined by the option coded with the COND=E keyword.

PRINT
causes all selected records to be displayed on the SYSLST data set.

COPY
causes all selected records to be transferred to the specified output data set. You can also write these records on the SYSLST data set by coding the PRTSYS keyword.

NEGOF
causes the OFFSET keyword value to be used as a negative offset from the end of the journal record. All records selected using this function are displayed on the SYSLST data set.

All of the following OPTION control statement keywords are optional:

OFFSET= or O=
defines the location in the record of the first byte of the field to be tested. The default is position 1 of the record.

number
can be in the range from 1 up to and including the length of the record under test. Maximum value is 32767 bytes, and no checking is performed to determine if the logical record length is exceeded.

Note: If DSECTs are used to locate values in control records or blocks, you must adjust the starting value for the OFFSET parameters. Most DSECTs start with a relative value of zero, while the value specified in the OFFSET keyword is always expressed as relative to byte 1. The first four bytes are the SAM Record Descriptor word. This contains the length of the record in the first two bytes followed by two bytes of zeros. The first data byte is then at OFFSET=5.

FLDTYP= or T=
defines the type of data in the VALUE=field.

X
data to be treated as hexadecimal pairs. The test data is packed (2 bytes into 1 to form hexadecimal equivalents). This is the default value.

Example: If VALUE=D9D6D6E3E2C5C7 (14 bytes) is specified with the FLDTYP=X parameter, the resultant VALUE= looks like this: ROOTSEG in EBCDIC characters or D9D6D6E3E2C5C7 in hexadecimal; in either case, the length is only 7 bytes.

C
data to be treated as EBCDIC characters. DFHJUP uses the data as coded in the OPTION statement, without alterations.

VALUE= or V=

defines those characters that comprise the test field. If you specify FLDTYPE=X, you must enter this data as hexadecimal character pairs. For a ‘test under mask’ condition, a single pair must represent the hexadecimal value for the test. If you specify FLDTYP=C, you must enter the value data as EBCDIC characters. However, if a blank or comma character is to be included in the value, you must specify FLDTYP=X, and code the value operand as hexadecimal characters, using `X'40'` for the blank and `X'6B'` for the comma, as appropriate.

string

cannot exceed 255 EBCDIC or 510 hexadecimal characters. The length of this field is set by the value of the FLDLEN= keyword and not by the number of non-null characters in this field.

FLDLEN= or L=

defines the number of characters to be used from the test field.

number

represents the actual number of bytes to be used, not the number of characters specified in the VALUE= keyword. The acceptable range of values for this field is from 1 up to and including 255. The default is 1.

COND= or C=

defines the type of test and its relationship to other tests in the group. If this keyword is not specified, the default is COND=E.

E

marks the last (or only) element in a test series. Any OPTION control statements appearing after this form a new series of tests. Coding an E to terminate a test series allows DFHJUP to perform various tests on each record, and each test series can be used on different fields within the record. Final output processing is determined by the OPTION function defined with this keyword value.

M

indicates that this is a multifield test. That is, more than one test is to be made on each input record. All tests in this series must be satisfied for record selection and output processing to begin.

T

causes the VALUE= byte to be used as a test-under-mask value, instead of as a compare field. Only the first byte (two hexadecimal characters if FLDTYP=X) of the VALUE= field is used. If FLDTYP=C is used, the hexadecimal equivalent of the EBCDIC character is the test value. If you code COND=T, you must not specify the FLDLEN= keyword and DFHJUP assumes a default length of 1.

Y

indicates that, for the test under mask to be considered satisfied, there must be a bit in the record test field for each corresponding bit of the test byte. This is equivalent to a branch-if-ones test.

N

indicates that, for the test under mask to be considered satisfied, there must not be a bit in the record test field for any of the corresponding bits of the test byte. This is equivalent to a branch-if-zeros test.

MT
defines a test-under-mask option as described above for T, but with the properties of a multifield test as described for M. Because the T parameter causes FLDLEN to default to 1, the MT parameter must be used for a multifield test that starts with a test-under-mask value.

ET
signifies that a multifield test series ends with a test-under-mask condition.

EXITR= or E=
specifies the entry point name of an exit routine that is to be given control when a candidate record has satisfied all selection criteria for the current test.

If multiple test groups have specified the same exit routine, DFHJUP loads a new copy of the routine into storage when it processes the next group, hence changes to the exit code storage are lost. Upon reaching end of file on input, a final call is made to the exit routine. You can determine if end of file was reached by checking for zeros in the parameter field.

The exit may have any valid AMODE and RMODE combination.

The interface to the exit routine is as follows:

ENTRY:

REGISTERS

R1 contains a pointer to a parameter list.
R13 points to an empty save area.
R14 contains a return address.
R15 contains the exit routine entry address.

PARMLIST

The parameter list consists of two words. The first is a pointer to the candidate record; the second (with the high order bit on) is a pointer to the SYSLST data set DTF.

EXIT:

Upon return from the exit routine, the contents of register 15 determine whether or not processing is to continue on this record.

A zero value indicates that this record is required, and output processing is now determined by the last OPTION statement encountered containing the COND=E keyword.

A return code of 8 or higher causes further processing of the OPTION to be terminated. A return code in the range 1–7 causes the record to be ignored.

DDNAME= or D=
defines the output data set used by the DL/I call trace journal record retrieval routine for whenever it has been specified as the user exit routine. A corresponding DLBL statement must be supplied.

PRTSYS= or P=
determines whether to print all the selected records on the SYSLST data set.

N
indicates that no printing of selected records is to be done.

Y

indicates that all records transferred to the output data set are also formatted and printed.

This keyword can be used only with the OPTION COPY function. N is the default.

END statement

When you have defined all tests for the current input file, use the END statement (see Table 11) to initiate the tests.

Positions 10 and upward can be used for comments.

Table 11. The DFHJUP END statement

1	10	16
END	[.....comments....]	

COMMENTS statement

The COMMENTS statement (see Table 12) is optional. If used, it causes the contents to be displayed on the SYSLST data set.

Table 12. The COMMENTS statement

1	10	16
*		

DFHJUP return codes

The following errors can occur in DFHJUP:

Return code ***Error description***

04 This return code can be issued for any of the following reasons:

- A syntax error is detected in a SYSIN record
- A syntax error is detected in an OPTION statement
- A SYSIN statement type is unknown
- An OPEN failure has occurred on DFHJRNL or DFHJOUT.
- An OPEN error has occurred on a file.

08 An I/O error has occurred on either DFHJRNL or DFHJOUT.

Examples of using DFHJUP

The following examples illustrate some of the ways in which DFHJUP can be used. Each makes reference to a CICS journal data set. However, this utility can be used with any data set that can be processed using SAM.

For clarity, all option keywords have been specified in their full form, and many are coded where the default could be taken. Use of the short form and keyword defaults will greatly reduce the required input. In each of the two main examples, the COMMENT statement has been used to describe the function being performed.

Example 1

Figure 12 shows the JCL and control statements required to print all the records from a CICS journal data set, which is a DASD file, to the output data set.

```
* $$ JOB JNM=JNLPRNT1,CLASS=A
* $$ LST CLASS=A
// JOB JNLPRNT1
// DLBL DFHJRNL,'CICSLOG'
// EXTENT SYSnnn,volname,1,0,nnnn,nnn
// ASSGN SYSnnn,DISK,VOL=volname,SHR
// EXEC DFHJUP, SIZE=DFHJUP
*------------------------------------------------------*
*    CONTROL STATEMENT : DEFAULTS                      *
*                INPUT = DFHJRNL                       *
*               OUTPUT = SYSLST                        *
* SELECTION QUALIFIERS :                               *
*           1. DEFAULT = ALL INPUT RECORDS             *
*------------------------------------------------------*
OPTION PRINT
END
*------------------------------------------------------*
/*
/&
* $$ EOJ
```

Figure 12. DFHJUP program, example 1. JCL and control statements to print journal records to the output dataset.

Example 2

Figure 13 shows the JCL and control statements required to copy all the records from a CICS journal data set, which is a disk file, to the optional output data set.

```
* $$ JOB JNM=JNLPRNT1,CLASS=A
* $$ LST CLASS=A
// JOB JNLPRNT1
// DLBL DFHJRNL,'CICSLOG'
// EXTENT SYSnnn,volname1,1,0,nnnn,nnn
// DLBL DFHJOUT,'EXAMPLE1.COPY1'
// EXTENT SYSmmm,volname2,1,0,nnnn,nnn
// ASSGN SYSnnn,DISK,VOL=volname1,SHR
// ASSGN SYSmmm,DISK,VOL=volname2,SHR
// EXEC DFHJUP,SIZE=DFHJUP
*------------------------------------------------------*
*    CONTROL STATEMENT : DEFAULTS                      *
*                INPUT = DFHJRNL                       *
*               OUTPUT = DFHJOUT                       *
* SELECTION QUALIFIERS :                               *
*           1. DEFAULT = ALL INPUT RECORDS             *
*------------------------------------------------------*
CONTROL CNTL BLKSIZE=27998
OPTION COPY
END
*------------------------------------------------------*
/*
/&
* $$ EOJ
```

Figure 13. DFHJUP program, example 2. JCL and control statements to copy journal records to an optional output data set.

Examples of the use of the OPTION parameters

The OPTION parameters can be used to select specific types of records from a journal. Referring to Table 13 on page 79 will help you construct these statements. The following passages contain code fragments which illustrate some of the types of record selection that can be achieved in this way.

Locating records using the system-type ID field: This example shows how to locate all records in a journal which are generated by the syncpoint program and mark the start of a unit of work. The two byte hexadecimal number, used with the VALUE parameter, is found from the tables under *CICS Journaling* in the *CICS Customization Guide*. The module identifier is located first, (in this case it is X'59'), and this is then used to find the relevant section in the function identifiers table. (Our example uses X'EF'.) In the journal, the system-type ID field is made up of the function identifier followed by the module identifier. The offset to the start of this field is 5, and it is a numeric (X) type of field, of length 2 bytes. The value we are going to use is X'EF59', as shown in the following code fragment:

```
OPTION PRINT OFFSET=5,FLDTYP=X,VALUE=EF59,FLDLEN=2,COND=E
END
/*
```

If all the file control records were to be found, for example, only the second half of the system-type ID would be used. The OPTION statement would have the following form:

```
OPTION PRINT OFFSET=6,FLDTYP=X,VALUE=11,FLDLEN=1,COND=E
END
/*
```

Using the task number: The task number appears as a three byte packed decimal value in a journal record. It must appear in the same form in the VALUE parameter. To do this take the actual task number, in this case 25, and turn it into a five digit decimal value by filling up the left hand side with zeros: 00025. Then add a capital letter C to the right hand end to show its a positive value: 00025C. The following statements will cause all records belonging to task 25 to be directed to the SYSLST data set:

```
OPTION PRINT OFFSET=16,FLDTYP=X,VALUE=00025C,FLDLEN=3,COND=E
END
/*
```

Finding all records for a particular transaction: The transaction identifier appears as a 4-byte hexadecimal field in the journal records. If FLDTYP=C is used in the parameters then this value can be entered in its character form as shown below, for a transaction called TRN1.

```
OPTION PRINT OFFSET=23,FLDTYP=C,VALUE=TRN1,FLDLEN=4,COND=E
END
/*
```

Alternatively, the hexadecimal equivalent for these characters could be used, with FLDTYP=X, as shown in the next example.

```
OPTION PRINT OFFSET=23,FLDTYP=X,VALUE=E3D5D9F1,FLDLEN=4,COND=E
END
/*
```

Finding all records with a particular time stamp: Here the time must be entered in the form hhmmsss+ as a series of decimal digits and where the + sign is represented by the letter F. The utility does not support the use of the ‘greater than’ or ‘less than’ logical operators, so searching using a time stamp value is of limited use.

```
OPTION PRINT OFFSET=19,FLDTYP=X,VALUE=1446591F,FLDLEN=4,COND=E
END
/*
```

Locating all records from a particular terminal: The terminal identifier is a 4-byte value which can be entered as four characters or their hexadecimal equivalent, in the same way as a transaction identifier. In this example all the records from terminal T004 are to be selected and printed.

```
OPTION PRINT OFFSET=27,FLDTYP=C,VALUE=T004,FLDLEN=4,COND=E
END
/*
```

Selection using more than one search parameter: Suppose you wanted to print all the start of unit of work records for a particular task. This needs two OPTION statements. The example shows how to search for all records which belong to task number 25 and have a system type ID of X'EF59'. The COND=M parameter performs the AND operation on the two statements.

```
OPTION PRINT OFFSET=16,FLDTYP=X,VALUE=00025C,FLDLEN=3,COND=M
OPTION PRINT OFFSET=5,FLDTYP=X,VALUE=EF59,FLDLEN=2,COND=E
END
/*
```

If more than one type of record is to be found then the form of the following example could be used. In this case all the file control records for task 48 are selected together with all the records generated by the TRN6 transaction. The first two OPTION statements are ‘ANDed’ together, whereas the third statement is a separate search because the second statement is terminated by COND=E.

```
OPTION COPY OFFSET=6,FLDTYP=X,VALUE=11,FLDLEN=1,COND=M
OPTION COPY OFFSET=16,FLDTYP=X,VALUE=00048C,FLDLEN=3,COND=E
OPTION COPY OFFSET=23,FLDTYP=C,VALUE=TRN6,FLDLEN=4,COND=E
END
/*
```

Using a mask to test specific bits in a record: Individual bit values may be used to select specific records from a journal in much the same way as the previous examples. The OFFSET and VALUE must be known. In this example the first option statement tests for all file control records. This is combined with the second statement to test for those records which were written by emergency restart. The OFFSET is 13, to the start of the JCSPFS field, plus 2 to move to the last byte in the field. You can find the value using the programming information in the *CICS Customization Guide*. The length field must not be included. COND=ET shows it is a test under mask and the end of a multifield test series.

```
OPTION COPY OFFSET=6,FLDTYP=X,VALUE=11,FLDLEN=1,COND=M
OPTION COPY OFFSET=15,FLDTYP=X,VALUE=20,COND=ET
END
/*
```

OPTION parameter values

Table 13. OPTION parameter values

Field name	OFFSET	FLDTYP	VALUE (example)	FLDLEN	Contents
Note: System header fields					
JCRLL	1	X	0037	2	Length of record
JCRBB	3	X	0000	2	Not used
JCRSTRID	5	X	EF59	2	System type ID
	5	X	EF	1	Function identifier
	6	X	59	1	Module identifier
JCRUTRID	7	X	12EF	2	User type ID
JCRLRN	9	X	002C	2	Record number within block
Note: Main system prefix fields					
JCSPLL	11	X	0014	2	Length of system prefix
JCSPFS	13	X	000002	3	Flags
JCSPTASK	16	X	00025C	3	Task number
JCSPTIME	19	X	1445123F	4	Time of request - hhmmsss+
JCSPTRAN	23	C	TRN1	4	Transaction identifier
	23	X	E3D9D5F1	4	alternative format
JCSPTERM	27	C	T004	4	Terminal identifier
	27	X	E3F0F0F4	4	alternative format

The automatic journal archiving utility

This section describes the function of the DFHJACDU utility program, which is used in conjunction with the automatic journal archiving option.

The DFHJACDU utility program

CICS maintains a journal archive control data set (JACD) for all CICS disk journals that have JOUROPT=AUTOARCH defined in their JCT entries. You can use the JACD utility program, DFHJACDU, to read or update the JACD. There are three DFHJACDU commands, which you can use to:

- **Check** the status of a specified journal at the start of a journal archive job
- **Update** the status of a specified journal at the end of a journal archive job
- **List** the journal archive control records from the JACD.

You would normally use the first two of these commands, CHECK and UPDATE, within an automatic journal archive job that is submitted by the journal archive submission program, DFHJASP. You should write automatic archive jobs in the form of skeletal JCL, in which you code the CHECK and UPDATE commands with symbolic parameters, leaving DFHJASP to substitute the required values. For details of the symbolic parameters that are supported by the archive submission program, see “Symbolic parameter substitution” on page 81.

You can also code the CHECK and UPDATE parameters manually to submit an archive job, but it is unlikely that you would want, or need, to do so.

You do not use symbolic parameters with the LIST command, which you code in a batch job for submission in the usual way.

The DFHJACDU commands

You specify the required function by coding the DFHJACDU command in a PARM parameter on the EXEC DFHJACDU statement. The format of each command and its parameters is described in the following sections.

The DFHJACDU CHECK and UPDATE commands

When you code the CHECK command in an automatic journal archive job, DFHJACDU reads the control record for the specified journal and checks the status. This command should be coded in the first step of an archive job, to ensure that the journal data set is in the correct state to be archived. DFHJACDU sets the return code to indicate the journal status as follows:

Return code	*Description*
0	NOTREADY, and the DATE and TIME parameters agree with the JACD data.
4	NOTREADY, and the DATE and TIME parameters do not agree with the JACD data.
8	READY.
12	CURRENT.
16	JACD I/O error.

The journal archive step should only run if the return code is zero.

When you code the UPDATE command, DFHJACDU again checks the status to make sure that it is still NOTREADY, and if this is the case, changes the status to READY. In an automatic journal archive job, you must use VSE conditional JCL to ensure that job steps only execute if the return code from the previous step is zero.

The full syntax of the commands is as follows:

CHECK|UPDATE JOURNAL=jfileid,DATASET=A|B[,DATE=yyyyddd][,TIME=hhmmsst]

Code the CHECK command to test the status of the specified journal data set. You specify which journal data set by coding the JOURNAL and DATASET parameters. You can optionally add the DATE and TIME parameters as an extra check where more than one archive job may be submitted for a given journal data set.

Code the UPDATE command to change the status of the specified journal data set, after successful archiving, from NOTREADY to READY.

JOURNAL In your skeletal JCL, code the JOURNAL operand with the symbolic parameter &JJ. DFHJASP substitutes the appropriate journal identifier in the range 01 through 99, where 01 is the system log and 02 through 99 are user journals.

DATASET In your skeletal JCL, code the DATASET operand with the symbolic parameter &D. DFHJASP substitutes the appropriate data set identifier, which can be A or B.

DATE In your skeletal JCL you can, optionally, code this with &ODATE, and DFHJASP substitutes the date the journal data set was opened. This provides additional identification of the journal data set.

TIME In your skeletal JCL you can, optionally, code this with &OTIME, and DFHJASP substitutes the time the journal data set was opened. This provides additional identification of the journal data set.

The DFHJACDU LIST command

LIST [JOURNAL=(jfileid1,jfileid2,...,jfileidn)]

Code the LIST command to request a formatted listing of the specified journal control records. You identify journals by the journal identifier (JFILEID in the JCT), where 01 is the CICS system log, and 02 through 99 are user journals. If you omit the JOURNAL parameter, DFHJACDU lists the entire JACD.

Symbolic parameter substitution

For an automatic journal archiving job, symbolic parameter substitution is available on all of the parameters. The following are the symbolic parameters that you can use with journal archive job submission:

Symbolic Parameter	Represents
&SYST	A 4-byte identifier of the CICS region issuing the journal archive job, derived from the SYSIDNT system initialization parameter.
&JJ	A 2-digit journal identification (JFILEID), in the range 01 through 99.
&D	A 1-character dataset identification: A or B.
&JOURDSN	A 1- to 44-byte journal dataset name.

&ODATE A 7-byte journal dataset open date (yyyyddd).

&OTIME A 7-byte journal dataset open time (hhmmsst).

&CDATE A 7-byte journal dataset close date (yyyyddd).

&CTIME A 7-byte journal dataset close time (hhmmsst).

&JACDDSN A 1- to 44-byte JACD dataset name.

&APPLID A 1- to 8-byte CICS APPLID. Note that for an XRF system this is the generic, not the specific, APPLID.

The journal archive submission program (DFHJASP) enables the substitution of variables by inserting VSE SETPARM variables after EVERY VSE // JOB statement that is read from the JCL skeleton. For example, the JCL skeleton containing:

```
$$$$ JOB JNM=ARCHTEST,CLASS=0
// JOB ARCHTEST Archive CICSTEST Journal Dataset
$$/* Step 1 : Check journal status
// DLBL DFHJACD,'&JACDDSN',,VSAM,CAT=VSESPUC
etc
$$/&
$$$$ EOJ
```

is changed by DFHJASP on submission to look something like:

```
* $$ JOB JNM=ARCHTEST,CLASS=0
// JOB ARCHTEST Archive CICSTEST Journal Dataset
// SETPARM SYST=CICT
// SETPARM JJ=01
// SETPARM D=A
// SETPARM JOURDSN='CICSTEST.JOURNAL.A'
// SETPARM ODATE=1998001
// SETPARM OTIME=0600013
// SETPARM CDATE=1998001
// SETPARM CTIME=1825142
// SETPARM JACDDSN='CICSTEST.DFHJACD'
// SETPARM APPLID=CICSTEST
/* Step 1 : Check journal status
// DLBL DFHJACD,'&JACDDSN',,VSAM,CAT=VSESPUC
etc
/&
* $$ EOJ
```

Creating and using a skeletal JCL member

You must catalog the skeletal JCL in a sublibrary using the VSE LIBR program. The member must have a member type of DFHJASP, and the name may be any valid member name.

You must ensure that the sublibrary appears in a LIBDEF SEARCH chain (that includes source libraries) in your CICS startup job stream.

To use your new JCL for a particular journal, you must ensure that the entry for the journal has the option AUTOARCH specified in the JOUROPT parameter, and the ARCHJCL parameter specifies the member name of your skeletal JCL member.

Sample DFHJACDU job stream

The job shown in Figure 14 illustrates the use of symbolic parameter substitution in a journal archive skeletal job. There are some notes following the sample job explaining some of the job control statements.

A sample job stream is supplied in PRD1.BASE as DFH$ARCH.J. To use the sample, copy it to your own library and rename it DFH$ARCH.DFHJASP. Then edit as required.

```
$$$$ JOB JNM=ARCHTEST,CLASS=0
// JOB ARCHTEST Archive CICSTEST Journal Dataset
// ON $RC GT 0 GOTO EOJ                                              1
$$/* Step 1 : Check journal status
// DLBL DFHJACD,'&JAACDDSN',,VSAM,CAT=VSESPUC
// EXEC DFHJACDU,PARM='CHECK JOURNAL=&JJ,DATASET=&D,DATE=&DATE,*     2
                 TIME=&OTIME'
$$/*
$$/* Step 2 : Copy journal to backup disk extent
$$/*
// DLBL DFHJRNL,'&JOURDSN'
// EXTENT SYS004,SYSWK2
// ASSIGN SYS004,DISK,VOL=SYSWK2,SHR
// DLBL DFHJOUT,'&JOURDSN..BACKUP',0
// EXTENT SYS005,SYSWK3,1,0,15,150
// ASSIGN SYS005,DISK,VOL=SYSWK3,SHR
// EXEC DFHJUP,SIZE=DFHJUP
OPTION COPY
END
$$/*
$$/* Step 3 : Update journal status
// DLBL DFHJACD,'&JACDDSN',,VSAM,CAT=VSESPUC
// EXEC DFHJACDU,PARM='UPDATE JOURNAL=&JJ,DATASET=&D,DATE=&ODATE,*   3
                 TIME=&OTIME'
$$/*
/. EOJ
$$/&
$$$$ EOJ
```

Figure 14. Sample DFHJACDU job to check and update the JACD

Notes:

1 The // ON statement ensures that a return code greater than zero will cause the job to stop.

2 The PARM statement on this EXEC DFHJACDU statement is coded with the CHECK command, to test the status of the specified journal data set. DFHJASP inserts the // SETPARM statements that assign values to the variables when the job is submitted.

3 The third job step is coded to update the status of the specified journal data set from NOTREADY to READY.

Chapter 8. Data Management Facility (DMF) Data Handler

This chapter describes the Data Management Facility (DMF) Data Handler. Information can be found under the following headings:

The Data Management Facility writes data collected by the monitoring (MN) and statistics (ST) domains to data sets on disk from where they can be processed further by DFH$MOLS (to format monitoring data), and DFHSTUP (to format statistics data).

The data is transferred from CICS-managed buffers to DMF by the CICS SVC monitoring service routine, DFHMNSVC. From DMF, it is buffered again before being written to one of 36 DMF data sets.

When the currently open DMF data set becomes full, DMF switches to a new one. You have to archive and clear the full data set (using the DMF utility DFHDFOU) before DMF can reuse it.

DMF runs in its own VSE/ESA partition, collecting monitoring and statistics data passed to it from any CICS system running under the same VSE/ESA image. It stores this data in a VSE data space[3]. DMF writes records from the data space to one of 36 DMF data sets. You must define each data set to VSAM as an ESDS file of fixed allocation, allowing spanned records with a maximum length of 32 767 bytes, and define them as REUSABLE. You can choose the names for the data sets which are stored in an assembled table, DFHDMFSU, together with other default parameter values used to control DMF activities.

The naming convention for DMF data sets is the same as that for VSE/VSAM file naming conventions, that is:

- The length of the data set must be between 1 and 44 characters.
- Characters must be alphanumeric (A-Z, 0-9, @, $, or #), hyphen (-), or plus zero (+0).
- After each group of eight (or fewer) characters, a period (.) must be inserted.
- Imbedded blanks cannot be used.
- The first character of the file name and the first character following a period must be alphabetic (A-Z) or @, $, or #.

[3] The VSE data space is created by DMF using the VSE DSPERV CREATE and ALSERV ADD macros.

See the *VSE/ESA System Control Statements* manual for more detailed information.

See section “The DFHDMF initialization table and parameters” on page 87 for more information about the DFHDMFSU table.

When the currently open DMF data set becomes full, DMF closes it and opens another. It makes sure that the new data set is empty before attempting to write to it.

You have to archive and clear the full data set manually using the DMF dump utility DFHDFOU. See section “The DMF dump utility (DFHDFOU)” on page 93 for more information.

You can specify the intervals at which data is written to disk; either as a timed interval, or when the data space becomes more than x% full.

Attention!

There is no alternative data repository to DMF; if DMF is not active, monitoring and statistical data will be lost.

You can query the status of DMF, for example, you can check if it is active or not, query which data set is currently being written to, and so on. You can also suspend the recording of data by DMF, resume recording, switch data sets, and shut down DMF. See section “DMF operator commands” on page 90 for more information about the different DMF operator commands.

Figure 15 is a schematic representation of DMF.

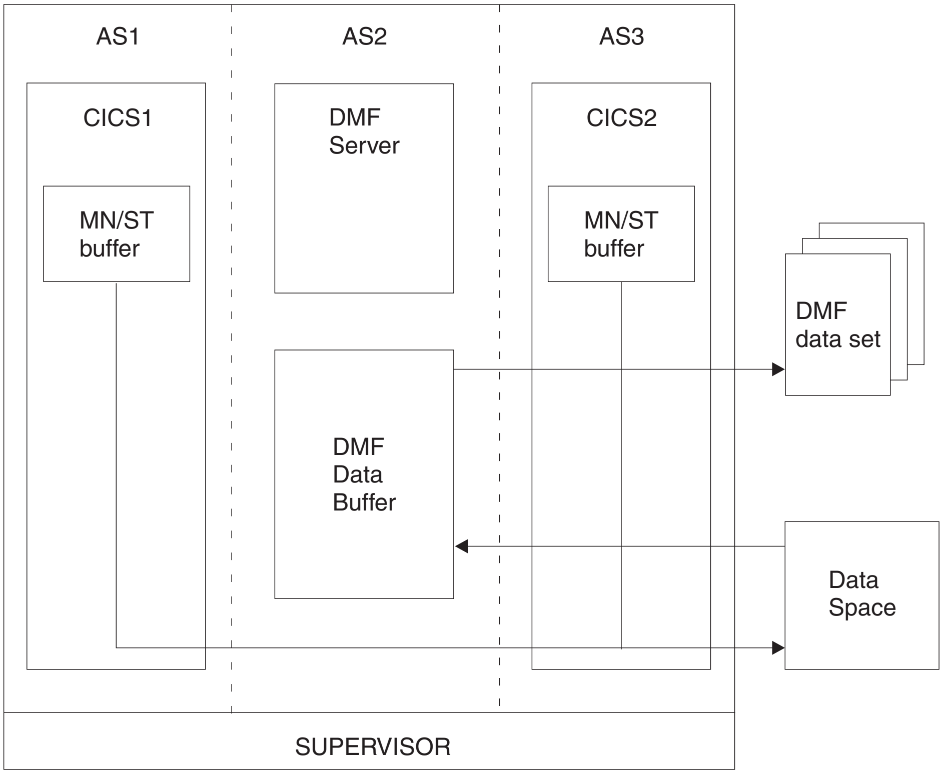


Figure 15. Schematic representation of DMF

The DMF initialization phase

The DMF initialization phase loads the programs representing each DMF function into storage. For more information on the DMF initialization phase, refer to the *CICS Diagnosis Reference* manual.

The DFHDMF initialization table and parameters

The assembled table, DFHDMFSU, contains the default settings for the DMF initialization parameters described in this section. If these defaults are not suitable for your needs, you can create a table of your own, suffixing it with a 2-character alphanumeric suffix, DFHDMF*xx*. MNOTEs are issued if the table suffix has been specified incorrectly.

