UNISYS

System 80 OS/3 Information Management System (IMS) System Support Functions

Programming Guide

OS/3 Release 13.0

January 1990

Priced Item

Printed in U S America
UP-11907 Rev. 1

The names, places, and/or events used in this publication are not intended to correspond to any individual, group, or association existing, living, or otherwise. Any similarity or likeness of the names, places, and/or events with the names of any individual, living or otherwise, or that of any group or association is purely coincidental and unintentional.

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THIS DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed Program Product License or Agreement to purchase or lease equipment. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such License or Agreement. Unisys cannot accept financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special, or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Correspondence regarding this publication should be forwarded to Unisys Corporation either by using the Business Reply Mail form at the back of this manual or by addressing remarks directly to Unisys Corporation, OS/3 Systems Product Information Development, P.O. Box 500, Mail Station E5-114, Blue Bell, Pennsylvania, 19424, U.S.A.

PAGE STATUS SUMMARY
ISSUE: UP-11907 Rev. 1

Part/Section	Page Number	Update Level
Cover		
Title Page/Disclaimer		
PSS	iii	
About This Guide	Tab Breaker v thru ix	
Contents	xi thru xxi	
Section 1	Tab Breaker 1 thru 7	
Section 2	Tab Breaker 1 thru 25	
Section 3	Tab Breaker 1 thru 11	
Section 4	Tab Breaker 1 thru 133	
Section 5	Tab Breaker 1 thru 19	
Section 6	Tab Breaker 1 thru 17	
Section 7	Tab Breaker 1 thru 24	
Section 8	Tab Breaker 1 thru 12	
Appendix A	Tab Breaker 1 thru 5	
Appendix B	Tab Breaker 1 thru 17	
Appendix C	Tab Breaker 1 thru 12	

Part/Section	Page Number	Update Level
Appendix D	Tab Breaker 1 thru 2	
Appendix E	Tab Breaker 1 thru 12	
Index	Tab Breaker 1 thru 18	
User Comments Form		
Back Cover		

Part/Section	Page Number	Update Level

About This Guide

Purpose

This guide is one of a series designed to instruct you in using the Unisys Information Management System (IMS) for Operating System/3 (OS/3) in a System 80 environment. It describes all aspects of generating an IMS system, creating a supporting ICAM network, and running your IMS system.

Scope

This programming guide provides detailed coding syntax and procedures for generating your IMS system and IMS network, configuring IMS, initiating and terminating an IMS session, using batch transaction processing, performing file recovery procedures, and performing IMS statistical reporting.

Before attempting to configure and generate your IMS system, you should develop a plan concerning the features and options needed in your IMS operating environment. You must also be thoroughly familiar with the configuration of your System 80 computer system and its associated communications network.

Audience

This guide is intended for the IMS administrator and systems analyst. It gives an overview of the IMS package and fully describes system preparation and system functions.

Prerequisites

Before making extensive use of this manual, you should have a fundamental understanding of the IMS theory, how it operates, and what you need to do to make it operational. This information is contained in the *IMS Technical Overview,* UP-9205. You should also be familiar with the contents of the other manuals listed in the subsection, "Related Product Information".

Organization

This document contains eight sections and five appendixes.

Section 1. Introduction

Discusses the purpose of IMS and the services it provides. Describes the functions you must perform to tailor IMS to your requirements and the modules in the IMS library that will become part of your IMS system. Explains IMS support of basic data management files in DTF mode and of consolidated data management files in CDM mode.

Section 2. Communications Support with ICAM

Describes the software interface between your IMS system and your terminals. Explains how to create an ICAM symbiont to support your terminal network and the necessary interfaces. Discusses the network definition macros that generate the ICAM symbiont.

Section 3. Pre-Online Processing

Lists the utilities and procedures you may execute before any online processing occurs in order to define your particular online operating environment. Tells you how to initialize and list the named record (NAMEREC) file, to which all IMS processors contribute records, and how to define passwords.

Section 4. Configuring IMS

Lists the five job steps that comprise the configuration process and describes how to configure IMS. Describes how to generate and execute the control streams that generate IMS. Also, describes in detail the 12 logical sections that make up your input to the IMS configurator.

Section 5. Initiating and Terminating IMS

Briefly tells you how to start and end an IMS session under normal and emergency conditions. Discusses how to load ICAM and establish a global network. Explains the job control stream needed to load IMS and describes how you can replace action programs while IMS is running. Describes how to bring IMS to a normal or emergency halt.

Section 6. Batch Processing of Transactions

Describes how:

- Batch processing works and when to use it
- Input and output messages are handled
- To set up the batch environment
- To prepare input for the batch processor
- To start and control batch processing
- To run IMS batch processing without a communications network
- To resume batch processing after a warm or cold restart

Section 7. File Recovery

Discusses online and offline file recovery functions. Describes the audit file, terminal output message file, and trace file. Explains what you need to do at configuration time and execution time in order to use these files. Describes IMS file recovery after a transaction is abnormally terminated or after a system restart. Tells how you can restore damaged files or files left in an inconsistent state after IMS termination or system failure.

Section 8. IMS Statistical Reporting

Gives an overview of the statistical reporting functions, and lists all statistical data printed. Describes the information you must supply to the configurator if you want IMS to print statistical data. Explains how and when statistical data is recorded, and describes the steps you must take to run the statistical print program.

Appendix A. Statement Conventions

Describes the conventions of IMS statement and parameter formats.

Appendix B. Estimating Main Storage Requirements for IMS

Provides guidelines for estimating main storage requirements for executing single-thread and multithread IMS systems under OS/3.

Appendix C. Operating Performance of IMS under OS/3

Discusses some ways to ensure the most efficient performance of IMS under OS/3.

Appendix D. UNIQUE Language Elements

Lists UNIQUE language elements and maximum length allowed for their replacements.

Appendix E. IMS Messages

Lists and describes error messages generated by the NAMEREC file utility, the configurator, and the batch transaction processor.

Related Product Information

As one of a series, this manual is designed to guide you in configuring and controlling IMS. Depending on your needs, you should also refer to the other manuals in the series. Complete manual names, their ordering numbers, and a general description of their contents and use are as follows:

Note: *Throughout this guide, when you are referred to another document, use the version that applies to the software level in use at your site.*

Information Management System (IMS) Technical Overview, **UP-9205**

Describes the basic concepts of IMS and the facilities it offers.

Information Management System (IMS) Action Programming in RPG II Programming Guide, **UP-9206**

Describes how to write action programs in RPG II, with extensive examples.

Information Management System (IMS) COBOL/Assembler Action Programs Programming Guide, **UP-9207**

Describes how to write action programs in COBOL and BAL, with extensive examples.

Information Management System (IMS) Operations Guide, **UP-12027**

Describes how terminal operators enter and receive messages at terminals. Includes IMS administrator information.

Information Management System (IMS) Data Definition and UNIQUE Programming Guide, **UP-9209**

Describes how to write data definitions and how to use UNIQUE commands to access defined files.

IMS/DMS Interface Programming Guide, **UP-8748**

Describes how to access a data base management system (DMS) data base from IMS.

The IMS administrator and systems analyst should also be familiar with the following OS/3 documents:

System 80 Models 3-6 and 8-20 OS/3 Installation Guide, UP-8839

System 80 Model 7E OS/3 Installation Guide, 7002 3858

System Messages Reference Manual, UP-8076

Job Control Programming Guide, UP-9986

Job Control Programming Reference Manual, UP-9984

Integrated Communications Access Method (ICAM) Technical Overview, UP-9744

Integrated Communications Access Method (ICAM) Operations Guide, UP-9745

Integrated Communications Access Method (ICAM) Programming Reference Manual, UP-9749

System Service Programs (SSP) Operating Guide, UP-8841

System Service Programs (SSP) Programming Reference Manual, UP-8842

Consolidated Data Management Programming Guide, UP-9978

Consolidated Data Management Macroinstructions Programming Guide, UP-9979

General Editor (EDT) Operating Guide, UP-9976

Interactive Services Operating Guide, UP-9972

Screen Format Services Technical Overview, UP-9977

Data Utilities Operating Guide, UP-8834

Supervisor Macroinstructions Programming Reference Manual, UP-8832

Assembler Programming Guide, UP-8913

Notation Conventions

Information on statement conventions used in this document is contained in Appendix A.

Contents

Section 4. Configuring IMS

Section 5. Initiating and Terminating IMS

Section 6. Batch Processing of Transactions

Appendix A. Statement Conventions

Appendix B. Estimating Main Storage Requirements for IMS

Appendix C. Operating Performance of IMS under OS/3

Figures

Figures

Tables

Section 1
Introduction

1.1. Overview

The Unisys Information Management System (IMS) is a transaction-oriented data file processing system that supports online inquiries from remote terminals for the purpose of examining or updating your files. For each message, IMS performs the requested processing and responds with an output message to the originating terminal. With IMS, your programmers do not have to be data communications experts because IMS provides you with the following services:

- Access to your data files
- Printing of transaction output
- Verifying, editing, and scheduling communications messages
- Scheduling processing sequences depending on message context
- Communications network access

IMS also provides for the data integrity and data security necessary in real-time processing. Data modifications are logged, and both online and offline data file recovery are available.

In addition to online transaction processing, IMS provides a batch transaction processor that you can use in offline or online mode for production runs involving batched transactions or for testing new applications.

IMS, as supplied by Unisys, is a collection of software components that you must adapt to the individual requirements of your installation. You create your own online IMS system in a configuration process (Section 4) in which you describe your communications network, your files, and your applications, and select the optional features you want included. The output of the configuration process is an IMS load module that operates as a communications user program interfacing with the integrated communications access method (ICAM). Generation procedures for an ICAM symbiont that supports IMS are described in Section 2.

In addition to the modules that are used to configure your online IMS system, IMS components include utility programs for pre-online processing of your applications and for offline recovery of your data files.

1.1.1. Transaction Processing

When you want to access your files, you enter a transaction code at a remote terminal. This initiates a transaction, which is a series of related input messages, file accesses, and responses. Each input message-output response sequence in the transaction is called an action. Transactions are handled by action programs that interact with the online IMS load module for system services. (They do not actually access your data files but call on IMS to do so.)

IMS provides a set of action programs called the Uniform Inquiry Update Element (UNIQUE) that allow you to retrieve, add, delete, and change records in your files by entering simple commands from terminals. To use UNIQUE, you must define your file structure and the restrictions you want to place on file updating in a data definition, described in the *IMS Data Definition and UNIQUE Programming Guide,* UP-9209.

If you have special file processing or message formatting needs that cannot be handled by UNIQUE, you can write your own action programs in basic assembly language (BAL), COBOL, or RPG II; these are described in the current versions of the *IMS COBOL/Assembler Action Programs Programming Guide,* UP-9207, and *IMS Action Programming in RPG II Programming Guide,* UP-9206. Action programs can access conventional indexed sequential access method (ISAM), multiple indexed random access method (MIRAM), direct access method (DAM), and sequential access method (SAM) disk files, MIRAM diskette files, and SAM tape files. Or you can access your files through the medium of defined files in which elements of your data files are logically combined to meet the requirements of a particular application. You do this by writing a data definition.

COBOL and basic assembly language programmers may define their own printer files within action programs (for multithread configurations only). These printer files may be spooled or nonspooled and use all the printer and spooling facilities offered by job control. By using printer function calls, you can print your own data and control forms positioning, spooler output, and the assignment of printer files to a terminal.

COBOL action programs can also access a data base management system (DMS) data base. You can use a data base as a source file in a data definition and then access the defined file you created through your COBOL, RPG II, or BAL action programs or UNIQUE. For more information, refer to the *IMS/DMS Interface Programming Guide,* UP-8748.

1.1.2. Single-Thread and Multithread IMS Systems

The IMS system you configure can be either single-thread or multithread. In a single-thread IMS system, only one action can be processed at a time, but actions for different transactions can be interspersed. Since the duration of an action is normally short, IMS can concurrently handle transactions originating from several terminals with very little delay in response time. Multithread IMS allows concurrent processing of actions for different transactions, providing better performance, but it requires more main storage.

Appendix B provides information on main storage requirements, and Appendix C discusses processing performance in single-thread and multithread IMS systems.

You can configure more than one IMS load module if your applications have varying requirements. That way, you can minimize main storage requirements by including only the options you need for each application and by configuring a single-thread or multithread load module as best suits the application.

Note: *Regardless of the type of IMS system you configure, your processor must be equipped with the storage protection feature.*

1.2. Generating an IMS System

1.2.1. Preparing for Online Processing

The process of generating a tailored IMS system starts at system generation (SYSGEN), when you must generate a supervisor that supports IMS, include the appropriate ICAM and IMS library modules in your system resident (SYSRES) disk volume (unless you use the OS/3 release volume (OS3REL) as your SYSRES), and generate an ICAM module that supports your IMS system. For details on the SYSGEN process, refer to the appropriate system installation guide for your system.

Depending on the requirements of your applications, you must perform some or all of the following functions in preparation for online processing:

- Initialize the named record (NAMEREC) file, which contains all the tables and records needed by online IMS. This can be done as part of the configuration process or separately with the NAMEREC file utility, ZP#NRU (Section 3).
- Generate passwords for use with UNIQUE, using the same NAMEREC file utility (Section 3).
- Write data definitions for defined files, required for UNIQUE and optional for use with user action programs. Data definitions are processed by the data definition processor, DT3DF, described in the *IMS Data Definition and UNIQUE Programming Guide,* UP-9209.
- Write action programs in BAL, COBOL, or RPG II to handle your applications for which you do not use UNIQUE. Action program preparation is described in the IMS action programming guides.
- Generate edit tables for use with your action programs, using the edit table utility, ZH#EDT (also described in the IMS action programming manuals). These tables simplify input message editing and validation.

- Configure an online IMS load module, using the IMS job control procedure (jproc), IMSCONF (Section 4). The configuration process also allocates and initializes the internal files needed by online IMS.

Generation of an IMS system is illustrated in the flowchart in Figure 1-1, which also shows online processing and offline recovery functions. The pre-online sequence shown is interchangeable, with three exceptions:

1. Supervisor generation must always be the first step.
2. The ICAM load module must be generated before configuration.
3. The NAMEREC file must be initialized before or during configuration and before processing of data definitions, passwords, or edit tables. If the NAMEREC file is reinitialized at any time, all pre-online processing, including configuration, must be repeated.

1.2.2. OS/3 System Generation Considerations

When you generate an OS/3 operating system to support IMS, you must consider a number of points in the SUPGEN and COMMCT phases of SYSGEN:

1. In the supervisor generation (SUPGEN) phase, you must:
   - Specify TIMER=MAX to include the GETIME and SETIME macros in the timer services available for your I/O devices.
   - Specify the number of communications adapters and communications lines to be supported, using the COMM keyword parameter.
   - Specify at least three transient areas (TRANS=3) for a single-thread IMS system; at least four for multithread. If other jobs will run with IMS, allocate at least five transient areas.
   - Specify at least two task priority levels (PRIORITY=2) for single-thread IMS; at least four for multithread.
   - Specify a symbiont priority (SYMBPRI=n) larger than the lowest priority used by IMS (higher numbers equal lower priorities). For single-thread, specify at least 2; for multithread, specify at least 4.
   - Allow for two job slots to provide the necessary storage keys for IMS. If other jobs will run with IMS, allocate at least three job slots.