The DFHDMF*xx* table is preassembled. Any errors detected at assembly time are communicated via MNOTES.

The initialization table is defined using the DFHDMFM macro.

DFHDMFM macro types
Mapping section - - DFHDMFM MAP
Table generation - - DFHDMFM TABLE

DFHDMFM macro types

The following macros map and define the DMF initialization table:

- DFHDMFM MAP produces an Assembler DSECT that describes the generated table.
- DFHDMFM TABLE generates the initialization table.

Mapping section - - DFHDMFM MAP

The DFHDMFM MAP macro produces an Assembler DSECT that describes the generated table.

name	DFHDMFM	MAP

where *name* identifies the name of the Assembler DSECT. This operand is required.

Table generation - - DFHDMFM TABLE

The DFHDMFM TABLE macro generates the initialization table.

DFHDMFM	TABLE ,SIZE=parm ,SUFFIX=XX [,CATALOG= data set name] [,FILELIST= (data set name LIST)| ,GENFILES=nn [,GENPREFIX=data set prefix]] [,INTERVAL=mmss] [,LISTDSN=YES|NO] [,SID=xxx] [,STATUS=ACTIVE|NOACTIVE] [,TRACE=YES|NO] [,TRTABSZ=nn] [,TYPE=(parms)|,NOTYPE=(parms)] [,USAGE=parm]

The parameters that can be coded are as follows. They can only be used by the DMF administrator, or the system programmer who has responsibility for initializing DMF.

CATALOG=data set name
Specifies the name of the catalog where the DMF data sets are defined. If omitted, the DMF searches for the DMF data sets in the catalog identified by the DLBL filename ISSYSUC.

FILELIST=(data set name list)
Specifies the names of the DMF data sets to be used for this run of DMF. A maximum of 36 data sets may be defined. The default is (CICS410.SYS1.MANY,CICS410.SYS1.MANZ)

GENFILES=*nn*
Specify this parameter as an alternative to FILELIST. This parameter causes ***nn*** entries to be generated for DMF to use. The data sets so generated have A-Z, 0-9 appended to the default prefix CICS410.SYS1.MAN. *nn* may be in the range 1-36.

GENPREFIX=data set prefix

Specify this parameter with GENFILES to override the default prefix of CICS410.SYS1.MAN with data set prefix. The maximum length for data set prefix is 43 characters.

INTERVAL=*mmss*

Specifies the intervals, in minutes and seconds and in the range 0001 through 5959, at which DMF is to receive a wake up call to write information to the currently open DMF data set. The default is 30 (3000) minutes.

You can modify the value of the INTERVAL parameter from the console while DMF is running.

LISTDSN=YES|LISTDSN=NO

Specifies whether you want DMF to generate data set status messages at startup time.

LISTDSN=YES

Generates the following status information:

- Name of the data set
- Status of the data set—active, alternate, or dump required
- Size of the data set, expressed as a number of 4096-byte blocks
- The amount of space in the data set that has been used up, expressed as a percentage (%)

LISTDSN=NO

You do not want data set status information generated.

NOTYPE=(parms)

All record types are to be collected except for the types specified on the NOTYPE option. *parms* takes the same form as for *TYPE*.

SID=*xxxx*

Specifies the system identifier to be used in all DMF records. *xxxx* can be any 4 alphanumeric and (or) special characters. The default is SID(3090™).

SIZE=*parm*

Defines the size of the DMF data space that DMF creates during the initialization phase in Mbytes in the range 4–2048. This parameter is mandatory.

STATUS={ACTIVE|NOACTIVE}

A specification of ACTIVE causes DMF to start recording data as soon as DMF is initialized. A specification of INACTIVE causes DMF to wait for an operator command (SETDMF ACTIVE) before recording data.

SUFFIX=xx

Code the SUFFIX parameter with the 2-character suffix representing the version of the DFHDMF table to be generated. The default is SUFFIX=SU, and creates the DMF default table, DFHDMFSU. SUFFIX=SU should not be used because it is used by IBM®.

TRTABSZ=nn

Specifies in Kilobytes the size of GETVIS storage that DMF accepts to hold its internal trace table. *nn* may be in the range 1024 to 8192.

TYPE=(parms)
Specifies the record types and subtypes to be recorded. These take the form [aa,bb:zz] where *aa*, *bb*, and *zz* are the decimal notations for each record type. Record types separated by *:* denote a range of record types, *bb:zz* would mean *bb* through *zz*.

The default is TYPE(0:255).

TRACE=YES|<u>TRACE=NO</u>
Specifies whether tracing is required for this run of DMF.

USAGE=parm
Specifies how full the data space must be (expressed as a percentage) before DMF starts writing records to the currently open DMF data set, regardless of the setting of the INTERVAL parameter (if set). The default is USAGE(50), meaning that the data space must be 50% full before DMF will start to write records.

Operator communications

The communications handler manages requests sent to DMF as a result of the central VSE operator issuing the VSE MSG(pn) command. See the *VSE/ESA System Control Statements* manual for more information about the MSG command.

The communications handler prompts you to enter a command, and checks it before acting upon it. If a command is specified incorrectly, a message is issued to the console asking you to correct the error and resubmit the command. This loop continues until the command and its associated value(s), if any, are correct.

DMF operator commands

The following operator commands control the activity of DMF:

? Displays general help information on DMF commands.

SETDMF Active
Causes DMF to start recording data in its data space.

SETDMF DEBUG{,ON|,OFF}
Causes DMF to start extra debug facilities (ON) or to stop them (OFF). This command should only be used under guidance from your IBM Service Representative.

SETDMF Flush
Causes DMF to write any remaining data in the data space to the currently open data set.

SETDMF Interval(*mmss*)
Causes DMF to change the wakeup interval to the value specified in *mmss*. The range may be *mm* in 0-99, *ss* in range 0-59, where *mm* represents minutes and *ss* represents seconds.

SETDMF NOActive
Causes DMF to temporarily suspend the recording of data in its data space, but without the termination of DMF.

SETDMF NOTrace
Causes DMF to stop recording trace information.

SETDMF SHutdown [,I]
Causes DMF to terminate, *I* indicates that the shutdown is immediate. During an immediate shutdown, DMF makes no attempt to write data currently in the data space to the open data set. Any such data is lost.

SETDMF Switch
Causes DMF to close the currently open data set, and open the next available data set in the file list.

SETDMF Trace
Causes DMF to start recording trace information in its internal trace table.

DISPLAY [O]
DISPLAY reflects the current state of the DMF system. Entered on its own, it causes DMF to display the name of the currently open data set. When the optional *O* is entered, it causes DMF to display its current settings, These include the Status, the name of the data space, the wakeup interval and the status of the trace.

Queue manager

For information about the DMF queue manager, refer to the *CICS Diagnosis Reference* manual.

File manager

The file manager manages all the data sets owned by DMF. For further information, refer to the *CICS Diagnosis Reference* manual.

DMF record format and record header

Each record written to a DMF data set has a standard SMF record header. The length and the field types for the record header varies depending upon the record written. For more information on the record format and header, refer to the *CICS Customization Guide*.

DMF statistics

Statistics relating to DMF activity are collected and reported on SYSLST when DMF shuts down.

```
Data Management Facility                                Page     1
Statistics Report                                  Date 11/10/1998
Write statistics
      Attempted write requests                           33
      Successful write requests               :          33
Read statistics
      Attempted read requests                 :          35
      Successful read requests                :          33
Data statistics
      Maximum data space usage (HWM)          :      195388
      Current data space usage                :           0
      Total count of bytes written (hex)      :       2FB3C
      Total count of bytes read (hex)         :       2FB3C
      Limit chosen by user                    :     2097150
```

Figure 16. Example of a DMF Statistics Report

DMF messages

DMF messages are prefixed with the component id, DF. For details of messages issued by the DMF file manager, refer to the *VSE/ESA Messages and Codes Volume 3*.

Error handler

For information about the DMF error handler, refer to the *CICS Diagnosis Reference* manual.

Abnormal termination of DMF

If DMF suffers an abnormal termination, a VSE SDUMPX is taken. DMF then terminates, but without writing the data currently stored in the dataspace to the open data set.

Termination task

There are two types of shutdown: normal and immediate.

Normal shutdown

During a normal shutdown, the termination task shuts down DMF in a controlled manner, ensuring that all the data stored in the data space is written out to the currently open DMF data set before closing all files and returning all facilities acquired to the operating system.

Immediate shutdown

During an immediate shutdown, the termination task shuts DMF down immediately. Data stored in the data space is lost because it will not be written to the currently open DMF data set. A dump is produced.

The DMF dump utility (DFHDFOU)

The offline DMF dump utility processes the data sets used by DMF, and allows the operator to copy records from the DMF data sets either to tape or to disk for subsequent processing.

The DMF dump utility:

- Initializes DMF data sets
- Allows you to copy the contents of multiple DMF data sets to sequential data sets either on tape or on disk
- Allows you to route different records to separate files
- Produces a summary activity report

DMF dump utility parameters

You can use the following parameters to tailor the way in which the DMF dump utility processes the DMF data sets. The parameters are specified in the SYSIPT datastream, and are checked by the DMF dump utility before being processed. Mistakes made when specifying the parameters cause error messages to be sent to SYSLST. When the parameters have been checked, the utility processes the records in the DMF data sets and writes them to tape or disk.

You may specify parameters on more than one input line. DFHDFOU accepts data from column 1 to column 72, and parameters may start in any column; no continuation characters are needed.

INDD(filename,OPTIONS({DUMP|CLEAR|ALL)})

The INDD parameter describes the input data set.

filename

Name of the input DMF data set.

OPTIONS

The following can be specified on the OPTIONS operand.

DUMP

Indicates that the input data set is to be read and copied without being reset. A summary activity report is written if at least one record is read or written.

CLEAR

Indicates that the input data set is to be reset and initialized. The information on the data set is **not** copied, and is therefore lost.

ALL

Indicates that both the DUMP and CLEAR options are applicable.

If the INDD parameter is not specified it defaults to

```
INDD(DUMPIN,OPTIONS(ALL))
```

Errors made coding the INDD parameter cause the DMF job to terminate with an associated error message.

If the INDD parameter is specified twice for the same filename, a message is issued to the SYSLIST data stream.

OUTDD(filename,{TYPE(list)|NOTYPE(list)})

The OUTDD parameter describes the output data set.

filename

Name of the output data set (either tape or disk).

TYPE

Indicates that the records specified in the list are to be included in the output data set.

NOTYPE

Indicates that the records specified in the list are to be excluded in the output data set.

list can be any record type, or combination of record types. The record types can be specified individually, or as a range. A range is specified by coding the lower and upper record types separated by a colon (:). For example, TYPE(2,4:7,9) indicates that record types 2,4,5,6,7 and 9 are to be included in the output data set.

If the OUTDD parameter is not specified by the user, it defaults to

```
OUTDD(DUMPOUT,TYPE(000:255))
```

in which all record types are included in the output data set.

If the OUTDD parameter is specified twice, with the TYPE option specified on the first instance, and the NOTYPE option specified on the second instance, for the same data set, the first valid specification is used.

If there is a syntax error in an OUTDD parameter, and its associated INDD parameter is coded with either ALL or CLEAR, the job is terminated. This protects the user from losing data, especially if the CLEAR option was specified on the INDD parameter.

Errors made coding the OUTDD parameter cause the DMF job to terminate with an associated error message.

DATE(yyyyddd,yyyyddd)

The DATE parameter specifies the start and end dates of the period for which records are to be written, where *yyyy* is the year, and *ddd* is the Julian date. The value of *ddd* must not exceed 366. If two dates are specified, only records recorded on or between them are written to the output data set.

If you do not specify the DATE parameter, it defaults to

```
DATE(1900001,9999366)
```

START(hhmm)

The START parameter specifies that only those records recorded after the START time, and before the END time are to be written, where *hh* is the hour and *mm* are the minutes in 24-hour clock. If you do not specify the START parameter, it defaults to

```
START(0000)
```

END(hhmm)

The END parameter specifies that only those records recorded after the time specified on the START parameter, and before the specified END time, are to be written, where *hh* is the hour and *mm* are the minutes in 24-hour clock. If you do not specify the END parameter, it defaults to

`END(2359)`

Errors made coding the END parameter result in the DMF dump utility assuming the default value.

SID(cccc)

The SID parameter specifies that only records written by the operating system with the specified system identification *cccc* are to be written to the output data set. The SID parameter can be specified for each VSE system that the DMF dump utility is expected to handle. (The DMF dump utility can process information from more than one VSE system.) If you do not code the SID parameter, records relating to **any** VSE system are written.

Format of a DMF dump header

After the DMF dump utility parameters have been processed, all the output files are opened and a DMF type 2 record is written to each output file to mark the beginning of a dump of a DMF data set. This happens before any records are processed and written to the output files.

The format of the DMF dump header is documented in the *CICS Customization Guide*.

Format of a DMF dump trailer

When all the input files have been processed, a type 3 record is written to each output file to mark the end of a dump of a DMF data set, and then the output file is closed.

The format of the DMF type 3 record dump trailer is documented in the *CICS Customization Guide*.

DMF dump utility statistics

Statistics are written to SYSLST when all the input parameters for the DMF dump utility have been processed.

Example of a DMF dump utility statistics report

Figure 17 is an example of a summary activity report.

```
SUMMARY ACTIVITY REPORT

START DATE-TIME 08/08/1998-00:00:00  END DATE-TIME 08/08/1998-23:00:00

                                      AVG.         MIN.         MAX.
RECORD       RECORDS      PERCENT     RECORD       RECORD       RECORD       RECORDS
TYPE         READ         OF TOTAL    LENGTH       LENGTH       LENGTH       WRITTEN

0            13           .01%        35.00        35           35           13
2            0                                                               1
3            0                                                               1
110          196,812      99.99%      500.00       500          500          196,812


TOTAL        196,825      100%        500.00       500          500          196,827
```

Figure 17. Example of a summary activity report

DFHEWTM macro

The DFHEWTM macro is the CICS Transaction Server for VSE/ESA Release 1 version of the SMF SMFEWTM macro. DFHEWTM is a dual-language macro consisting of a PLX version and an Assembler version.

For more information, refer to the *CICS Customization Guide*.

Getting ready to run DMF

The following steps are required to successfully start DMF.

1. Create DMF start-up table (optional if you want to use the default table
2. Define DMF data sets
3. Initialize DMF data sets
4. JCL required to start DMF

Create DMF start-up table

If you do not want to use the default files as defined in DFHDMFSU, or want to change any of the other options, you must create your own start-up table. Sample JCL to do this is:

```
// JOB ASSEMBLE DMF TABLE
// LIBDEF *,SEARCH=PRD1.BASE
// LIBDEF PHASE,CATALOG=user sublibrary
// OPTION CATAL
// EXEC ASMA90,SIZE=(ASMA90,50K)             (col 72 V)
         DFHDMFM TABLE,                              *
               SUFFIX=xx,                            *
               SIZE=nn,                              *
                  :
/*
// EXEC LNKEDT
/*
```

Define DMF data sets

Having defined the DMF table, you must define the VSAM data sets that DMF will use. The following sample shows typical VSAM commands to define the default data sets.

```
// JOB DEFINE DMF DATA SETS
// EXEC IDCAMS,SIZE=AUTO
   DEFINE CLUSTER (NAME(CICS410.SYS1.MANY) -
          NONINDEXED                       -
          VOLUME(vvvvvv)                   -
          CYLINDERS(10)                    -
          REUSE                            -
          RECORDSIZE(125 32767)            -
          SPANNED                          -
          NOCOMPRESSED                     -
          SHAREOPTIONS(2)                  -
          CONTROLINTERVALSIZE(4096)        -
          CATALOG(user catalog name)
   DEFINE CLUSTER (NAME(CICS410.SYS1.MANZ) -
          NONINDEXED                       -
          VOLUME(vvvvvv)                   -
          CYLINDERS(10)                    -
          REUSE                            -
          RECORDSIZE(125 32767)            -
          SPANNED                          -
          NOCOMPRESSED                     -
          SHAREOPTIONS(2)                  -
          CONTROLINTERVALSIZE(4096)        -
          CATALOG(user catalog name)
/*
```

Initialize DMF data sets

You initialize the DMF data sets using the DMF dump utility DFHDFOU. The following sample JCL shows how to initialize the two VSAM files created above.

```
// JOB INITIALIZE DMF DATA SETS
// DLBL catname,'user catalog name',,VSAM
// DLBL INFILE1,'CICS410.SYS1.MANY',,VSAM,CAT=catname
// DLBL INFILE2,'CICS410.SYS1.MANZ',,VSAM,CAT=catname
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC DFHDFOU,SIZE=DFHDFOU
   INDD(INFILE1, OPTIONS (CLEAR))
   INDD(INFILE2, OPTIONS (CLEAR))
/*
```

JCL required to start DMF

The following sample JCL can be used to start DMF. It assumes that the default table is being used.

```
// JOB START DMF
// LIBDEF DUMP,CATALOG=system dump sublibrary      1
// OPTION SYSDUMP                                  1
// DLBL IJSYSUC,'user catalog name',,VSAM
// LIBDEF *,SEARCH=(PRD1.BASE,user sublibrary)
// EXEC DFHDFSIP,SIZE=DFHDFSIP,OS390               2
   SUFFIX=SU
/*
```

Note: 1 If SYSDUMP is not specified either in STDOPT or as part of the OPTION statement, DMF does not issue any SDUMPX request, which results in the loss of diagnostic information if an error occurs. DMF informs you of this condition by issuing message DFHDF0032.

2 Optionally, you may request DMF to issue all messages in upper case. To do this, code a parm statement on the EXEC card:

```
// EXEC DFHDFSIP,SIZE=DFHDFSIP,PARM='CASE(UPPER)',OS390
```

Additionally, you may request that DMF uses a specific language to display its messages. To do this, code a parm statement on the EXEC card:

```
// EXEC DFHDFSIP,SIZE=DFHDFSIP,PARM='CASE(NATLANG(x)',OS390
```

where *x* may be:

E = English
G = German

The full syntax for the parm value is:

```
PARM='NATLANG(x) CASE(ccccc)'
```

where *x* is:

E = English
G = German

and *ccccc* is:

MIXED = **MIXED** case output
UPPER = UPPER case output

and NATLANG and CASE may be specified in any order.

Offloading DMF data sets to sequential files

Once DMF has started, by starting and running CICS TS for example, and you have some records written to it, you will want to offload the data from DMF to sequential data sets for further processing.

To ensure that the data has been moved to the VSAM data sets, enter SETDMF FLUSH to the DMF server, as described in “Operator communications” on page 90.

In order to be able to offload the VSAM data sets, they must be closed to DMF, so enter the command SETDMF SWITCH to the DMF server, as described in “Operator communications” on page 90.

Alternatively, you can shutdown DMF by entering SETDMF SHUTDOWN to the DMF server, as described in “Operator communications” on page 90.

Now you are ready to offload the DMF data sets to a sequential file. You can offload to a DASD file (using either SAM or a file in managed VSAM space) or to an unlabelled tape. #

The following sample JCL shows how to offload a DMF data set to a sequential file.

```
// JOB OFFLOAD DMF, DATA SET
// DLBL IJYSUC,'user catalog name',,VSAM
// DLBL INFILE,'CICS410.SYS1.MANY',,VSAM,CAT=IJSYSUC
// DLBL OUTFILE,'CICS410.DMF.DATA',0,SD
// EXTENT SYSnnn,vvvvvv,1,0,mm,nn
// ASSGN SYSnnn,DISK,VOL=vvvvvv,SHR
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC DFHDFOU,SIZE=DFHDFOU                         1
   INDD (INFILE, OPTIONS (DUMP))
   OUTDD (OUTFILE, TYPE(0:255))
/*
```

A similar sample shows how to offload to a VSAM managed SAM file.

```
// JOB OFFLOAD DMF, DATA SET
// DLBL IJYSUC,'user catalog name',,VSAM
// DLBL INFILE,'CICS410.SYS1.MANY',,VSAM,CAT=IJSYSUC
// DLBL OUTFILE,'CICS410.DMF.DATA',0,VSAM,CAT=IJSUSUC,          *
               DISP=(NEW,KEEP),RECORDS=2000,RECSIZE=5750
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC DFHDFOU,SIZE=DFHDFOU                         1
   INDD (INFILE, OPTIONS (DUMP))
   OUTDD (OUTFILE, TYPE(0:255))
/*
```

A similar sample shows how to offload to an unlabelled tape.

```
// JOB OFFLOAD DMF, DATA SET
// DLBL IJYSUC,'user catalog name',,VSAM
// DLBL INFILE,'CICS410.SYS1.MANY',,VSAM,CAT=IJSYSUC
// TLBL OUTFILE,'CICS410.DMF.DATA'
// ASSGN SYSnnn,cuu                                  2
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC DFHDFOU,SIZE=DFHDFOU                         1
   INDD (INFILE, OPTIONS (DUMP))
   OUTDD (OUTFILE, TYPE(0:255))
/*
```

Once the offload has worked, you can reinitialize the file, as described in “Initialize DMF data sets” on page 97.

You may also input a previously offloaded sequential file along with a newly recorded DMF data set to accumulate data into one sequential file, as shown in the following example JCL.

```
// JOB APPEND NEW DMF DATA
// DLBL IJYSUC,'user catalog name',,VSAM
// DLBL INFILE1,'CICS410.SYS1.MANY',,VSAM,CAT=IJSYSUC
// DLBL INFILE2,'CICS410.DMF.DATA',0,VSAM,CAT=IJSUSUC,          *
               DISP=(OLD,KEEP)
// DLBL OUTFILE,'CICS410.DMF.ACCUM',0,VSAM,CAT=IJSUSUC,         *
               DISP=(NEW,KEEP),RECORDS=4000,RECSIZE=5750
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC DFHDFOU,SIZE=DFHDFOU                         1
   INDD (INFILE1, OPTIONS (DUMP))
   INDD (INFILE2, OPTIONS (DUMP))
   OUTDD (OUTFILE, TYPE(0:255))
/*
```

Note: **1** Optionally, you may request DFHDFOU to issue all messages and output in upper case. To do this, code a parm statement on the EXEC card:

```
// EXEC DFHDFOU,SIZE=DFHDFOU,PARM='CASE(UPPER)'
```

The full syntax for the parm value is:

```
PARM='CASE(ccccc)'
```

where *ccccc* is:

MIXED = **MIXED** case output
UPPER = UPPER case output

# **Note:** **2** In the samples above, SYSnnn must be SYS020.

Migration and coexistence considerations

Attention!

Any user-written or vendor products that analyze monitoring or statistics data need to be rewritten.

- All monitoring and statistics data are written to DMF data sets. There is no alternative repository.
- Monitoring and statistics record formats are changed.
- Account class and global performance data are no longer collected. Account class data is provided by task-related performance data. Global performance data is provided through statistics and task-related information.
- Each DMF record contains data from only one monitoring class, either PERFORMANCE or EXCEPTION.

  In CICS for VSE/ESA Version 2.3, dictionary, accounting, performance, and exception records are mixed within a single SMF 110-format record, with section headers and descriptors to define which type follows in the record. This means that programs have to process the whole record to see if it contains any data of interest. In CICS Transaction Server for VSE/ESA Release 1, a flag in the record header defines the class of data (DICTIONARY, PERFORMANCE or EXCEPTION) that the record contains. Section headers or descriptors **are not** produced.
- The two TRUEs which receive control when monitoring data records are about to be written to the monitoring files in CICS for VSE/ESA Version 2.3 are replaced by the new monitoring global user exit (GLUE), XMNOUT.
- There is a new statistics GLUE, XSTOUT. XSTOUT is called before a statistics record is written to DMF and allows you to suppress the writing of the DMF record.
- If you are coexisting with CICS for VSE/ESA Version 2.3, remember that your CICS for VSE/ESA Version 2.3 system cannot send its monitoring and statistics data to DMF. You should continue to process monitoring and statistics data using CICS for VSE/ESA Version 2.3 methods.

Chapter 9. Statistics utility program, DFHSTUP

The statistics utility program, DFHSTUP, prepares and prints reports offline, using the CICS statistics data recorded on Data Management Facility (DMF) Data Handler data sets.

To enable the CICS statistics domain to record statistics on these DMF data sets, you must specify the STATRCD=ON system initialization parameter. The other statistics record types (unsolicited, requested, and end-of-day) are written regardless of the STATRCD setting. For information about DMF data sets, see the *CICS System Definition Guide* manual. For information about what CICS data is recorded on the DMF data sets, and about interpreting CICS statistics output in the DFHSTUP report, see the *CICS Performance Guide*.

For a description of the STATRCD system initialization parameter, see the *CICS System Definition Guide*.

For a description of DMF itself, see Chapter 8, “Data Management Facility (DMF) Data Handler” on page 85.

Use the version of the DFHSTUP program from the same release of CICS as the data that it is to process. This chapter describes the CICS Transaction Server for VSE/ESA Release 1 version of the DFHSTUP program, which you should use for CICS Transaction Server for VSE/ESA Release 1 data ***only***. If you want to prepare and print reports offline of data from another release of CICS, use the version of the DFHSTUP program for that release.

The statistics recording status

The statistics recording status is set at CICS startup by the system initialization parameter STATRCD.

STATRCD	**Meaning**
OFF (default)	Interval statistics are not collected. 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.

You can use the global user exit XSTOUT in the statistics domain, to give control to an exit program before each statistics record is written to the DMF data set, and can access the relevant statistics record. You may want to use an exit program at this exit point to examine the statistics record and suppress the writing of unwanted records. For information about the XSTOUT global user exit, see the *CICS Customization Guide*.

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 by:

- Changing the STATRCD system initialization parameter as a SIT override during a warm or emergency restart.
- Using CEMT or the EXEC CICS SET STATISTICS command while CICS is running.

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 previous reset time up to the time that you issue the PERFORM STATISTICS command.

The last reset time is either of the following:

- The beginning of the current interval
- The logical end-of-day collection time
- The time that you last issued a CEMT or EXEC CICS SET or PERFORM STATISTICS command specifying RESETNOW.

For details of how to use the CEMT PERFORM STATISTICS and SET STATISTICS commands, see the *CICS-Supplied Transactions* manual. For programming information about the equivalent EXEC CICS commands, see the *CICS System Programming Reference* manual.

Whenever you use a CEMT or EXEC CICS SET command to change the statistics recording status, the changed status is recorded in the global catalog for use in a warm or emergency restart.

"Job to run the DFHSTUP program," below, gives information about how to use the DFHSTUP program to select and format CICS statistics.

Restriction in the use of some output devices

Statistics data is written by CICS to the DMF data sets in a mixture of upper and lower case English characters. By default, the DFHSTUP program outputs the data in a mixture of upper and lowercase characters. If the keyword UPPERCASE=YES is coded in the SYSIPT data stream, all data is output in uppercase only. This provides support for Katakana devices.

Job to run the DFHSTUP program

The job shown in Figure 18 on page 104 comprises three job steps, of which the second step is optional. The job steps are:

1. Unload the DMF data set (or data sets) containing the CICS statistics that you want to process.

   Note: For other examples of jobs to unload the DMF data set, see "Offloading DMF data sets to sequential files" on page 98.

2. Optionally, run the journal utility program to extract from the dumped DMF data set(s) the statistics record types only. If you are interested in the statistics for a single CICS region only, you can add another OPTION statement to select the statistics records for the CICS APPLID you want.

   Note: Using the journal utility to select APPLIDs is optional; the statistics utility program also allows you to select by specific APPLID. In this example we are selecting records using the generic APPLID at offset 47. If you want to select records for the specific APPLID of a CICS region running with XRF, you should specify OFFSET=55.

3. Run the statistics utility program to sort, format, and print the statistics data.

```
// JOB STUP
* *********************************************************************
* Step 1: Unload data from the DMF data sets
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE,PRD2.SCEEBASE)
// DLBL catname,'user.cat.name',,VSAM
// DLBL INDD1,'CICS410.DMF.MANx',,VSAM,CAT=catname
// DLBL INDD2,'CICS410.DMF.MANy',,VSAM,CAT=catname
// DLBL OUTDD1,'CICS410.DMF.DATA',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// EXEC DFHDFOU,SIZE=DFHDFOU
INDD ( INDD1, OPTIONS (DUMP))
INDD ( INDD2, OPTIONS (DUMP))
OUTDD ( OUTDD1, TYPE( 000:255)        )
/*
* *********************************************************************
* Step 2: Optionally, select and copy the statistics records only
*         to a journal-type data set using the juornal utility
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE,PRD2.SCEEBASE)
// DLBL DFHJRNL,'CICS410.DMF.DATA',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// DLBL DFHJOUT,'CICS410.DMF.STATS',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// EXEC DFHJUP,SIZE=DFHJUP
OPTION  COPY  OFFSET=6,FLDTYP=X,VALUE=6E,FLDLEN=1
OPTION  COPY  OFFSET=23,FLDTYP=X,VALUE=0002,FLDLEN=2
OPTION  COPY  OFFSET=47,FLDTYP=C,VALUE=genapplid,FLDLEN=8
* The next test group is only needed to select a second APPLID
OPTION  COPY  OFFSET=6,FLDTYP=X,VALUE=6E,FLDLEN=1
OPTION  COPY  OFFSET=23,FLDTYP=X,VALUE=0002,FLDLEN=2
OPTION  COPY  OFFSET=47,FLDTYP=C,VALUE=genapplid2,FLDLEN=8
END
/*
* *********************************************************************
* Step 3: Sort, format and print the statistics records
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE,PRD2.SCEEBASE)
// DLBL DFHSTAT,'CICS410.DMF.STATS',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// DLBL DFHSTWK,'SORT.WORK.FILE.0',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// DLBL SORTWK1,'SORT.WORK.FILE.1',0,SD
// EXTENT SYS0nn,Volser,1,0,nnn,mm
// ASSGN SYS0nn,DISK,VOL=volser,SHR
// EXEC DFHSTUP,SIZE=2M,OS390                        1
SELECT APPLID=(applid1,applid2)
COLLECTION TYPE=ALL
SORT WORK=1
/*
/&
```

Figure 18. Example job to extract and print statistics data

Note: 1 A size of 2M is shown for DFHSTUP. This is shown as a guideline only; you may have to adjust this value depending on which sort package is installed.