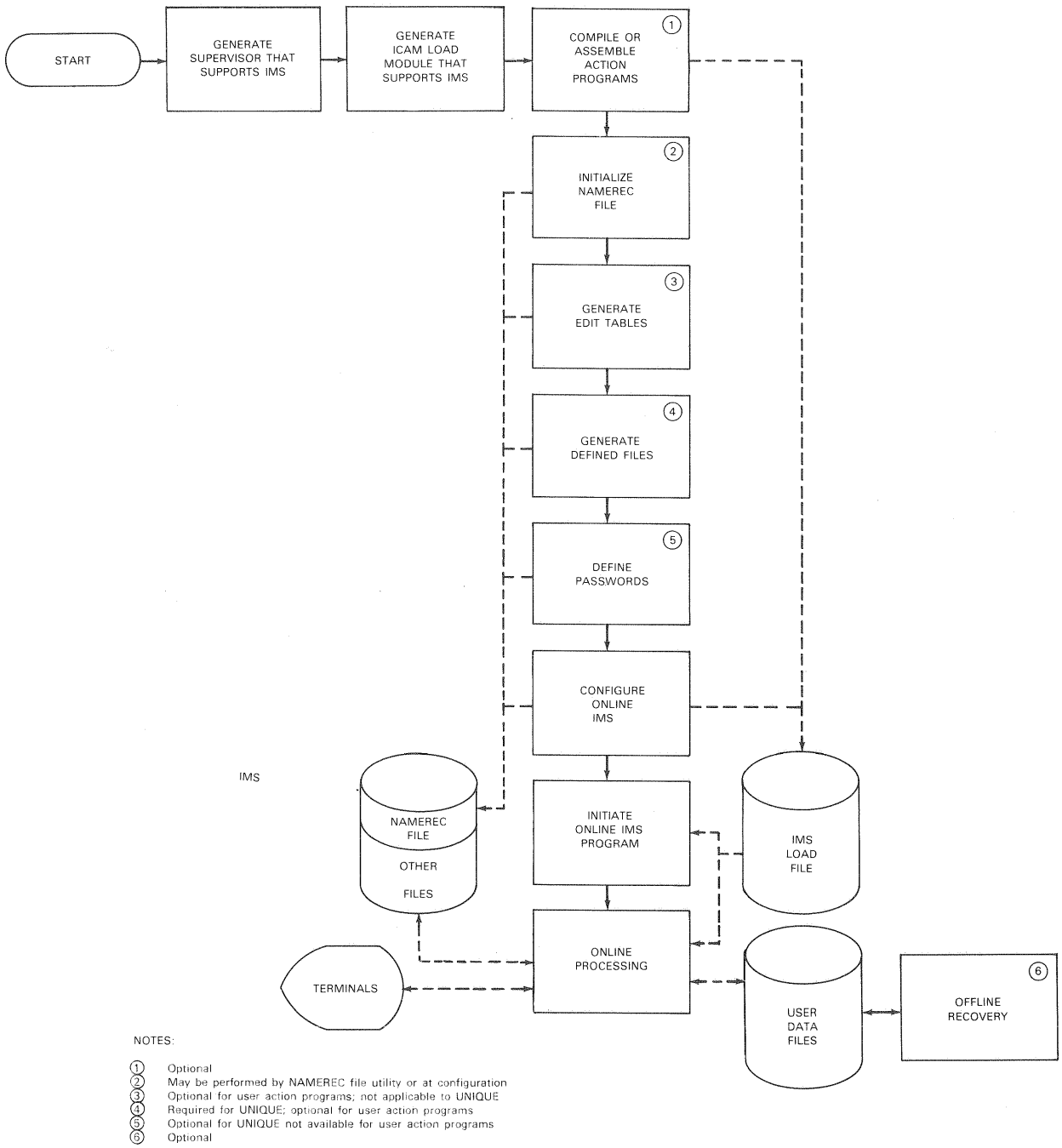


Figure 1-1. Flowchart for IMS Generation and Processing

- For System 80 models 7E, 8, 10, 15, and 20, specify DTF, CDM, or mixed data management mode (DMGTMODE parameter) based on which version of IMS (DTF or CDM) you intend to use.

If you intend to process batched transactions and have only one printer available, you also must include the appropriate spooling capability in your supervisor. Additional SUPGEN specifications that can improve processing performance under IMS are discussed in Appendix C.

2. You must generate an ICAM load module that supports IMS in the COMMCT phase. You can do this at the same time you generate a supervisor or later in a separate SYSGEN. Details on generating an ICAM module that supports IMS are in Section 2.

1.3. IMS Component Structure

The IMS library that you receive as part of the OS/3 software contains a collection of source, object, and load modules. Most of these modules become part of your online IMS system; they perform such functions as applications management and internal message control. Some modules are included in every IMS system; others are optional.

In addition to the components that become part of your online IMS system, the IMS library includes the following programs:

- Configurator programs: ZS#CNF for single-thread IMS, ZQ#CNF for multithread. The configurator program is executed by an IMS jproc, IMSCONF.
- The NAMEREC file utility, ZP#NRU, which initializes the NAMEREC file and generates passwords
- The data definition processor, DT3DF, with which you create defined files
- The edit table generator, ZH#EDT
- The offline recovery utility, ZC#TRC
- The tape copy routine, ZC#TCP
- The STATFIL print program, ZC#ZSF

The offline recovery utility and tape copy routine, which are described in Section 7, are provided in the form of object modules that you must link with modules produced by the configuration process and other modules to create executable programs.

1.4. DTF or CDM Mode

IMS accesses your data files through OS/3 data management. OS/3 actually has two data management systems, and the one you use depends on your hardware, the way your data files are organized, and options you select at system generation.

- **Basic data management** accesses files in DTF mode. In DTF mode, data files are defined by DTF macroinstructions. (You don't have to write the DTF macroinstructions -- IMS generates them from your specifications in the configurator FILE section.) IMS supports direct access method (DAM), multi-indexed random access method (MIRAM), indexed sequential access method (ISAM), and sequential access method (SAM) files in DTF mode. IMS also supports indexed random access method (IRAM) files but accesses them through MIRAM. You must define your IRAM files as MIRAM files in the configurator FILE section.

- **Consolidated data management** accesses files in CDM mode. In CDM mode, data files are defined by CDIB and RIB macroinstructions. (IMS generates CDIB and RIB macroinstructions from your specifications in the configurator FILE section.) In CDM mode, all disk and diskette data files are organized using MIRAM. You can also use sequential tape files in CDM mode.

If you have a System 80 model 3, 4, 5, or 6, your files are always accessed in CDM mode. If you have a System 80 model 7E, 8, 10, 15, or 20, you can use either the DTF mode with DAM, ISAM, MIRAM, and SAM files or the CDM mode with MIRAM files. You select DTF mode, CDM mode, or mixed mode for your operating system at supervisor generation.

At configuration, you tell IMS whether to use the DTF or CDM mode with the CDM parameter of the IMSCONF jproc. You can't define both modes in the same IMS configuration.

For more information about OS/3 data management and the CDM mode, refer to the *Consolidated Data Management Programming Guide,* UP-9978.

Section 2
Communications Support with ICAM

2.1. General

When you use IMS online, it becomes part of a communications system. At one end of the system, shown in Figure 2-1, is your configured IMS load module (Section 4). At the other end are your terminals. In between are two types of interfaces. The first, the communications adapter, is a hardware device that takes messages coming in from your network and puts them in a form usable by the operating system, and vice versa. The other interface, the integrated communications access method (ICAM), is the subject of this section. It's a software component that controls the input from and output to the terminals. Because of ICAM, IMS is device independent -- you can have any arrangement of supported lines and terminals without IMS or your action programs being aware of what hardware they're working with.

ICAM is designed to support a number of different types of programs. Because of this, ICAM has four different interfaces and two types of networks -- dedicated and global. You'll be using an interface created for use by IMS, the transaction control interface (TCI). You can use it with a dedicated network or a global network. A dedicated network supports only your IMS program. All the lines and terminals defined in the network definition are dedicated to IMS for the life of a session. A global network can be used by your IMS program and other communications user programs at the same time. Communications lines can be shared, and terminals can be attached to and detached from IMS and other programs during the session.

2.2. Creating ICAM

To use the communications system, you must create an ICAM symbiont that supports your network of terminals and the interfaces needed by IMS. You define the ICAM support needed with a set of ICAM network definition macros. These macros are then run through the SG$PARAM and SG$COMMK procedures of system generation (SYSGEN), either when you create your supervisor or in a separate SYSGEN. To create your ICAM symbiont, you'll need the current versions of the following manuals:

- *Integrated Communications Access Method (ICAM) Technical Overview,* UP-9744

  Describes how ICAM works.

- *Integrated Communications Access Method (ICAM) Operations Guide,* UP-9745

  Describes the macros used to configure an ICAM module.

- *System 80 Models 3-6 and 8-20 OS/3 Installation Guide,* UP-8839, or *System 80 Model 7E OS/3 Installation Guide,* 7002 3858

  Refer to the appropriate installation guide for your site, which provides a general description of how to use the ICAM network definition macros to create an ICAM symbiont during system generation.

The ICAM symbiont you create must be resident.

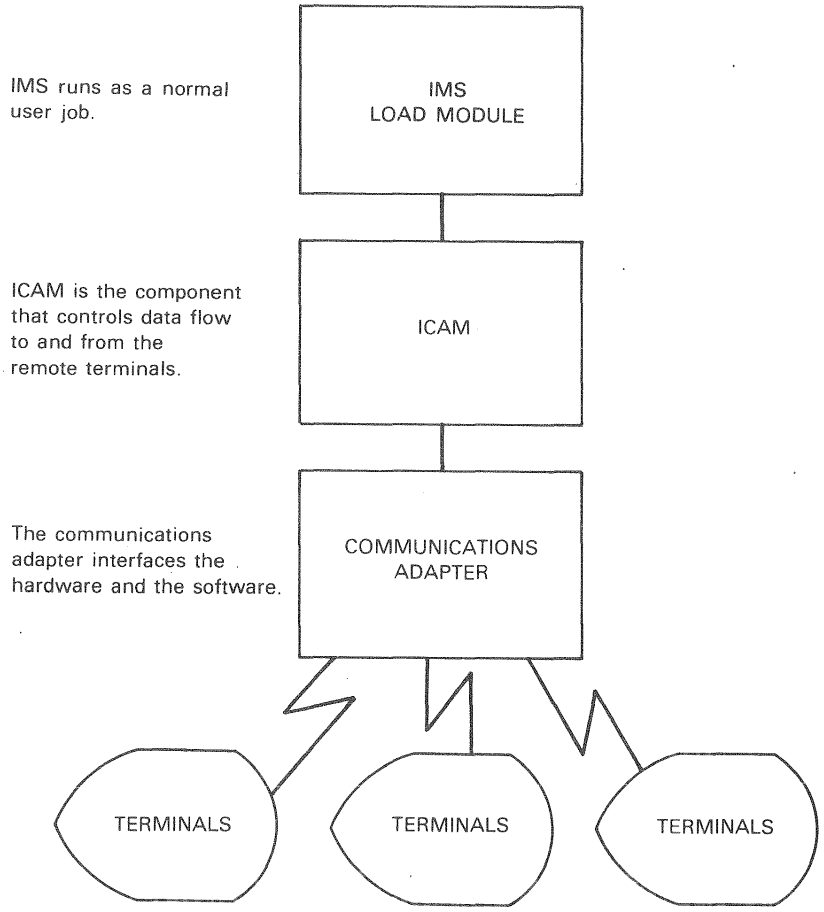


Figure 2-1. IMS and the OS/3 Communications System

When you run the ICAM network definition macros through the SG$COMMK phase of system generation, an ICAM object module is created and stored in the system object library (YOBJ) on the system resident volume (SYSRES). This object module, called the communications control area (CCA), is used for two purposes. When linked by SG$COMMK, it becomes part of the ICAM symbiont. Also, a link of the CCA object module is done during the first step of the IMSCONF jproc. This allows the configurator to determine whether the network is dedicated or global and, if global, to validate the LOCAP name against the name specified on the CUP parameter of the NETWORK section.

2.3. Network Definition

2.3.1. ICAM Network Definition Macros

Network definition macros for an ICAM supporting IMS are summarized in Table 2-1. For details on generating ICAM, refer to the *ICAM Operations Guide,* UP-9745. In your network definition, you may also use the message processing procedure specifications (MPPS) macros described in the *ICAM Programming Reference Manual,* UP-9749. Additional features that apply to the transaction control interface, including journaling, are documented in UP-9745.

Table 2-1. ICAM Network Definition Macros

Macro	Function	Remarks
CCA	Generates the control section for this network	IMS requires TYPE=(TCI) for a dedicated network; TYPE=(GBL,,node) for a global network.
BUFFERS	Identifies buffer and activity request packet requirements	Required.
LOCAP	Creates an interface between a global network and IMS	Required in a global network; IMS requires TYPE=(TCI).
LINE	Defines the characteristics of a line	Must be repeated for every line in the network.
TERM	Defines the characteristics of a terminal. For resident ICAM, also creates output queues	TERM macros describing terminals on a particular line must immediately follow the LINE macro describing that line.

continued

Table 2-1. ICAM Network Definition Macros (cont.)

Macro	Function	Remarks
SESSION	Sets up a static session path between a terminal and IMS or between a process file and IMS	Required for process files and static session terminals in a global network.
PRCS	Creates a process file	This feature is used only for an IMS system using unsolicited output.
DISCFILE	Identifies disk files to be used for disk buffering	Required to identify disk buffer file. Also identifies output disk queueing files, if used.
ENDCCA	Marks the end of the network definition	Required

The ICAM network definition macros basically define three things: the type of ICAM interface, your network of lines and terminals, and input/output interfaces.

1. You use the CCA macro to identify a global network or a dedicated network that uses the transaction control interface. In a global network, you use the LOCAP macro to define a transaction control interface.

2. Your network is the arrangement of your terminals for the collection of data going to IMS and the distribution of information coming out of it. You must define this arrangement for ICAM with LINE and TERM macros. In a global network, you also use SESSION macros to define static session terminals -- those terminals that are attached to IMS during an entire IMS session.

3. Figure 2-2 shows the input/output requirements for a resident ICAM. In a resident system, ICAM stores input messages in main storage buffers and output messages in either main storage or a disk queueing file. IMS may request that an input message be temporarily stored in a disk buffer file if main storage buffer space is not available. You define the disk buffer and disk queueing files with DISCFILE macros and a set of network buffers with the BUFFERS macro. When you configure an IMS system that supports unsolicited output, you also create a special file for unsolicited messages with the PRCS macro. In a global network supporting unsolicited output, you must include a SESSION macro for each process file.

Messages coming into the central processor from the terminals are put into a main storage or disk buffer by ICAM. When IMS requests a message from ICAM, ICAM takes it from the buffer and passes it to IMS.

When IMS outputs a message, ICAM puts the message into a main storage or disk queue. After ICAM finishes processing the message, it is transmitted to the appropriate terminal.