Note: If the input data set, DFHSTAT, is on disk, use

```
// DLBL DFHSTAT, ...
// EXTENT SYS011, ...
// ASSGN SYS011,DISK, ...
```

and if on tape, use

```
// TLBL DFHSTAT, ...
// ASSGN SYS020,cuu
```

Each control parameter in the SYSIPT data stream should start on a new line and is terminated by a blank. If you need to continue a control parameter for more than one line, you must ensure that the line to be continued ends with a comma in column 1 through 71, there is a non-blank character in column 72 of the line to be continued, and start each continuation line in column 16. For example:

```
   Column  16 v                                               72 v
SELECT APPLID=(CICSIDA,CICSIDB,CICSIDC,CICSIDD,CICSIDE,CICSIDF,        /
               CSIDG,CICSIDN,CICSID)
COLLECTION TYPE=ALL
```

The available parameters are described in “Control parameters of the DFHSTUP program.”

Control parameters of the DFHSTUP program

This section describes the parameters that you can use to control the output of statistics data by the DFHSTUP program. If you want the statistics output in uppercase only, you must code the UPPERCASE=YES parameter first in the parameter list.

If you do not code any parameter, the DFHSTUP program formats all the collection types for all APPLIDs, to a page size of 60 lines.

COLLECTION TYPE={ALL|[,INT][,EOD][,REQ][,RRT][,USS]}

Code this parameter to select the statistics records to be included in the formatted reports for the selected APPLIDs, according to their collection type. The parameter keywords must be coded as shown, with one blank between the two words. The operands are as follows:

ALL Format the statistics for all types of collection, for all the specified APPLIDs: this is the default.

INT Format the statistics that were collected at specified intervals, for all of the selected APPLIDs.

EOD Format end-of-day type statistics for all of the selected APPLIDs. The end-of-day collection type is simply another interval collection, being the final collection of statistics for the last interval at the time of shut down or logical end-of-day. Because the specified interval over which interval statistics are being collected may not have elapsed, the actual interval spanning the end-of-day collection can be a short interval.

REQ Format requested statistics for all of the selected APPLIDs.

RRT Format requested reset statistics for all of the selected APPLIDs. These are statistics asked for by using the CEMT or EXEC CICS PERFORM STATISTICS RESETNOW command or, when changing the statistics recording status, the CEMT or EXEC CICS SET STATISTICS ON|OFF RECORDNOW RESETNOW command.

USS Format unsolicited statistics for all of the selected APPLIDs. CICS collects unsolicited statistics for:

Autoinstall Whenever an autoinstalled terminal entry in the TCT is deleted (after the terminal logs off), CICS collects statistics covering the autoinstalled period since the last interval. This period covers any delay interval specified on the system initialization parameter AILDELAY.

If an autoinstall terminal logs on again before the expiry of the delay interval, then the accumulation of statistics continues until the next interval. At that interval, the accumulation of statistics is restarted.

FEPI connections Unsolicited connection statistics are produced when a connection is destroyed. This could occur with a DISCARD TARGET, DISCARD NODE, DISCARD POOL, DELETE POOL, DISCARD NODELIST, or DISCARD TARGETLIST command is used.

FEPI pools Unsolicited pool statistics are produced when a pool is discarded (DISCARD POOL or DELETE POOL).

FEPI targets Unsolicited target statistics are produced when a target is destroyed or removed from a pool. This occurs when a DELETE POOL, DISCARD POOL, DISCARD TARGET, or DISCARD TARGETLIST command is used.

Files Whenever CICS closes a file, CICS collects statistics covering the period from the last interval.

LSRPOOL When CICS closes the last file in an LSRPOOL, CICS collects the statistics for the LSRPOOL. The following peak values are reset to the current value at each interval collection:

- Peak number of requests waiting for a string
- Maximum number of concurrent active file control strings.

The other statistics, which are not reset at an interval collection, cover the entire period from the time the LSRPOOL is created (when the first file is opened) until the LSRPOOL is deleted (when the last file is closed).

Programs Whenever an installed program definition is discarded, CICS collects the statistics covering the installed period since the last interval.

System dumps Whenever a system dump table entry is deleted.

Transactions Whenever an installed transaction definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Transaction classes Whenever an installed transaction class definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Transaction dumps Whenever a transaction dump table entry is deleted.

DATE START=mm/dd/yyyy,STOP=mm/dd/yyyy

Code this to force the DFHSTUP program to print only statistics collected during the specified period (START date to STOP date). This parameter should be used in conjunction with the TIME parameter. If no TIME parameter is coded, statistics collected at any time during the specified period are printed. The parameter keywords must be coded exactly as shown, with one blank between the first two words, and with both START and STOP dates specified. The start and stop dates must be specified as:

mm/dd/yyyy where:

mm = month of the year
dd = day of the month
yyyy = year of the century

DEVICE={<u>DISK</u>|TAPE}

Code this parameter to specify disk or tape input.

<u>DISK</u> Input will be from a data set on disk. This is the default.

TAPE Input will be from a data set on an unlabelled tape.

DSPSIZE=*nnnn*

Code this parameter to specify that your SORT package performs the sort using a data space of size *nnnn*. *nnnn* can is in the range 0000-2048, and represents the number of megabytes of data space to be used for data space sorting.

IGNORE APPLID={applid|(applid1[,applid2]..[,applidN])}

Code this parameter with the APPLIDs of the CICS regions for which you want the statistics ignored. The parameter keywords must be coded as shown, with one blank between the two keywords. Code only one SELECT APPLID parameter or one IGNORE APPLID parameter, with up to 40 APPLIDs. If you specify two or more APPLIDs, you must enclose them in parentheses, and separate them by commas. If you specify more than 40 APPLIDs, the results are unpredictable.

If your CICS regions are defined with both generic and specific APPLIDs, it is the specific APPLID that you must specify on the IGNORE APPLID parameter.

If you do not code this parameter, the DFHSTUP program reports statistics for all APPLIDs found in the DFHSTAT data set, according to the SELECT APPLID parameter.

IGNORE TYPE={type|(type1[,type2]...[,typeN])}

Code this parameter with the resource types for which you want the statistics ignored. The parameter keywords must be coded as shown, with one blank between the two keywords. If you specify two or more resource types, you must enclose them in parentheses, and separate them by commas. (The resource types that you can code on this parameter are listed below.)

Code either the SELECT TYPE parameter or the IGNORE TYPE parameter but not both.

If you do not code this parameter, the DFHSTUP program reports statistics for the resource types found in the DFHSTAT data set, depending on the SELECT TYPE parameter.

The following resource types can be selected or ignored by DFHSTUP:

- AUTOINSTALL
- CONNECTION
- DISPATCHER
- DTB
- FEPI
- FILE
- JOURNAL
- LSRPOOL
- MONITOR
- PROGAUTO
- PROGRAM
- STATS
- STORAGE
- SYSDUMP
- TABLEMGR
- TCPIPSERV #
- TRANCLASS or TCLASS
- TDQUEUE
- TERMINAL
- TRANDUMP
- TRANSACTION
- TSQUEUE
- USER
- VTAM

PAGE SIZE=number

Code this with the number of lines to be formatted per page, in the range 20 to 99. The default page size is 60 lines.

SELECT APPLID={applid|(applid1[,applid2]..[,applidN])}

Code this parameter with the applids of the CICS regions for which you want statistics to be formatted and printed. The parameter keywords must be coded as shown, with one blank between the two words. Code only one SELECT APPLID parameter or one IGNORE APPLID parameter, with up to 40 APPLIDs. If you specify more than 40 APPLIDs, the results are unpredictable.

If your CICS regions are defined with both generic and specific APPLIDs, it is the specific APPLID that you must specify on the SELECT APPLID parameter.

If you do not code this parameter, the DFHSTUP program reports statistics for all APPLIDs found in the DFHSTAT data set, other than those APPLIDs specified on an IGNORE APPLID parameter.

SELECT TYPE={type|(type1[,type2]...[,typeN])}

Code this parameter with the resource types for which you want statistics to be formatted and printed. The parameter keywords must be coded as shown, with one blank between the two words. If you specify two or more resource types, you must enclose them in parentheses, and separate them by commas. (The resource types that you can code on this parameter are listed below.)

Code either the SELECT TYPE parameter or the IGNORE TYPE parameter but not both.

If you do not code this parameter, the DFHSTUP program reports statistics for all resource types found in the DFHSTAT data set, other than those resource types specified on an IGNORE TYPE parameter.

SORT WORK=*n*
Code this parameter to specify the number of SORT work files you want to use. The number you specify in *n* must have matching SORTWK1 thru SORTWKn DLBL statements specified.

SUMMARY
Code this if you require the DFHSTUP program to produce a summary report for each APPLID selected. A summary report is composed by adding together the statistics contained in the interval, requested reset, end-of-day, and unsolicited collections. The summary report statistics are listed in almost the same order as interval and end-of-day reports.

The summary report lists statistics records in the following type order:

Statistics domain
Transaction manager
Transaction class
Dispatcher
Monitoring
Storage Manager DSA
Storage Manager task subpool
Storage Manager domain subpool
Loader
Temporary storage
Transient data
VTAM
Terminal Autoinstall
Program Autoinstall
System dump
Transaction dump
Dynamic transaction backout
Table manager
Transaction
Program
File
Transient data queue
Journal
LSR Pool
Terminal
ISC/IRC system and mode entry
FEPI pool
FEPI connection
FEPI target
User domain

Note: The statistics produced in the summary report for SELECT TYPE(LSRPOOL) do not contain buffer information for individual LSR pools.

If the DMF data set (or data sets) contains CICS statistics from several runs of CICS with the same applid, you must use the TIME and, if necessary, DATE parameters to produce the summary report for one run of CICS. If you do not use the TIME and DATE parameters to specify one of several runs of CICS, the results are unpredictable.

You can save a lot of paper if you code this parameter and omit the COLLECTION TYPE parameter.

If this parameter is not coded, a summary report is not produced.

TIME START=hh.mm.ss,STOP=hh.mm.ss,{ELAPSED|DAILY}

Code this to cause the DFHSTUP program to print only the statistics collected during the specified period (START time to STOP time). This period is determined by a combination of the START to the STOP time, the ELAPSED|DAILY keyword and the DATE parameter as follows:

ELAPSED This is the default. If ELAPSED is coded, DFHSTOP processes every statistics record between the DATE START and TIME START until the DATE STOP and TIME STOP.

DAILY If DAILY is coded, DFHSTOP processes every statistics record between the TIME START and TIME STOP for each day between the specified DATE START and DATE STOP.

If no DATE parameter is coded, statistics collected during the specified period are printed regardless of the date on which they were collected. The parameter keywords must be coded exactly as shown, with one blank between the first two words, and with both START and STOP times specified. The start and stop times must be specified as:

hh.mm.ss where:

hh = number of hours (24 hour clock notation)
mm = number of minutes
ss = number of seconds

Note: If the specified period (START time to STOP time) spans across midnight, the DATE parameter must also be coded.

UPPERCASE=YES

Code this parameter if you want the statistics output in uppercase only. The parameter must be coded as shown in uppercase characters with no spaces between words. The parameter must be the first one coded in the parameter list. If you want output in mixed case (the default), do not code this parameter.

Chapter 10. Monitoring utility programs

CICS provides two programs for processing any CICS monitoring data that is written to Data Management Facility (DMF) data sets. These two programs are:

DFHMNDUP A utility program that generates a monitoring dictionary record, in a sequential data set, for use with monitoring data extracted from DMF data sets.

DFH$MOLS A print program for CICS monitoring data. DFH$MOLS is a sample program to show you how you can code your own monitoring utility program to print CICS monitoring data.

This chapter describes:

- The purpose of the DFHMNDUP program in "The monitoring dictionary utility program, DFHMNDUP," with a sample job to run it.
- The steps required to extract and print monitoring data using the DFH$MOLS sample program, or a utility program of your own modelled on the DFH$MOLS sample program, in "The sample monitoring data print program, DFH$MOLS" on page 115.
- The abend codes and error messages which can be issued by the DFH$MOLS program, in "Abend codes and error messages for the DFH$MOLS program" on page 126.

For more information about DMF itself, see Chapter 8, "Data Management Facility (DMF) Data Handler" on page 85.

The monitoring dictionary utility program, DFHMNDUP

When CICS monitoring is switched on, and you activate the monitoring performance class (system initialization parameter MNPER=ON), CICS first writes a performance dictionary record to the current DMF data set, and then begins to write the monitoring performance data records. A new dictionary record, which always precedes the monitoring data it relates to, is written whenever you:

- Start CICS with the performance class active, and CICS monitoring on
- Change the status of the monitoring performance class from inactive to active, with CICS monitoring on. If monitoring is off and the monitoring performance class is switched from inactive to active, a dictionary record is scheduled from the next time monitoring is activated.

Any monitoring utility program that processes performance data must read the dictionary record that relates to the data being processed before attempting to analyze the data. However, if DMF switches data sets during the period when CICS monitoring is writing performance data, CICS does not write a new dictionary record. A CICS performance dictionary record is not, therefore, the first monitoring performance record on the new DMF data set. The DFHMNDUP program provides a solution to the problem posed by DMF data sets that do not contain a dictionary record.

The performance dictionary record

A performance dictionary record holds specific information about each data field in a performance data record. It derives its information from predefined CICS fields, and from any user-defined fields in the MCT specified for the CICS run. For programming information about the performance dictionary and data records, see the *CICS Customization Guide*. For information about the MCT definitions, see the *CICS Resource Definition Guide* manual.

To enable you to process DMF data sets that contain performance data records but not a dictionary record, DFHMNDUP writes a dictionary record to a sequential data set. The dictionary record is written to a data set specified on a DLBL statement with a file name of SYSUT4.

You then use this file with filename INPUT01 as input to DFH$MOLS.

You provide control information for the DFHMNDUP program in the SYSIPT data stream so that it can generate the correct dictionary record for the performance data you are processing. The values that are specified are used to contruct the fields in the SMF Header and Product section. The relationship of DFHMNDUP SYSIPT parameters to SMF fields is shown in Table 14.

Table 14. Relationship of DFHMNDUP SYSIPT parameters to SMF fields

SYSIPT PARM	SMF DSECT field	Meaning
DATE	SMFMNDTE	Date record moved
GAPPLID	SMFMNPRN	Product name (Generic APPLID)
JOBDATE	SMFMNRSD	Job execution date
JOBNAME	SMFMNJBN	Jobname of CICS job
JOBTIME	SMFMNRST	Job execution time
SAPPLID	SMFMNSPN	Specific APPLID
SYSID	SMFMNSID	System identification
TIME	SMFMNTME	Time record moved
UPPERCASE	N/A	Uppercase output
USERID	SMFMNUIF	User identification

You specify control information for the DFHMNDUP program on the following parameters:

DATE=yyyyddd Code this with the Julian date to be included in the dictionary record. If you do not specify a date, the current date is used.

GAPPLID=name Code this with the generic VTAM APPLID of the CICS region for which you are analyzing performance data.

JOBDATE=yyyyddd
Code this with the Julian date, as seven numeric characters, of the VSE job to be included in the dictionary record.

yyyy represents the year

ddd represents the day, in the range 1 through 366.

If you do not specify a date, the current date is used.

JOBNAME=xxxxxxxx
Code this with a VSE job name for the CICS region to be included in the dictionary record.

JOBTIME=hhmmss
Code this with a time stamp, as six numeric characters, for VSE job to be included in the dictionary record.

hh the number of hours, in the range 00 through 24.

mm
the number of minutes, in the range 00 through 59.

ss the number of seconds, in the range 00 through 59.

If you do not specify a time, the current time is used.

MCT=xx Code this with the suffix of the monitoring control table (MCT) used in the CICS run for which you are analyzing performance data. If your CICS region ran with the system initialization parameter MCT=NO (which results in a default MCT dynamically created by CICS monitoring domain) you should specify MCT=NO for DFHMNDUP also. Alternatively, you can indicate that your CICS used a default MCT by specifying 'MCT=' or 'MCT=,' where appropriate (see below).

SAPPLID=name Code this with the specific VTAM APPLID of the CICS region for which you are analyzing performance data. If you omit this parameter, the value you specify on the GAPPLID parameter is taken as the specific APPLID also.

SYSID=xxxx Code this with the system identifier of the VSE system that owns the DMF data sets.

TIME=hhmmss Code this with a time stamp for the dictionary record. If you do not specify a time, the current time is used.

UPPERCASE Code this parameter if you want the statistics output in uppercase only. If you want output in mixed case (the default), do not code this parameter.

USERID=xxxxxxxx
Code this with eight alphanumeric characters that represent the user identification of the VSE job to be included in the dictionary record.

You can enter each parameter on a separate line, with the parameter keyword starting in column one. Alternatively, you can enter all of the parameters on a single line, starting in column one, with each parameter separated by a comma. If your CICS used a default MCT, you can enter the MCT parameter as 'MCT=NO', 'MCT=', or 'MCT=,'.

For example, you can use the following three methods to specify the same control information for the DFHMNDUP program:

- (MCT=NO)

  ```
  MCT=NO
  GAPPLID=DBDCCICS
  SAPPLID=DBDCCIC1
  DATE=1996256
  TIME=000001
  /*
  ```

- (MCT=)

  ```
  MCT=
  GAPPLID=DBDCCICS
  SAPPLID=DBDCCIC1
  DATE=1996256
  TIME=000001
  /*
  ```

- (MCT=,)

  ```
  MCT=,GAPPLID=DBDCCICS,SAPPLID=DBDCCIC1,DATE=1996256,TIME=000001
  /*
  ```

Sample job illustrating the use of DFHMNDUP

The sample job in Figure 19 shows how you can use the DFHMNDUP program in conjunction with the DFH$MOLS program to print monitoring performance data.

```
// JOB DFHMNDUP
* *********************************************************************
* Step 1: Create new dictionary record
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE,user.sublib,PRD2.SCEEBASE)
// DLBL SYSUT4,'CICS410.MNDUPREC',0,SD
// EXTENT SYS0SS,volser,1,0,nnn,mm
// ASSGN SYS0SS,DISK,VOL=volser,SHR
// EXEC DFHMNDUP,SIZE=DFHMNDUP
MCT=NO
SYSID=3090
GAPPLID=DBDCCICS
SAPPLID=DBDCCIC1
DATE=1998102
TIME=000100
/*
* *********************************************************************
* Step 2: Unload the DMF data set containing CICS records
* *********************************************************************
// DLBL catname,'user.cat.name',,VSAM
// DLBL INDD1,'CICS410.DMF.MANx',,VSAM,CAT=catname
// DLBL OUTDD1,'CICS410.DMF.DATA',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// EXEC DFHDFOU,SIZE=DFHDFOU
INDD ( INDD1, OPTIONS (DUMP))
OUTDD ( OUTDD1, TYPE( 110))
/*
* *********************************************************************
* Step 3: Run DFH$MOLS to print the CICS monitoring data,
*         using the new dictionary record from step 1
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE)
// DLBL INPUT01,'CICS410.MNDUPREC',0,SD
// EXTENT SYS021,volser,1,0,nnn,mm
// ASSGN SYS021,DISK,VOL=volser,SHR
// DLBL INPUT02,'CICS410.DMF.DATA',0,SD
// EXTENT SYS022,volser,1,0,nnn,mm
// ASSGN SYS022,DISK,VOL=volser,SHR
// EXEC DFH$MOLS,SIZE=2M
 CONCAT 02
/*
/&
* $$ EOJ
```

Figure 19. Sample job stream to run DFHMNDUP

The sample monitoring data print program, DFH$MOLS

This section describes how you can unload monitoring data from DMF data sets, for one or more CICS regions, and print the data using the CICS sample utility program, DFH$MOLS. This utility is a CICS-supplied batch program which you can modify or adapt to your own purposes. The job steps involved in processing CICS monitoring data are:

1. Unload the DMF data set(s) so that the DMF data is available for processing by a CICS utility. For information about unloading DMF data sets, see Chapter 8, “Data Management Facility (DMF) Data Handler” on page 85.

2. Run the DFH$MOLS program to print monitoring records, which you can optionally select and sort by means of control statements. See “Control statements of the DFH$MOLS program” on page 118 for further information.

Overview of the DFH$MOLS program

The DFH$MOLS program is a data reduction program designed to produce reports from the data collected by the CICS monitoring domain (MN), and written to DMF data sets. You run the DFH$MOLS program in a batch region to process any CICS SMF type 110 monitoring records that are present in an unloaded DMF data set, which you can write to either a temporary or cataloged data set. You can determine the scope of the report(s) by supplying control statements in the SYSIPT datastream.

You can specify a sort option for the selected data. The DFH$MOLS program sorts the data by means of a link to the VSE sort program, DFSORT™, passing parameters to the sort, and using the sort exits E15 and E35. You can use any standard sort utility provided it has these E15 and E35 exits. For further information about the DFSORT program, see the *DFSORT/VSE Application Programming Guide*.

The program reads, formats, and prints the CICS monitoring data, which is packaged in the following format:

[SMF HEADER].[SMF PRODUCT SECTION].[CICS DATA SECTION]

The CICS data section in a monitoring record is one of the following:

1. A dictionary data section, consisting of a sequence of dictionary entries
2. An exception data section, consisting of a single exception record
3. A performance data section, consisting of a sequence of field connectors followed by one or more performance records

For programming information about the structure of CICS SMF type 110, and how the monitoring data is packaged within the SMF records, see the *CICS Customization Guide*. The DFH$MOLS program removes the DMF data and moves individual CICS monitoring data fields to the print area. If you want to analyze the data using your own routines, this is the point at which you can link to a user-written analysis program.

Monitoring dictionary records

The DFH$MOLS program requires a monitoring dictionary record to process monitoring performance data. When it locates a dictionary record, it builds an in-store dictionary and processes the subsequent (if any) performance data using this dictionary. Whenever it reads a new dictionary record, the current dictionary is released and a new in-store dictionary is built. The dictionary record must appear before any related performance data, otherwise the DFH$MOLS program abends. Note that monitoring exception records do not require a dictionary and so they can precede the first dictionary record and still be successfully processed. For more information about monitoring dictionary records, see “The performance dictionary record” on page 112.

Volume of output

The DFH$MOLS program prints about one page per task, so take care to specify only those items that you need using the DFH$MOLS program control statements. For details of the selection options, see “Control statements of the DFH$MOLS program” on page 118.

Job Control statements to Assemble DFH$MOLS

The sample program DFH$MOLS is supplied in the sublibrary PRD1.BASE as a member of DFH$MOLS.A. If you want to change this member to reflect your own requirements, we recommend that you copy the member to another sublibrary before doing the modifications.

Figure 20 shows a sample job for Assembling and link-editing DFH$MOLS.

```
// JOB ASSEMBLE DFH$MOLS
// STEP 1 DFH$MOLS ASSEMBLE STEP
// LIBDEF PHASE,CATALOG=user.sublibrary
// LIBDEF *,SEARCH=(PRD1.BASE,PRD1.MACLIB),TEMP
// OPTION CATAL
// EXEC ASMA90,SIZE=(ASMA90,50K)
        COPY DFHGDGFS
* $$ SL1 MEM=DFH$MOLS.A,S=(source.sublibrary)
/*
// STEP 2 LINKEDIT DFH$MOLS
// EXEC LNKEDT
/*
```

Figure 20. Sample job for Assembling and link-editing DFH$MOLS.

Sample job stream for the DFH$MOLS programs

Figure 21 shows a sample job with the DMF unload step as well as the monitoring report step.

```
// JOB DFHMNDUP
* *********************************************************************
* Step 1: Unload the DMF data set containing CICS records
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE,PRD2.SCEEBASE)
// DLBL catname,'user.cat.name',,VSAM
// DLBL INDD1,'CICS410.DMF.MANx',,VSAM,CAT=catname
// DLBL OUTDD1,'CICS410.DMF.DATA',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// EXEC DFHDFOU,SIZE=DFHDFOU
INDD ( INDD1, OPTIONS (DUMP))
OUTDD ( OUTDD1, TYPE( 110))
/*
* *********************************************************************
* Step 2: Optionally, select and copy the monitoring records only
*         to a journal-type data set using the juornal utility
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE,PRD2.SCEEBASE)
// DLBL DFHJRNL,'CICS410.DMF.DATA',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// DLBL DFHJOUT,'CICS410.DMF.MONITOR',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// EXEC DFHJUP,SIZE=DFHJUP
OPTION  COPY  OFFSET=23,FLDTYP=X,VALUE=0001,FLDLEN=2
END
/*
* *********************************************************************
* Step 3: Run DFH$MOLS to print the CICS monitoring data
* *********************************************************************
// LIBDEF *,SEARCH=(PRD1.BASE,PRD2.SCEEBASE)
// DLBL INPUT01,'CICS410.DMF.MONITOR',0,SD
// EXTENT SYS0ss,volser,1,0,nnn,mm
// ASSGN SYS0ss,DISK,VOL=volser,SHR
// DLBL SORTWK1,'SORT.WORK.FILE',0,SD
// EXTENT SYS0nn,volser,1,0,nnn,mm
// ASSGN SYS0nn,DISK,VOL=volser,SHR
// EXEC DFH$MOLS,SIZE=2M
SORT
/*
/&
* $$ EOJ
```

Figure 21. Sample job to unload and process CICS data from DMF data sets.

Control statements of the DFH$MOLS program

You control the processing that the DFH$MOLS program performs by specifying the following control statements:

***** [user comments]

BREAK Use this statement to group SELECT/IGNORE statements into logical groups.

CONCAT nn

CONTROL STOPAFT=nnnnnnnn

DATE START=mm/dd/yyyy,STOP=mm/dd/yyyy

IGNORE APPLID=xxxxxxxx[,yyyyyyyy,...]

TERMID=xxxx[,yyyy,...]

TRANID=xxxx[,yyyy,...]

USERID=xxxxxxxx[,yyyyyyyy,...]

You can use any of these IGNORE options in conjunction with SELECT statements to form SELECT/IGNORE groups (see the BREAK control statement).

PRINT EXC,PER,ALL

SELECT APPLID=xxxxxxxx[,yyyyyyyy,...]

TERMID=xxxx[,yyyy,...]

TRANID=xxxx[,yyyy,...]

USERID=xxxxxxxx[,yyyyyyyy,...]

You can use any of these SELECT options in conjunction with IGNORE statements to form SELECT/IGNORE groups (see the BREAK control statement).

SORT Use this statement to sort the input monitoring data before processing.

TIME START=hh.mm.ss,STOP=hh.mm.ss

TIMEOFF Use this statement to suppress testing for data output chronologically.

Rules for coding control statements

Control statements are free format, each statement having an operation keyword followed by a parameter that can have one or more operands. The DFH$MOLS program pads out the operands with trailing blanks. Multiple operands are separated by commas, but if you specify only a single operand, a blank indicates the end of the operand. You can code any characters (except comma and blank) in the operands.

There are no continuation statements; you can specify multiple occurrences of the same control statement keyword, eliminating the need for continuations.

The DFH$MOLS program prints each control statement before analyzing it. If the DFH$MOLS program detects an error, it is associated with the last statement printed. Control statement errors are followed by an abend U101, without a dump.

If you do not specify any control statements, the DFH$MOLS program produces a default listing of the monitoring data, using default values.

Control statement descriptions

*** (asterisk)**

An asterisk appearing in column 1 means that the statement is a comment only and has no effect on processing. It is printed without any further analysis.

BREAK

Code this control statement to end a SELECT/IGNORE group. The BREAK control statement logically terminates the previous (ungrouped) SELECT/IGNORE statements, and the DFH$MOLS program forms them into a logical SELECT/IGNORE group. A BREAK statement has no meaning unless preceded by one or more SELECT/IGNORE statements. You can form multiple

SELECT/IGNORE groups by including BREAK statements at the appropriate points.

The main intent of BREAK is to allow SELECT statements to be processed as a logical OR function instead of as a logical AND function if they are in the same group.

If you do not specify a BREAK statement after the last SELECT/IGNORE statement, the DFH$MOLS program assumes one by default. This means that there is always at least one SELECT/IGNORE group, however few SELECT or IGNORE statements you specify.

If you specify only one SELECT/IGNORE group, either implicitly or by including a BREAK statement, the SELECT/IGNORE logic is the same as described for the individual SELECT/IGNORE statements.

The DFH$MOLS program processes multiple SELECT/IGNORE groups using the following rules in the order listed:

1. Processing starts with the first group.
2. The DFH$MOLS program processes the SELECT/IGNORE groups in the order in which you specify them in SYSIPT.
3. APPLID, USERID, TERMID, and TRANID parameters may all be included in the same run.
4. You cannot specify SELECT and IGNORE for the same parameter in the same SELECT/IGNORE group. For example, SELECT TERMID and IGNORE TERMID is invalid.
5. If you specify SELECT for more than one parameter in a SELECT/IGNORE group, the SELECT statements form a logical AND function.
6. If you specify IGNORE for more than one parameter in a SELECT/IGNORE group, the IGNORE statements form a logical OR function.
7. The DFH$MOLS program processes ***all*** SELECT statements in a group before any IGNORE statements in the same group.
8. If a data record satisfies ***all*** of the SELECT statements in a group, it is selected (but subject to any following IGNORE statements).
9. If a selected record (either by default in the absence of any SELECT statements, or explicitly because it satisfies selection criteria) also satisfies an IGNORE test, the record is excluded.
10. If a record is not included or excluded after all of the SELECT/IGNORE statements in a group have been processed, it is processed by the next group.
11. If a record is not specifically included or excluded after all of the SELECT/IGNORE groups have been processed, one of the following events occurs:
    - If you do not specify a group with IGNORE statements only, to specifically exclude the record, it is excluded by default.
    - If you specify one or more groups with IGNORE statements only, to specifically exclude ***other*** records, the record is included.

      Note: You can specify one or more groups with IGNORE statements only to specifically exclude records. However, any record not included or excluded, after all the SELECT/IGNORE and IGNORE-only groups, is included.

Examples:

The following control statements select records for transaction id TSK1 which were entered from terminal id T040:

```
SELECT TRANID=TSK1
SELECT TERMID=T040
```

The following control statements select records for all records for transaction id TSK1, and all records from terminal id T040. The BREAK statement effectively creates two SELECT/IGNORE groups, and any record satisfying group 1 (the transaction id is TSK1) or group 2 (the terminal id is T040) is selected:

```
SELECT TRANID=TSK1
BREAK
SELECT TERMID=T040
```

The following control statements select records for transaction ids TSK1 and TSK2, but excluding those that were entered from terminal id T040:

```
SELECT TRANID=TSK1,TSK2
IGNORE TERMID=T040
```

The following control statements select all records for transaction id TSK1 (SELECT group 1) and all records for transaction id TSK2 excuding those entered from terminal id T040 (SELECT/IGNORE group 2):

```
SELECT TRANID=TSK1
BREAK
SELECT TRANID=TSK2
IGNORE TERMID=T040
```

If you also have records for terminal ids T050 (for transaction ids TSK1 and TSK3) and T060 (for transaction id TSK3 only), you can use the following IGNORE-only group to exclude all records entered from terminal id T050:

```
IGNORE TERMID=T050
```

In this case, records for terminal id T060 are included because you have not specifically excluded them.

To exclude the records from terminal ids T050 and T060, you can do one of the following:

- Do not specify any IGNORE-only groups; the records for terminal ids T050 and T060 are excluded by default.
- Specify one or more IGNORE-only groups, to specifically exclude records from terminal ids T050 and T060, for example:

  ```
  IGNORE TERMID=T050,T060
  ```

  In this case, if you later add another terminal, its records are included unless you specify the terminal id in an IGNORE-only group.

CONCAT nn

Specify the CONCAT statement with its associated *nn* parameter to specify how many input data sets you want to process. DFH$MOLS appends

01,02..nn to filename INPUT and processes each file in sequence. If you do not specify CONCAT, DFH$MOLS processes only 1 file, called INPUT01

DFH$MOLS also assigns different SYS numbers to each input file, starting with SYS021 and adding 1 for each input file.

CONTROL STOPAFT=nnnnnnnn

Specify the CONTROL statement with its associated STOPAFT parameter to control the number of records you want to process.

STOPAFT=nnnnnnnn

Specify this parameter to limit the number of SMF type 110 records you want the DFH$MOLS program to process. The DFH$MOLS program terminates after processing the number of SMF 110 records specified by nnnnnnnn.

DATE

Specify the start and stop dates on this statement which, in conjunction with the TIME statement (if specified), enables you to select records for a particular period only. (See also the TIME control statement.)

START=start date

Specify the date of the beginning of the period for which you want records processed, in the form `mm/dd/yyyy`.

STOP=stop date

Specify the date of the end of the period for which you want records processed, in the form `mm/dd/yyyy`.

You do not have to specify both START and STOP; you can specify START without STOP, and STOP without START. If you omit the DATE statement, records for all dates present in the input file are processed. You can specify only one DATE statement (and associated TIME statement) in SYSIPT.

IGNORE {APPLID|TERMID|TRANID|USERID}

Code an IGNORE control statement to exclude all records from a CICS region that has the specified generic APPLID, or all records that have a specified transaction, terminal, or user identifier.

APPLID=xxxxxxxx[,yyyyyyyy,.,.]

Specify one or more generic APPLIDs to exclude monitoring data from a CICS region, or regions.

TERMID=xxxx[,yyyy,.,.]

Specify one or more terminal identifiers to exclude monitoring data associated with these terminals.

TRANID=xxxx[,yyyy,.,.]

Specify one or more transaction identifiers to exclude monitoring data for these transactions.

USERID=xxxxxxxx[,yyyyyyyy,.,.]

Specify one or more user identifiers to exclude monitoring data for transactions submitted by these users.

You can specify each of the APPLID, TERMID, TRANID, and USERID parameters in the same SELECT/IGNORE GROUP, but you cannot specify an IGNORE and SELECT for the same type of parameter. For example, you ***can*** specify SELECT APPLID= and IGNORE TERMID=, but you ***cannot*** specify SELECT APPLID= and IGNORE APPLID=.

The DFH$MOLS program pads out the operands with trailing blanks. You cannot continue control statements on another line, but the program logically chains multiple control statements of the same keyword in the same IGNORE group (see the BREAK control statement). If you specify IGNORE for more than one parameter, those IGNORE statements form a logical OR function.

Examples:

If you specify:

```
IGNORE TRANID=CEMT
IGNORE USERID=OP7
```

the program excludes all records for transaction CEMT (regardless of user ID), and exclude all records containing userid OP7 (regardless of transaction ID). It includes all other records.

If you specify:

```
SELECT TRANID=CEMT
IGNORE TERMID=TRM3
```

the program includes only records for transaction CEMT, except for those from terminal TRM3.

PRINT {<u>ALL</u>|EXC|PER|EXC,PER}
Specify on this statement the type of monitoring data record that you want printed.

<u>ALL</u> lists all of the monitoring SMF type 110 records that are selected by any other control statement options. This is the default if you omit the PRINT statement.

EXC
lists only the monitoring exception class records that are selected by any other control statement options.

PER
lists only the monitoring performance class records that are selected by any other control statement options.

Note that the SMF headers, SMF product sections, and CICS dictionary records are always printed.

SELECT {APPLID|TERMID|TRANID|USERID}
The SELECT control statement includes all records of the specified generic APPLIDs, transaction, terminal, or user identifiers.

APPLID=xxxxxxxx[,yyyyyyyy,.,.]
Specify one or more generic APPLIDs to include monitoring data from the CICS regions identified by these APPLIDs.

TERMID=xxxx[,yyyy,.,.]
Specify one or more terminal identifiers to include monitoring data associated with these terminals.

TRANID=xxxx[,yyyy,.,.]
Specify one or more transaction identifiers to include monitoring data for these transactions.

USERID=xxxxxxxx[,yyyyyyyy,.,.]
Specify one or more user identifiers to include monitoring data for transactions submitted by these users.

You can specify each of the APPLID, TERMID, TRANID, and USERID parameters in the same SELECT/IGNORE GROUP, but you cannot specify IGNORE and SELECT for the same type of parameter. For example, you ***can*** specify SELECT APPLID= and IGNORE TERMID=, but you ***cannot*** specify SELECT APPLID= and IGNORE APPLID=.

You cannot continue control statements on another line, but the program logically chains multiple control statements of the same keyword in the same SELECT group. (See the BREAK control statement for details of how to terminate a SELECT/IGNORE group.) If you specify SELECT for more than one parameter, those SELECT statements form a logical AND function.

Examples:

If you specify:

```
SELECT TERMID=TRM3
SELECT TRANID=CEMT
```

the program includes only records with a transaction identifier of CEMT ***and*** with a terminal identifier of TRM3. It does not include any other records.

If you specify:

```
SELECT APPLID=DBDCCICS
SELECT TRANID=CEMT
IGNORE TERMID=TRM3
```

the program includes only those records that are from the CICS region with the generic APPLID DBDCCICS and are for transaction CEMT, but do ***not*** have the terminal identifier TRM3.

SORT
Specify SORT to request a sort of the input monitoring data before the records are processed for output. If you specify a sort, the DFH$MOLS program links to the standard VSE SORT program, DFSORT. (For information about the standard sort used, see the *DFSORT/VSE Application Programming Guide*.)

The SORT utility sorts the monitoring data into the following sequence:

1. Generic APPLID at position 47
2. SMF record sub-type at position 23
3. SMF record date at position 11
4. SMF record time at position 7

These sort fields are built into the DFH$MOLS program, and therefore the SORT statement does not require any parameters. However, if you want to perform a stand-alone sort before running the sample utility, you should use the following SORT statements:

```
SORT FIELDS=(48,8,CH,A,23,2,BI,A,11,4,PD,A,7,4,BI,A),EQUALS
RECORD TYPE=V
```

Notes:

1. SORT is the recommended option when you are processing data from multiple DMF data sets, and must be used when processing data for multiple CICS regions.
2. If you specify SORT, you must supply a DLBL for SORTWK1.

TIME

Specify the start and stop times on this statement which, in conjunction with the DATE statement (if specified), enables you to select records for a particular SMF time period only. (The time stamp against which the DFH$MOLS program compares is the SMF time in the SMF header, not the time in individual performance records. This means that the program may select performance records for times that may be a few minutes outside the specified period because of the way they are buffered for writing to DMF.)

Note: A TIME statement without a DATE statement causes The DFH$MOLS program to select data for the specified time period for ***all*** dates present in the input data set.

START=start time

Specify the start time of the period for which you want records processed, in the form `hh.mm.ss` or `hhmmss`. A start time is optional, and if omitted the report includes all records for the start date, irrespective of time.

STOP=stop time

Specify the end time of the period for which you want records processed, in the form `hh.mm.ss` or `hhmmss`. An end time is optional, and if omitted the report includes all records for the stop date, irrespective of time.

You do not have to specify both START and STOP; you can specify START without STOP, and STOP without START. You can specify only one TIME statement (and associated DATE statement) in SYSIPT

TIMEOFF

Specify the TIMEOFF statement to suppress testing for data being out of chronological sequence.

By default, the DFH$MOLS program checks the date/time sequence of the data to prevent incorrect processing caused by data being out of sequence. This can occur if you do not specify the SORT option of the DFH$MOLS program, and one of the following is true:

- The input data is from a data set incorrectly sorted prior to running the DFH$MOLS program.
- The unsorted data is from multiple DMF data sets that are not concatenated in time ordered sequence, or are unloaded in the wrong sequence.

The default sequence check causes the DFH$MOLS program to terminate if data is out of sequence. You are recommended to omit the TIMEOFF statement, unless you have some reason for suspecting that the data is out of chronological sequence, and want to obtain a full listing of all the data.

Abend codes and error messages for the DFH$MOLS program

The DFH$MOLS program can fail with one of the following error messages:

INVALID, DUPLICATE, OR MISSING CONTROL CARD INFORMATION
The DFH$MOLS control statement scan routines produce this message. It occurs immediately after processing an invalid control card, and the last control statement printed contains the error. Some of the causes are:

- Misspelled control card operation
- Misspelled control card operand
- Invalid operand delimiter
- Operand length too long

INCOMPATIBLE CONTROL CARD OPTIONS REQUESTED
Some of the selected control statement options are in conflict.

SORT ENDED WITH NON-ZERO RET CODE; REPORT IS TERMINATED
The DFSORT utility has returned an error code after being invoked. This may be due to missing or incorrect job control statements. The SORT utility writes a message to SYSOUT defining the error.

NO DATA FOR THIS CICS DATA SECTION; REPORT IS TERMINATED
The DFH$MOLS program has detected a CICS 110 monitoring record that does not contain any data rows within the data section. The DFH$MOLS program detects this by validating the contents for field SMFMNDRN in the SMF product section.

UNKNOWN CLASS IN CICS DATA SECTION; REPORT IS TERMINATED
The DFH$MOLS program has detected a CICS 110 monitoring record that does not contain a valid data section. The DFH$MOLS program detects this by validating the contents for field SMFMNCL in the SMF product section, which should indicate that the data section contains either dictionary data, performance data, or exception data.

NO VALID MONITORING RECORDS WERE READ; REPORT IS TERMINATED
There is no valid CICS monitoring data in the input data set.

NO MONITORING RECORDS WERE SELECTED FOR PROCESSING; REPORT IS TERMINATED
Valid CICS monitoring records have been read, but no records are eligible for processing because of SELECT/IGNORE control statements.

X‘nnnn’ - CONNECTOR NOT DEFINED; REPORT IS TERMINATED
A CICS 110 monitoring performance class record has a connector number that is not in the current dictionary.

NO DICTIONARY ESTABLISHED; REPORT IS TERMINATED
The first CICS 110 monitoring performance class record read by the DFH$MOLS program has not been preceded by the corresponding dictionary record. The DFH$MOLS program must establish the dictionary record before it can correctly analyze any associated monitoring performance class records. (See “Monitoring dictionary records” on page 116.)

NO DICTIONARY ESTABLISHED FOR NEW APPLID xxxxxxxx; REPORT IS TERMINATED
The DFH$MOLS program has detected a change of APPLID in CICS 110 monitoring performance class data before it has read the corresponding

dictionary record for the new APPLID. The DFH$MOLS program must establish the new dictionary record for the new APPLID before it can correctly analyze any associated monitoring performance class records.

DATE/TIME STEPDOWN ERROR ON INPUT; REPORT IS TERMINATED
The DFH$MOLS program has detected that the data is out of sequence. CICS writes monitoring records in time sequence, and the date/time fields should continually increase in value. (The TIMEOFF control statement suppresses the date/time field check.)

UNABLE TO LOAD SORT PACKAGE; REPORT IS TERMINATED
The DFH$MOLS program has attempted to load the SORT phase but this has failed. Ensure that the sublibrary containing your SORT package is in the LIBDEF PHASE search chain.

Chapter 11. Trace utility print programs

There are two destinations for CICS region trace data:

1. A table in main storage, when you specify INTTR=ON and SYSTR=ON as system initialization parameters
2. The CICS auxiliary trace data sets, when you specify AUXTR=ON and SYSTR=ON as system initialization parameters

You can also obtain trace entries at these destinations while CICS is running, by means of the CETR trace transaction or the equivalent EXEC CICS SET commands.

This chapter describes how you can print the CICS region trace data from the CICS auxiliary trace data sets, using the CICS trace utility program, DFHTU410.

The CICS trace utility program, DFHTU410

The CICS utility program, DFHTU410, extracts all or selected trace entries from the A or B auxiliary trace data set, and formats and prints the data. You specify the type of entries to be processed by this program on trace selection parameters supplied in either of the following:

- A PARM parameter on the EXEC DFHTU410 statement
- The SYSIPT data stream

You can specify that all entries are to be processed, or select specific entries for processing. For example, you can select trace entries:

- Written to the auxiliary trace data set within a specified period of time
- Written for a specified terminal
- With a specified trace identifier
- With specified trace entry sequence numbers[4]
- Associated with a specified transaction identifier
- Associated with a specific instance of a transaction identifier (task)
- Associated with a selected kernel task
- That are for exception trace only

You can select which trace entries you want to highlight in your formatted output by specifying the time interval between one trace entry and the next being written.

If more than the specified interval elapses before the next trace entry is written, then this next trace entry is formatted and printed with an asterisk (*) to draw your attention to this entry.

# You can use the job control statements shown in Figure 22 on page 130 to invoke
# the utility program for each auxiliary trace data set. If the auxiliary trace data set is
# on a tape, use the job control statements in Figure 23 on page 130. Only use the
# trace utility program to print auxiliary trace data sets that you have opened in the
# most recent run of CICS. If you did not open an auxiliary trace data set during the

[4] The sequence number is given in each trace entry, and can be determined from a summary trace point.

most recent run of CICS, the trace utility program either prints records from a previous run or cannot recognize the records. If you opened the auxiliary trace data set A in the most recent run of CICS, but did not open auxiliary data set B, you can print data set A, but not print B.

```
* $$ JOB JNM=PRTAX2,CLASS=x,DISP=D
* $$ LIST CLASS=A
// JOB C410AUXP   PRINT AUX TRACE
// DLBL DFHAUXT,'user.label',0,SD
// EXTENT SYS001,volname,1,0,nnnn,nnn
// ASSGN SYS001,DISK,VOL=volname,SHR
// LIBDEF     *,SEARCH=PRD1.BASE
// EXEC DFHTU410,SIZE=DFHTU410
* ** TRACE SELECTION PARAMETERS  ** *
  ⋮
/*
// EXEC LISTLOG
/*
* $$ EOJ
```

Figure 22. Sample to invoke DFHTU410 for each trace data set

```
* $$ JOB JNM=DFHAUXPR,DISP=D,CLASS=0
// JOB DFHAUXPR   PRINT CICS AUXILIARY FROM TAPE
*
* STEP6: REPLY 'END/ENTER' TO RUN THIS JOB ANALYZING TRACED DATA
*
* FOR PRINT OPTIONS REFER TO THE CICS OPERATIONS AND UTILITIES GUIDE
*
// TLBL DFHAUXT,'CICS.AUXTRACE'
// ASSGN SYS009,380
// PAUSE ---> PLEASE FOLLOW GENERAL INSTRUCTIONS ABOVE
// EXEC DFHTU410,SIZE=1880K,OS390
* ** TRACE SELECTION PARAMETERS ** *
/*
  ⋮
/&
* $$ EOJ
```

Figure 23. Sample to invoke DFHTU410 when trace data set is on tape

The trace selection parameters

You code the trace selection parameters to define the number of lines to be printed on a page and to define which trace records you want to select for printing either via SYSIPT or in the PARM parameter. For example:

```
// EXEC DFHTU410,SIZE=DFHTU410,PARM='selection_parameter,selection_parameter,...'
```

ABBREV|<u>FULL</u>

Code this statement to indicate how much of each trace entry you want printed. If you specify this statement it must always be the first statement in either the PARM parameter or the SYSIPT data stream.

ABBREV indicates that you require the abbreviated, one-line-per-entry, form of trace print.

FULL indicates that you want a fully formatted print of all the data in each entry. This is the default.

ALL
This statement specifies that all trace entries in the auxiliary trace data set are to be printed. This is the default.

ENTRY_NUM=({nnnnnn|nnnnnn-nnnnnn}[,{nnnnnn|nnnnnn},.,.,.])
This statement specifies the sequence numbers of one or more trace entries that you want to print. Each sequence number can be up to six digits in length. If you specify a range of sequence numbers, by using xxxxxx-yyyyyy, the second sequence number (yyyyyy) must be larger than the first (xxxxxx).

EXCEPTION
This statement specifies that only exception trace entries in the auxiliary trace data set are to be printed.

INTERVAL={0.0128|number of seconds}
This parameter specifies the interval between auxiliary trace entries after which entries highlighted with an asterisk as follows:

- In abbreviated trace format, the asterisk appears to the left of the sequence number.
- In full trace format, the asterisk appears as the next character after the printed time interval.

If successive auxiliary trace entries are written at intervals equal or greater than this limit, they are highlighted in the same manner.

If successive auxiliary trace entries are written at intervals less than this limit, they are not highlighted. They are, however, written, formatted and printed.

If you specify no INTERVAL value, a default of 0.0128 seconds applies.

You can specify interval values in the range zero seconds (where all trace entries would be highlighted) through 99.9999999999 seconds.

Note: The interval extends to ten decimal places. Zeros are padded from the right.

KE_NUM=(xxxx[,xxxx,.,.])
This statement specifies that only the entries for tasks with the specified hexadecimal kernel task numbers are printed.

PAGESIZE=(value)
This statement specifies the number of lines printed on a page. You can specify a value in the range 20 through 9999 lines per page. If you specify an incorrect value, CICS issues an error message and stops the trace. The default value is 55.

TASKID=({id|id-id}[,,{id|id-id},.,..])
This statement specifies the task identifiers (id) of one or more tasks for which trace entries are to be printed. An id value can be in any of the following forms, to compare with the task field in the formatted trace data:

- Any number up to 5 decimal digits long
- Any of the character strings JAS, J01 through J99, III, TCP, or DSTCB

- Any non-numeric two-character domain ID of the attaching domain (for non-TCA) tasks

You can specify a range of task identifiers of the five decimal digit form by using a hyphen (for example, TASKID=nnnnn-nnnnn).

TERMID=(tttt[,tttt,.,.,.])
This statement specifies the terminal identifiers (tttt) of one or more terminals for which trace entries are to be printed.

If you use the TERMID parameter to specify the trace entries you want formatted, the DFHTU410 program selects all the trace entries that are associated with any transaction-attach trace entries it finds containing the terminal identifier(s) you specify. For more information about how trace entries for tasks are associated with transaction-attach trace entries, see "Identifying trace entries from their transaction-attach entries" on page 133.

TIMERG=(hhmmss-hhmmss[,hhmmss-hhmmss,.,.,.])
This statement specifies the time period or periods for which trace entries are to be printed. Time periods are shown by pairs of values represented as hours (hh), minutes (mm), and seconds (ss) separated by a hyphen. The ending value of each pair must be later than the starting value.

The DFHTU410 program converts the store-clock (STCK) values in the trace entries to whole seconds prior to comparing against the time range you specify. Fractions of a second are ignored; that is, all times are rounded down to the nearest whole second, which means in effect that the minimum time span can be up to two seconds. For example, if you specify TIMERG=153000-153001, the DFHTU410 program includes all trace entries with times of 153000.00000000 to 153001.99999999 inclusive.

TIMESTAMP
This statement specifies that time stamps (maximum interval of 9.999999 seconds) are printed instead of the trace sequence numbers in an abbreviated trace.

TRANID=(tttt[,tttt,.,.,.])
This statement specifies the transaction identifiers of one or more transactions for which trace entries are to be printed.

If you use the TRANID parameter to specify the trace entries you want formatted, the DFHTU410 program selects all the trace entries that are associated with any transaction-attach trace entries it finds that contain the transaction identifier(s) you specify. For more information about how trace entries for tasks are associated with transaction-attach trace entries, see "Identifying trace entries from their transaction-attach entries" on page 133.

TYPETR=({ddxxxx|ddxxxx-xxxx}[,{ddxxxx|ddxxxx-xxxx}])
This statement specifies the trace entry identifiers for the particular domain entries, specified by the domain id and a point id within the domain.

dd represents the domain identifier (listed in Table 15 on page 133).

xxxx represents the point ID within the domain in the form of a four-character hexadecimal value (0000-FFFF). You can specify a range of point IDs by using a hyphen.

Table 15. CICS domains and their identifiers

Domain ID	Domain
AP	Application
DD	Directory manager
DM	Domain manager
DS	Dispatcher
DU	Dump
EX	External CICS interface
GC	Global catalog
KE	Kernel
LC	Local catalog
LD	Loader
LM	Lock manager
ME	Message
MN	Monitoring
PA	Parameter manager
PG	Program manager
SM	Storage manager
ST	Statistics
TI	Timer
TR	Trace
US	User
XM	Transaction manager
XS	Security manager

UPPERCASE
This statement specifies that you want the trace output in uppercase only. If you want trace output in mixed case (the default), do not code this parameter.

Identifying trace entries from their transaction-attach entries

The AP domain writes a trace entry each time a transaction is attached for execution. It is this transaction-attach trace entry that contains the terminal and transaction identifiers. It also contains the task identifier that is unique to a particular instance of a transaction. This is illustrated in the diagram shown in Figure 24 on page 135.

If you select trace entries by specifying the TRANID or TERMID parameters, the DFHTU410 program searches for any transaction-attach trace entries that contain the specified TERMID or TRANID. It then formats any associated trace entries, identified by the TASKID found in the transaction-attach trace entry data.

For example, if the entries in your auxiliary trace data set are as illustrated in Figure 24 on page 135, you can obtain formatted trace output for task IDs 00123 and 00124 by specifying either the TERMID or TRANID parameters. This is possible because the associated transaction-attach trace entries are present (see record numbers 2 and 7 in the diagram). However, you cannot obtain formatted trace output for task ID 00120 by specifying a TERMID or TRANID, because the auxiliary trace data does not contain the transaction-attach trace entry for that task.

Rules for coding trace selection parameters

If you enter the control statements in the SYSIPT data stream, enter them in columns 1 to 71 of the 80-character records. Leading blanks are ignored, and no imbedded blanks are allowed. The first blank in a line terminates the statements on that line; you can include comments after the first blank. The TERMID and TRANID entries are padded with blanks to four characters if necessary.

You can specify each control statement one or more times; for example,

```
TASKID(xxxx,zzzz,yyyy,aaaa,bbbb,cccc,dddd,eeee,ffff,gggg,hhhh,iiii,jjjj),
TASKID(kkkk,rrrr-uuuu,wwww)
```

You must use commas to separate keywords and entries in a list. Continuation to another record is allowed after any comma ***that separates keywords***, provided the comma is in column 71 or is followed by a blank. Continuation records can start in any column.

For example, the following statements can be coded in the SYSIPT data stream:

```
TERMID=LP1,                  [Select entries for terminal LP1
TRANID=(ABRW,AORD,MYTR),     [Select entries for tranids ABRW, AORD, & MYTR
TIMERG=(123000-150000))      [Select entries timed between 1230 and 1500
```

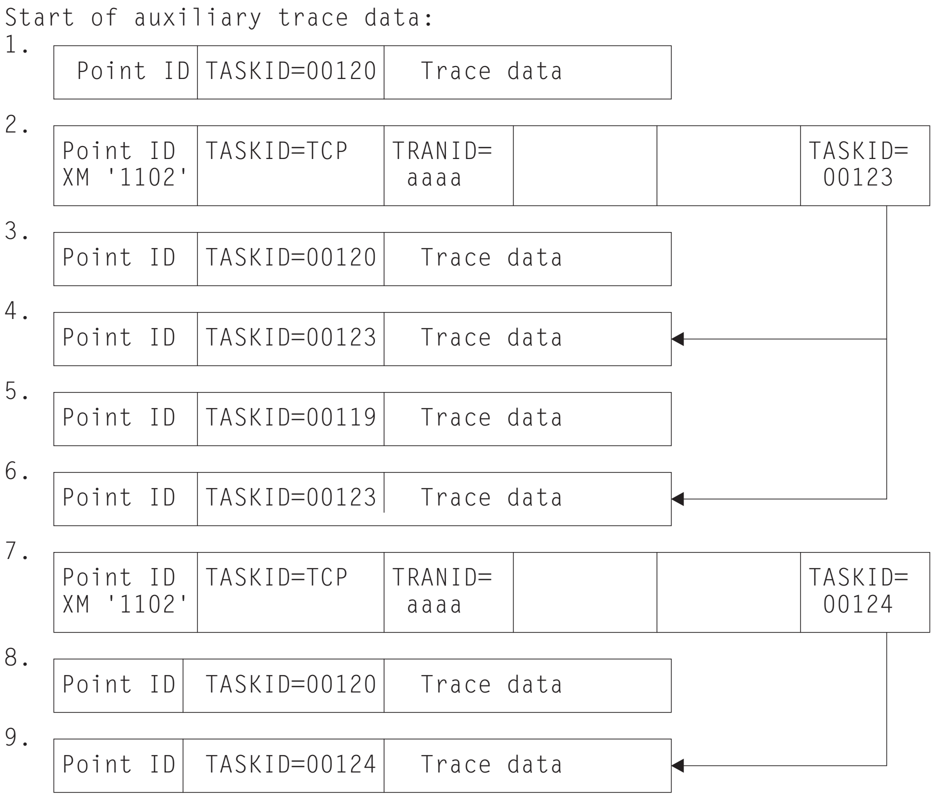
The same example could be coded in the PARM parameter as follows:

```
        col 16 v                                                col 72 v
// EXEC DFHTU410,SIZE=DFHTU410,PARM='TERMID=LP1,TRANID=(ABW,AORD,MYTR),*
               TIMERG=(123000-150000)'
```

VSE PARM values are coded up to column 71. A non-blank character must be coded in column 72, and the values must resume in column 16 of the next JCL statement.