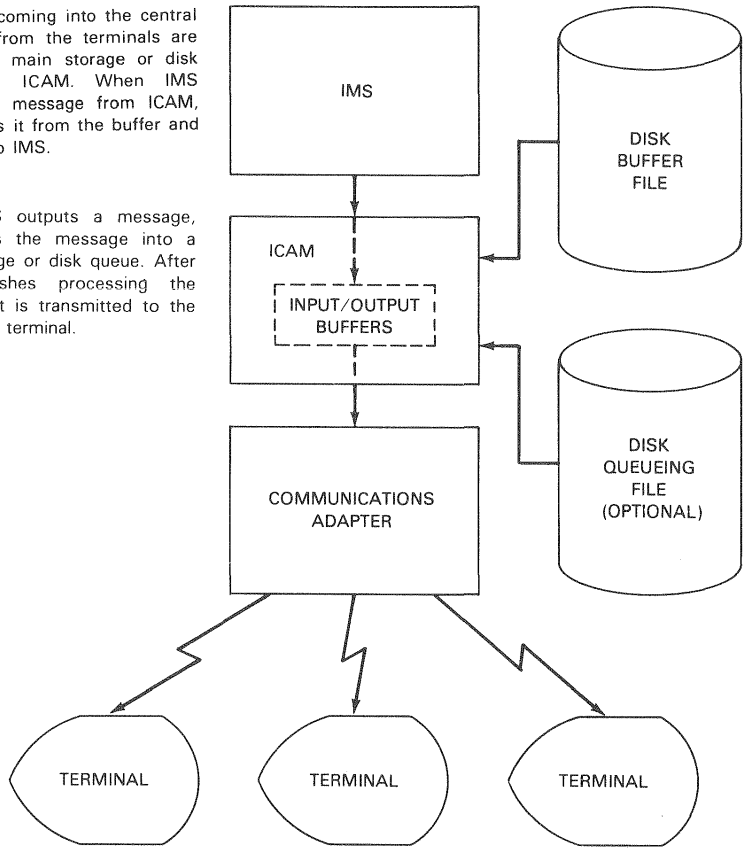


Figure 2-2. How IMS Inputs and Outputs Messages through ICAM

2.3.2. Resident Networks for Single-Thread or Multithread IMS

When you use a resident ICAM to support IMS, you can have the following features:

- A dedicated or global network
- Up to eight communications lines, each on a separate single-line communications adapter
- Any number of terminals, which may consist of:
  - UNISCOPE®100 Display Terminals
  - UNISCOPE 200 Display Terminals

UNISCOPE is a registered trademark of Unisys Corporation.

  - Directly or remotely connected workstations
  - Unisys UTS 400 Universal Terminal System terminals
  - Unisys UTS 4000 Universal Terminal System terminals (a family of terminals including UTS 10, 20, and 40)
  - TELETYPE® (TTY) Terminals
  - IBM® 3270 Terminal System terminals in interactive mode

  In addition, you can attach these auxiliary devices to those terminals that support them:

  - Unisys 8541 Communications Output Printer (COP)
  - Unisys Model 800 Terminal Printer (TP)
  - Unisys 0786 and 0791 Printer Subsystems
  - Unisys Model 610 Tape Cassette System (TCS)
  - Unisys 8406 Diskette Subsystem

- Unattended answering and automatic dialing

With a resident ICAM, you have to create more than one network buffer with the BUFFERS macro. The number of buffers you'll need and their size depends on the amount of activity you anticipate and the sizes of most of your messages. You must create at least one output queue for each terminal. You can have the messages associated with a particular queue held in either main storage buffers or a disk queueing file. If you have heavy activity, you may want to use disk queueing to save main storage space. However, you'll sacrifice some processing speed.

TELETYPE is a registered trademark of Teletype Corporation.
IBM is a registered trademark of International Business Machines Corporation.

When you code the ICAM network definition macros, you must meet the following specifications:

- For a dedicated network, you must specify TYPE=(TCI) on the CCA macro to get a transaction control interface ICAM module. (For a global network, you specify TYPE=(GBL,,node) and GAWAKE=YES on the CCA macro and TYPE=(TCI) on the LOCAP macro.) The name of the network given on the label of the CCA macro must match the names given with the NAME parameter in the NETWORK section of the configurator input and the CCA parameter of the IMSCONF jproc. The default name for both keywords is IMS4. Similarly, the password name specified with the PASSWORD operand in the CCA macro must match the password name given in the PASSWORD parameter of the NETWORK input to the configurator. You must also specify SAVE=YES to save the CCA object module, which is needed at configuration time. If the network definition is for a single-thread IMS system, you must specify FEATURES=(OUTDELV) for shutdown processing. You should also include TRACEMAX on the FEATURES operand when debugging, but not in a production environment.

- On the BUFFERS macro, you declare the number of network buffers and their size, in words. The network buffer size should be a minimum of 256 words.

- If you use local workstations and have an output message size larger than 2400 bytes (including control characters), specify the LBL operand on the LINE macros. Refer to the OUTSIZE parameter of the configurator ACTION section to calculate message size. The default size for the LBL operand is 600 words (2400 bytes). If you use screen format services, specify at least 900 words. Refer to *Screen Format Services Technical Overview*, UP-9977.

- On the TERM macros, you use the LOW operand to create a queue for each terminal. (You can also use the HIGH and MEDIUM parameters, but this is not usually necessary unless your IMS system supports unsolicited output.) Do not specify DICE=OFF for an IBM 3270 terminal or for any terminal from which you intend to use one of the following IMS features:

  - UNIQUE
  - CHTBL, DITBL, DLMSG, DLOAD, JI, SWTCH, and ZSTAT transaction codes
  - ZZCLS, ZZOPN, and ZZTST master terminal commands
  - Screen format services

- You use the DISCFILE macro to identify the disk files that ICAM is to use for the input buffers and for output disk queueing (if used). You must allocate and initialize these files through job control sometime before the first time you execute online IMS or at initial start-up (Figure 5-1). The names for the disk files must match the names on the LFD statements when the files are created.

Figure 2-3 shows a dedicated network definition for a resident ICAM module that supports a single-thread or multithread IMS system without unsolicited output. Figure 2-4 shows the communications network.

```
                        Macro                                    Explanation

1       10      16                                  72
--------------------------------------------------//---
IMS1    CCA     TYPE=(TCI),                         X       Gives name of network and type of
                                                              interface
                PASSWORD=IMC,                       X       Gives network password
                SAVE=YES,                           X       Specifies that CCA object module is
                                                              saved in $Y$OBJ
                FEATURES=(OPCOM,OUTDELV)                    Allows console operator communications
                                                              with ICAM and includes module needed
                                                              for shutdown processing in
                                                              single-thread

        BUFFERS 40,                                 X       Creates 40 main storage output buffers
                64,                                 X       Specifies size of buffers as 64 words
                 5,                                 X       Gives a threshold value of 5 buffers
                                                              (12.5%)
                 ARP=25                                     Gives the number of ARPs to be
                                                              available to ICAM

LNE1    LINE    DEVICE=(UNISCOPE),                  X       Specifies type of terminals on line
                TYPE=(2400,UNAT,SWCH,SYNC),         X       Gives characteristics of the line
                ID=04                                       Gives port number of communications
                                                              adapter that this line is to use

TRM1    TERM    ADDR=(28,51),                       X       Gives address of terminal 1
                FEATURES=(U200,1920),               X       Identifies terminal as UNISCOPE 200
                                                              with screen size of 1920 characters
                LOW=MAIN                                    Creates queue to be used with main
                                                              storage buffers

TRM2    TERM    ADDR=(28,52),                       X       Gives address of terminal 2
                FEATURES=(U100,960),                X       Identifies terminal as UNISCOPE 100
                                                              with size of 960 characters
                 LOW=MAIN                                   Creates queue to be used with main
                                                              storage buffers

TRM3    TERM    ADDR=(29,53),                       X       Gives address of terminal 3
                FEATURES=(U100,960),                X       Identifies terminal as UNISCOPE 100
                                                              with screen size of 960 characters
                LOW=MAIN                                    Creates low-level queue to be used
                                                              with main storage buffers

TRM4    TERM    ADDR=(29,54),                       X       Gives address of terminal 4
                FEATURES=(U100,960),                X       Identifies terminal as UNISCOPE 100
                                                              with screen size of 960 characters
                LOW=MAIN                                    Creates queue to be used with main
                                                              storage buffers

LNE2    LINE    DEVICE=(TTY,33),                    X       Specifies type of terminal on line
                TYPE=(110,SWCH),                    X       Gives characteristics of line
                CALL=5424100                                Specifies phone number of line
```

Figure 2-3. Network Definition for Resident ICAM (Part 1 of 2)

```
                    Macro                                      Explanation

1          10    16                                72
-----------------------------------------//------
TRM5       TERM  FEATURES=(TTY),                   X    Identifies terminal as TELETYPE
                 LOW=MAIN                               Creates queue to be used with main
                                                          storage buffers

LNE3       LINE  DEVICE=(UNISCOPE),                X    Specifies type of terminals on line
                 TYPE=(2400,SWCH,SYNC,UNAT)             Gives characteristics of line
                 ID=06

TRM6       TERM  ADDR=(29,51),                     X    Gives address of terminal 6
                 FEATURES=(U400,1920),             X    Identifies terminal as UTS 400 with
                                                          screen size of 1920 characters
                 LOW=MAIN                               Creates queue to be used with main
                                                          storage buffers

TRM7       TERM  ADDR=(29,52),                     X    Gives address of terminal 7
                 FEATURES=(U400,1920),             X    Identifies terminal as UTS 400 with
                                                          screen size of 1920 characters
                 LOW=MAIN                               Creates queue to be used with main
                                                          storage buffers

TCIDTF     DISCFILE MSGSIZE=1920                        Identifies file to be used for input
                                                          buffer; maximum message size is 1920
                                                          bytes.

           ENDCCA                                       Ends ICAM definition
```

Figure 2-3. Network Definition for Resident ICAM (Part 2 of 2)

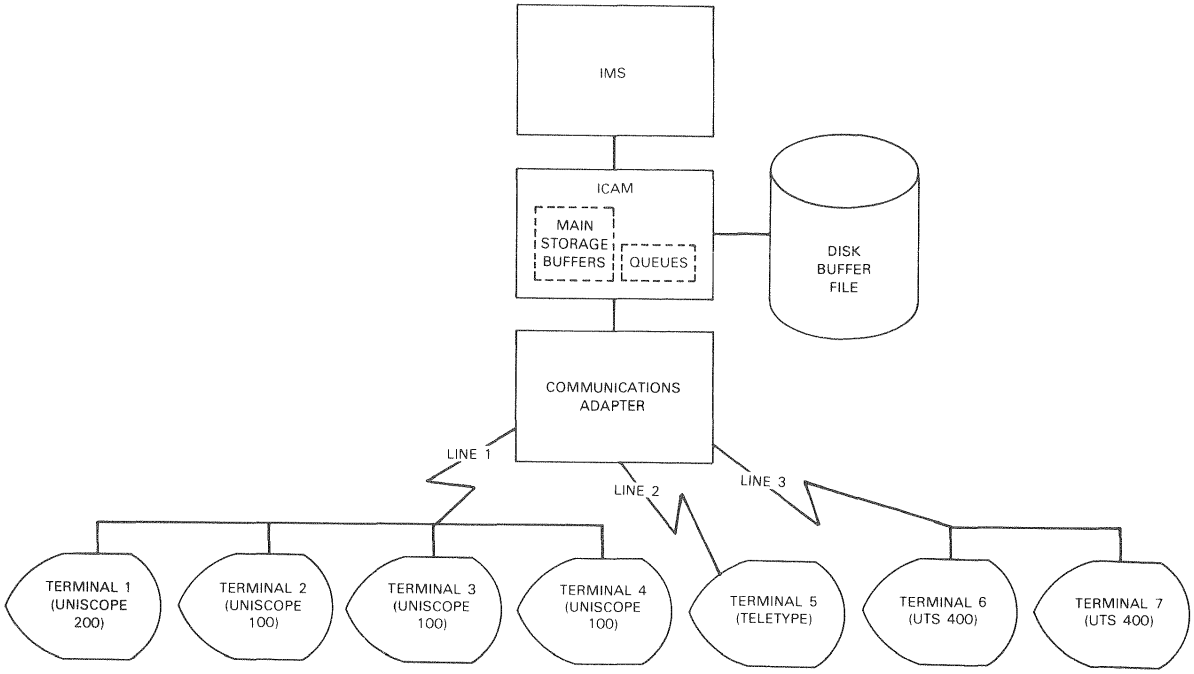


Figure 2-4. Sample Dedicated Communications Network for Resident ICAM

Network Definition for a System Supporting Unsolicited Output

If you configure an IMS system that supports unsolicited output, you must create a resident ICAM module that supports unsolicited output when you want to use:

- The SEND function
- The SWTCH transaction
- The ZZTST terminal command
- Continuous output
- Downline loading

In addition to the requirements already mentioned for a resident ICAM, you must meet the following additional specifications in your ICAM network definition:

- Using the HIGH, MEDIUM, and LOW parameters on the TERM macros, you must create three output message queues for each terminal. ICAM will output all the messages in the high-level queue before it begins to transmit messages from the medium-level queue, then the low-level queue. It is recommended that you have the messages for the low-level queue kept on disk to keep the size of your ICAM module down.
- Using the PRCS macro, you must create a special queue called a process file. Unsolicited messages are kept buffered until the end of the action. It is recommended that you create an additional disk buffer file for this purpose. When using multithread IMS systems in which multiple actions use unsolicited output concurrently, create at least the same number of process files as specified in the PRCSNUM keyword in the NETWORK section of the IMS configuration.
- You must specify FEATURES=(OUTDELV) on the CCA macro for a multithread system supporting continuous output. (It is always required for a single-thread system.)

Figure 2-5 shows the network definition macros for a resident ICAM module that supports an IMS system with unsolicited and continuous output. It defines the communications network shown in Figure 2-6. The network definition is the same as in Figure 2-4, except that a communications output printer, a terminal printer, and specifications required for unsolicited and continuous output have been added. The macros and parameters shown in the previous example (Figure 2-3) are shaded; the new specifications are unshaded.

```
                 Macro                                      Explanation

1        10     16                                72
--------------------------------------------//----
IMS1     CCA    TYPE=(TCI),                       X
                PASSWORD=IMC,                     X
                SAVE=YES,                         X
                FEATURES=(OPCOM,OUTDELV)
         BUFFERS 40,                              X
                  64,                             X
                   5,                             X
                  ARP=25

LNE1     LINE   DEVICE=(UNISCOPE),                X
                TYPE=(2400,UNAT,SWCH,SYNC),       X
                ID=04

TRM1     TERM   ADDR=(28,51),                     X
                FEATURES=(U200,1920),             X
                HIGH=MAIN,                        X     Creates high-level queue to be used
                                                          with main storage buffers
                MEDIUM=MAIN,                      X     Creates medium-level queue to be used
                                                          with main storage buffers
                LOW=DQFILE1                             Creates low-level queue to be used
                                                          with disk file buffer
TRM2     TERM   ADDR=(28,52),                     X
                FEATURES=(U100,960),              X
                HIGH=MAIN,                        X     Creates high-level queue to be used
                                                          with main storage buffers
                MEDIUM=MAIN,                      X     Creates medium-level queue to be used
                                                          with main storage buffers
                LOW=DQFILE1                             Creates low-level queue to be used
                                                          with disk file buffer
TRM3     TERM   ADDR=(29,53),                     X
                FEATURES=(U100,960),              X
                HIGH=MAIN,                        X     Creates high-level queue to be used
                                                          with main storage buffer
                MEDIUM=MAIN,                      X     Creates medium-level queue to be used
                                                          with main storage buffers
                LOW=DQFILE1                             Creates low-level queue to be used
                                                          with disk file buffer
TRM4     TERM   ADDR=(29,54),                     X
                FEATURES=(U100,960),              X
                AUX1=(COP,73),                    X     Terminal has communications output
                                                          printer
                HIGH=MAIN,                        X     Creates high-level queue to be used
                                                          with main storage buffers
                MEDIUM=MAIN,                      X     Creates medium-level queue to be used
                                                          with main storage buffers
                LOW=DQFILE1                             Creates low-level queue to be used
                                                          with disk file buffer
```