Note: The following example would not work, because the list has been split *within* the keyword rather than between keywords:

```
TRANID=(ABRW,AORD,   [Select entries for tranids ABRW, AORD,
MYTR),                     & MYTR
```



Note: The trace point for transactions that have been task-attached is XM 1102. The trace point for transactions that have been terminal-attached is AP 1730.

Figure 24. Association of transaction-attach trace entries with task entries

Chapter 12. Dump utility programs

Information can be found under the following headings:

- “The CICS transaction dump utility program, DFHDU410”
- “Using Info/Analysis to format and analyze CICS dumps” on page 143

CICS produces two types of dump: transaction dumps that CICS writes to CICS transaction dump data sets; and VSE system dumps (produced using the VSE SDUMP macro) which are written to the VSE SYSDUMP library. CICS provides two utility programs to help you analyze these dumps. These are:

1. The transaction dump utility program, DFHDU410, for processing CICS transaction dumps. For details of this CICS batch utility program, see “The CICS transaction dump utility program, DFHDU410.”
2. The INFOANA dump exit, DFHPD410, that is invoked by INFOANA (Info/Analysis) for processing the VSE SDUMP dumps that CICS takes. DFHPD410 may also be used to analyze CICS partition dumps produced with a VSE standalone dump or with the attention routine DUMP command. However, as these partition dumps do not contain storage from outside of the CICS partition, some areas will not be formatted.

   Note: For CICS to produce an SDUMP that can be formatted by the DFHPD410 INFOANA dump exit, the following conditions must all be met:

   - The job control option SYSDUMP must be specified for the CICS job. Failure to specify the SYSDUMP option results in an unformatted dump on SYSLIST.
   - // LIBDEF DUMP statement must be active for the CICS partition. If no // LIBDEF DUMP is defined, any SDUMP requests issued by CICS fail and no dump is produced.
   - There must be sufficient space in the SYSDUMP library to hold the dump. If there is insufficient space, VSE either:
     – Suppresses the dump if the job control option SYSDUMPC is specified
     – Produces an unformatted dump on the SYSLIST.

   For further information on the SYSDUMP and SYSDUMPC job control options, refer to the *VSE/ESA System Control Statements* manual.

The CICS transaction dump utility program, DFHDU410

The output from the CICS dump domain is written to DASD or to tape, depending on which you specified when the transaction dump data sets were created and defined to CICS. The transaction dump utility program, DFHDU410, prepares the transaction dump output for printing and prints the formatted information.

Selecting transaction dump output for the DFHDU410 program

You govern the selection of the dumps that you want printed by control statements in the SYSIPT data stream. In addition to the dumps you select, the DFHDU410 program always writes a summary in the form of an index of the dumps that are on the data set being processed. The index information is taken from transaction dump header records, and is written to a separate print file addressed by logical unit, SYS009.

Format of the SYSIPT control statements

```
SELECT TYPE={OR|NOTOR|AND|NOTAND|SCAN}
[TRANID=({value|generic-value}[,value|generic-value}],.,.)]
[DUMPCODE=({value|generic-value}[,{value|generic-value}],.,.)]
[DUMPID=({value|value-range}[,{value|value-range}],.,.)]
[PAGESIZE=(value)]
[TIME=({time|time-range}[,{time|time-range}],.,.)]
[UPPERCASE=YES]
END
```

Figure 25. SYSIPT control statements for the DFHDU410 program

If SYSIPT does not contain any control statements, all dumps in the DFHDMPDS data set are printed.

Code only one SELECT statement and one END statement, but you may code multiple TRANID, DUMPCODE, DUMPID, and TIME control statements. Each control statement must be on a separate line, but can start in any column. For example:

```
SELECT TYPE=OR
DUMPCODE=(MY*)
TRANID=ABCD
END
```

The descriptions of the statements you can code in SYSIPT are as follows:

SELECT TYPE={OR|NOTOR|AND|NOTAND|SCAN}
This control statement, which is mandatory if you are specifying any of the other selection control statements, must be the first in SYSIPT. Code the TYPE parameter with one of the following selection operands:

OR Print only those dumps that match at least one of the fields defined in any TRANID, DUMPID, DUMPCODE, or TIME control statements that follow the SELECT statement. This is the default if you omit the TYPE parameter.

NOTOR Print only those dumps that do ***not*** match any of the fields defined in any TRANID, DUMPID, DUMPCODE, or TIME control statements that follow the SELECT statement.

AND Print only those dumps that match all of the fields defined in any TRANID, DUMPID, DUMPCODE, or TIME control statements that follow the SELECT statement.

NOTAND Print only those dumps that do ***not*** match the combination of the fields defined in any TRANID, DUMPID, DUMPCODE, or TIME control statements that follow the SELECT statement.

SCAN Do not print any dumps, but write only the summary, either to the DFHTINDX data set, or to the SYSLST data set if the DFHTINDX DLBL statement is missing. If you code SCAN, all other statements in the SYSIPT data stream (apart from the END statement) are ignored.

If you code any of the following control statements, they must appear in the SYSIPT data stream after a SELECT statement, and before the END statement. Each control statement must be on a separate line, but can start in any column.

TRANID=({value|generic-value}[,value|generic-value}],.,.)
Code this control statement to select dumps by their transaction identifier (ID), coding one or more four-character transaction IDs. You can code up to 20 transaction IDs on the TRANID statements; but any further transaction IDs are ignored. Code the transaction IDs either as explicit IDs, or as a generic form using plus (+) or asterisk (*) symbols as arbitrary characters. If you code a transaction ID of fewer than four characters, and without any arbitrary characters, it is assumed to be filled with trailing blanks (up to the limit of four characters for a transaction ID).

A + symbol represents any single character other than blank, and should be used to specify a single arbitrary character. For example:

TRANID=ABC
specifies a 3-character transaction ID of ‘ABC’.

TRANID=AB+
specifies a 3-character transaction ID, where the first two are ‘AB’, and the third is any character (other than blank).

TRANID=CD+F
specifies a 4-character transaction ID, where the first two are ‘CD’, the third is any character (other than blank), and the fourth is ‘F’.

An asterisk (*) symbol represents any character string not containing blanks, for example:

TRANID=XY*
specifies a transaction ID, where the first two characters are ‘XY’, the third character can be any character other than a blank, and the fourth can be any character.

All of the above examples can be coded on the following TRANID statement:

TRANID=(ABC,CD+F,XY*,AB+)

DUMPCODE=({value|generic-value}[,{value|generic-value}],.,.)
Code this to select dumps by a transaction dump code, which is either the four-character abend code or your own explicitly defined code if you requested the dump. You can code up to 20 dump codes on the DUMPCODE statement(s); but any further dump codes are ignored. Code the dump codes either as explicit codes, or as a generic form using plus (+) or asterisk (*) symbols as arbitrary characters. See the TRANID control statement for details of how to use the arbitrary character symbols.

DUMPID=({value|value-range}[,{value|value-range}],.,.)
Code this to select dumps by a 6- to 9-character dump identifier. You can code up to 10 dump identifiers or ranges of dump identifiers on the DUMPID

statement(s); but any further dump identifiers are ignored. The format of a dump identifier is xxxx/yyyy where xxxx represents the dump run number and yyyy is the dump count. You must code the slash (/) symbol as a separator character between the dump run number and the dump count.

Note: The DFHDU410 program checks only that the DUMPID operand is valid in length, and contains only numeric and / characters. If you specify a wrong numeric dump run number or dump count, or specify the wrong number of / characters, DFHDU410 will not find a matching dump.

The dump identifier operands are defined as follows:

Dump run number
A number in the range 1 to 9999. (Leading zeros are not used for this number, which is why the dump id can vary from 6 to 9 characters.) The dump run number begins at 1 when you first start CICS with a newly-initialized CICS local catalog (DFHLCD), and is incremented by 1 each time you restart CICS.

Note: The dump run number is saved in the CICS local catalog when you perform a normal shutdown, but is reset if you start CICS with a START=COLD system initialization parameter.

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

You can code the DUMPID parameter as a single value, as a range of values, or as a combination of both. If you specify a range of DUMPIDs, you must specify the lower value first. For example:

DUMPID=10/0005
This specifies a single dump identified as the fifth dump taken during dump run number 10.

DUMPID=125/0001-125/9999
This specifies all the dumps taken during dump run number 125.

DUMPID=(125/0001-125/0003,125/0019)
This specifies the first three dumps taken during dump run number 125, plus dump count number 19.

PAGESIZE=(value)
Code this parameter with the number of lines printed on a page. You can code values in the range 20 through 9999 lines per page. If you specify an incorrect value, CICS issues an error message and uses the default page size (60).

TIME=({time|time-range}[,{time|time-range}],.,.)
Code this parameter to select dumps by the time at which a dump was taken. You can code up to ten time values or range of times on the TIME statements; but any further times are ignored. Code either a time value or a range of times, or any combination of both, specifying the time in hours and minutes only, ignoring the seconds. (If CICS takes more than one transaction dump in the same minute, all dumps matching the hour and minute are selected.)

The format for time is `hh.mm` or `hh:mm`, and you specify a range of times as `hh.mm–hh.mm` or `hh:mm–hh:mm`. You must specify the hours and minutes as two digits, in the range 00 through 24 and 00 through 59 respectively.

UPPERCASE=YES
Code this parameter if you want the data output in uppercase only. The parameter must be coded as shown in uppercase characters with no spaces between words. If you want output in mixed case (the default), do not code this parameter.

END
This statement is optional and terminates the SELECT group. All statements following the END statement are ignored. If you omit the END statement, the SELECT group is terminated by the end of the SYSIPT data stream.

Job control statements to run the DFHDU410 program

The job stream to run the DFHDU410 program **should** include DLBL statements for the following data sets:

ddname Description

DFHDUMP (mandatory) The input data set, from which the dump data is to be processed.

DFHPRNT (optional) The output data set, usually a printer, to which the dump data is written. If not specified, the output data is written to SYSLIST.

DFHINDX (optional) The output data set to which the dump index summary is written.

SYSIN (optional) The input data set in which you define the control statements for the DFHDU410 program needed if you want to code specific selection parameters. Alternatively, use SYSIPT.

See Figure 26 for a sample job stream for the DFHDU410 program.

```
  // JOB DFHDU410
  // DLBL DFHDUMP,'CICS410.applid.DFHDMPA',0,SD
  // EXTENT SYS001,volser,1,0,nnn,mm
  // ASSGN SYS001,DISK,VOL=volser,SHR
# // ASSGN SYS009,SYSLST                       1
# // LIBDEF     *,SEARCH=(PRD1.BASE)
# // EXEC DFHDU410,SIZE=DFHDU410,PARM='TRANSLATE=FOLD',OS390
  /*
  /&
```

Figure 26. Sample job for DFHDU410

# **Note:** **1** // ASSGN SYS009 statement can be included if DLBL statement for
# DFHINDX (optional) has been omitted.

The PARM options of the EXEC statement are:

DOUBLE|SINGLE
For SINGLE, the transaction dump output is printed single-spaced. For DOUBLE the output is printed with a blank line between the printed lines.

NOABBREV

Prevents the trace entries from being formatted in the abbreviated format in the transaction dump.

NOFULL

Prevents the trace entries from being formatted in the full format in the transaction dump.

Trace formatting may be suppressed by using both the NOABBREV and NOFULL options. The default action is that the trace will be printed in ABBREVIATED followed by FULL formats (see the ABBREV and FULL keywords of the DFHTU410 utility program for details).

Note: ABBREV and FULL are not valid keywords of the DFHDU410 utility program.

TRANSLATE=LC|FOLD|UC

For LC, lowercase letters are printed as lowercase and uppercase letters as uppercase in the interpreted output on the right side of the dump output. For FOLD, all lowercase letters are converted to uppercase in the interpreted output. For UC, only the uppercase letters are printed in the interpreted output.

Before printing a transaction dump data set, you should close the data set. Use the CEMT SET DUMPDS SWITCH command, or the corresponding EXEC CICS command, to switch to another dump data set and close the active dump data set. You can now print the closed dump data set. You can use the CEMT SET DUMPDS AUTOSWITCH command, or the corresponding EXEC CICS SET command, to make CICS perform this switch automatically when the dump data set is full.

The contents of a transaction dump data set are never erased, but they may be overwritten when the data set is next opened for use. This happens only when:

- The data set is opened during initialization.
- You switch to the data set by using the CEMT SET DUMPDS SWITCH command, or by the corresponding EXEC CICS SET command.
- The data set is opened explicitly by the CEMT SET DUMP OPEN command, or by the corresponding EXEC CICS SET command.

Any dumps taken during the current run of CICS overwrite the contents of the data set in use.

When a dump data set is closed, an EOF marker is added after the last dump record written. If you switch from one dump data set to another without taking a dump, the EOF marker is added as the first record of the closed dump data set.

If you use the dump utility program to print a dump data set that is not in use by CICS (that is, has been opened during the current run but is now closed), any dumps written during the current run are printed (up to the EOF marker).

If you use the dump utility program to print a dump data set that is still in use by CICS (that is, before the data set has been closed), any transaction dumps written during the current run are printed. These may be followed by an unidentified partial transaction dump from a previous run, whose header has been overwritten during the current run. Any such partial transaction dumps may be followed by further transaction dumps from the previous run (up to the EOF marker from that run).

Do not use the dump utility program to print a dump data set that has not been opened during the current run of CICS. If you try to, either transaction dumps from a previous run are printed, or the program is unable to recognize the records on the data set.

Using Info/Analysis to format and analyze CICS dumps

You use Info/Analysis to produce a symptom record dump, and to call the offline CICS dump exit, DFHPD410, which produces the formatted dump.

You cannot use DFHPD410 to format dumps produced by CICS systems earlier than CICS Transaction Server for VSE/ESA Release 1.

The syntax of the CICS exit parameters is shown in Figure 27 and described in "The CICS dump exit parameters."

For some examples of using INFOANA to process CICS SDUMPs, see page 145.

```
[JOB={jobname|CURRENT}]
[UPPERCASE]

[,DEF={0|1|2|3}]
[,keyword [=levelnumber]]
 where    – keyword specifies the CICS component ID (see page 145
                for a full list of all the keywords).
          – levelnumber specifies the level of data to be output,
                either to a terminal or to a printer
(see page 145 for details of
                the available level numbers).
```

Figure 27. The CICS dump exit parameters

The CICS dump exit parameters

The CICS dump exit parameters are as follows:

UPPERCASE

This optional parameter specifies that you want the dump data output in uppercase only. If you want output in mixed case (the default), do not code this parameter.

DEF={0|1|2|3}

This parameter specifies a default level for the formatting of data from the dump data set. The DEF parameter is effective only for those components that are not included in a list of dump component keywords.

The possible levels that you can specify are as follows:

Level Meaning

0 **Suppress**

For those components not in a specified list of keywords, suppress all component formatting. If you specify DEF=0, but do not specify any component keywords, you still get the dump summary and, if appropriate, the error message index.

1 **Summary**
For those components not in a specified list of keywords, and where applicable, produce only a formatted summary from the control blocks. (A summary is not available for all components; see the level numbers available for the individual keywords for which a summary of dump information is available.)

2 **Full**
For those components not in a specified list of keywords, format all the control block information in full.

3 **Summary and full**
For those components not in a specified list of keywords, format all control blocks and (where applicable) the summary information.

The effects of omitting the DEF parameter are as follows:

- If you omit the DEF parameter and ***do not*** specify any component keywords, the result is as if you specified DEF=3. That is, DFHPD410 formats all the available summary and control block information. For the trace (TR) component, both the abbreviated and full trace are produced. The control block index is produced (see the IND component keyword) and, if appropriate, an error message index.

If you omit the DEF parameter ***and*** specify one or more component keywords, the result is as if you specified DEF=0. For example, KE=1' produces a summary of the kernel dump data, plus the dump summary and, if appropriate, an error message index, but suppresses formatting of data for all other dump components.

Exceptions to the scope of the DEF parameter

The two parts of a CICS system dump that are not governed by component keywords, and are therefore outside the scope of the DEF parameter, are:

1. The dump summary
2. The error message index

The dump summary is always formatted, even if you specify DEF=0 and no component keywords. The error message index is produced only if an error or information message is output while the CICS dump exit is formatting the dump data, even if you specify DEF=0 and no component keywords. For example:

> 'DEF=2,DS=0' suppresses formatting of the dispatcher (DS) domain; the dump summary is formatted, and all other components are formatted for level 2 only. The error message index is only produced if an error or information message is output while the CICS dump exit is formatting the dump data.

For details of the dump summary and the error message index, see page 147.

Examples of the use of level numbers

'KE=1,DS' prints a summary of the kernel domain data, and all available information for the dispatcher domain.

'DEF=2,KE=1,DS=0' prints a summary of the kernel domain data, and the control blocks for all other components except for the dispatcher domain, which is suppressed.

A job stream without any parameters produces summary and control block output for all the CICS components in the dump.

The DFHPD410 dump exit component keywords

The component keywords specify which functional areas of the CICS dump you want the DFHPD410 exit to format, and the level number operand specifies the amount of data you want formatted. If you omit all of the component keywords, and provided you have not specified DEF=0, the CICS dump exit formats dump data for all components.

The syntax of the component parameter is as follows:

component-keyword[=0|1|2|3]

Specify the component keyword from the list below. The level number operand has the same function as the level number on the DEF parameter, but with different default rules, as follows:

0 **Suppress** all output for the component.

1 **Summary** only, but available only for certain components. If you code level 1 for a component that does not have a summary, it defaults to level 0 (that is, all formatting is suppressed). Note that level 1 has a special meaning for the trace and index components, as follows:

TR If you specify level 1 for the TR (trace) component, you get the abbreviated trace only (see the ABBREV keyword on the DFHTU410 utility program for details).

IND If you specify level 1 for the IND (index) keyword, you get the control block index sorted by address.

2 **Full** control block formatting. Level 2 has a special meaning for the trace and index components, as follows:

TR If you specify level 2 for the TR (trace) component, you get full trace output (see the FULL keyword on the DFHTU410 utility program for details).

IND If you specify level 2 for the IND (index) keyword, you get the control block index sorted by name.

3 **Summary and full**, that is, both level 2 and (where available) level 1. If you code level 3 for a component that does not have a summary, it defaults to level 2.

Note: If you omit the level number, it defaults to level 3 for those components that have a summary, and level 2 for those that do not.

For details about the CICS dump component keywords, see the *CICS Problem Determination Guide*.

Keyword	***Functional area***
AI [={0|2}]	Autoinstall model manager
CC [={0|2}]	CICS catalog domain
CP [={0|2}]	CPI static storage area
CSA [={0|2}]	CICS common system area
DD[={0|1|2|}]	Directory manager domain
DF [={0|1|2|3}]	Data Management Facility (DMF)
DM [={0|1|2|3}]	Domain manager
DS [={0|1|2|3}]	Dispatcher domain
DU [={0|2}]	Dump domain
FCP [={0|2}]	File control program
ICP [={0|2}]	Interval control program
IND [={0|1|2|3}]	Page number indexes for the formatted output
JCP [={0|2}]	Journal control program
KE [={0|1|2|3}]	CICS kernel
LD [={0|1|2|3}]	Loader domain
LM [={0|1|2|3}]	Lock manager domain
ME [={0|2}]	Message domain
MN [={0|1|2|3}]	Monitoring domain
MRO [={0|2}]	CICS multiregion operation
PA [={0|2}]	Parameter manager domain
PCT [={0|2}]	Program control table
PG[={0|1|2|3}]	Program manager domain.
PR [={0|2}]	Partner resource manager (PRM) static area and table
RC [={0|2}]	Report Controller
RM [={0|2}]	Recovery manager
SM [={0|1|2|3}]	Storage manager domain
SSA [={0|2}]	Static storage areas
ST [={0|1|2|3}]	Statistics domain
SZ [={0|1}]	Front End Programming Interface[5]
TCP [={0|1|2|3}]	Terminal control program
TDP [={0|2}]	Transient data program
TI [={0|1|2|3}]	Timer domain
TMP [={0|2}]	Table manager program

[5] The SZ keyword is only applicable if you have the CICS Front End Programming Interface (FEPI) feature installed. For information about the FEPI feature, see the *CICS Front End Programming Interface User's Guide*.

TR [={0|1|2|3}] Trace domain

TRS[={<trace selection parameter(s)>}]
Trace entries within SDUMP (See “Trace entry selection.”)

TSP [={0|1|2|3}] Temporary storage program

UEH [={0|2}] User exit handler

US[={0|1}] User domain

XM [={0|1|2|3}] Transaction manager

XRF [={0|2}] Extended recovery facility

Trace entry selection: The TRS component keyword allows you to exercise much the same choice over the formatting and printing of trace entries written in a trace internal to a system dump as you may exercise over the formatting and printing of trace entries in an auxiliary trace.

Note: The TRS keyword is effective only if the TR keyword value is 1, 2, or 3.

The trace selection parameters may be any valid trace selection parameters available to DFHTU410 for the formatting of CICS auxiliary trace entries, except the parameters PAGESIZE, ABBREV, and FULL. You may, as with DFHTU410, select any number of parameters from those available. (For descriptions of available parameters, see “The trace selection parameters” on page 130.)

Note: You must use angled brackets around the parameter, or sequence of parameters, that you specify. The format and default values of parameters used to select trace entries from an internal SDUMP trace, are the same as those that apply when you use DFHTU410 to format auxiliary trace entries.

The dump summary and error index

The CICS dump exit always produces a dump summary, even if you suppress all the component areas by specifying DEF=0 without component keywords. The dump summary always appears at the head of the dump, and contains the following:

- Dump identifier
- Dump code
- Date and time at which the SDUMP was taken
- Message text associated with the dump
- Symptom string
- Dump title
- Caller
- Address space ID

Figure 28 on page 148 gives an example of a dump summary.

```
=== DUMP SUMMARY
   DUMPID:   1/0001
   DUMPCODE: AP0001
   DATE/TIME:  7/01/98 16:18:08 (LOCAL)
   MESSAGE:  DFHAP0001 applid AN ABEND (CODE 0C1/AKEA) HAS OCCURRED AT OFFSET
                                           X'00000076' IN MODULE DFHAPDM.
   SYMPTOMS: PIDS/564805400 LVLS/410 MS/DFHAP0001 RIDS/DFHAPDM PTFS/ULnnnnn
                                          AB/S00C4 AB/UAKEA ADRS/00000076
   TITLE:    (NONE)
   CALLER:   (NONE)
   ASID:     X'001D'
```

Figure 28. Example dump summary

An error message index is produced if an error or information message is output while the CICS dump exit is formatting the dump data. The error message index is organized as follows:

- It is sorted by page number, giving the page numbers on which error or information messages have been output.
- There are separate indexes for the information and error messages.
- It contains a summary giving totals of messages.

Sample jobs to process a CICS SDUMP using the CICS dump exit

The JCL required to format selected areas of a SDUMP is shown below:

```
// JOB INFOANA
// DLBL SYSDUMP,'vse.dump.library',0,SD
// EXTENT SYSnnn,volid,1,0,rtrk,ntrks
// BLNDMF ,'info.ana.dump.mgmt.file',0,SD
// EXTENT SYSmmm,volid,1,0,rtrk,ntrks
// BLNXTRN ,'info.ana.ext.rtns.file',0,SD
// EXTENT SYSyyy,volid,1,0,rtrk,ntrks
// ASSGN SYSnnn,cuu
// ASSGN SYSmmm,cuu
// ASSGN SYSyyy,cuu
// LIBDEF PHASE,SEARCH=(PRD1.BASE,IJSYSRS,SYSLIB)
// EXEC INFOANA,SIZE=1024K
DUMP NAME SYSDUMP.PROD.DF300000
SELECT DUMP VIEWING
CALL DFHPD410 DATA DEF=2,KE=1,DS=0
/*
/&
```

The dump exit component keywords are specified via the INFOANA DATA keyword.

You must use commas to separate keywords in a list. Continuation to another DATA record is allowed after any comma that separates the keywords, as long as the comma is in column 71 or is followed by a blank. A non-blank character must be coded in column 72. For example:

```
                               col 72 v
CALL DFHPD410 DATA DEF=2,             *
              DATA KE=1,              *
              DATA DS=0
```

Chapter 13. System definition file utility program, DFHCSDUP

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

This chapter describes:

- "Sharing the CSD between CICS Transaction Server for VSE/ESA Release 1 and earlier releases" on page 152
- "Input to the DFHCSDUP program" on page 152
- "Output from the DFHCSDUP program" on page 153
- "Invoking DFHCSDUP as a batch program" on page 153
- "Commands for the DFHCSDUP program" on page 156

  In the section on commands, this chapter outlines
  - The syntax of commands for the DFHCSDUP program, on page 156
  - DFHCSDUP command processing following an internal error, on page 157

  This chapter then describes the commands that allow you to use the DFHCSDUP program to:
  - ADD a group to the end of a named list in a CSD file (see page 158).
  - ALTER attributes of an existing resource definition (see page 159).
  - APPEND a group list from one CSD file to a group list in another, or in the same, CSD file (see page 161).
  - COPY all of the resource definitions in one group or several generically named groups to another group or several other generically named groups in the same, or in a different, CSD file (see page 163).
  - DEFINE a single resource, or a group of resources, on the CSD (see page 165).
  - DELETE from the CSD a single resource definition, all of the resource definitions in a group, or all of the group names in a list (see page 167).
  - EXTRACT data from the CSD and pass it to a user program for processing (see page 169).
  - INITIALIZE a new CSD file, and add to it CICS-supplied resource definitions (see page 171).
  - LIST selected resource definitions, groups, and lists (see page 172).
  - MIGRATE the contents of a table from a CICS load library to a CSD file (see page 174).
  - REMOVE a single group from a list on the CSD file (see page 177).
  - SERVICE a CSD file when necessary (see page 178).
  - UPGRADE the CICS-supplied resource definitions in a primary CSD file for a new release of CICS (see page 179).
  - VERIFY a CSD file by removing internal locks on groups and lists (see page 182).

The DFHCSDUP program is invoked as a batch program (see page 153). It cannot be invoked from a user program.

Sharing the CSD between CICS Transaction Server for VSE/ESA Release 1 and earlier releases

If you want to share the CSD between CICS regions at different release levels, to enable you to share common resource definitions, you must update the CSD from the higher level region, that is CICS Transaction Server for VSE/ESA Release 1.

In CICS Transaction Server for VSE/ESA Release 1, some attributes are obsolete, and are removed from the CSD definitions. Using the ALTER command on definitions that specify obsolete attributes does not cause the loss of these attributes in CICS Transaction Server for VSE/ESA Release 1, so you can safely update resource definitions from a CICS Transaction Server for VSE/ESA Release 1 region.

If you are sharing the CSD between a CICS Transaction Server for VSE/ESA Release 1 region and a CICS for VSE/ESA Version 2.3 region, you can use the CICS Transaction Server for VSE/ESA Release 1 CSD utility, DFHCSDUP, to update resources that specify obsolete attributes. A compatibility option is added for this purpose, which you must specify on the PARM parameter on the EXEC DFHCSDUP statement. You indicate the compatibility option by specifying COMPAT or NOCOMPAT. The default is NOCOMPAT, which means that you cannot update obsolete attributes. (See Figure 30 on page 153.)

The *CICS Migration Guide* discusses sharing your CSD between different releases. It also summarizes obsolete attributes and discusses compatibility groups.

Note: You cannot use the EXTRACT command of the CICS Transaction Server for VSE/ESA Release 1 DFHCSDUP utility when the COMPAT option is specified.

Input to the DFHCSDUP program

Input to the DFHCSDUP program (see Figure 29 on page 153) is from:

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

Output from the DFHCSDUP program

The result of running the DFHCSDUP program (see Figure 29) may be an updated primary file, or a print file.