Figure 2-5. Network Definition for ICAM Supporting Unsolicited Output (Part 1 of 2)

```
              Macro                                  Explanation

1        10     16                         72
-------------------------------------//----
LNE2     LINE   DEVICE=(TTY,33),            X
                TYPE=(110,SWCH),            X
                CALL=5424100
TRM5     TERM   FEATURES=(TTY),             X
                HIGH=MAIN,                  X    Creates high-level queue to be used
                                                   with main storage buffers
                MEDIUM=MAIN,                X    Creates medium-level queue to be used
                                                   with main storage buffers
                LOW=DQFILE1                 X    Creates low-level queue to be used
                                                   with disk file buffer
LNE3     LINE   DEVICE=(UNISCOPE),          X
                TYPE=(2400,SWCH,SYNC,UNAT)
                ID=06                       X
TRM6     TERM   ADDR=(29,51),               X
                FEATURES=(U400,1920),       X
                HIGH=MAIN,                  X    Creates high-level queue to be used
                                                   with main storage buffers
                MEDIUM=MAIN,                X    Creates medium-level queue to be used
                                                   with main storage buffers
                LOW=DQFILE1                 X    Creates low-level queue to be used
                                                   with disk file buffer
TRM7     TERM   ADDR=(29,52),               X
                FEATURES=(U400,1920),       X
                AUX1=TP,77)                 X    Terminal has terminal printer
                HIGH=MAIN,                  X    Creates high-level queue to be used
                                                   with main storage buffers
                MEDIUM=MAIN,                X    Creates medium-level queue to be used
                                                   with main storage buffers
                LOW=DQFILE1                 X    Creates low-level queue to be used
                                                   with disk file buffer

P001     PRCS   LOW=DQFILE2                      Creates a single-level process file
                                                   used with disk file buffer

DQFILE1  DISCFILE  FILEDIV=8                     Identifies file to be used for output
                                                   disk file buffer

DQFILE2  DISCFILE  FILEDIV=8                     Identifies file to be used with
                                                   process-file for unsolicited output

TCIDTF   DISCFILE  MSGSIZE=1920                  Ends ICAM definition

         ENDCCA
```

Figure 2-5. Network Definition for ICAM Supporting Unsolicited Output (Part 2 of 2)

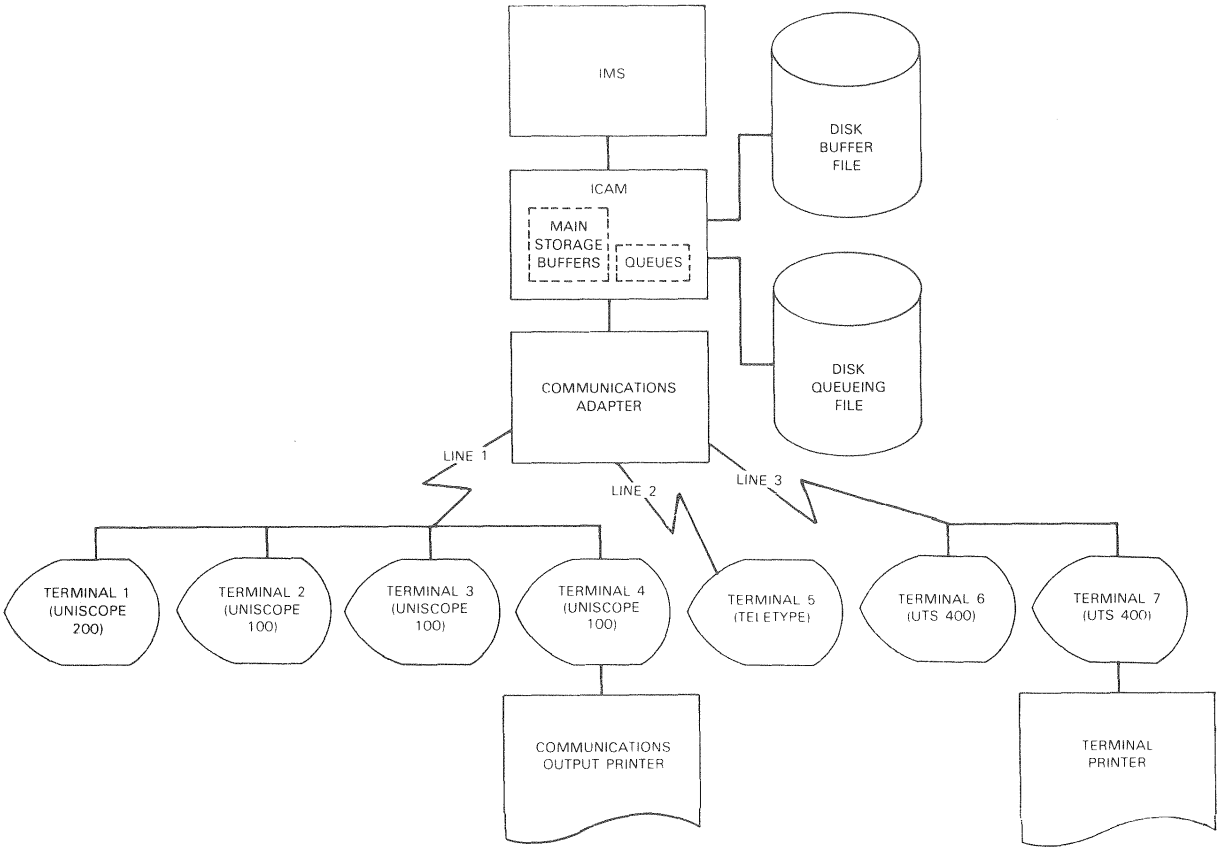


Figure 2-6. Sample Communications Network for ICAM Supporting Unsolicited Output

Network Definition for a Global Network

You can use a global network for more than one IMS program and for other communications user programs that use the ICAM standard interface. A global network requires a resident ICAM. To use a global network, you must include the CUP parameter in the configurator NETWORK section.

You write a network definition for a global network the same way as for a dedicated network, with several modifications:

- On the CCA macro, you must specify TYPE=(GBL,,node) to identify a global network and to name the computer system using the network. You must also specify GAWAKE=YES to allow dynamic sessions.

- You use the LOCAP macro to identify your online IMS program as a local application. The label on the LOCAP macro must match the CUP parameter specification in the configurator NETWORK section. You must specify TYPE=(TCI) to get a transaction control interface. This specification is available to IMS users only and is not documented in ICAM manuals.

- You use SESSION macros to identify static session terminals. Static session terminals are always attached to your IMS program when it is online. Terminals that you identify with TERM macros but do not name in SESSION macros are dynamic session terminals. Dynamic session terminals can be attached to and detached from your IMS program (or any other program using the global network definition) while your program is running. For information on attaching and detaching dynamic terminals, refer to the *IMS Operations Guide,* UP-12027.

  You also use SESSION macros to identify one or more process files in static sessions with IMS. This is required for an IMS system with unsolicited output.

Figure 2-7 shows the network definition macros for the global communications network shown in Figure 2-8. The network definition is the same as in Figure 2-5, except for the global specifications. The macros and parameters previously shown are shaded; the new entries are unshaded.

```
                    Macro                                          Explanation

1         10     16                                      72
----------------------------------------------------//----
IMS1      CCA    TYPE=(GBL,, NOD1),                      X     Identifies a global network and the
                 PASSWORD=IMC,                           X       local node
                 SAVE=YES,                               X
                 FEATURES=(OPCOM,OUTDELV),               X
                 GAWAKE=YES                                    Allows dynamic sessions
          BUFFERS 40,                                    X
                   64,                                   X
                   5,                                    X
                   ARP=25
IMS9      LOCAP  TYPE=(TCI)                                    Gives name of IMS program
                                                                 and type of interface
LNE1      LINE   DEVICE=(UNISCOPE),                      X
                 TYPE=(2400,UNAT,SWCH,SYNC),             X
                 ID=04
TRM1      TERM   ADDR=(28,51),                           X
                 FEATURES=(U200,1920),                   X
                 HIGH=MAIN,                              X
                 MEDIUM=MAIN,                            X
                 LOW=DQFILE1

TRM2      TERM   ADDR=(28,52),                           X
                 FEATURES=(U100,960),                    X
                 HIGH=MAIN,                              X
                 MEDIUM=MAIN,                            X
                 LOW=DQFILE1
```

Figure 2-7. Global Network Definition (Part 1 of 2)

```
                    Macro                                    Explanation

1         10    16                                    72
                                                      //
TRM3      TERM  ADDR=(29,53)                          X
                FEATURES=(U100,960),                  X
                HIGH=MAIN,                            X
                MEDIUM=MAIN,                          X
                LOW=DQFILE1

TRM4      TERM  ADDR=(29,54),                         X
                FEATURES=(U100,960),                  X
                HIGH=MAIN,                            X
                MEDIUM=MAIN,                          X
                LOW=DQFILE1

LNE2      LINE  DEVICE=(TTY,33),                      X
                TYPE=(110,SWCH),                      X
                CALL=5424100

TRM5      TERM  FEATURES=(TTY),                       X
                HIGH=MAIN,                            X
                MEDIUM=MAIN,                          X
                LOW=DQFILE1

LNE3      LINE  DEVICE=(UNISCOPE),                    X
                TYPE=(2400,SWCH,SYNC,UNAT)
                ID=06

TRM6      TERM  ADDR=(29,51),                         X
                FEATURES=(U400,1920),                 X
                HIGH=MAIN,                            X
                MEDIUM=MAIN,                          X
                LOW=DQFILE1

TRM7      TERM  ADDR=(29,52),                         X
                FEATURES=(U400,1920),                 X
                HIGH=MAIN,                            X
                MEDIUM=MAIN,                          X
                LOW=DQFILE1

P001            PRCS   LOW=DQFILE2

DQFILE1         DISCFILE FILEDIV=8

DQFILE2         DISCFILE FILEDIV=8

          SESSION EU1=(IMS9),EU2=(TRM1)          Identifies TRM1 as a static session
                                                   terminal
          SESSION EU1=(IMS9) EU2=(TRM2)          Identifies TRM2 as a static session
                                                   terminal
          SESSION EU1=(IMS9),EU2=(TRM3)          Identifies TRM3 as a static session
                                                   terminal
          SESSION EU1=(IMS9),EU2=(TRM4)          Identifies TRM4 as a static session
                                                   terminal
          SESSION EU1=(IMS9),EU2=(P001)          Identifies P001 as a static process
                                                   file
          ENDCCA
```

Figure 2-7. Global Network Definition (Part 2 of 2)

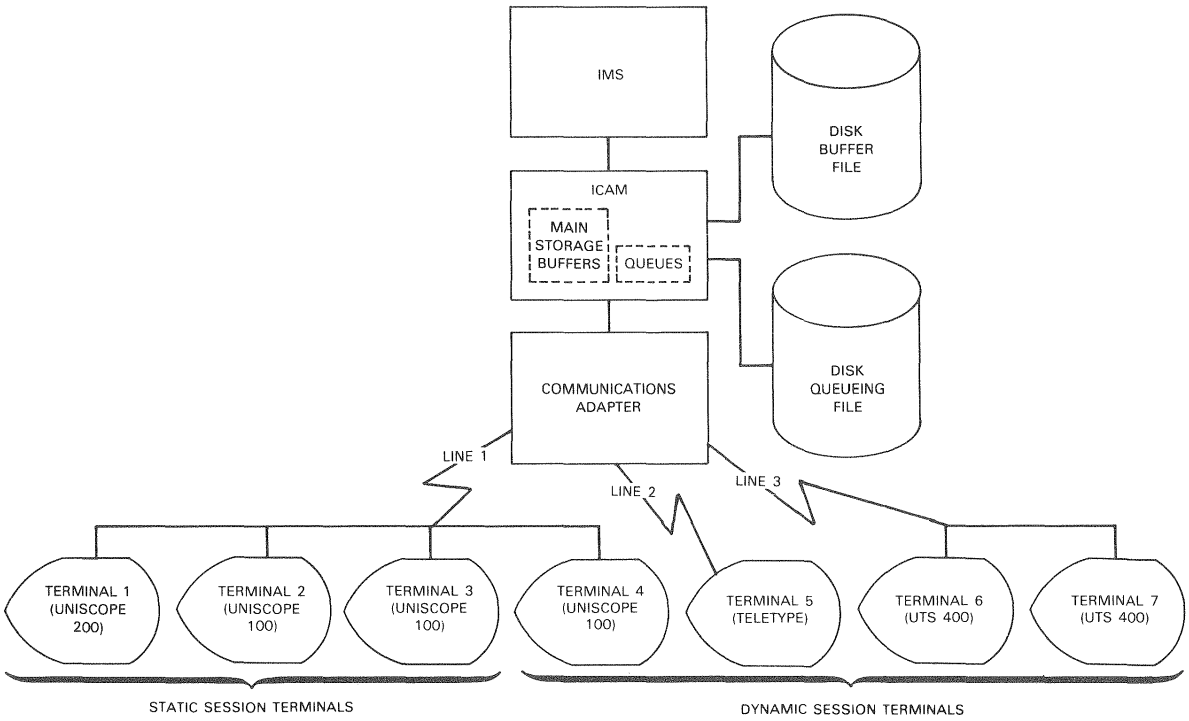


Figure 2-8. Sample Global Communications Network

Network Definition for a Global Network Using Local and Remote Workstations

Figures 2-9 and 2-10 illustrate a global network that uses three local workstations and one remote workstation. Notice in the network definition that there is only one local workstation per line. The remote workstation has a screen bypass device that is defined to ICAM as a separate terminal. Both terminals are on one line and form one polling group.

This network supports three other applications in addition to IMS, through the ICAM standard interface. One of the local workstations, TRM1, is defined in a SESSION macro as a static terminal for CUPP and is not available to IMS. The other three workstations are dynamic terminals and can be attached to IMS or any of the other applications.

Note that this network does not support unsolicited or continuous output. To support unsolicited and continuous output, it would require the FEATURES=(OUTDELV) operand on the CCA macro, three queues for each terminal, and a process file in static session with IMS.

```
                        Macro                                   Explanation

1       10    16                                  72
--------------------------------------------------//-----
C007    CCA   TYPE=(GBL,,A),                      X
              SAVE=YES,                         X
              FEATURES=(OPCOM),                 X
              GAWAKE=YES
        BUFFERS 20,512,1,ARP=42,                X
              STAT=YES

IMS1    LOCAP TYPE=(TCI)                            Names IMS program using TCI interface
CUPN    LOCAP TYPE=(STDMCP),LOW=MAIN                Identifies an application using the
CUPO    LOCAP TYPE=(STDMCP),LOW=MAIN                  interface
CUPP    LOCAP TYPE=(STDMCP),LOW=MAIN
LNE1    LINE  DEVICE=(LWS),STATS=YES                Defines type of terminal on LNE 1 as
                                                      local workstation
TRM1    TERM  FEATURES=(LWS),ADDR=(315),  X         Gives address of terminal and creates
              LOW=MAIN,HIGH=MAIN,         X           low and high queues. This terminal is
              INPUT=PRF1                              not available to IMS. (See SESSION
                                                      macro.)
LNE2    LINE  DEVICE=(LWS),STATS=YES,     X         Defines type of terminal on LNE2 as
              LBL=1000                                local workstation and gives line
                                                      buffer length of 1000 words

TRM2    TERM  FEATURES=(LWS),ADDR=(316),  X         Gives address of terminal and creates
              LOW=MAIN,HIGH=MAIN                      low and high queues. TRM2, TRM3, and
                                                      TRM4 are available to IMS.
LNE3    LINE  DEVICE=(LWS),STATS=YES,     X
              LBL=1000

TRM3    TERM  FEATURES=(LWS),ADDR=(317),  X
              LOW=MAIN,HIGH=MAIN
        PGROUP PGID=28                              Identifies polling group

LNE4    LINE  DEVICE=(RWS),               X         Specifies that the communications line
              TYPE=(9600,SWCH,SYNC)       X           LNE4 uses remote workstation protocol
              ID=9

TRM4    TERM  FEATURES=(U40,,,PRIMARY),   X         Specifies that the remote workstation is
              ADDR=(28,51),LOW=MAIN,      X           the primary screen of a UTS 40.
              HIGH=MAIN,AUX1=(COP,73)                 Identifies address of workstation in
                                                      polling group

TRM5    TERM  FEATURES=(U40,,,SECONDARY), X         Specifies that the remote workstation is
              ADDR=(28,52),LOW=MAIN,      X           a UTS40 screen bypass device.
              HIGH=MAIN,AUX1=(COP,73)                 Identifies address of screen bypass
                                                      device in polling group

PRF1    PRCS  LOW=MAIN

        SESSION EU1=(CUPP),EU2=(TRM1)               Identifies TRM1 as a static session
                                                      terminal for CUPP
        SESSION EU1=(PRF1),EU2=(CUPP)               Identifies PRF1 as a static process file
                                                      for CUPP
        ENDCCA
```

Figure 2-9. Network Definition for Global ICAM with Workstations

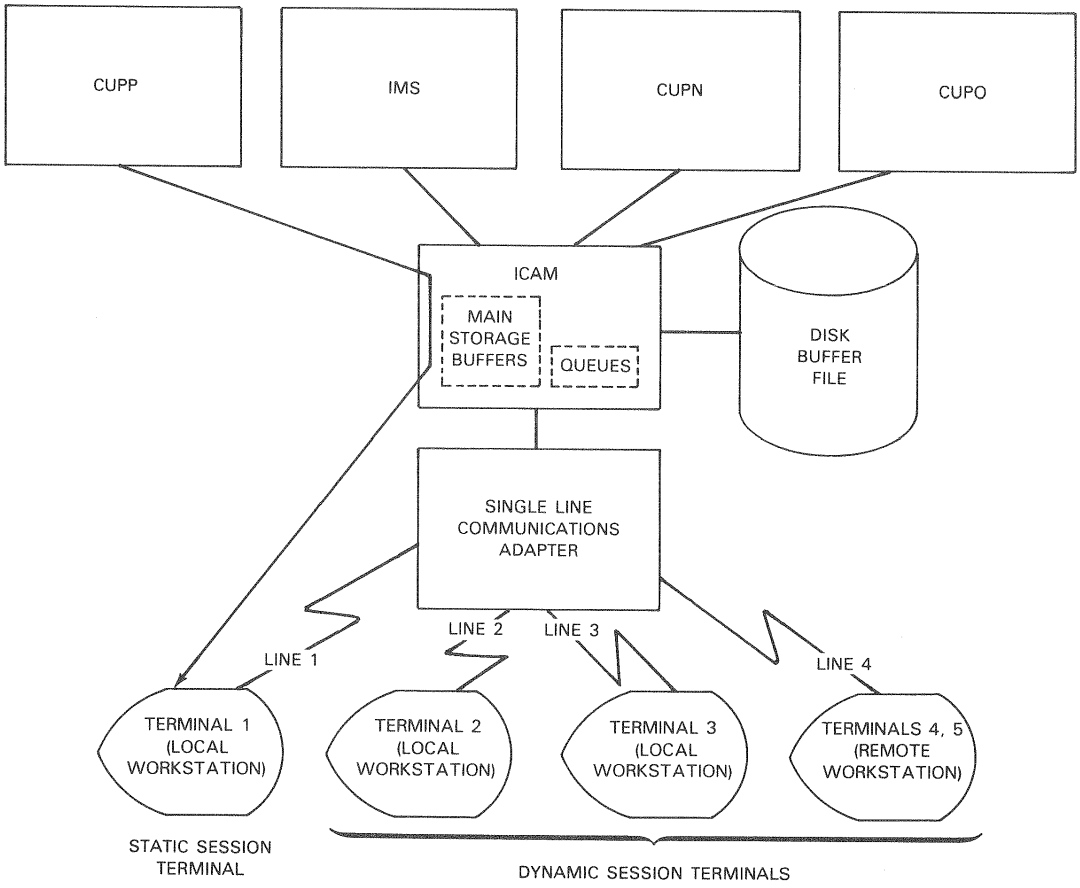


Figure 2-10. Sample Communications Network for Global ICAM with Workstations

Network Definition for a Global Network Supporting Distributed Data Processing

Figures 2-11 and 2-12 illustrate a global network that supports distributed data processing. The network contains two computers. Either computer can route transactions to the other for processing or process transactions it receives from the other. Figure 2-11 illustrates the definition of the primary computer, and Figure 2-12 illustrates the definition of the secondary computer.

```
                    Macro                                     Explanation

1          10    16                          72
---------------------------------------------//---
C3N1       CCA   TYPE=(GBL,,NODA),            X
                 FEATURES=(OUTDELV,OPCOM),    X
                 PASSWORD=IMSNET01,           X
                 GAWAKE=YES,                  X
                 SAVE=YES,                    X
                 DCA=YES                           Indicates support of distributed
                                                     communications architecture

           BUFFERS 100,                       X
                 64,                          X
                 1,
                 ARP=50,                      X
                 LINKPAK=(50,80,10)                Specifies the number of 80-word link
                                                     buffers used with distributed data
                                                     processing and threshold value

IMS9       LOCAP TYPE=(TCI),                  X
                 LOW=MAIN,                    X
                 MEDIUM=MAIN,                 X
                 HIGH=MAIN

C2L1       LINE  DEVICE=(UNISCOPE),           X
                 TYPE=(2400,SWCH,SYNC)        X
                 CALL=3589                         Specifies the telephone number used to
                                                     dial a terminal

TRM1       TERM  ADDR=(28,51),                X
                 FEATURES=(U100,960),         X
                 HIGH=MAIN,                   X
                 MEDIUM=MAIN,                 X
                 LOW=MAIN,                    X
                 INPUT=(YES)                       Creates an input message queue for this
                                                     terminal

TRM2       TERM  ADDR=(28,52),X               X
                 FEATURES=(U100,960),         X
                 HIGH=MAIN,                   X
                 MEDIUM=MAIN,                 X
                 LOW=MAIN,                    X
                 INPUT=(YES)

TRM3       TERM  ADDR=(29,53),                X
                 FEATURES=(U100,960),         X
                 HIGH=MAIN,                   X
                 MEDIUM=MAIN,                 X
                 LOW=MAIN,                    X
                 INPUT=(YES)
```

Figure 2-11. Network Definition for Global ICAM Supporting Distributed Data Processing (Primary Computer) (Part 1 of 3)

Macro				Explanation
1	10	16	72	
TRM4	TERM	ADDR=(29,54),	X	
		FEATURES=(U100,960),	X	
		HIGH=MAIN,	X	
		MEDIUM=MAIN,	X	
		LOW=MAIN,	X	
		INPUT=(YES)		
LNE6	LINE	DEVICE=(UNISCOPE),	X	
		TYPE=(9600,SYNC),	X	
		ID=15		
TRME	TERM	ADDR=(28,51),	X	
		FEATURES=(U200,1920)	X	
		HIGH=MAIN,	X	
		MEDIUM=MAIN,	X	
		LOW=MAIN,	X	
		INPUT=(YES)		
P001	PRCS	LOW=MAIN		Creates a process file for temporarily storing messages
	SESSION	EU1=(IMS9),	X	Specifies the name of a local end user defined in the label of a LOCAP macro
		EU2=(P001)		Specifies the name of a local end user defined in the label of a PRCS macro
V3N1	VLINE	DEVICE=(ABM,PRIMARY),	X	Specifies this line is used for communication between two OS/3 systems and that this processor is the primary computer
		ID=10,	X	Identifies the line number of the line communications adapter
		TYPE=(9600,FLDQ),	X	Specifies the line speed
		CMADDR=3,	X	Specifies the universal data link control frame level address used to transmit commands and receive responses
		RSPADDR=1		Specifies the universal data link control frame level address used to transmit responses and receive commands

Figure 2-11. Network Definition for Global ICAM Supporting Distributed Data Processing (Primary Computer) (Part 2 of 3)

Macro				Explanation
1	10	16	72	
	LPORT	EU1=IMS9,	X	LPORT creates a remote session entry table for each single logical port; EU1= specifies the label of the LOCAP macro defining the local end user.
		EU2=IMSR,	X	Specifies the label of the LOCAP macro defining the remote end user
		USERTP=STDMCP,	X	Indicates DDP operates through ICAM's standard interface
		CATP=C,	X	Indicates error recovery by port flow control and provision of assurance units by the DDP interface
		LINE=V3N1,	X	The label of the VLINE macro
		REMOTE=NODB,	X	The name of the destination node
		PORT=4,	X	The port number
		RWNDW=2		The port window level
IMSR	LOCAP	TYPE=(TCI),	X	
		REMOTE=(NODB)		The name of the remote computer node where this locap file is located
	ENDCCA			

Figure 2-11. Network Definition for Global ICAM Supporting Distributed Data Processing (Primary Computer) (Part 3 of 3)

```
                    Macro                                   Explanation

1         10    16                                  72
-------------------------------------------------//------
C3N2      CCA   TYPE=(GBL,,NODB),                   X
                FEATURES=(OUTDELV,OPCOM),           X
                PASSWORD=IMSNET01,                  X
                GAWAKE=YES,                         X
                SAVE=YES,                           X
                DCA=YES
          BUFFERS 100,                              X
                64,                                 X
                1,                                  X
                ARP=50,                             X
                LINKPAK=(50,80,10)

IMSR      LOCAP TYPE=(TCI),                         X
INE5      LINE  DEVICE=(UNISCOPE),                  X
                TYPE=(2400,SYNC,SWCH,UNAT),         X
                CALL=4408,                          X
                ID=12

TRMA      TERM  ADDR=(28,51),                       X
                FEATURES=(U400,1920),               X
                HIGH=MAIN,                          X
                MEDIUM=MAIN,                        X
                LOW=MAIN,                           X
                INPUT=(YES)

TRMB      TERM  ADDR=(28,52),                       X
                FEATURES=(U400,1920),               X
                HIGH=MAIN,                          X
                MEDIUM=MAIN,                        X
                LOW=MAIN,                           X
                INPUT=(YES)

TRMC      TERM  ADDR=(28,53),                       X
                FEATURES=(U400,1920),               X
                HIGH=MAIN,                          X
                MEDIUM=MAIN,                        X
                LOW=MAIN,                           X
                INPUT=(YES)

TRMD      TERM  ADDR=(28,54),                       X
                FEATURES=(U400,1920),               X
                HIGH=MAIN,                          X
                MEDIUM=MAIN,                        X
                LOW=MAIN,                           X
                INPUT=(YES)
```

Figure 2-12. Network Definition for Global ICAM Supporting Distributed Data Processing (Secondary Computer) (Part 1 of 2)

Macro				Explanation
1	10	16	72	
LNE6	LINE	DEVICE=(UNISCOPE),	X	
		TYPE=(9600,SYNC),	X	
		ID=15		
TRME	TERM	ADDR=(28,51),	X	
		FEATURES=(U400,1920),	X	
		HIGH=MAIN,	X	
		MEDIUM=MAIN,	X	
		LOW=MAIN,	X	
		INPUT=(YES)		
P001	PRCS	LOW=MAIN		
	SESSION	EU1=(IMSR),	X	
		EU2=(P001)		
V3N2	VLINE	DEVICE=(ABM,PRIMARY),	X	
		ID=05,	X	
		TYPE=(9600,FLDQ),	X	
		CMADDR=1,	X	Specifies universal data link control frame level address used to transmit commands and receive responses
		RSPADDR=3		Specifies universal data link control frame level address used to transmit responses and receive commands
	LPORT	EU1=IMSR,	X	The label of the LOCAP macro defining the local end user
		EU2=IMS9,	X	The label of the LOCAP macro defining the remote end user
		USERTP=TCI,	X	
		CATP=C,	X	
		LINE=V3N2,	X	The label of the VLINE macro
		REMOTE=NODA,	X	The name of the destination node
		PORT=4,	X	
		RWNDW=2		
IMS9	LOCAP	TYPE=(TCI),	X	
		REMOTE=(NODA)		The name of the remote computer node where this locap file is located
	ENDCCA			

Figure 2-12. Network Definition for Global ICAM Supporting Distributed Data Processing (Secondary Computer) (Part 2 of 2)

2.4. Communications Message Format Control

Several methods are available to the action programmer for controlling the formatting of input and output messages at remote terminals. The most convenient of these are device-independent control expressions (DICE) and field control characters (FCCs).

DICE and FCC sequences in message text are detected and processed by ICAM. If you specify editing of input messages to the configurator (EDIT and FCCEDIT keyword parameters in the ACTION section), DICE and FCC sequences are deleted from input messages before they are passed to your action program.

2.4.1. DICE Sequences

The device-independent control expression (DICE) is a 4-character hexadecimal sequence inserted into the text of an output message to control positioning and functions on remote terminal devices. DICE functions provided by ICAM include carriage return, cursor positioning, forms control, line control, tab control, line feed, and screen erasure. They are documented in the IMS action programming manuals, with examples of their use.

2.4.2. Field Control Characters

If you have UTS terminals or local workstations in your network, you can use field control character (FCC) sequences in your action program or your UTS program. The FCC is a 5-byte hexadecimal sequence that begins with the character US (hexadecimal 1F). It is similar to DICE but provides additional control over data formatting. With FCC sequences in the output message, your program can highlight selected elements of data to automatically reject the wrong type of data or prevent the entry or change of any data in a given field. You may also use FCCs to specify that only changed or variable data be transmitted.

For details on the use of FCC sequences, refer to the appropriate reference manual for your terminal system.

Note: *Output messages sent to a console configured for transaction processing are not edited. Most bytes of DICE and FCC sequences are processed as blanks, but in a few cases they are printed as characters on the console screen.*

Section 3
Pre-Online Processing

3.1. General

The term *pre-online processing* refers to all of the IMS utilities and procedures that you may execute beforehand to define your particular online operating environment. The term excludes other activities such as creating the conventional ISAM, MIRAM, SAM, or DAM files that are to be processed by your IMS application; those you create offline. It also does not include the programming necessary to produce your COBOL, BAL, or RPG II action programs.

Pre-online processing includes the following:

- Allocating and initializing the named record (NAMEREC) file, using the IMS utility program, ZP#NRU
- Password definition, using the same NAMEREC file utility, ZP#NRU
- Data definition, using the data definition processor (DT3DF)
- Edit table generation, using the IMS utility, ZH#EDT
- Configuring the online IMS system

The NAMEREC file must be initialized before any other pre-online processing can take place, because all of the IMS processors, including the configurator, contribute records to this file. You initialize the file either by executing the NAMEREC file utility or as part of the configuration process, in which case you must configure before doing any other pre-online processing.

The other pre-online processing functions may be accomplished in any convenient order, and all except configuration are optional, depending on your individual requirements. Password definition is available only for UNIQUE. Data definition is required only if you use UNIQUE or if your action programs access defined files. Edit table generation is not applicable to UNIQUE and is optional for user action programs.

Initialization of the NAMEREC file and password definition are described in this section and configuration in Section 4. Data definition is described in the *IMS Data Definition and UNIQUE Programming Guide*, UP-9209, and edit table generation is discussed in the IMS action programming manuals.

3.2. The Named Record File Utility

You may use the named record (NAMEREC) file utility, ZP#NRU, to:

- Initialize the NAMEREC file
- Define new password definition records and change existing passwords
- Delete data definition records and password records
- List the contents of the NAMEREC file

The NAMEREC file can be initialized at configuration time with the INIT parameter of the IMSCONF job control procedure (jproc). IMSCONF calls in the ZP#NRU program to initialize, or scratch and reinitialize, the NAMEREC file, but it does not permit password definition or produce a listing of entries in the NAMEREC file. If you initialize the NAMEREC file at configuration, however, you can still execute the ZP#NRU utility to define passwords or to obtain a listing of records in the file.