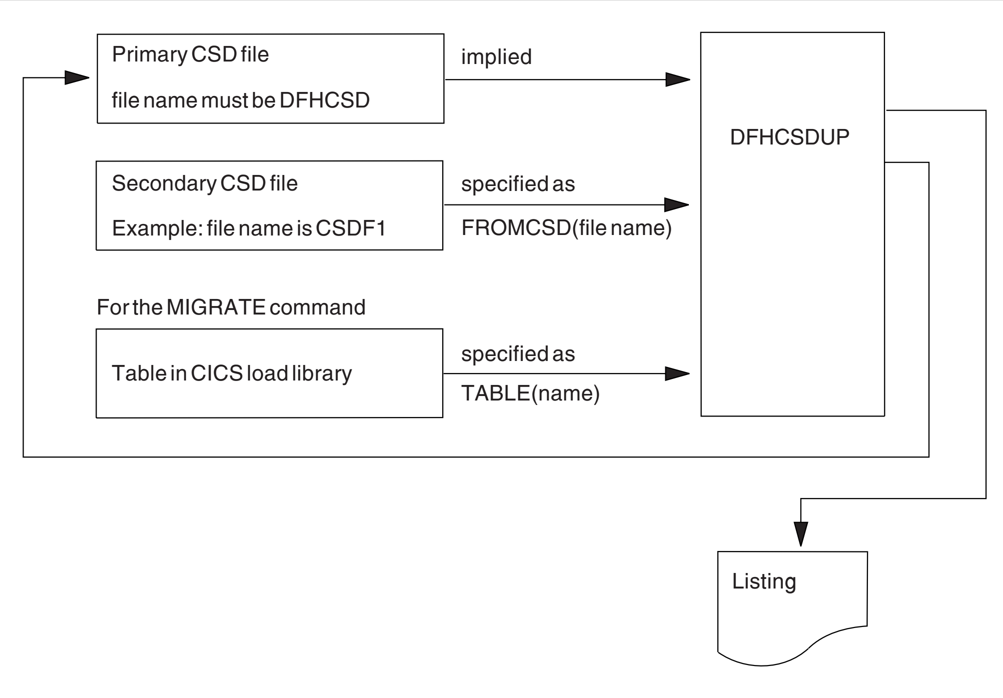


Figure 29. The DFHCSDUP offline utility program

Invoking DFHCSDUP as a batch program

The job in Figure 30 shows you an example of the job control statements you can use to invoke DFHCSDUP as a batch program.

```
// JOB CSDJOB   Job to execute DFHCSDUP
// DLBL usercat,'usercat',,VSAM
// DLBL DFHCSD,'user.applid.dfhcsd',,VSAM, CAT=usercat
// DLBL csdfll,'user.applid.dfhcsd2',,VSAM  1
// LIBDEF PHASE,SEARCH=PRD1.BASE, 2
*
*
// EXEC DFHCSDUP,SIZE=DFHCSDUP,PARM=COMPAT  3
        .
     DFHCSDUP commands
        .
/*
/&
```

Figure 30. Sample job to run DFHCSDUP

Notes:

1 Include the *// DLBL csdfll* statement if you are using a second CSD.

2 If you run DFHCSDUP with the MIGRATE or EXTRACT command, you may have to alter the LIBDEF PHASE search chain to include the sublibrary containing

your CICS PHASE tables (MIGRATE command), or user programs (EXTRACT command).

In addition, if you compile your user programs with the COBOL RES option, you also need to include the COBOL run-time libraries in the search chain. For example,

```
// LIBDEF PHASE,SEARCH=(cicstab.libname,userprog.libname,PRD1.BASE,PRD2.DBASE)
```

If you are using DFHCSDUP with the EXTRACT command, you must include the name of the sublibrary containing your user programs in the SEARCH chain. For example,

```
// LIBDEF PHASE,SEARCH=(PRD1.BASE,user.libname)
```

3 DFHCSDUP is approximately 280K in size.

However, for the:

- MIGRATE command, the table to be migrated is loaded into GETVIS space. Message DFHCA5148 is issued if insufficient storage is available to load a table.
- EXTRACT command, all USERPROGRAMs are loaded into GETVIS. Message DFHCA5282 is issued if insufficient storage is available to load a user program.

Therefore ensure that the partition you run the job in is large enough to provide sufficient program storage for the DFHCSDUP utility program itself while, at the same time, leaving enough GETVIS space for the VSAM-managed CSD, and any tables to be migrated or EXTRACT USERPROGRAM's.

If you specify the EXTRACT command, you might need to include a DLBL statement for any input data set that is defined in your user program. For example, the CICS-supplied user programs, DFH$CRFA, DFH$CRFP, and DFH0CRFC, need DLBL statements with a file name of CRFINPT. This input file is needed to supply the list of attribute keywords for which you want a cross reference listing.

You can use the sample job shown in Figure 31 to create the input file, CRFINPT, for the DFH$CRFA, DFH$CRFP, and DFH0CRFC samples.

```
// JOB INFOSET
// DLBL CRFINPT,'crfinput.data',0,SD
// EXTENT SYSnnn,volid,1,0,rtrk,ntrks
// ASSGN SYSnnn,DISK,VOL=volid,SHR
// UPSI 1
// EXEC DITTO
$$DITTO SET HEADERPG=NO
$$DITTO CS FILEOUT=CRFINPT,RECFMOUT=F,BLKSIZE=80
TERMINAL
LANGUAGE
/*
$$DITTO EOJ
/*
/&
```

Figure 31. Sample job to create input file, CRFINPT

Include a DLBL statement for any ouput data sets that receive output from your EXTRACT user program. The sample programs available in CICS Transaction

Server for VSE/ESA Release 1 need DLBL statements for the data set names shown in Table 16.

Table 16. DLBL statements required by EXTRACT sample programs

Program name	Required DLBL and EXTENT
DFH0CRFC	// DLBL CRFOUT,'name',0,SD // EXTENT SYS003,volid,1,0,rtrk,ntrks
DFH0CBDC	// DLBL CBDOUT,'name',0,SD // EXTENT SYS002,volid,1,0,rtrk,ntrks

Sample programs DFH$CRFA and DFH$CRFP direct their output to SYSLST, therefore they do not need DLBL statements to define the output destination.

The output data sets in these examples are opened and closed for each EXTRACT command specified in SYSIPT. For programming information about the CICS-supplied user programs, see the *CICS Customization Guide*.

Note that the PARM parameter can be used to specify any of the following options:

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

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

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

NOCOMPAT or **COMPAT** specifies whether the DFHCSDUP utility program is to run in compatibility mode (that is, whether it can update definitions that are obsolete in CICS Transaction Server for VSE/ESA Release 1). The default is NOCOMPAT, which means that you cannot update obsolete attributes. For further information about this option, see “Sharing the CSD between CICS Transaction Server for VSE/ESA Release 1 and earlier releases” on page 152.

Syntax

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

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

Commands for the DFHCSDUP program

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

Rules for the syntax and preparation of commands

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

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

```
ALter Connection(name) Group(groupname)
```

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

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

Blank records between commands are ignored.

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

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

An example of a valid sequence of commands is shown in Figure 32 on page 157. Other examples of commands are given in the command descriptions that follow.

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

Figure 32. Sample commands of the DFHCSDUP program

Command processing following internal error detection

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

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

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

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

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

ADD a group to a list

```
►►──ADd──Group(groupname1)──LIst(listname)──►◄
```

The ADD command adds a group to a list.

Group(groupname1)
The name of the group to be added. The name must not already exist in the list. A generic group name is not accepted. If you don't specify a group, the current group name is added.

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

Examples of the ADD command

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

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

Add another group to list LA01:

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

LA01 now looks like this:

GA001
GA002

ALTER a resource definition

```
►►──ALter──┬─Connection(name)───┬──Group(groupname)────────────────────►
           ├─File(name)─────────┤
           ├─Lsrpool(name)──────┤
           ├─Mapset(name)───────┤
           ├─PARTItionset(name)─┤
           ├─PARTNer(name)──────┤
           ├─PROFile(name)──────┤
           ├─PROGram(name)──────┤
           ├─Sessions(name)─────┤
           ├─TErminal(name)─────┤
           ├─TRANClass(name)────┤
           ├─TRANSaction(name)──┤
           └─TYpeterm(name)─────┘

►──attribute list(new value)──►◄
```

The ALTER command changes some or all of the attributes of an **existing** resource definition. It accepts both generic resource names and group names.

For information about the attributes that you can specify on the ALTER command for the various resource types, and for a description of the attributes and default values of each resource type, see the *CICS Resource Definition Guide*.

Resource(name)
The resource whose attributes you want to alter. You can specify a generic name by using the characters **+** or *****

Group(groupname)
The name of the group containing the resource to be altered. You can specify a generic name by using the characters **+** or *****

Attribute list
The attributes to be altered.

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

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

  If a keyword, for which you have specified a null value, has a default, the default value is used. If a keyword does not have a default value, the definition acts as if the keyword has never been specified. In this example, if you list the resource definition for file TEST, it is shown with DESCRIPTION().
- Changes to resource definitions in the CSD do not take effect until you install the group in which the resource definition resides.

Examples of the DFHCSDUP ALTER command

To make a program resident:

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

To make all the programs in group GENMOD resident:

```
ALTER PROGRAM(*) GROUP(GENMOD) RESIDENT(YES)
```

APPEND a list to another list

```
►►──APpend──List(listname1)──To(listname2)──FRomcsd(dlblname)──►◄
```

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

List(listname1)
Specifies the name of the list that is appended. Do not use a generic list name.

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

To(listname2)
Specifies the name of the list to which you want the group names appended. If you are appending from another CSD, you can give this list the same name as the one you are appending from. Do not use a generic list name.

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

FRomcsd(dlblname)
Identifies the DLBL of the secondary CSD from which you are appending listname1.

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

Example of the APPEND command

A list called LISTA contains the following groups:

GB001
GB002
GB003

A list called LISTB contains the following groups:

G001
G002
G003

Append LISTB to LISTA, like this:

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

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

GB001
GB002
GB003
G001
G002
G003

and LISTB still contains:

G001
G002
G003

COPY a resource definition

```
►►──Copy──Group(groupname1)──To(groupname2)──┬─────────┬──────────────────►
                                             ├─Replace─┤
                                             └─MErge───┘
►──FRomcsd(dlblname)──►◄
```

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

Group(groupname1)
Specifies the name of the group to be copied. You can specify a generic name by using an asterisk (*). See “Generic naming in the COPY command” on page 164 for details.

To(groupname2)
Specifies the name of the group to which the definitions are copied. If you are copying from another CSD, you can give this group the same name as the one you are copying from. You can specify a generic name by using an asterisk (*). See “Generic naming in the COPY command” on page 164 for details.

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

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

FRomcsd(dlblname)
Identifies the DLBL of the secondary CSD from which you are copying groupname1.

The COPY command copies all the resource definitions in *groupname1* to *groupname2*. The group to be copied (*groupname1*) can be on the primary CSD, or it can be on the CSD specified by the FROMCSD parameter.

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

Generic naming in the COPY command

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

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

You can use the asterisk (*) symbol to copy from generically-named groups to other generically-named groups or from generically-named groups to a specific group, as shown in Figure 33.

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

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

Examples of the COPY command

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

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

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

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

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

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

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

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

Figure 33. Examples of the COPY command

DEFINE a resource definition

```
►►──DEFine──┬─Connection(name)───┬──Group(groupname)──────────────►
            ├─File(name)─────────┤
            ├─LSRpool(name)──────┤
            ├─Mapset(name)───────┤
            ├─PARTItionset(name)─┤
            ├─PARTNer(name)──────┤
            ├─PROFile(name)──────┤
            ├─PROGram(name)──────┤
            ├─Sessions(name)─────┤
            ├─TErminal(name)─────┤
            ├─TRANClass(name)────┤
            ├─TRANSaction(name)──┤
            └─TYpeterm(name)─────┘

►──┬────────────────┬──►◄
   └─Attribute list─┘
```

The DEFINE command creates new resource definitions.

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

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

Attribute list
The attribute list depends on the resource type being defined; some resources have attributes that must be included in the definition. For a description of the attributes and default values of each resource type, Attributes that you do not specify are given default values.

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

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

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

Examples of the DEFINE command

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

```
    DEFINE TERMINAL(CON0)     GROUP(CONTERMS)
           CONSNAME(SYS1)      TYPETERM(DFHCONS)
           DESCRIPTION(VSE MASTER CONSOLE)

    DEFINE TERMINAL(CON1)     GROUP(CONTERMS)
           CONSNAME(USER1)     TYPETERM(DFHCONS)
           DESCRIPTION(ICCF CONSOLE)
```

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

Figure 34. Examples of the DEFINE command

DELETE a resource definition, group, or list

```
►►──DELete──┬─List(listname)───────────────────────────┬──►◄
            └─Group(groupname)──┬────────────────────┬─┘
                                ├─All(name)──────────┤
                                ├─Connection(name)───┤
                                ├─File(name)─────────┤
                                ├─Lsrpool(name)──────┤
                                ├─Mapset(name)───────┤
                                ├─PARTItionset(name)─┤
                                ├─PARTNer(name)──────┤
                                ├─PROFile(name)──────┤
                                ├─PROGram(name)──────┤
                                ├─Sessions(name)─────┤
                                ├─TErminal(name)─────┤
                                ├─TRANClass(name)────┤
                                ├─TRANSaction(name)──┤
                                └─TYpeterm(name)─────┘
```

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

List(listname)
Specifies the name of the list to be deleted. Do not use a generic list name.

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

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

This parameter can be used only with the GROUP keyword.

Deleting a resource definition is different from removing a group from a list (see "REMOVE a group from a list" on page 177). A deleted resource definition really does disappear from the CSD.

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

When a group is deleted, the group is removed from all lists that contained it unless you were running UPGRADE.

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

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

Example of the DFHCSDUP DELETE command

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

GB001
GB002

Group GB001 contains the following resource definitions:

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

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

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

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

```
DELETE GROUP(GB002)
```

This leaves only group GB001 in the group list LISTA.
The following command deletes all group names in the group list LISTA:

```
DELETE LIST(LISTA)
```

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

Figure 35. Examples of the DELETE command

EXTRACT resource definition data

```
►►──EXtract──┬─Group(groupname)─┬──────────────────────────────────────►
             └─LIst(listname)───┘

►──┬─USerprogram(DFHxCRFy) Objects─────────────────┬──►◄
   ├─USerprogram(DFH0CBDC)─┬─────────┬─────────────┤
   │                       └─Objects─┘             │
   └─USerprogram(user-written program)─┬─────────┬─┘
                                       └─Objects─┘
```

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

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

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

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

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

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

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

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

CICS supplies two types of sample user program: DFHxCRFy and DFH0CBDC. The letter x in the program name is $ for assembler or PL/I and 0 for COBOL. The letter y in the program name denotes the programming language, where y=A is the assembler version, y=C is the COBOL version, and y=P is the PL/I version. Note that DFH0CBDC is supplied as COBOL only. All other user programs are available in source form in the VSE/ESA sublibrary PRD1.BASE, and the assembler versions are also available in pregenerated form in the VSE/ESA sublibrary PRD1.BASE.

Example of the EXTRACT command

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

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

Figure 36. Example of the EXTRACT command

INITIALIZE a new CSD with the required CICS resource definitions

```
►►──INITialize──►◄
```

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

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

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

LIST resource definitions

```
          ┌─All─┐
►►──LIst──┼─────┴────────────┬──┬─────────┬──►◄
          ├─Group(groupname)─┤  └─Objects─┘
          └─LIst(listname)───┘
```

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

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

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

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

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

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

- LIST ALL

  – Names of defined lists and groups
  – Summary of lists
  – Summary of groups

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

- LIST ALL OBJECTS

  – Names of defined lists and groups
  – Summary of lists
  – Summary of groups
  – Objects in groups

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

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

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

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

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

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

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

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

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

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

MIGRATE macro-defined resource definitions from tables to the CSD

```
►►──MIgrate──TAble(tablename)──┬──────────────────────────────┬──►◄
                               └─TYpesgroup(typesgroupname)─┘
```

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

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

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

The contents of a table are transferred as one group, or as a set of several groups, containing definitions. When migrating large tables, make sure you allocate a sufficiently large region for the largest table to be loaded.

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

  MIgrate TAble(tablename)

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

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

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

- **To transfer a PCT or PPT** , the format is:

  MIGRATE TABLE(tablename) [TOGROUP(groupname)]

  The result is a set of groups containing, for a PCT, transaction and profile definitions or, for a PPT, program, map set, and partition set definitions. You can specify each group by the macro:

  ```
  DFHPCT|DFHPPT TYPE=GROUP,GROUP=xxxxxxxx
  ```

  which you insert in the PCT or PPT source instructions before you assemble them for migration. All definitions after such a TYPE=GROUP macro, up to the next TYPE=GROUP macro, go into the group named by `GROUP=xxxxxxxx`. Definitions that occur before the first such TYPE=GROUP macro are migrated to the default group. You can also specify that definitions are to be migrated to the default group by inserting the following macro in the PCT or PPT before the definition entries:

  ```
  DFHPCT|DFHPPT TYPE=GROUP,GROUP=*DEFAULT
  ```

  You can use the TOGROUP parameter of the MIGRATE command to assign a specific name to the default group. If you do not specify TOGROUP, the name of the default group is derived from the table name. For example, if the migrated table name is DFHPPT24, the name of the group created is PPT24.

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

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

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

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

The result is:

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

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

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

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

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

   The existing TYPETERMs checked are:

   – TYPETERMs in the GROUP currently being created

   – TYPETERMs in the group specified in the TYPESGROUP parameter of the MIGRATE command.

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

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

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

Example of the MIGRATE command

```
An FCT, DFHFCTS$, from CICS for VSE/ESA Version 2.3 contains the following
definitions:
     DFHFCT TYPE=FILE,ACCMETH=VSAM,FILE=DDNAME1,JREQ=ALL
     DFHFCT TYPE=FILE,ACCMETH=VSAM,FILE=DDNAME2,JREQ=ALL
     DFHFCT TYPE=FILE,ACCMETH=VSAM,FILE=DDNAME3,JREQ=ALL
```

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

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

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

```
MIGRATE TABLE(DFHFCTS$)
```

Figure 37. Examples of the MIGRATE command

REMOVE a group from a list

```
►►──Remove──LIst(listname)──Group(groupname)──►◄
```

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

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

Group(groupname)
The name of the group to be removed. Do not use a generic group name.

Example of the REMOVE command

A list LL02 contains the following groups:

G001 G002 G003 G004

To remove group G003:

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

This leaves:

G001 G0023 G004

Figure 38. Examples of the REMOVE command

SERVICE a CSD

```
►►──Service──FRomcsd(dlblname)──LEvel(nnn)──►◄
```

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

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

FRomcsd(dlblname)
Specifies the DLBL of the current CSD, which for the purposes of the command is treated as the secondary CSD.

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

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

UPGRADE a CSD

```
►►──UPgrade──┬──────────────────┬──►◄
             └─USing(filename)─┘
```

You can use the UPGRADE command to change the CICS-supplied resource definitions in a primary CSD. If your existing CSD was initialized at CICS for VSE/ESA Version 2.3 or earlier, you must UPGRADE it using DFHCSDUP to obtain the RDO extensions in CICS Transaction Server for VSE/ESA Release 1. Upgrading ensures that the definitions in the DFH-groups are brought up to the level of CICS Transaction Server for VSE/ESA Release 1 function.

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

USing(filename)

Upgrading a CSD for CICS Transaction Server for VSE/ESA Release 1 does not require you to use the USING operand. All IBM-supplied resource definitions from **any** release are deleted and then the CSD is initialized, so you do not need to say which release you came from. However, UPGRADE USING(filename) is used to install IBM optional functions or features onto CICS. For example, UPGRADE USING(DFHCURCF) is used to place the report controller definitions onto the CSD.

USERDEFINE a resource definition

```
►►──USerdefine──┬─Connection(name)───┬──Group(groupname)──────────────────►
                ├─File(name)─────────┤
                ├─Lsrpool(name)──────┤
                ├─Mapset(name)───────┤
                ├─PARTItionset(name)─┤
                ├─PARTNer(name)──────┤
                ├─PROFile(name)──────┤
                ├─PROGram(name)──────┤
                ├─Sessions(name)─────┤
                ├─TErminnal(name)────┤
                ├─TRANClass(name)────┤
                ├─TRANSaction(name)──┤
                └─TYpeterm(name)─────┘

►──attribute list(newvalue)──►◄
```

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

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

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

You could have the following resources in your USERDEF group:

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

Attribute list
The attribute list depends on the resource type being defined, some resources have attributes that must be included in the definition. See the *CICS Resource Definition Guide* for a description of the attributes and default values of each resource type. Attributes that you do not specify are given default values.

Group(groupname)
The name of the group to be defined.

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

Examples of the USERDEFINE command

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

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

```
DEFINE PROGRAM(USER) GROUP(USERDEF) LANGUAGE(ASSEMBLER)
```

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

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

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

By renaming your USER to something else and defining your own dummy resource definition, you can use your own default values.
Normally, however, your installation probably agrees on default values for standardization reasons, and puts a LOCK on the USERDEF GROUP.

VERIFY a CSD

```
►►──VERIFY──►◄
```

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

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

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

Chapter 14. Identify macro-level programs utility program, DFHMSCAN

To convert your CICS applications to command level, you first have to identify your macro-level programs. To help you do this, CICS provides the program DFHMSCAN to scan a load module library and identify programs that use CICS macros.

This chapter describes:

- How to use DHMSCAN to produce a summary report (see page 183)
- How to use DHMSCAN to produce a detailed report (see page 184)
- Restrictions on using DFHMSCAN (see page 184)

You need the JCL shown in Figure 39 to invoke DFHMSCAN.

```
// JOB DFHMSCAN
// ASSGN SYS001,printer-name
// ASSGN SYS002,printer-name
// LIBDEF PHASE,SEARCH=sublib.list  1
// EXEC DFHMSCAN,PARM='parameters'  2
```

Figure 39. JCL to invoke the DFHMSCAN utility program

Notes for Figure 39:

1 sublib.list is the sublibrary to be scanned. This must also include the sublibrary where DFHMSCAN itself resides, somewhere in the chain.

2 `parameters` are either "`specified.sublibrary,$SUMMARY`" or "`specified.sublibrary,name1,name2...,name20`"

The `specified.sublibrary` parameter must match the primary search value on the LIBDEF statement. If `$SUMMARY` is specified, it must be the second and final parameter. For specified names, do not include the ".PHASE" extension.

Return code 99 is given for parameter errors. If LIBR does not load successfully, the load return code is returned.

Summary report

If you specify `PARM='$SUMMARY'`, DFHMSCAN:

1. Scans each module in the load library for BALR (assembler-language branch and linkon register) 14,14 and BALR 14,15 instructions.
2. Analyzes the instructions preceding identified BALR instructions, to see if they match sequences used by CICS macro requests or EXEC CICS commands.
3. Prints, for each module in the scanned library, its name, size, language (if determined), the number of CICS macro-level statements, the number of CICS command-level statements, and the number of unrecognized BALR instructions.
4. Prints the total number of modules in the library and the number of macro-level programs of each type (assembler-language, COBOL, and PL/I).

Detailed report

If you specify `PARM='name1,name2,...'`, DFHMSCAN scans the named modules and for each module:

1. Prints a line for each BALR found, giving its offset from the start of the module, some of the code that precedes it, and what it appears to be.
2. For macro-level statements, attempts to identify the macro type.

Restrictions

DFHMSCAN:

- Does not scan CICS modules and tables in the load library.
- Does not separately identify CHECK macros.
- Cannot identify forms of the DFHBIF macro that do not produce a BALR, or that produce code indistinguishable from that generated by EXEC CICS commands.
- DFHMSCAN cannot find all EXEC CICS commands for C, COBOL and PL/I, because the code depends on the compiler. The main purpose of DFHMSCAN is to find macro-level programs.
- Can only find code patterns that are similar to those generated by CICS macros. A module can contain such code without having a CICS macro in its source.

Chapter 15. Resource manager in-doubt resolution program, DFH$INDB

This chapter contains an overview of DFH$INDB on page "Overview of the DFH$INDB utility" and explains possible inconsistencies between the report produced by DFH$INDB and actual in-doubt records (see page 186), and information on:

- "JCL to assemble and link-edit the DFH$INDB utility" on page 186
- "JCL to run the DFH$INDB utility" on page 186
- "Return codes from the DFH$INDB utility" on page 187
- "Source code for the DFH$INDB utility" on page 187

For an example of the output from DFH$INDB, see page 187

You can use the sample program DFH$INDB in recovery situations to resolve manually any incomplete updates to external resource records about which the resource manager, such as DB2® for VSE (SQL), is in doubt whether to commit or back out. It produces a list of the in-doubt external resource records, and the action needed to resolve each update.

This manual resolution is needed if CICS is cold-started after the resource manager is abnormally disconnected from CICS.

You must run the DFH$INDB utility using the set of system logs, starting with the system log in use when CICS connected to the resource manager and ending with the system log in use when CICS disconnected abnormally from the resource manager. However, ***do not*** include system logs that are used for the subsequent cold start of CICS. This means that you may have to run the DFH$INDB utility ***before*** you cold start CICS (for example, if your JCT options mean that the system log is reused on the cold start, beginning at the point where the previous CICS run ended). In any case, you should run the DFH$INDB utility before CICS reconnects to the resource manager. You should use the report produced by the DFH$INDB utility to resolve the in-doubt updates manually by using operator commands before you reconnect CICS to the resource manager.

Overview of the DFH$INDB utility

The DFH$INDB utility is a batch program that reads a series of archived CICS system log data sets and, for a given resource manager, produces a list of in-doubt records and the actions required to resolve them. One line is produced on the report for each in-doubt record, giving the:

- Recovery token
- Name of the CICS transaction being used
- CICS task number
- ID of the terminal on which the transaction was running
- ID of the operator running the transaction
- Time at which the commit processing for the update began
- Action required to resolve the in-doubt.

Inconsistencies between the report and in-doubt records

Most in-doubt records for a given resource manager appear on the report produced by the DFH$INDB utility. However, in two cases, a resource manager can be in doubt about a record, but for there to be no corresponding entry in the report produced by the DFH$INDB utility. These two cases are:

1. The in-doubt record may have been created before the previous cold start of CICS. In this case, you must run the DFH$INDB utility against the archived copies of the logs used prior to the cold start to determine the correct action to resolve the in-doubt record.
2. The resource manager completes its prepare process before CICS writes a 'prepare' record to its log. Therefore, if the resource manager disconnects abnormally from CICS at this point, the resource manager is in doubt, but CICS has no record of it. The correct resolution in this case is always to back-out.

The DFH$INDB utility can also list in-doubt records that the resource manager does not know about. This can occur if CICS is disconnected abnormally from the resource manager in the time between the resource manager resolving the in-doubt record and CICS writing the 'forget' record to its log. It can also occur if the DFH$INDB utility is run after resource manager operator commands have been issued to resolve in-doubt records. In these cases, no further action is required.

JCL to assemble and link-edit the DFH$INDB utility

Before you can run the DFH$INDB utility, you must assemble and link-edit it into your CICS environment. You should use your own JCL to assemble and link-edit the DFH$INDB utility.

A sample job stream is given in Figure 40.

```
// JOB INDBJOB
// OPTION CATAL,NODECK,ALIGN
// LIBDEF    *,SEARCH=(cicsusr.source,PRD1.BASE),CATALOG=cicsusr.phase
   PHASE DFH$INDB,*
   MODE AMODE(24),RMODE(24)
// EXEC ASMA90
  ⋮
      DFH$INDB source statements
  ⋮
/*
// EXEC LNKEDT
/*
```

Figure 40. Sample job to assemble and link-edit the DFH$INDB utility

JCL to run the DFH$INDB utility

Figure 41 on page 187 gives a sample job stream to run the DFH$INDB utility. Specify the name of the resource manager, for which you wish to produce the listing of in-doubt records, in the PARM field of the EXEC statement. The resource manager name must match the value in the field JCSRMNME in the log record.

The job stream to run the DFH$INDB utility should include DLBL statements for the following data sets:

DLBL name **Defines**

INPUT The archived copies of the CICS system logs that are to be read. Start the sequence of statements for the archived logs with the copy of the log that was in use at the time CICS connected to the resource manager. Include, in chronological order, a statement for a copy of every log data set up to, and including, the log data set that was in use when CICS disconnected abnormally from the resource manager.

OUTPUT The data set that the listing of in-doubt records is written to, is usually defined to SYSOUT.

```
// JOB DFH$INDB
// LIBDEF     *,SEARCH=(PRD1.BASE,user.sublib)
// DLBL INPUT,'CICS410.LOGA.COPY1',0,SD
// EXTENT SYS001,volser,1,0,nnn,mm
// ASSGN SYS001,DISK,VOL=volser,SHR
// EXEC DFH$INDB,SIZE=4M,PARM='rmname'
/*
/&
```

Figure 41. Sample job stream to run the DFH$INDB utility

Example of output from the DFH$INDB utility

Figure 42 shows an example of the output from the DFH$INDB utility.

```
       IN-DOUBT IDENTIFICATION REPORT FOR RESOURCE MANAGER rmname
LATEST COLD START TOKEN - A17C1A05FEE07100
RECOVERY TOKEN   TRANID TASK_NUM TERMID OPID   FAIL TIME  RESOLUTION ACTION
A17D7A9BBA150401  DLID        31  2B77  0001   12.56.41     COMMIT
A17D6A14FD4D6A01  DLID        35  1E77  0057   11.40.44     BACK OUT
A17D697ED8CB0101  DLUP        32  2D77  X999   11.37.23     COMMIT
 END OF REPORT
```

Figure 42. Example report output by the DFH$INDB utility

Return codes from the DFH$INDB utility

Code **Meaning**

0 Successful completion.

4 Resource manager name not specified.

8 Supplied resource manager name is more than 8 characters.

Source code for the DFH$INDB utility

For more information about the DFH$INDB utility, or to create your own version, see the source code in the VSE/ESA sublibrary, PRD1.BASE.

Chapter 16. In-doubt window resolution utility, DFH$IWUP

This chapter describes the utility program, DFH$IWUP, which you can use to help determine the resources that have been changed by transactions using ISC or MRO for tasks that are considered to have been in-doubt after a CICS region failure.