3.2.1. Initializing the NAMEREC File (INIT)

Initializing the NAMEREC file with the ZP#NRU utility allows you to:

- Define the exact location on disk where the file is to reside (This is not possible when the NAMEREC file is initialized by IMSCONF.)
- Process data definitions, password definitions, and edit tables before configuration

To initialize the NAMEREC file with the ZP#NRU utility, you must include the INIT function parameter in the job control stream. Its format is:

```
INIT BLKSZE=nnnnn
```

where:

`INIT`
: Is the function parameter and must occupy columns 1-4.

`BLKSZE=nnnnn`
: Defines the block size of the NAMEREC file in decimal bytes ranging from 1024 to 12,800. This specification must be a multiple of 256 and must not exceed the track size of the disk subsystem on which the NAMEREC file resides.

The job stream in Figure 3-1 illustrates the execution of the ZP#NRU program to allocate and initialize the NAMEREC file. The file name in the LFD statement must be ISAMNRF. The EXT statement defines the file type -- MIRAM (MI) for a CDM system, ISAM (IS) for a DTF system -- and specifies the size of the file in either cylinders or blocks. For a discussion on determining the size needed for the NAMEREC file, see C.2.2. If you are initializing a previously allocated NAMEREC file, you should omit the EXT statement; however, you must specify INIT on the LFD statement. Omission of the INIT parameter causes ZP#NRU to abnormally terminate with an error code of 62.

```
// JOB NAMEREC
// DVC 20  // LFD PRNTR
// DVC 50  // VOL DS1234
// EXT MI,C,,CYL,20
// LBL NAMEREC  // LFD ISAMNRF,,INIT
// EXEC ZP#NRU
/$
INIT   BLKSZE=6144
/*
/&
// FIN
```

Note:

In the shaded line above, replace MI with IS in a DTF system.

Figure 3-1. Sample Control Stream to Allocate and Initialize the NAMEREC File

After initializing the NAMEREC file for its first use in your pre-online processing, you may need to scratch it and reinitialize it. This becomes necessary if execution of the configurator, data definition processor, edit table generator, or a password run of the NAMEREC utility is abnormally terminated, or if you have any other reason to suspect that the file is compromised. You can scratch and reinitialize the NAMEREC file with the INIT parameter of the configurator jproc, IMSCONF, or with the ZP#NRU utility and the SCR job control statement. After you reinitialize the NAMEREC file, you must rerun all pre-online processors.

Figure 3-2 shows the job control statements needed to scratch and reinitialize an existing NAMEREC file with the ZP#NRU utility and the SCR job control statement. Two device assignment sets are needed for the NAMEREC file: one to deallocate; the other to reallocate and reinitialize it. You can use any file name except ISAMNRF in the LFD statement for the file that is to be scratched; you must use the same file name in the SCR statement. The file identifier in the LBL statement must agree with the LBL name assigned to the file when it was originally allocated.

```
// JOB NEWNMRC
// DVC 20  // LFD PRNTR
// DVC 50  // VOL DS1234  // LBL NAMEREC
// LFD SCRNMRC
// SCR SCRNMRC
// DVC 50  // VOL DS1234
// EXT MI,C,,CYL,20
// LBL NAMEREC  // LFD ISAMNRF,,INIT
// EXEC ZP#NRU
/$
INIT  BLKSZE=6144
/*
/&
// FIN
```

Note:

In the shaded line above, replace MI with IS in a DTF system.

Figure 3-2. Sample Control Stream to Scratch and Reinitialize a Compromised NAMEREC File

3.2.2. Listing the Records in the NAMEREC File (LIST)

The LIST parameter of ZP#NRU lists the keys of all records in the NAMEREC file. The first byte of the key is printed as a hexadecimal value and indicates record type. Possible values are:

Value	Description
C1	Configuration Internal Action Control Record Created during configuration.
C3	Configuration Control Record or Configurator Error Control Record
C6	Configurator Internal File Control Table
C9	Configurator CCA Control Record
D3	Configurator Internal Language/Lexicon Control Record
D7	Configurator Internal Program Record
D9	Restart Record Created during configuration and updated during multithread shutdown; used during restart.
E2	Start Record Created during configuration; used during start-up.

Value	Description
E3	Configurator Internal Terminal/Transaction Record
F1	Dummy Record Created by named record file utility.
FA	Expanded Editing Record Created by edit table generator.
FC	Password Definition Record Created by named record file utility or data definition processor.
FD	Data Definition Record Created by data definition processor.
FE	UNIQUE Lexicon Record Created during configuration.

In addition, the configurator may create other records for internal use.

For configuration records, bytes 2, 3, and 4 of the key specify the control table name; they're printed as alphanumeric characters. Possible values are:

Value	Description
TCT	Terminal Control Table
FCT	File Control Table
PCT	Program Control Table
ACT	Action Control Table
TRN	Transaction Control Table
LCP	LOCAP Control Table

The remaining four bytes of the key contain the binary representation of n in the specification CONFID=n (4.3.1). They're printed as blanks.

Configuration records such as D9 and E2 are printed in sets depending on the number of configurations in the NAMEREC file. For each unique CONFID=n specified to the configurator, a set of D9, E2, and other internal configuration records are generated into the NAMEREC file.

The format of the LIST function parameter is:

```
LIST
```

The LIST parameter must be coded in columns 1-4. See Figure 3-3 for an illustration of LIST specified in a password definition run of the ZP#NRU utility.

3.2.3. Password Definition

The password definition process provides security for your data files when you use UNIQUE. Terminal operators using UNIQUE must enter the appropriate password with the OPEN command to gain access to a defined file or subfile, and you can change this password as frequently as you choose. The password facility is not available to user action programs.

You can define passwords in two different ways during pre-online processing:

1. Using the PASSWORD option in the DEFINED FILE and SUBFILE statements in the data definition. This allows all terminal operators to access the defined file using the defined file name or subfile name.

2. Executing the NAMEREC file utility with the PASSWORD function parameter. This allows you to restrict access to specific terminals and to change passwords.

The password capability of the data definition processor is limited. You either include the PASSWORD option or you omit it. When you include PASSWORD, all terminals can access the defined file or subfile by its actual name (but you can void the use of this name with the NAMEREC utility). When you omit PASSWORD, no terminals can access the file until you define a password with the NAMEREC utility. The PASSWORD option of the data definition processor is discussed in the *IMS Data Definition and UNIQUE Programming Guide,* UP-9209.

When you use the NAMEREC utility to define passwords, you have several options:

1. You can allow a password to be entered from all terminals.

2. You can restrict access to specific terminals.

3. You can void a password defined in the data definition or in a previous run of the NAMEREC utility.

4. You can define multiple passwords for the same defined file or subfile, restricting each to the desired terminals.

Because a defined file or subfile can have more than one password, when you want to change passwords you should delete or void the old password in addition to defining a new one. If you used the PASSWORD option in the data and you now want to restrict access to specific terminals, you must first delete or void the old password and then define a new password, restricting its use to the desired terminals.

To define passwords with the NAMEREC file utility, place the PASSWORD function parameter in the job control stream, in the following form:

```
PASSWORD PID=password-id DDN=data-def-rec-name FN=def-filename

          [ALL                                    ]
          [NONE                                   ]
TID=      {term-1[{Δ} term-2 ]...                 }
          [{,}                                    ]
```

where:

PASSWORD
: Is the function parameter and must be coded in columns 1-8.

PID=password-id
: Designates the 1- to 7-character password name, the first character of which must be alphabetic.

DDN=data-def-rec-name
: Identifies the data definition record in which the defined file or subfile this password accesses is defined. The name of the data definition record is the same as the defined file name.

FN=def-filename
: Identifies the defined file or subfile to which this password authorizes access.

TID=ALL
: Stipulates that all terminals configured in this IMS communications network are authorized to submit the password name for access to the file.

TID=NONE
: Stipulates that no terminal is authorized to submit the password name. Use this specification to delete a previously authorized password. You can also delete passwords with the DELETE parameter (3.2.4).

```
TID=term-1[{Δ} term-2 ]...
          [{,}       ]
```

: Identifies those terminals that are authorized to submit this password. These terminal names must match terminal names specified in the ICAM network definition. Terminal names may be one to four characters in length. If less than four characters, terminal names must be delimited by either commas or spaces; if exactly four characters, they may be coded with or without delimiters (commas or spaces). More than one blank character indicates the end of the terminal definitions, unless a nonblank character is coded in column 72.

You must include all keyword parameters, whether a password is to be added or changed.

Except that the function parameter PASSWORD must be coded in columns 1-8, the parameters may be punched freeform. A nonblank character in column 72 indicates continuation, and continuation lines may start in any column except column 1. Sequence numbers in columns 73-80 are optional. You can define any number of passwords in a single execution.

Figure 3-3 illustrates the execution of the ZP#NRU utility to define passwords and to list the keys of records in the NAMEREC file.

```
// JOB IMPAS
// DVC 20 // LFD PRNTR
// DVC 50 // VOL DS9999 // LBL NAMEREC,DS9999 // LFD ISAMNRF
// EXEC ZP#NRU
/$
PASSWORD  PID=WARES   DDN=WARE   FN=PRODFIL  TID=ALL
PASSWORD  PID=PAYROL   DDN=TAXRCRD  FN=PAYFILE   TID=T1,T2,T3
PASSWORD  PID=CUST  DDN=CUSTM  FN=CUSTFIL  TID=T001T002T003T004T005    X
              T007T008T009T0010T011T012T013T014T015
LIST
/*
/&
// FIN
```

Figure 3-3. Sample Control Stream for Password Definition and Listing of Records in the NAMEREC File

The first PASSWORD statement assigns the password WARES to the defined file PRODFIL and specifies that all terminals may have access to this file. The second statement specifies that only terminals T1, T2, and T3 are authorized to access the defined file PAYFILE by submitting the password PAYROL. The third statement shows the splitting of terminal identifiers on two lines. Notice that the terminal names are four characters in length and have no delimiters; they could optionally have been separated by commas or spaces.

To void an existing password definition record, whether it was created and placed in the NAMEREC file by the data definition processor or by a previous execution of the ZP#NRU utility, you can employ the TID=NONE option in a subsequent run (or use the DELETE function parameter). To change an existing password, you can use the TID=NONE option to delete the previous password and establish the new password with another specification of the PASSWORD function parameter. Notice, in Figure 3-4, that both may be done in the same execution of ZP#NRU.

```
// JOB CHGPAS
// DVC 20 // LFD PRNTR
// DVC 50 // VOL DS9999 // LBL NAMEREC,DS9999 // LFD ISAMNRF
// EXEC ZP#NRU
/$
PASSWORD  PID=WARES  TID=NONE DDN=WARE  FN=PRODFIL
PASSWORD  PID=SESAME    DDN=WARE  FN=PRODFIL  TID=ALL
/*
/&
// FIN
```

Figure 3-4. Sample Control Stream for Changing an Established Password

Figure 3-4 illustrates the method of changing an existing password; it voids the password name, WARES, assumed to have been previously established for the defined file PRODFIL by the example in Figure 3-3 and establishes the new password SESAME for the same file.

3.2.4. Deleting Data Definition and Password Records (DELETE)

The DELETE function parameter logically deletes data definition and password records from the NAMEREC file and allows you to physically delete these records using an OS/3 utility program, without reformatting the NAMEREC file. To delete data definition and password records with the NAMEREC file utility, include the DELETE function parameter in your job control stream in the following format:

```
DELETE  {DDN=name-1 [,....,name-n]}
        {PSW=name-1 [,....,name-n]}
```

where:

`DELETE`
Is the function parameter and must be coded in columns 1 through 6.

`DDN=name-1 [,....,name-n]`
Specifies the data definition records to be deleted. Record names should not exceed seven characters.

`PSW=name-1 [,....,name-n]`
Specifies the password names to be deleted. Record names should not exceed seven characters.

Record names can be continued on succeeding lines. A nonblank character in column 72 indicates continuation, and continuation lines may begin in any column except column 1.

The NAMEREC file utility logically deletes records by placing a hexadecimal FF in the first byte of data (position 8) of the record. To physically delete the records, you should use the deletion option of the utility disk-to-disk (UDD) data utility. You can restore a password record that was incorrectly deleted by using the NAMEREC utility PASSWORD parameter.

Figure 3-5 illustrates the execution of the ZP#NRU utility, which deletes a password and lists the keys of the records in the NAMEREC file.

```
 // JOB DELPS
// DVC 20 // LFD PRNTR
// DVC 50 // VOL DS9999 // LBL NAMEREC,DS9999 // LFD ISAMNRF
// EXEC ZP#NRU
/$
DELETE PSW=WARES
LIST
/*
/&
// FIN
```

Figure 3-5. Sample Control Stream for Password Deletion and Listing of Records in the NAMEREC File

The DELETE statement deletes the password WARES. The LIST statement lists the keys of all records in the NAMEREC file.

3.2.5. NAMEREC Utility Error Processing

When the ZP#NRU utility encounters a password definition input error, a message is printed and processing continues. Valid passwords are written to the NAMEREC file; only the password definitions containing errors must be resubmitted. I/O errors cause program termination; an error message is also printed.

The NAMEREC utility also generates error messages when you use the DELETE parameter to delete data definition and password records.

See Table E-1 in Appendix E for diagnostic messages, their causes, and corrective action.

3.3. Generating Input Edit Tables

The edit table utility program, ZH#EDT, offers a convenient means for converting freeform input received from terminal operators into fixed formats required by your action programs and checking this input for types of data, value ranges, and presence of required fields. The ZH#EDT utility must be executed after the NAMEREC file has been initialized; the edit tables generated are placed in the NAMEREC file. The edit table generator is described in the IMS action programming manuals.

3.4. Data Definition

If you are going to use UNIQUE in your IMS application, you must define the files to be accessed by your terminal operators by using the data definition processor (DT3DF). Your action programs can also access these defined files. The data definition records produced by the data definition processor are written to the NAMEREC file. Data definition is discussed in the *IMS Data Definition and UNIQUE Programming Guide,* UP-9209.

Section 4
Configuring IMS

4.1. Overview of the Configuration Process

One unified configuration process serves for both single-thread and multithread operations in IMS under OS/3. A simple, flexible IMS-supplied job control procedure (jproc), IMSCONF, generates and executes an IMS configurator program for either single-thread or multithread IMS, produces and links the ultimate load module for the online IMS system configured, and stores it in the designated load library. The characteristics of the configured IMS system are determined by the parameters you supply in the configuration input sections.

The IMSCONF jproc consists of five job steps necessary to generate an online IMS system:

- CCA Linkage Step

  Links the user communications control area (CCA) and stores the output in a load library; the generated load module is loaded in the configuration step in order to validate network-related configuration input.

- Initialization Step

  Allocates and initializes (or scratches and reinitializes) any or all IMS internal files (AUDFILE, AUDCONF, CONDATA, NAMEREC and STATFIL). This step calls the procedure IMS#INT, which contains two subordinate jprocs: IMS#NTZ for file initialization; IMS#SCR for scratching files.

- Configuration Step

  Executes the configurator program and generates records in the NAMEREC file. Performs the following functions:

  - Generates an assembler source module containing all configured user and IMS file specifications and other online requirements of IMS

  - Generates a linker source module containing the link stream for the configured IMS online module

  - Generates tables and records supporting online IMS

- Preformats AUDFILE and CONDATA files, as required for a multithread IMS configuration
- Preformats an AUDCONF file for a single-thread IMS configuration

- Assembly Step

  Assembles the configurator-generated IMS source module and DTFs or CDIB/RIBs for your data files.

- Online Module Linkage Step

  Links IMS object modules and the configurator output into an executable IMS load module and places it in a load library.

The IMSCONF jproc does not always generate all five control streams. You select the control streams to be run by means of the CNFJCS and INIT keyword parameters (4.2.6 and 4.2.9). In addition, if a fatal error should occur during the configurator execution step, the job terminates, and the assembly and online module linkage steps are not executed. Figure 4-1 portrays the 5-step configuration process performed by IMSCONF.

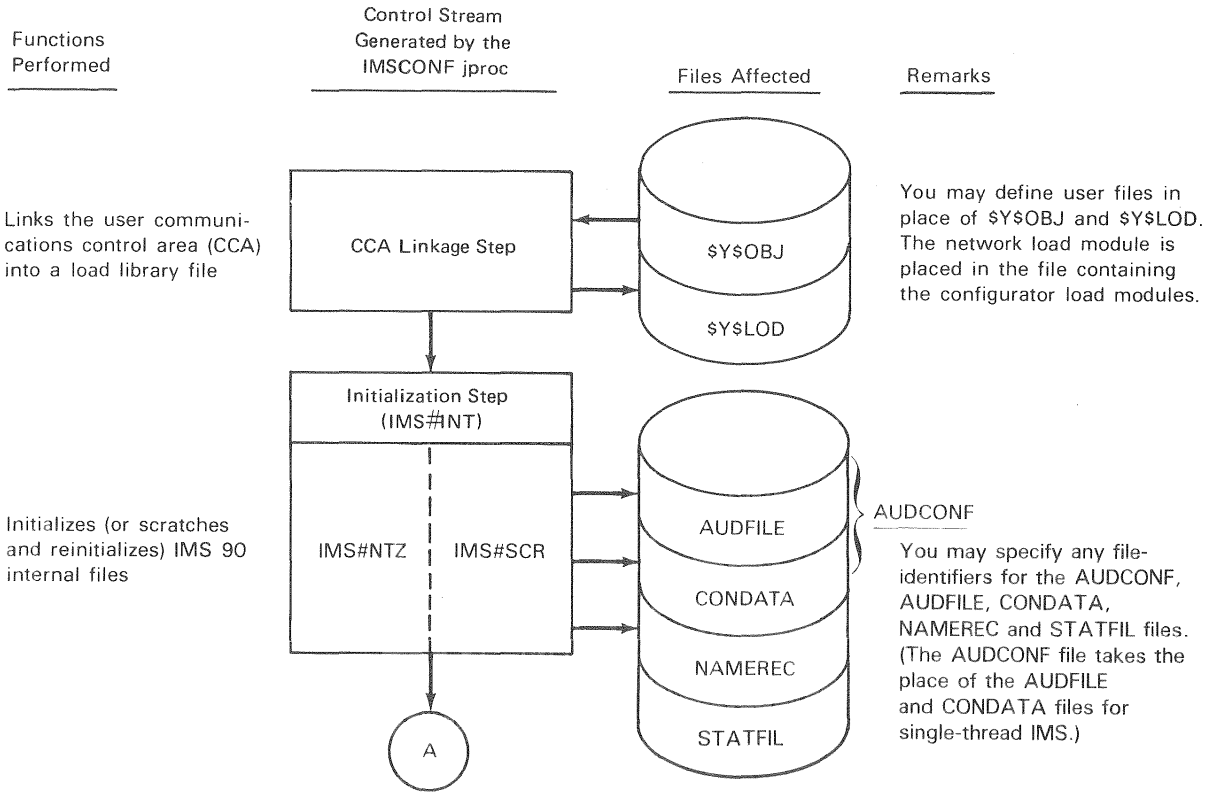


Figure 4-1. Functional Flow of IMSCONF Configuration Process (Part 1 of 2)

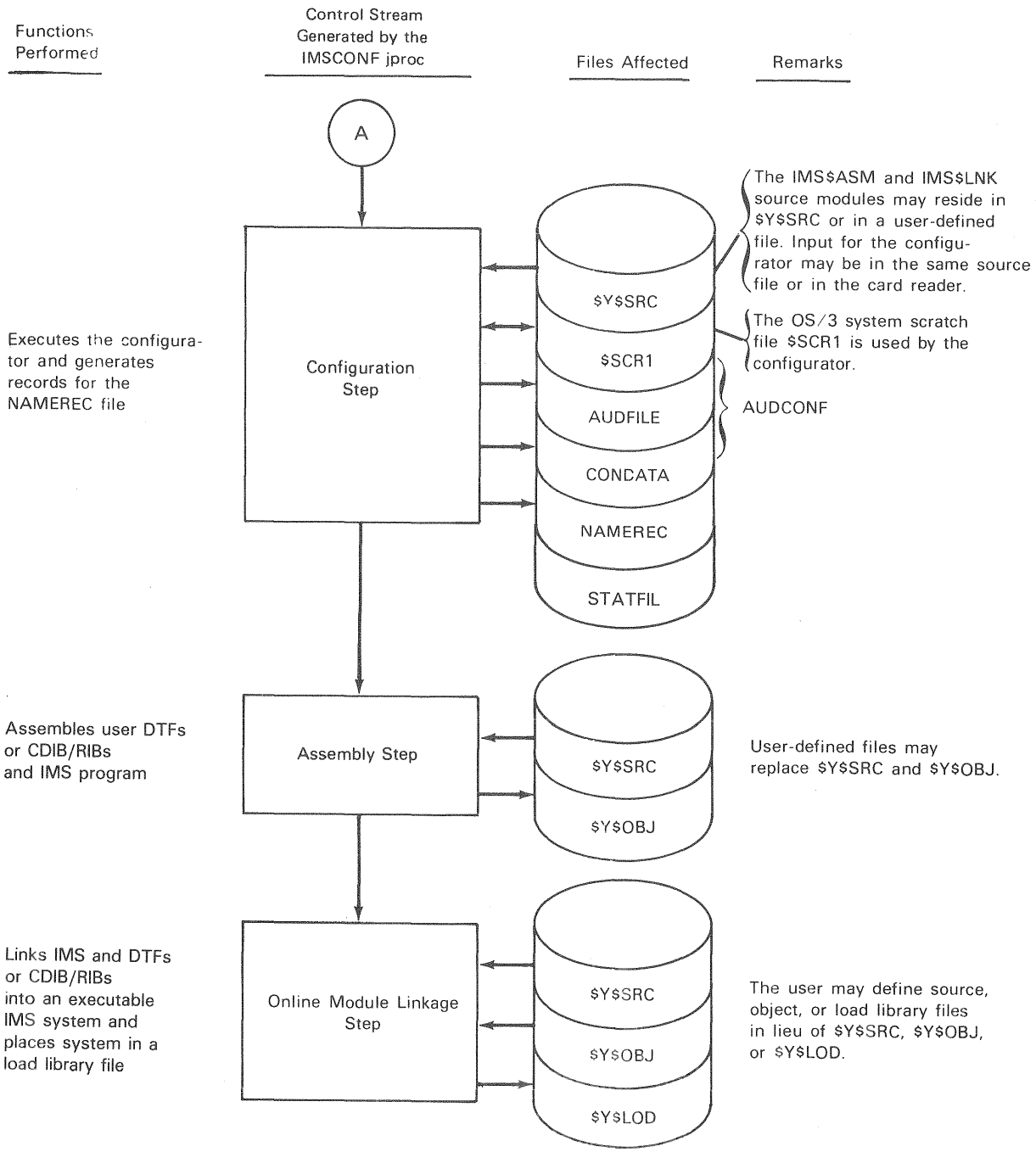


Figure 4-1. Functional Flow of IMSCONF Configuration Process (Part 2 of 2)

4.1.1. Procedure for Configuring IMS

The way you configure an online IMS system depends on whether you use card input or enter your job from a workstation.

To configure your IMS system using card input:

1. Prepare configurator input, as described in 4.3. Follow configurator cards with a // FIN card. If desired, file the configurator input in the library named on the LIBS parameter of the IMSCONF jproc, using the librarian. See the *System Service Programs (SSP) Operating Guide,* UP-8841.

2. Prepare the job control stream:

   a. Set up the IMSCONF jproc call line (4.2).

   b. Compute storage requirements for the JOB job control statement (4.2.14).

   c. Place the // IMSCONF card behind the // JOB card; follow it with /& and // FIN cards.

   d. If desired, file the run stream in the system job control stream library file (YJCS) by keying in the command FI at the console. If configurator input is from cards or if ALTER job control statements are required, you must file the job control stream in YJCS.

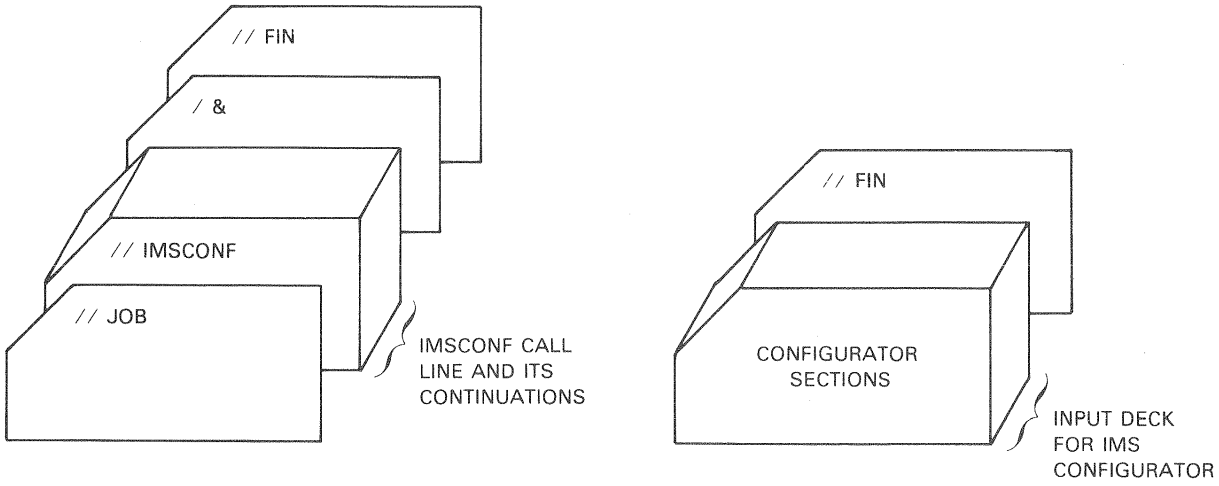
3. Run configurator job:

   a. If either the job control stream or the configurator input is on cards, place the deck in the card reader.

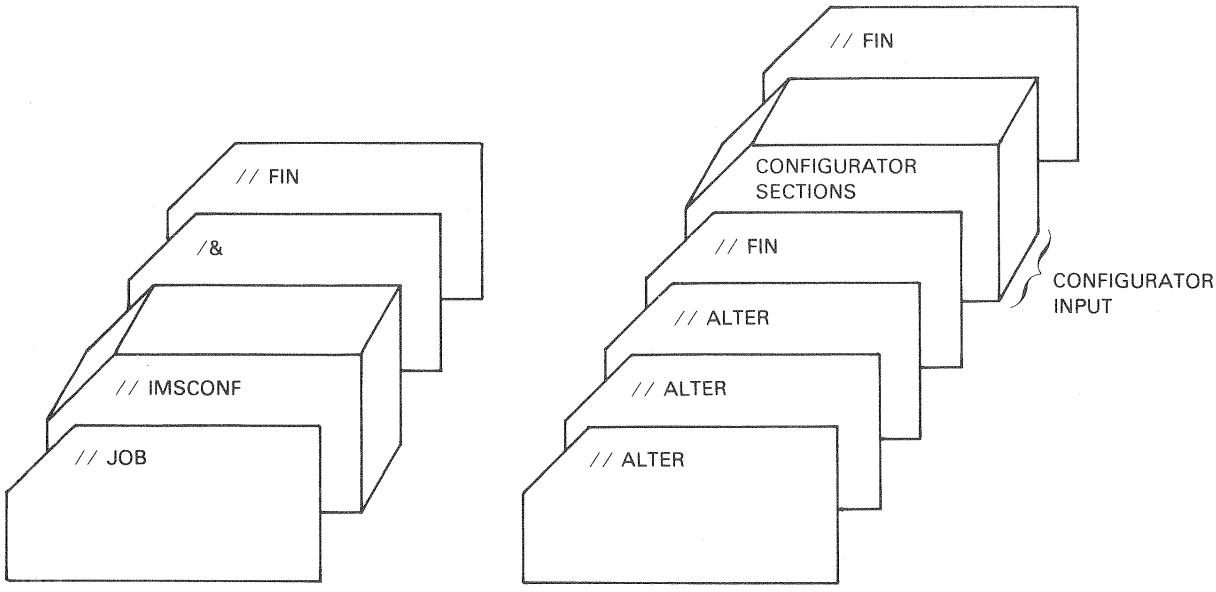
   b. If there are any ALTER job control statements (4.2.11), follow them with a // FIN card and place in a card reader; if configurator input is on cards, the ALTER deck precedes the configurator deck.

   c. Key in the command *RUN jobname* at the console.

Figure 4-2 illustrates the job decks required for the configurator job. At least one of the job decks must be filed before running the configurator job.



a. Configuration Job Decks with no ALTER Statements



b. Configuration Job Decks when ALTER Job Control Statements Are Used

Figure 4-2. Configuring IMS Using Card Input

To run your IMS configuration job from a workstation (Figure 4-3):

1. Prepare configurator input (4.3) using the general editor, and file it in the library named on the LIBS parameter of the IMSCONF jproc. Refer to the *General Editor Operating Guide,* UP-9976.

2. Prepare the job control stream:

   a. Select IMSCONF jproc parameters (4.2) and compute storage requirements for the JOB statement (4.2.14).

   b. Perform the logon procedure, as described in the *Interactive Services Operating Guide,* UP-9972.

   c. Use the general editor to build the job control stream and file it in the YJCS system library.

3. Key in the command *RV jobname*.

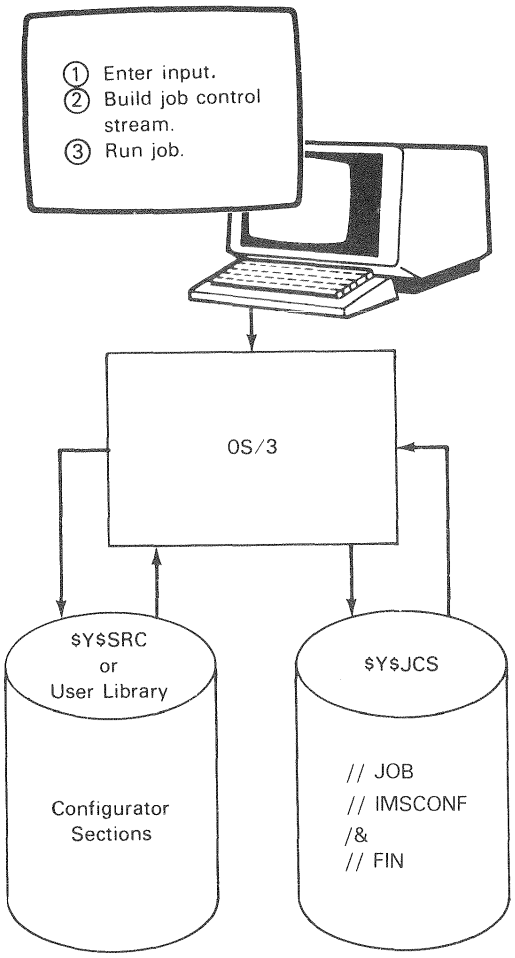


Figure 4-3. Configuring IMS from a Workstation

4.1.2. IMS Internal Files

A number of the internal files used online by IMS are described briefly in this paragraph because they must be considered at several points in the configuration process: both when you are building the call line for the IMSCONF jproc and when you are preparing input to the configurator. The file names listed are reserved for IMS files and cannot be used for user data files.