This chapter:

- Explains how an in-doubt window can arise (see “Overview of in-doubt windows”)
- Provides an overview of how DFH$IWUP can help to resolve in-doubt windows (see “Overview of the in-doubt window resolution utility program, DFH$IWUP” on page 190)
- Explains the limitations of the DFH$IWUP program (see “Limitations of the DFH$IWUP utility program” on page 190)
- Gives an example of the job control statements needed to prepare DFH$IWUP for running in your CICS environment (see “Job control statements to assemble and link-edit the DFH$IWUP program” on page 191)
- Gives an example of the job control statements needed to run DFH$IWUP (see “Job control statements to run the DFH$IWUP program” on page 192)
- Describes the error messages produced by the DFH$IWUP utility program (see “Error messages produced by the DFH$IWUP program” on page 196)
- Outlines how to produce your own version of the DFH$IWUP utility program (see “Creating your own DFH$IWUP utility program” on page 197)

Overview of in-doubt windows

If a CICS region fails during a syncpointing process, other CICS regions with which it communicates do not know whether the failed CICS region has backed out or committed its updates. The period from the last syncpoint to the time of failure is known as the **in-doubt window**. For this period, the resource data is potentially inconsistent between the failed CICS region and other CICS regions that also use the same resources.

The danger is that after the failed CICS region has restarted, new transactions can use the inconsistent data; they may read or update the data, and in either case the results of these new transactions can be corrupted.

If a CICS region fails, it issues message DFHZN2101 to identify any units of work, relating to specific tasks, which are considered to be in-doubt. When the CICS region recovers from the failure, it issues further messages to resolve the in-doubt tasks to “In sync” or “Out of sync”. (The latter case is reported by message DFHZN2103.) The messages indicate that data integrity may be exposed, but they do not indicate which resources and records are exposed. Note that only CICS resources are reported upon; no other resource types are reported upon.

Overview of the in-doubt window resolution utility program, DFH$IWUP

The in-doubt window resolution utility program, DFH$IWUP, helps you to resolve in-doubt resource updates.

The DFH$IWUP utility program is an exit program to the CICS journal utility program, DFHJUP. It takes as input a list of unit-of-work IDs as they appear in the DFHZN210x messages generated to the CSMT log.

Note: You can copy the CSMT log and edit it to produce your own list of unit-of-work IDs to be investigated.

The DFH$IWUP utility program scans the journal, and outputs a report of all journal records for tasks relating to the unit-of-work IDs. The report lists, by task, for each journal record found:

- The resource type and type of request.

  This enables you to differentiate spurious messages from significant ones. (Spurious warning messages of in-doubt problems are issued by CICS regions that have not updated any resources.)

- The user data, including resource name and record key (where appropriate).

  This helps you to investigate the messages that report loss of integrity, enabling you to quantify the effect of the failure.

To resolve the in-doubt resource update operations, you must interpret the journal data to determine whether or not there has been a loss of integrity.

You can use the DFH$IWUP utility program to resolve:

- DFHZN2103 messages (for databases out of synchronization) after CICS has reported the result of its recovery logic
- Any units of work whose outcome is unknown after a recovery failure
- Operator action which has prevented CICS from resolving DFHZN2101 messages.

You should run the DFH$IWUP utility against copies of the system logs for all CICS regions involved in the units of work which are in-doubt.

Note: You cannot use the DFH$IWUP utility program to resynchronize resources.

Limitations of the DFH$IWUP utility program

You cannot use the DFH$IWUP utility program to resynchronize resources.

The DFH$IWUP utility program does not list unit-of-work IDs for CICS regions connected using LU6.1 protocols. That is, it is unable to resolve any in-doubt resources updated using LU6.1 protocols.

Job control statements to assemble and link-edit the DFH$IWUP program

Before you can run the DFH$IWUP program, you must assemble and link-edit it into your CICS environment. You should use your own JCL to compile and link-edit the DFH$IWUP program.

Figure 43 shows sample job control statements that you can use assemble and link-edit the DFH$IWUP utility program.

```
// JOB IWUPJOB
// OPTION CATAL,NODECK,ALIGN
// LIBDEF    *,SEARCH=(cicsusr.source,PRD1.BASE),               *
               CATALOG=cicsusr.phase,  1 2
   PHASE DFH$IWUP,*
   MODE AMODE(24),RMODE(24)
// EXEC ASMA90
  ⋮
      DFH$IWUP source statements
  ⋮
/*
// EXEC LNKEDT
/*
```

Figure 43. Sample job to assemble and link-edit the DFH$IWUP utility

Notes:

1 Change `cicusr.source` to the library containing your version of DFH$IWUP source.

2 Change `cicusr.phase` to the library to which the assembled and link-edited DFH$IWUP program is to be written.

Job control statements to run the DFH$IWUP program

To run the DFH$IWUP utility program, edit and submit the sample JCL shown in Figure 44. Figure 45 on page 193 shows a sample of the report produced by the DFH$IWUP utility program.

```
// JOB DFH$IWUP
// DLBL UOWIDN,'CICS410.UOWIDN',0,SD
// EXTENT SYS001,volser,1,0,nnn,mm
// ASSGN SYS001,DISK,VOL=volser,SHR
// EXEC DITTO
$$DITTO CS  FILEOUT=UOWIDN,RECFMOUT=F,BLKSIZE=80
lllllllllllllllll.X'tttttttttttt'.nnnnn
lllllllllllllllll.X'tttttttttttt'.nnnnn
/*
$$DITTO EOJ
/*
// LIBDEF     *,SEARCH=(PRD1.BASE,user.sublib)
// DLBL SYSUT1,'CICS410.applid.DFHJ01A',0,SD
// EXTENT SYS001,volser,1,0,nnn,mm
// ASSGN SYS001,DISK,VOL=volser,SHR
// DLBL UOWIDIN,'CICS410.UOWIDIN',0,SD
// EXTENT SYS001,volser,1,0,nnn,mm
// ASSGN SYS001,DISK,VOL=volser,SHR
// EXEC DFHJUP,SIZE=DFHJUP
OPTION PRINT EXITR=DFH$IWUP
END
/*
/&
```

Figure 44. Sample JCL to run the DFH$IWUP utility program

Notes: The JCL to run the DFH$IWUP utility program uses the following DLBL statements:

DLBLname Defines

SYSUT1
The journal data set (in use at the time of failure) to be processed.

The output data set to which the formatted print records and control messages are to be sent by the DFH$IWUP utility program.

UOWIDIN
The input list of unit-of-work IDs to be scanned for. Each unit-of-work ID must be entered on a separate line, starting in column 1. Each unit-of-work ID is of the format:

```
lllllllllllllllll.X'tttttttttttt'.nnnnn
```

where:

```
lllllllllllllllll  is the LU name
tttttttttttt       is a hexadecimal token/clock value
nnnnn              is the sequence number
```

Any file of unit-of-work IDs specified on the UOWID DLBL statement must have an 80-byte record format.

SYSIPT
The input control data set for DFHJUP. This file must be in 80-byte record format.

```
* OPTION PRINT
OPTION PRINT EXITR=DFH$IWUP
*** INVALID UOWID FOUND :    GBIBMIYA.CICSB .X'999999999999'.00001
DFHJC4531 END OF FILE ON INPUT.
DFH$IWFX: CICS JOURNAL PRINT - FORMATTING EXIT
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
***  UNIT OF WORK ID :   GBIBMIYA.CICSB2.X'B0D53EABCFFC'.00001
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
***  NO RECORDS FOUND FOR THIS UOWID :
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
***  UNIT OF WORK ID :   GBIBMIYA.CICSE3.X'AFC6273F5111'.00002
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
EF59------SP UNIT OF WORK ID  MODULE-------SYNCH.POINT  RECORD LENGTH------00055
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--Y  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00001E 000000  180FC7C2 C9C2D4C9 E8C14BC3 C9C3E2C5  F3AFC627 3F511100 02
--------------------------------------------------------------------------------
C311------------FC WRITE ADD  MODULE------FILE CONTROL  RECORD LENGTH------00064
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    FILE NAME-------FILEA     BASE RBA------0000000000
    OFFSET FROM 0  RECORD ID  LENGTH---------------006
DFH$IWFX: CICS JOURNAL PRINT - FORMATTING EXIT
    00002A 000000  C9D1D2D3 D4D5
    OFFSET FROM 0  DATA       LENGTH-------------00016
    000030 000000  00100000 40C9D1D2 D3D4D5C9 D1D2D3D4
--------------------------------------------------------------------------------
C311------------FC WRITE ADD  MODULE------FILE CONTROL  RECORD LENGTH------00064
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    FILE NAME-------FILEA     BASE RBA------0000000000
    OFFSET FROM 0  RECORD ID  LENGTH---------------006
    00002A 000000  D6D7D8D9 E2E3
    OFFSET FROM 0  DATA       LENGTH-------------00016
    000030 000000  00100000 40D6D7D8 D9E2E3D6 D7D8D9E2
--------------------------------------------------------------------------------
C311------------FC WRITE ADD  MODULE------FILE CONTROL  RECORD LENGTH------00064
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    FILE NAME-------FILEA     BASE RBA------0000000000
    OFFSET FROM 0  RECORD ID  LENGTH---------------006
    00002A 000000  E4E5E6E7 E8E9
    OFFSET FROM 0  DATA       LENGTH-------------00016
    000030 000000  00100000 40E4E5E6 E7E8E9E4 E5E6E7E8
--------------------------------------------------------------------------------
```

Figure 45 (Part 1 of 4). Sample output from the DFH$IWUP program

```
F159--------SP LOGICAL START  MODULE-------SYNCH.POINT  RECORD LENGTH------00030
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
--------------------------------------------------------------------------------
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00002  LAST SPR OUTBOUND--00000  OPERATOR ID----------
    IN-DOUBT---------------N  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------N
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00020000 01000000 C9E2F340 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00002  LAST SPR OUTBOUND--00000  OPERATOR ID----------
    IN-DOUBT---------------N  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------N
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00020000 01000000 C9E2F240 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
DFH$IWFX: CICS JOURNAL PRINT - FORMATTING EXIT
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00001  LAST SPR OUTBOUND--00001  OPERATOR ID----------
    IN-DOUBT---------------N  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------N
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00010001 01000000 C9E2F140 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00002  LAST SPR OUTBOUND--00000  OPERATOR ID----------
    IN-DOUBT---------------Y  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------N
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00020000 90000000 C9E2F340 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00002  LAST SPR OUTBOUND--00000  OPERATOR ID----------
    IN-DOUBT---------------Y  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------N
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00020000 90000000 C9E2F240 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
```

Figure 45 (Part 2 of 4). Sample output from the DFH$IWUP program

```
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00000  LAST SPR OUTBOUND--00002  OPERATOR ID----------
    IN-DOUBT---------------Y  SENT SYS PFX ROOT------Y  SUCCESSFUL ABORT-------N
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00000002 D0000000 C9E2F140 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00002  LAST SPR OUTBOUND--12624  OPERATOR ID----------
    IN-DOUBT---------------N  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------N
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00023150 02000000 C9E2F140 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
C111----------FC READ UPDATE  MODULE------FILE CONTROL  RECORD LENGTH------00064
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
DFH$IWFX: CICS JOURNAL PRINT - FORMATTING EXIT
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    FILE NAME-------FILEA     BASE RBA------0000000000
    OFFSET FROM 0  RECORD ID  LENGTH---------------006
    00002A 000000  E4E5E6E7 E8E9
    OFFSET FROM 0  DATA       LENGTH-------------00016
    000030 000000  00100000 40E4E5E6 E7E8E9E4 E5E6E7E8
--------------------------------------------------------------------------------
C111----------FC READ UPDATE  MODULE------FILE CONTROL  RECORD LENGTH------00064
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    FILE NAME-------FILEA     BASE RBA------0000000000
    OFFSET FROM 0  RECORD ID  LENGTH---------------006
    00002A 000000  D6D7D8D9 E2E3
    OFFSET FROM 0  DATA       LENGTH-------------00016
    000030 000000  00100000 40D6D7D8 D9E2E3D6 D7D8D9E2
--------------------------------------------------------------------------------
C111----------FC READ UPDATE  MODULE------FILE CONTROL  RECORD LENGTH------00064
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    FILE NAME-------FILEA     BASE RBA------0000000000
    OFFSET FROM 0  RECORD ID  LENGTH---------------006
    00002A 000000  C9D1D2D3 D4D5
    OFFSET FROM 0  DATA       LENGTH-------------00016
    000030 000000  00100000 40C9D1D2 D3D4D5C9 D1D2D3D4
--------------------------------------------------------------------------------
F159--------SP LOGICAL START  MODULE-------SYNCH.POINT  RECORD LENGTH------00030
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
--------------------------------------------------------------------------------
F459------SP BAD END OF TASK  MODULE-------SYNCH.POINT  RECORD LENGTH------00030
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
--------------------------------------------------------------------------------
```

Figure 45 (Part 3 of 4). Sample output from the DFH$IWUP program

```
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
    LAST SPR INBOUND---00001  LAST SPR OUTBOUND--00000  OPERATOR ID----------
    IN-DOUBT---------------N  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------Y
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00010000 20000000 C9E2F340 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
F559-------SP SPR LOG RECORD  MODULE-------SYNCH.POINT  RECORD LENGTH------00067
    TASK NUMBER--------00034  TRANSACTION---------ISC2  TERMINAL------------1F77
    PHYSICAL START OF TASK-N  LOGICAL START OF TASK--N  RECORD IN-FLIGHT-------N
    EMERGENCY RESTART------N  DL/I CHECKPOINT--------N  USER PREFIX PRESENT----N
DFH$IWFX: CICS JOURNAL PRINT - FORMATTING EXIT
    LAST SPR INBOUND---00001  LAST SPR OUTBOUND--00000  OPERATOR ID----------
    IN-DOUBT---------------N  SENT SYS PFX ROOT------N  SUCCESSFUL ABORT-------Y
    SESSION FAILED---------N  WAIT FOR SESS.RECVY----N
    OFFSET FROM 0  DATA       LENGTH-------------00025
    00002A 000000  00010000 20000000 C9E2F240 180FC7C2  C9C2D4C9 E8C14BC3 C9
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
***  NO MORE RECORDS TO PRINT FOR THIS UOWID
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
***  END OF SEARCH. NO MORE RECORDS TO PRINT
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
DFHJC4532 OPTION COMPLETE.
DFHJC4527 END OF JOB.
```

Figure 45 (Part 4 of 4). Sample output from the DFH$IWUP program

Error messages produced by the DFH$IWUP program

The following messages may be issued by the DFH$IWUP utility program:

ERRMSG02 PROGRAM HAS TERMINATED- NO VALID unit-of-work IDS SUPPLIED.

Explanation: Either no unit-of-work IDS were specified or the unit-of-work IDs supplied were not of the correct format.

System Action: Processing terminates.

User Response: Correct the unit-of-work IDs and resubmit the DFH$IWUP job.

Destination: SYSLST

Job statement: UOWIDIN DLBL statement

ERRMSG04 PROGRAM HAS TERMINATED- OUT OF MEMORY. UNABLE TO BUILD THE LIST OF UOWIDS.

Explanation: While building the list of UOWIDs the DFH$IWUP utility program ran out of memory.

System Action: Processing terminates.

User Response: Resubmit the DFH$IWUP job with either a larger SIZE parameter or a smaller list of UOWIDs.

Destination: SYSLST

Job statement: EXEC PGM=DFHJUP statement or UOWIDIN DLBL statement

ERRMSG08 OUT OF MEMORY. UNABLE TO SAVE FURTHER RECORDS. PROGRAM WILL TERMINATE AFTER PRINTING ITEMS FOUND SO FAR

Explanation: While searching the journal, the DFH$IWUP utility program ran out of memory.

System Action: Processing terminates.

User Response: Resubmit the DFH$IWUP job with either a larger region or a smaller list of UOWIDs.

Destination: SYSLST

Job statement: EXEC PGM=DFHJUP statement or UOWIDIN DLBL statement

ERRMSG16 PROGRAM HAS TERMINATED- UNABLE TO OPEN UOWIDIN FILE.

Explanation: An error occurred while opening the UOWIDIN file.

System Action: Processing terminates.

User Response: Check the system console log for any associated messages from the OPEN macro. Correct and resubmit the DFH$IWUP job.

Destination: SYSLST

Job statement: UOWIDIN DLBL statement

ERRMSG32 PROGRAM HAS TERMINATED- UNABLE TO READ ITEM FROM UOWIDIN FILE.

Explanation: An input/output error occurred while reading the UOWIDIN file.

System Action: Processing terminates.

User Response: Check the system console log for any associated messages. Correct and resubmit the DFH$IWUP job.

Destination: SYSLST

Job statement: UOWIDIN DLBL statement

Creating your own DFH$IWUP utility program

You can create your own DFH$IWUP utility program. Source code is supplied in member DFH$IWUP of the VSE/ESA sublibrary, PRD1.BASE.

Chapter 17. Stagger end-of-day time sample utility program, DFH$STED

Statistics intervals can occur simultaneously across many CICS regions in the same VSE system, and this can degrade performance. To prevent this, you should vary the statistics interval occurrence time for each CICS region. You can use the sample utility program, DFH$STED, to vary automatically the end-of-day time of each CICS region it is installed in, and therefore vary the interval occurrence time of these CICS regions.

You should run the DFH$STED program in the third phase of CICS initialization (that is, during the second phase of PLT processing).

You can use the following three parameters to control how the end-of-day time is amended. These parameters are part of the source of DFH$STED. To change them you will have to modify the source of DFH$STED, which is located in the VSE/ESA sublibrary, PRD1.BASE.

EODDRIFT Specifies the end-of-day **drift time**; that is, the maximum allowable drift from the original end-of-day time.

This enables you to stagger the end-of-day time of each of your CICS regions by a pseudo-random amount (based upon the time of day at which the program is executing), up to a user-specified maximum value. Since intervals are calculated using the end-of-day as a base time, the occurrence of intervals are staggered by this pseudo-random drift time. The default is ten minutes.

EODTIME Specifies whether the end-of-day time before amendment by the drift value should take the current value (that is, 00:00:00 if COLD started, or the value at previous CICS shutdown if AUTO or WARM started).

You should set this field to CURRENT if you need the current end-of-day time, or FIXED if you need a new end-of-day time. If you specify FIXED, you should specify the new time on the EODFIXED parameter. The default value of the EODTIME parameter is FIXED.

EODFIXED Specifies the new logical end-of-day time, in the form `hhmmss`, as a hexadecimal value in the range `X'000000'` through `X'235959'`. Specify the EODFIXED parameter only if you also specify the EODTIME=FIXED parameter. When used in conjunction with a finite value of EODDRIFT, the drift value specified by the EODDRIFT parameter is applied to the new end-of-day specified by the EODFIXED parameter. When a zero value is specified for the EODDRIFT parameter, the end-of-day time is as specified for the EODFIXED parameter. The default setting is `X'000000'` (midnight).

Note: If a CICS region is brought up with START=AUTO on several subsequent occasions, and you specify EODTIME=CURRENT, the end-of-day time is never reset, and the drift accumulates.

Example: You could specify the following values for the parameters of the DFH$STED program if:

- All your CICS regions collect and write their statistics at hourly intervals
- You want to see statistics for all the CICS regions over the same period, but without performance degradation.

```
EODDRIFT=5            (5 minutes maximum drift time)
EODTIME=FIXED         (a new end-of-day time)
EODFIXED=X'000000'    (end-of-day time is midnight)
```

This would vary the statistics intervals by a pseudo-random amount, from midnight, up to maximum of five minutes:

```
Region 1 - statistics taken at 12.00.00
Region 2 - statistics taken at 12.04.10
Region 3 - statistics taken at 12.01.45
  ⋮
Region n - statistics taken at 12.00.27
```

Chapter 18. Local catalog storage program, DFHSMUTL

The local catalog storage manager domain subpool record manipulation program, DFHSMUTL, is a VSE batch program that adds or removes storage manager domain subpool records to or from the CICS local catalog data set. The records are used to store tuning information that CICS uses to determine the optimum sizes of the subpools, and to indicate to CICS which subpools are to have the self-tuning mechanism enabled.

If the local catalog is re-initialized, DFHSMUTL should be run again to add the required subpool records to the local catalog.

Input

Control statements are read from SYSIN that specify the subpool records to be added to the local catalog data set.

The format of the control statements is as follows:

1. All commands must start in column 1.
2. An asterisk in column 1 indicates a comment. Everything else on the line is ignored.
3. To add a new subpool record to the local catalog, code ADD SUBPOOL=**name**, where **name** is the name of the subpool the record is added to (for example, ADD SUBPOOL=ZCTCTUA). There can only be one blank after ADD, and the rest of the line following the subpool name must be blank. The subpool name is not checked for validity.
4. To delete a subpool record from the local catalog, code DEL SUBPOOL=**name**, where **name** is the name of the subpool the record is deleted from (for example, DEL SUBPOOL=ZCTCTUA). There can only be one blank after DEL and the rest of the line following the subpool name must be blank.
5. To print a list of subpool records from the local catalog, code LST.

Deleting and adding a subpool record resets the tuning information for that subpool.

Output

Storage manager domain subpool records that are written to the CICS local catalog data set.

Messages

Messages, including errors, are written to SYSPRINT. The DFHSM0300 DFHSMUTL REPORT shows:

ADD SUBPOOL=xxxxxxxx PROCESSED SUCCESSFULLY (ADD SUBPOOL=xxxxxxxx has been processed successfully.)

DEL SUBPOOL=xxxxxxxx PROCESSED SUCCESSFULLY (DEL SUBPOOL=xxxxxxxx has been processed successfully.)

FOUND DFHLCD RECORD SMSUBPOL=xxxxxxxx (Subpool record found by the LST command.)

ERROR OPENING DFHLCD (An error has occurred opening the local catalog data set. The program is terminated.)

UNRECOGNISED VERB xxx IN INPUT (Only ADD, DEL and LST are allowed. The statement is ignored.)

UNRECOGNISED OPERAND xxxxxxxx IN INPUT (Only ADD SUBPOOL=xxxxxxxx or DEL SUBPOOL=xxxxxxxx are allowed. the statement is ignored.)

ERROR PROCESSING 'ADD SUBPOOL=xxxxxxxx'.
R15 = X'yy'.
RPL FEEDBACK CODE = X'zz'.

(A VSAM error has occurred while processing an ADD SUBPOOL=xxxxxxxx command. For the meaning of the VSAM codes, refer to "VSE/VSAM codes" in *VSE/ESA Messages and Codes, Volume 2*. The program is terminated.)

ERROR PROCESSING 'DEL SUBPOOL=xxxxxxxx'.
R15 = X'yy'.
RPL FEEDBACK CODE = X'zz'.

(A VSAM error has occurred while processing a DEL SUBPOOL=xxxxxxxx command. For the meaning of the VSAM codes, refer to "VSE/VSAM codes" in *VSE/ESA Messages and Codes, Volume 2*. The program is terminated.)

END OF DFHSMUTL REPORT (Report trailer.)

Job control statements to run the DFHSMUTL program

```
// JOB DFHSMUTL
//***********************************************************
//*
//* Use DFHSMUTL to add or remove storage manager domain
//* subpool records to or from the local catalog data set
//*
//***********************************************************
// DLBL CICSUCT,'usercat',,VSAM
// DLBL DFHLCD,'CICS410.applid.DFHLCD',,VSAM,CAT=CICSUCT   1
// LIBDEF PHASE,SEARCH=(PRD1.BASE)
// EXEC DFHSMUTL,SIZE=DFHSMUTL
ADD SUBPOOL=name-of-subpool-to-add                         2
DEL SUBPOOL=name-of-subpool-to-remove                      3
LST
/*
```

Notes:

1. **1** Change CICS410.applid.DFHLCD to the name of the local catalog.
2. **2** Example, to enable the self-tuning mechanism for the ZCTCTUA subpool, specify ADD SUBPOOL=ZCTCTUA.
3. **3** Example, to disable the self-tuning mechanism for the ZCTCTUA subpool, specify DEL SUBPOOL=ZCTCTUA.

Part 3. Appendix

Appendix A. Operating procedures

This appendix describes the suggested procedures to follow when starting and running an online CICS region.

Information can be found under the headings:

- "Departmental and job titles"
- "CICS online sessions log sheet" on page 207
- "CICS incident report" on page 209
- "Data set/database and transaction table" on page 211
- "Transaction and program table" on page 213
- "Error procedures" on page 213

In the descriptions of the operating procedures, various departmental and job titles are used. You may not have a department or job with each of the titles used, so definitions are also given in "Departmental and job titles" to enable you to identify the department or individual that performs the function in your organization.

Various forms and tables are illustrated to help you operate the master terminal. These are for your guidance; you may need to modify them for your organization. Some of the forms are intended for your database and data communication administration people. Others are for you to complete to document the online session. The forms shown in this chapter are:

- CICS online sessions log sheet
- CICS incident report
- Data set, database, and transaction table
- Transaction and program table

Departmental and job titles

The following are definitions of the departmental and job titles that are used in subsequent descriptions of operating procedures:

Application supervisor
The person responsible for resolving errors in designated applications on production systems.

Database (DB) administration
The department responsible for the design, development, operation, integrity, maintenance, and use of databases.

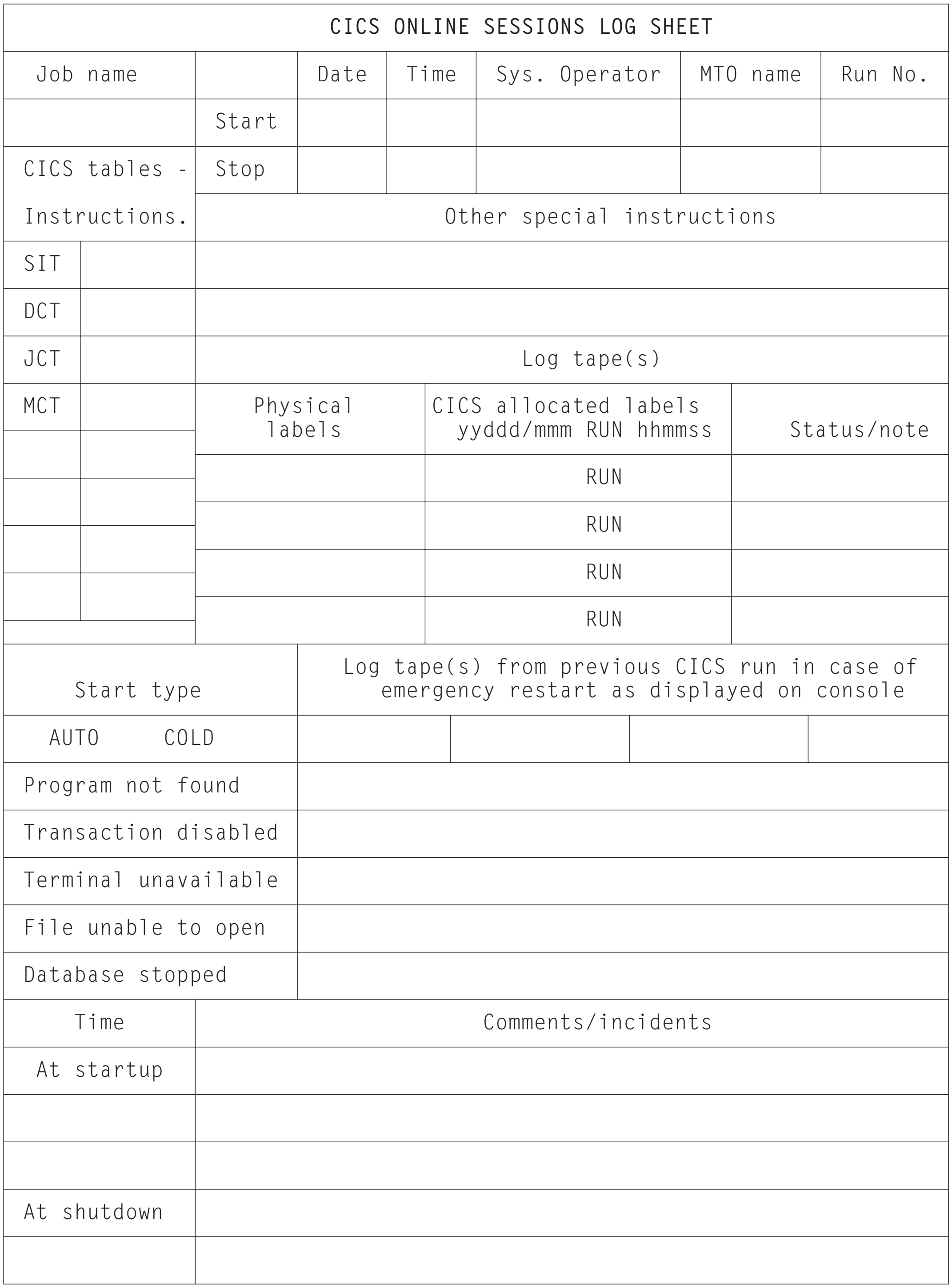
Data communication (DC) administration
The department responsible for the administration of online interactive systems.

Job scheduler
The person responsible for determining the order in which jobs are run, taking into account priorities and interdependences.

Network controller
The person responsible for the provision and maintenance of the hardware equipment (for example, terminals and lines) through which connections are made between terminals and hosts.

User liaison

The department responsible for communication between data processing functions and end users.



CICS ONLINE SESSIONS LOG SHEET						
Job name		Date	Time	Sys. Operator	MTO name	Run No.
	Start					
CICS tables -	Stop					
Instructions.	Other special instructions					

SIT				
DCT				
JCT		Log tape(s)		
MCT		Physical labels	CICS allocated labels yyddd/mmm RUN hhmmss	Status/note
			RUN	
			RUN	
			RUN	
			RUN	

Start type	Log tape(s) from previous CICS run in case of emergency restart as displayed on console			
AUTO COLD				
Program not found				
Transaction disabled				
Terminal unavailable				
File unable to open				
Database stopped				

Time	Comments/incidents
At startup	
At shutdown	

Figure 46. CICS online sessions log sheet

CICS online sessions log sheet

Use this log sheet to document each online session. You should complete it every time you start up CICS with the information needed to restart the system.

This CICS online sessions log sheet is submitted to operations by DC Administration, or the job scheduler, with some information already filled in. This information typically includes the job name, date, run number, physical labels of available log tapes, and any special instructions for the startup.

DC Administration specifies the start type required (usually START=AUTO) and, if START=AUTO results in an emergency restart, write in the label(s) of any log tape(s) from the previous run. This information comes from the log sheet of the previous run.

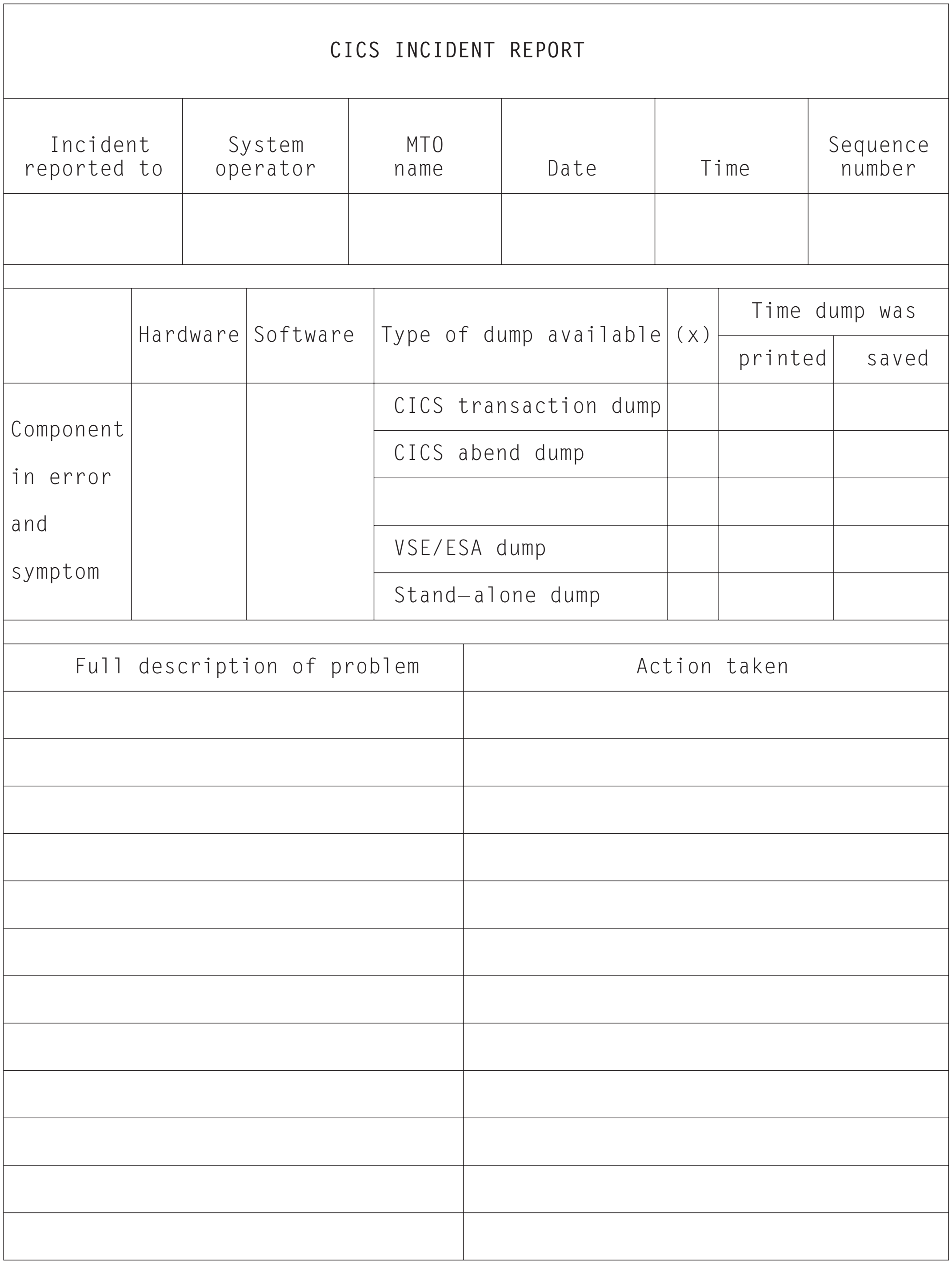
When starting up the system, you should do the following:

1. Record the start time and your name.
2. If you are using ***unlabeled tape journals***, a scratch tape volume is requested when the journal file is being opened for output. Mount the first tape from the available log tape list; CICS allocates a label for it. Record the allocated label on the log sheet. Do not forget to prepare a sticker with this label for use when the tape is unloaded during or at the end of the session. You may use more than one log tape.
3. If you perform an emergency restart and use tape rather than disk data sets for the system log, you need the system log tapes from the previous session. Start by mounting the last log tape from the previous CICS run.

   For unlabeled tape journals, DC Administration should already have entered the label of this tape (taken from the previous log sheet), on the current log sheet. When you mount the tape, CICS displays the first record label on the console for you to verify.

Document other useful information; for example, programs not found in the library, disabled transactions, files that could not be opened, or network problems.

Use the bottom portion of the form, “Comments/incidents,” to record any unusual incidents that occur at startup, shutdown, or during the session. Note here incidents such as hardware errors on lines or terminals, or abending transactions. Cross-refer to the CICS incident report where necessary. Some of these incidents appear on the system console and others are reported as CICS messages to a destination determined by the installation. This may be the console, another terminal, or a hard-copy device.



CICS INCIDENT REPORT

Incident reported to	System operator	MTO name	Date	Time	Sequence number

	Hardware	Software	Type of dump available	(x)	Time dump was	
					printed	saved
Component in error and symptom			CICS transaction dump			
			CICS abend dump			
			VSE/ESA dump			
			Stand–alone dump			

Full description of problem	Action taken

Figure 47. CICS incident report

CICS incident report

You should complete this form in duplicate to record any of the following error conditions:

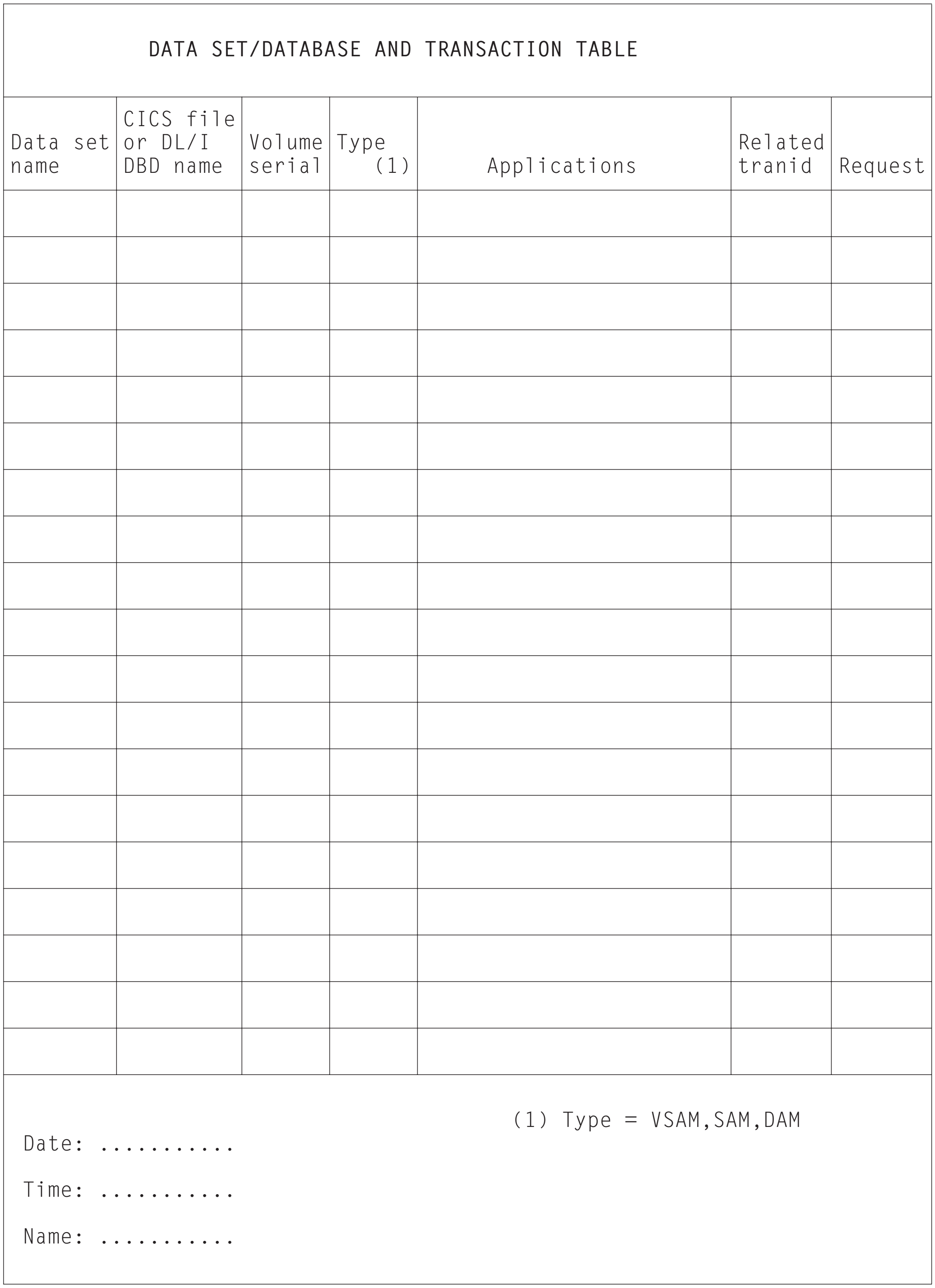
- VSE abend
- CICS abend
- Application abend
- Line/terminal hardware problem
- I/O errors on databases
- I/O errors on journal files
- Response time problems reported by users

Note the sequence number of the CICS incident report on the CICS online log sheet for that session.

In the middle block under the headings “Hardware” and “Software”, briefly describe the symptoms of the problem. If a dump is available, indicate the type of dump and the time it was taken, and, once it has been listed, the time of printing. If possible, describe the problem more fully in the bottom part of the form, and any action you have taken. This “Action” column may also be used by the recipient.

Keep one copy of the form for follow-up purposes and give the other copy to one of the following:

- Your network control people (if the incident was caused by a line or terminal error)
- The appropriate supervisor (if the incident was caused by an application problem)
- Your DC administration (in all other cases)



DATA SET/DATABASE AND TRANSACTION TABLE

Data set name	CICS file or DL/I DBD name	Volume serial	Type (1)	Applications	Related tranid	Request

(1) Type = VSAM,SAM,DAM

Date: ....

Time: ....

Name: ....

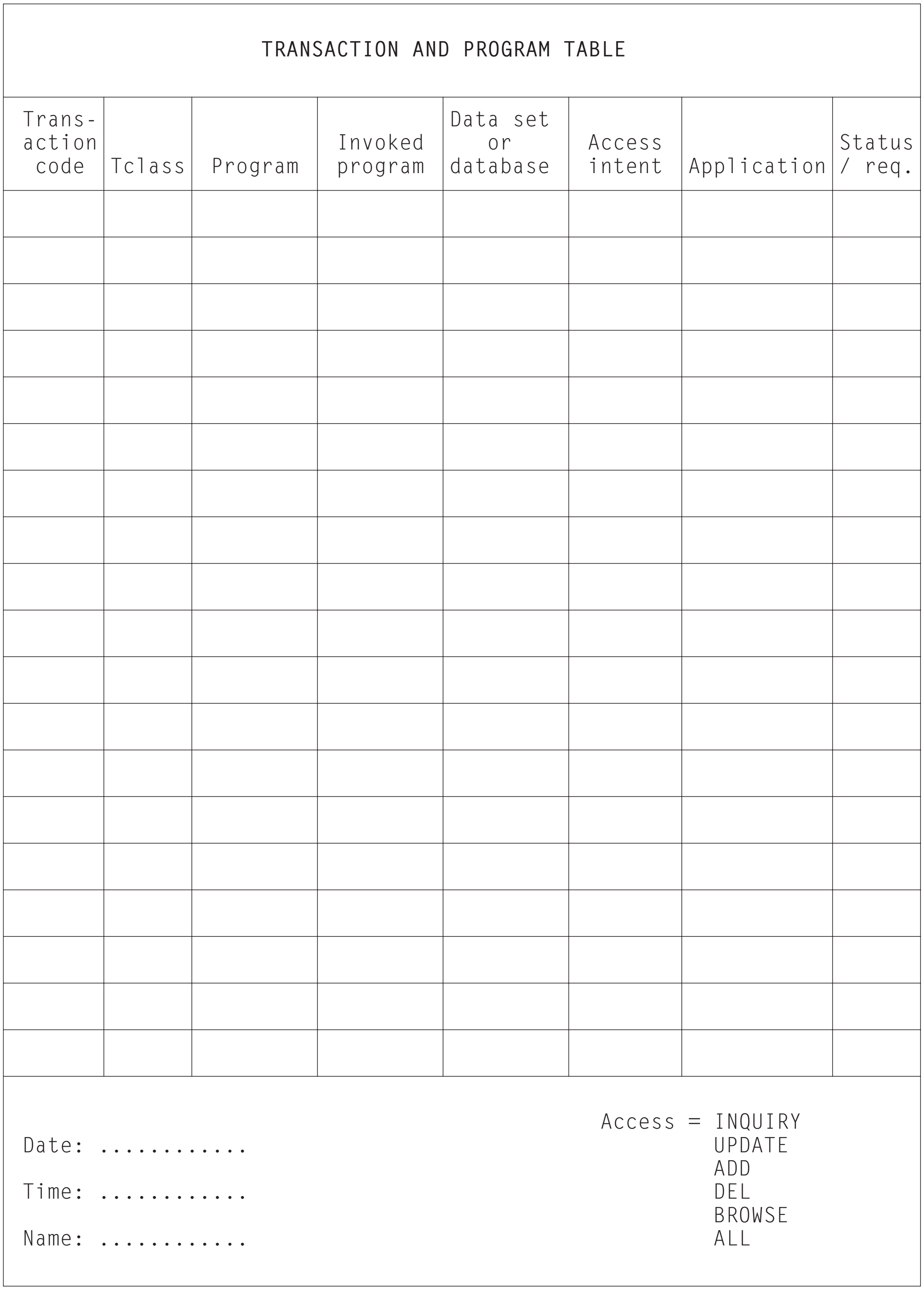
Figure 48. Data set/database and transaction table

Data set/database and transaction table

This table is prepared initially by DC Administration. Your user liaison people should also use it, for example, whenever they ask you to disable transactions that use a particular data set or database.

The table shows you the relationship between the data set name and the DL/I database names or the CICS file names, the volume serial numbers of the packs containing those data sets, the access method type, and the applications that use them. It also shows the relationship of all transactions to each data set or database.

From this table you can tell which transactions to disable before taking a data set or database offline. The date, time, and name at the lower left corner are used when you disable a related transaction at the request of your user liaison people.



TRANSACTION AND PROGRAM TABLE

Trans-action code	Tclass	Program	Invoked program	Data set or database	Access intent	Application	Status / req.

Date: ....

Time: ....

Name: ....

Access = INQUIRY
UPDATE
ADD
DEL
BROWSE
ALL

Figure 49. Transaction and program table

Transaction and program table

This table is prepared by DC Administration.

It shows you the relation between transaction codes and programs, the databases used, and whether a transaction accesses the data sets or databases for update or read-only purposes. It also shows the task class, the programs invoked by a transaction, the application, and the status of the program, for example, “Enabled” or “Disabled.”

The date, time, and name at the lower left hand corner is used when you disable a transaction at the request of your user liaison people.

Error procedures

Some user problems and system failures that require you to establish error procedures are illustrated in the following flowcharts:

Flowchart	**Procedure for**
A-1 (see page 214)	Machine check, power failure,
VSE loop, or abend	
A-2 (see page 216)	Action on user problem.

Chart A-1: Machine check, power failure, VSE loop, or abend

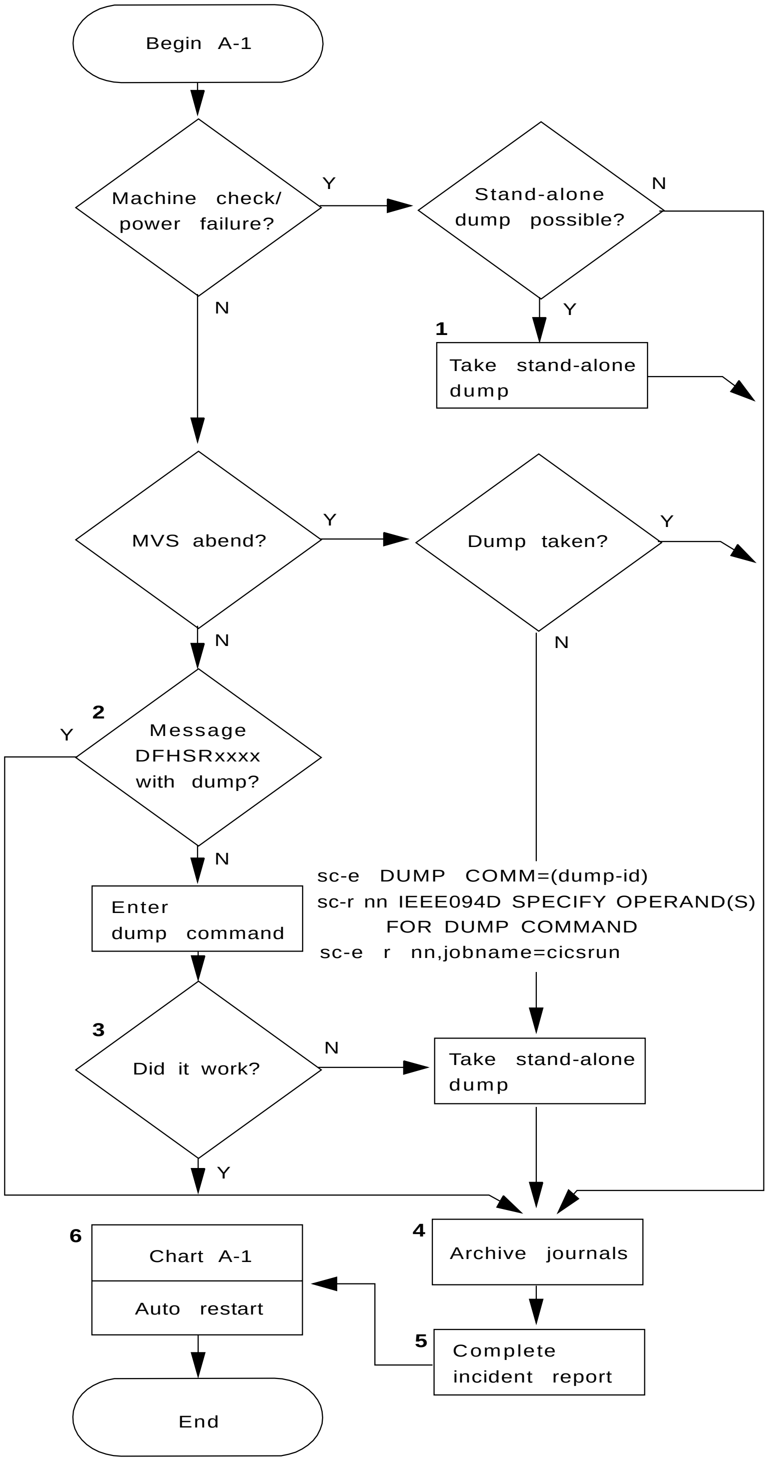


Figure 50. Chart A-1: Machine check, power failure, VSE loop, or abend

Notes for chart A-1

1. In the case of a machine check, a dump is useful for customer engineers and program service representatives to determine the cause of the problem.

2. Using the message DFHSRxxxx displayed on the operating system console, check in the *VSE/ESA Messages and Codes Volume 3* manual to see whether the dump is provided on the system dump data set when CICS is abnormally terminated. You should print the resulting dump before starting the next CICS session.

3. The operating system messages

   ```
   OS30I DUMP STARTED MEMBER=name  DUMP IN SUBLIB=sublibraryname
   1ISII DUMP COMPLETE.
   ```

   tells you that the dump is successfully written to the system dump.

4. You may want to archive to tape any crucial disk journals (for example, the CICS system log) at this point. If you do, make sure that the tapes are correctly labeled and entered on the online sessions log sheet. If you are using the CICS automatic archiving facility, this step is unnecessary, because CICS automatically submits the necessary archive jobs during the subsequent emergency restart.

5. The CICS incident report is used here for follow-up by DC Administration.

   Complete the CICS online sessions log sheet with stop date, time, and your name. Make sure that the “allocated label” column is correctly and completely filled in. State in the “Comments/incidents” area that the next startup must be an emergency restart. The information on this CICS online sessions log sheet is used in the following CICS startup.

6. If a machine check or power failure occurs, you can perform an emergency restart (START=AUTO resulting in an emergency start) as soon as the machine is up again, unless a disk failure is suspected.

Chart A-2: Action on user problem

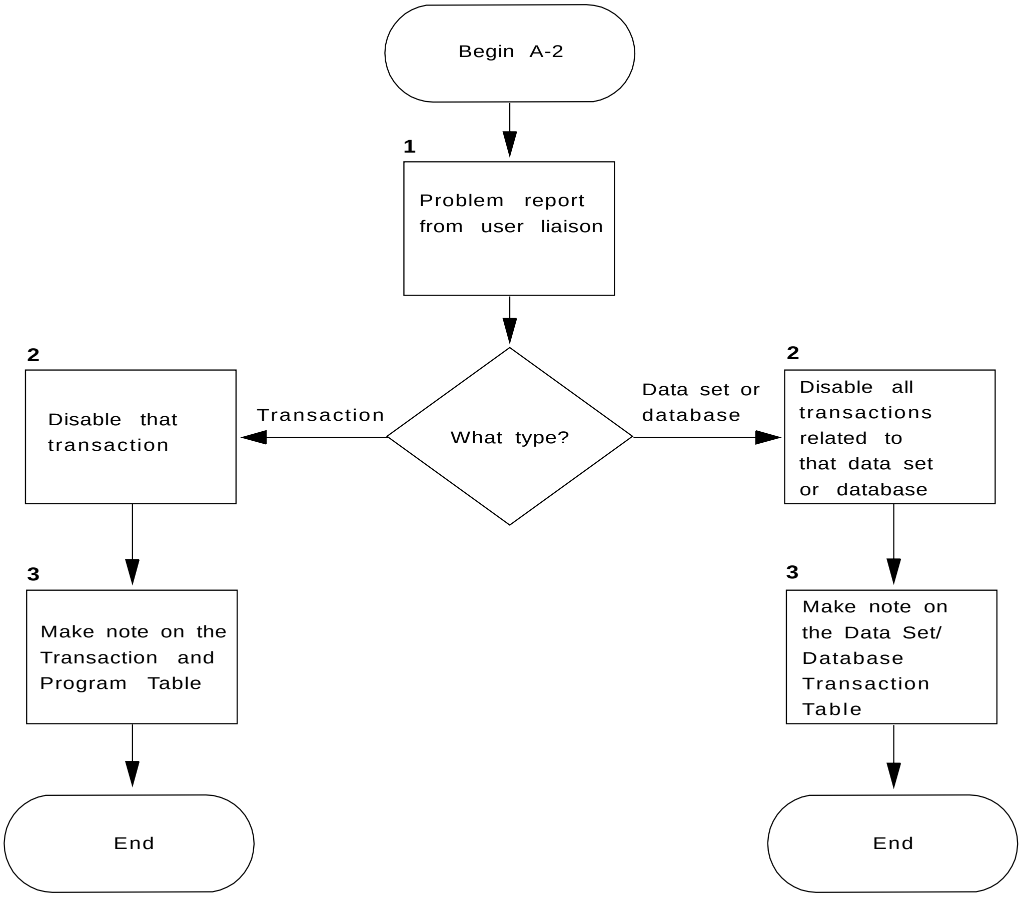

Begin A-2
1
Problem report from user liaison
2
Disable that transaction
Transaction
What type?
Data set or database
2
Disable all transactions related to that data set or database
3
Make note on the Transaction and Program Table
3
Make note on the Data Set/ Database Transaction Table
End
End

Figure 51. Chart A-2: Action on user problem

Notes for chart A-2

1. When a remote terminal operator or end user is experiencing difficulty, that person should contact user liaison. Your user liaison people can then determine the cause of the problem, and decide on the action to be taken. If the problem is related to transactions, data sets, or CICS local databases, user liaison may ask you to disable some transactions, by submitting a formal request. This request could be on the “Transaction and program table” for a transaction error, or on the “Data set/database and transaction table” for a data set or database error (see Figure 48 on page 210 and Figure 49 on page 212 for tables that you can use to process these requests).

2. Disable the transaction(s), in accordance with the instructions received, by entering a CEMT SET TRANSACTION DISABLED command. Alternatively, depending on the complexity of the transactions that access the various databases or file control data sets, you might prefer to stop the databases or disable the CICS file control files. For details of CEMT SET master terminal commands, see the *CICS-Supplied Transactions* manual.

3. After you have disabled the transaction(s), do not forget to complete the request by filling in the date, time, and your name on the form submitted to you.

Bibliography

CICS Transaction Server for VSE/ESA Release 1 library

Evaluation and planning	
Release Guide	GC33-1645
Migration Guide	GC33-1646
Report Controller Planning Guide	GC33-1941
General	
Master Index	SC33-1648
Trace Entries	SC34-5556
User's Handbook	SC34-5555
Glossary (softcopy only)	GC33-1649
Administration	
System Definition Guide	SC33-1651
Customization Guide	SC33-1652
Resource Definition Guide	SC33-1653
Operations and Utilities Guide	SC33-1654
CICS-Supplied Transactions	SC33-1655
Programming	
Application Programming Guide	SC33-1657
Application Programming Reference	SC33-1658
Sample Applications Guide	SC33-1713
Application Migration Aid Guide	SC33-1943
System Programming Reference	SC33-1659
Distributed Transaction Programming Guide	SC33-1661
Front End Programming Interface User's Guide	SC33-1662
Diagnosis	
Problem Determination Guide	GC33-1663
Messages and Codes Vol 3 (softcopy only)	SC33-6799
Diagnosis Reference	LY33-6085
Data Areas	LY33-6086
Supplementary Data Areas	LY33-6087
Communication	
Intercommunication Guide	SC33-1665
CICS Family: Interproduct Communication	SC33-0824
CICS Family: Communicating from CICS on System/390	SC33-1697
Special topics	
Recovery and Restart Guide	SC33-1666
Performance Guide	SC33-1667
Shared Data Tables Guide	SC33-1668
Security Guide	SC33-1942
External CICS Interface	SC33-1669
XRF Guide	SC33-1671
Report Controller User's Guide	GC33-1940
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 2.4 base program libraries

VSE/ESA Version 2 Release 4

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
Unattended Node Support	SC33-6712

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 2.4 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	SC09-2393
Database Administration	GC09-2389
Installation	GC09-2391
Interactive SQL Guide and Reference	SC09-2410
Operation	SC09-2401
Overview	GC08-2386
System Administration	GC09-2406

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
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

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Trademarks and service marks

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

ACF/VTAM
CICS
DB2
DFSORT
IBM
IMS
Language Environment
OS/390
RETAIN
VSE/ESA
VTAM
3090

Index

Sending your comments to IBM

CICS® Transaction Server for VSE/ESA™

Operations and Utilities Guide

SC33-1654-09

If you want to send to IBM any comments you have about this book, please use one of the methods listed below. Feel free to comment on anything you regard as a specific error or omission in the subject matter, and on the clarity, organization or completeness of the book itself.

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:

  IBM UK Laboratories
  Information Development
  Mail Point 095
  Hursley Park
  Winchester, SO21 2JN
  England

- 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)
  - Email: idrcf@hursley.ibm.com

Whichever method you use, ensure that you include:

- The publication number and title
- The page number or topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Program Number: 5648-054

SC33-1654-09