- NAMEREC

  The named record file is always required. It contains various control tables generated by the configurator and records from pre-online processing. The same NAMEREC file may contain records from as many as 255 different IMS configurations. However, if you have more than one IMS system online at the same time, they can't access the same NAMEREC file. The NAMEREC file is in ISAM format in a DTF system; in MIRAM format in a CDM system.

- AUDFILE

  The audit file is a system access technique (SAT) file created for online recovery in multithread IMS systems. It consists of pre-update before-images of files. These images are used to roll back the file when a CANCEL command (UNIQUE) or a ZZCNC terminal command is executed. The audit file is also used for writing IMS control information at termination time and therefore is required whether or not updating is configured.

- CONDATA

  The continuity data file is also a SAT file whose block size is determined by the configurator. It contains data that must be passed from one action to another within a transaction. The CONDATA file is required in a multithread configuration if you use UNIQUE, if any of your action programs use a continuity data area, or if you specify TOMFILE=YES.

- TRCFILE

  The trace file is created online for use in offline recovery if the FUPDATE and RECOVERY parameters are both specified in the configurator OPTIONS section. Before-images, after-images, or both, are recorded in the trace file, depending on the RECOVERY parameter specifications. The trace file is a SAT file, but unlike the other internal files, it may be on disk, diskette, or magnetic tape.

- AUDCONF

  The audit and continuity data file is the single-thread IMS counterpart to both the AUDFILE and CONDATA files used in multithread IMS systems. Like them, the AUDCONF file is a SAT file. It is always required.

- STATFIL

  The statistical data file is used to contain statistical data recorded during online processing. It is used in both single-thread and multithread IMS.

- TOMFILE

  The terminal output message file is actually a partition of the single-thread AUDCONF file or the multithread CONDATA file. It is created if you include the TOMFILE=YES parameter. The TOMFILE records output messages generated at rollback points and at transaction termination to aid terminal operators during online recovery.

  These output messages may also be recorded in the trace file so as to permit offline reconstruction of the TOMFILE.

- LDPFILE

  The fast loader file is a system access technique (SAT) file that is required when you include the FASTLOAD=YES parameter in the OPTIONS section. Action programs are copied from the action program load library, LOAD, the first time they are called by a transaction and then are loaded from the LDPFILE.

Of these files, only a NAMEREC file and the STATFIL can be used for both single-thread and multithread IMS load modules - none of the others can, and you should assign them accordingly. For example, the same AUDCONF file may be used for several single-thread IMS load modules, but must never be assigned to a multithread IMS system.

If the same AUDCONF or AUDFILE file is to be used for more than one load module, the configuration for each load module must have the same value for the AUDITNUM parameter in the GENERAL section. In addition, the largest BLKSIZE specified for any file in a FILE section must be the same value in each configuration, and the number of terminals designated by the TERMS parameter in the NETWORK section must be the same.

4.2. Calling the IMSCONF Job Control Procedure

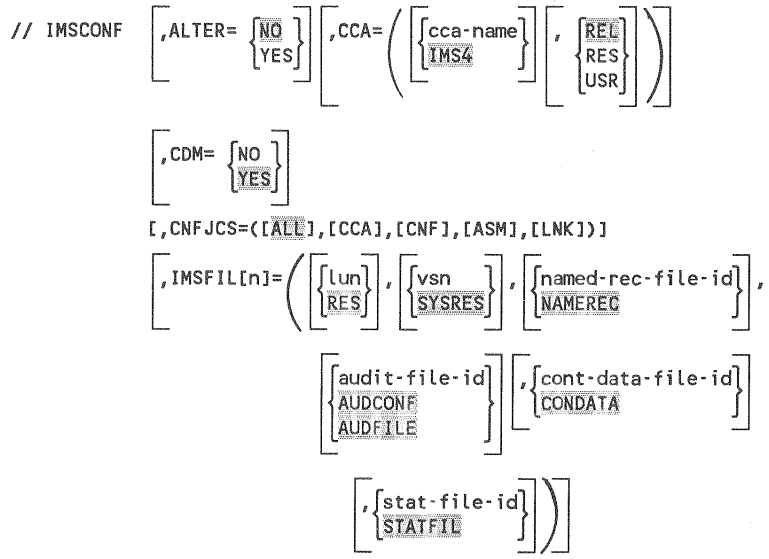
The IMSCONF jproc generates and executes the necessary control streams to perform system generation functions for a unique single-thread or multithread online IMS system. The keyword parameters of the IMSCONF jproc call line determine the manner in which the configuration process is to be performed. You may specify:

- The volume on which the various files reside and their file identifiers
- Whether input is in the card reader or in a source library
- A multithread or single-thread IMS system

- DTF mode or CDM mode
- Which control streams are to be run
- Listing options
- The names of your online load module and your communications control area
- The format and size of the IMS internal files
- Whether temporary changes must be made to any of the IMS configurator load modules

All of the IMSCONF keyword parameters are optional. They all have default specifications, except for the INIT parameter, which must be coded if internal files are to be initialized.

The IMSCONF jproc call line is illustrated in the following format. The keyword parameters are shown here in alphabetic order, but they may be coded in any convenient order, following the coding conventions described in the *Job Control Programming Guide,* UP-9986.



(continued)

(continued)

Multithread only

```
[,INIT=(  [I]  ,[blksize-namerec]  ,[blocks-namerec]  ,
          [R]   [NO              ]   [75             ]
          [S]   [6144            ]
      [cyl-audit-file]  [,cyl-condata]  [,cyl-statfil]))]
      [NO            ]  [NO         ]  [NO          ]
      [15            ]  [5          ]  [10          ]
      [10            ]
```

Single-thread only

```
INIT=(  [I]  ,[blksize-namerec]  ,[blocks-namerec]
        [R]   [NO              ]   [75             ]
        [S]   [3072            ]
          [,cyl-audconf-file]  [,cyl-statfil]  )
          [NO              ]   [NO          ]
          [15              ]   [10          ]
```

```
[,INPUT= [src-module-name]]
         [CARD           ]

[,LIBL=(  [lun]  ,[vsn    ]  [,load-lib-name]))]
          [50 ]   [OS3REL ]   [$Y$LOD       ]
          [RES]   [SYSRES ]

[,LIBO=(  [lun]  ,[vsn    ]  [,obj-lib-name]))]
          [50 ]   [OS3REL ]   [$Y$OBJ      ]
          [RES]   [SYSRES ]

[,LIBS=(  [lun]  ,[vsn    ]  [,src-lib-name]))]
          [50 ]   [OS3REL ]   [$Y$SRC      ]
          [RES]   [SYSRES ]

[,LOADM= online-load-module-name]

[,LST= [(A)                ]]
       [NO                 ]
       [([C] [,D][,O][,S])  ]
       [(C,D,S)            ]

[,SWPRI=priority]

[,ZCNF=(  [MT]  [,REL]  )]
          [ST]  [,RES]
                [,USR]
```

4.2.1. Assigning Configurator Library Files (LIBS, LIBO, LIBL)

The LIBS, LIBO, and LIBL keyword parameters control the assignment of files to contain the source, object, and load libraries needed by the configurator.

The source library file receives output streams from the configuration step of the IMSCONF jproc; these are later input to the assembly and linkage steps. It must also contain the configurator input if such input is not in the card reader.

The object library file receives output from the assembler and the configurator in the assembly step. This file must contain the IMS object modules provided on your OS/3 release volume; these modules are linked with the configurator output in the linkage step. In the linkage step, IMSCONF also includes data management and system modules from the YOBJ library file on SYSRES. Because these modules must be current, your SYSRES volume must be updated to the current OS/3 release level before you configure IMS.

The load library receives output from the IMS linkage step -- the online IMS load module. Output from the CCA linkage step goes to the file indicated on the ZCNF parameter.

The flowchart in Figure 4-4 illustrates a functional sequence for selecting the keyword parameters that need to be coded. The same sequence is followed in the subsequent paragraphs, which describe each keyword parameter and its functions. Coding examples are included.

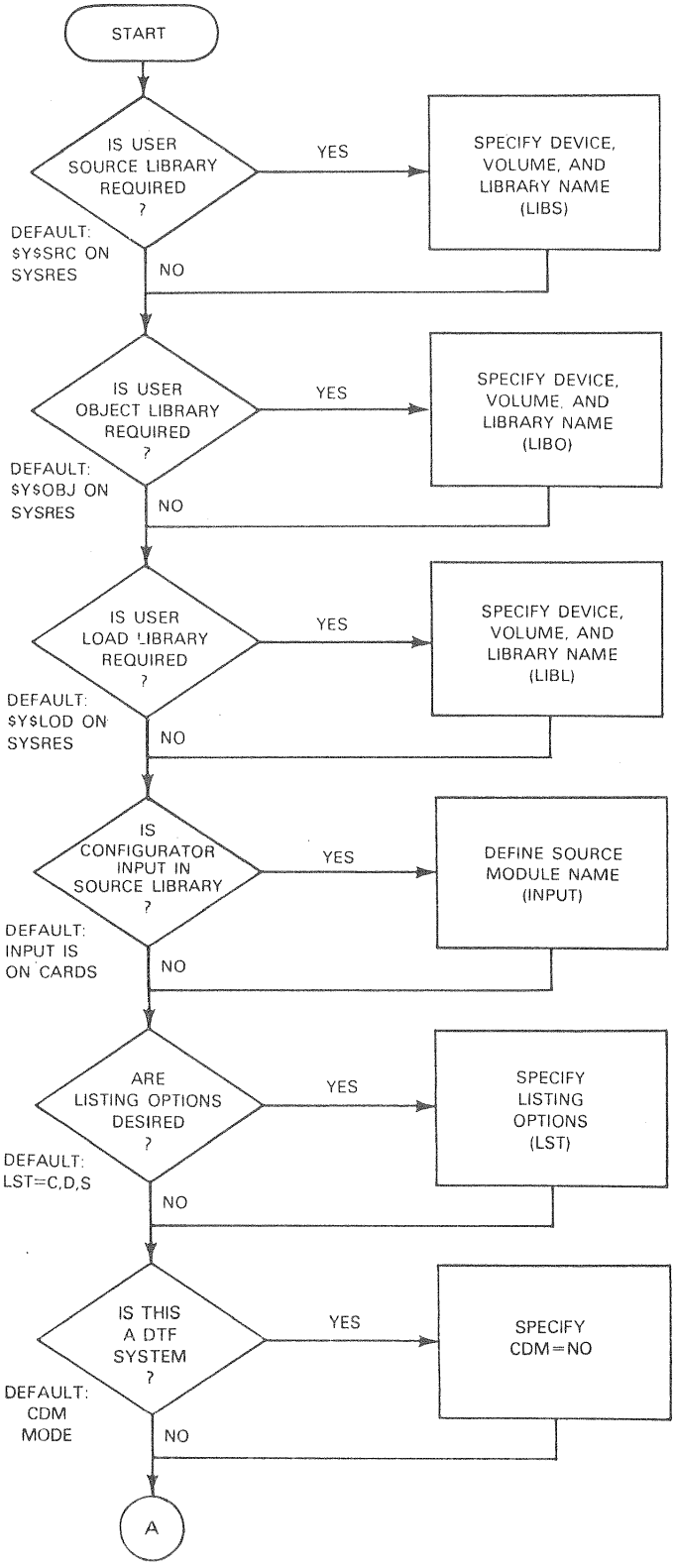
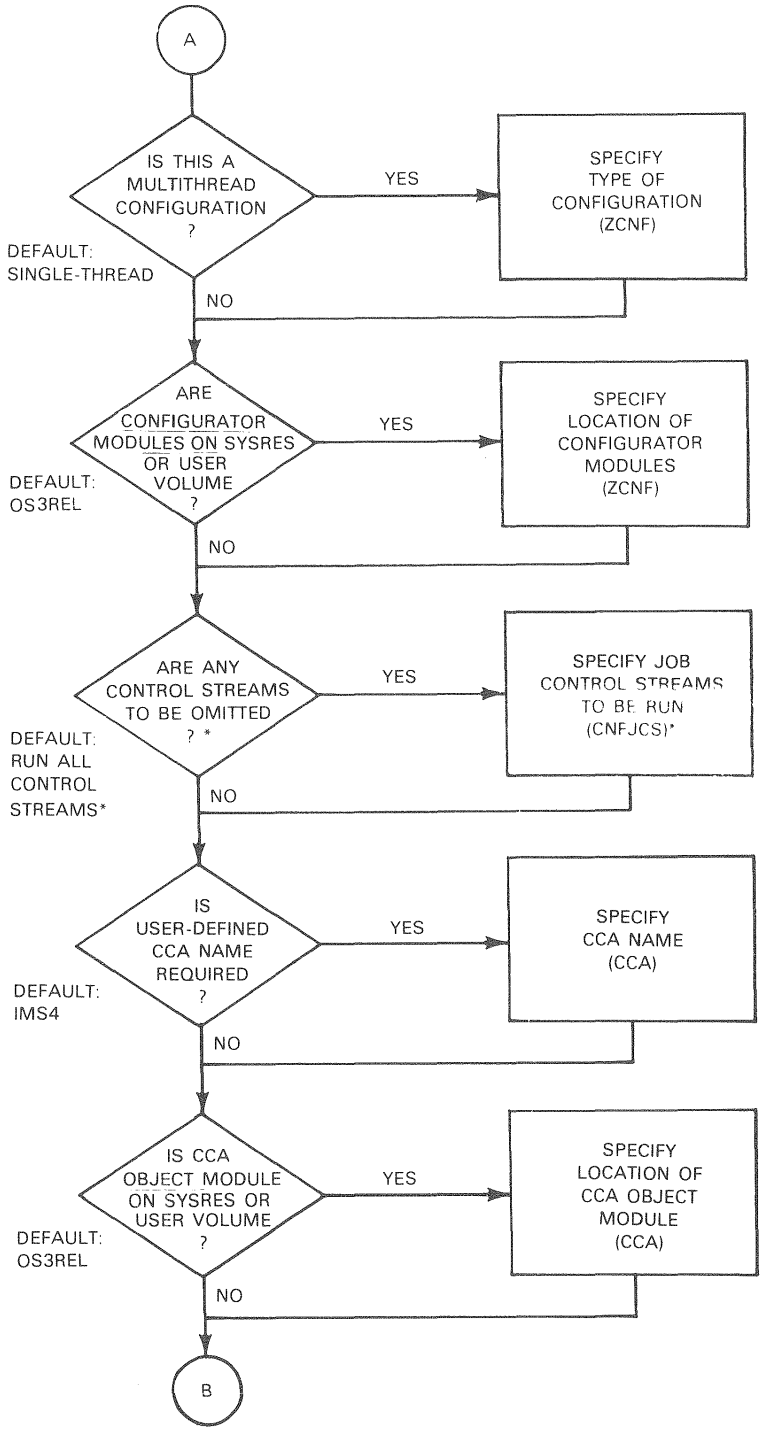


Figure 4-4. Flowchart for Selecting IMSCONF Jproc Parameters (Part 1 of 3)



*Does not apply to IMS#INT control stream, which is governed by the INIT keyword parameter.

Figure 4-4. Flowchart for Selecting IMSCONF Jproc Parameters (Part 2 of 3)

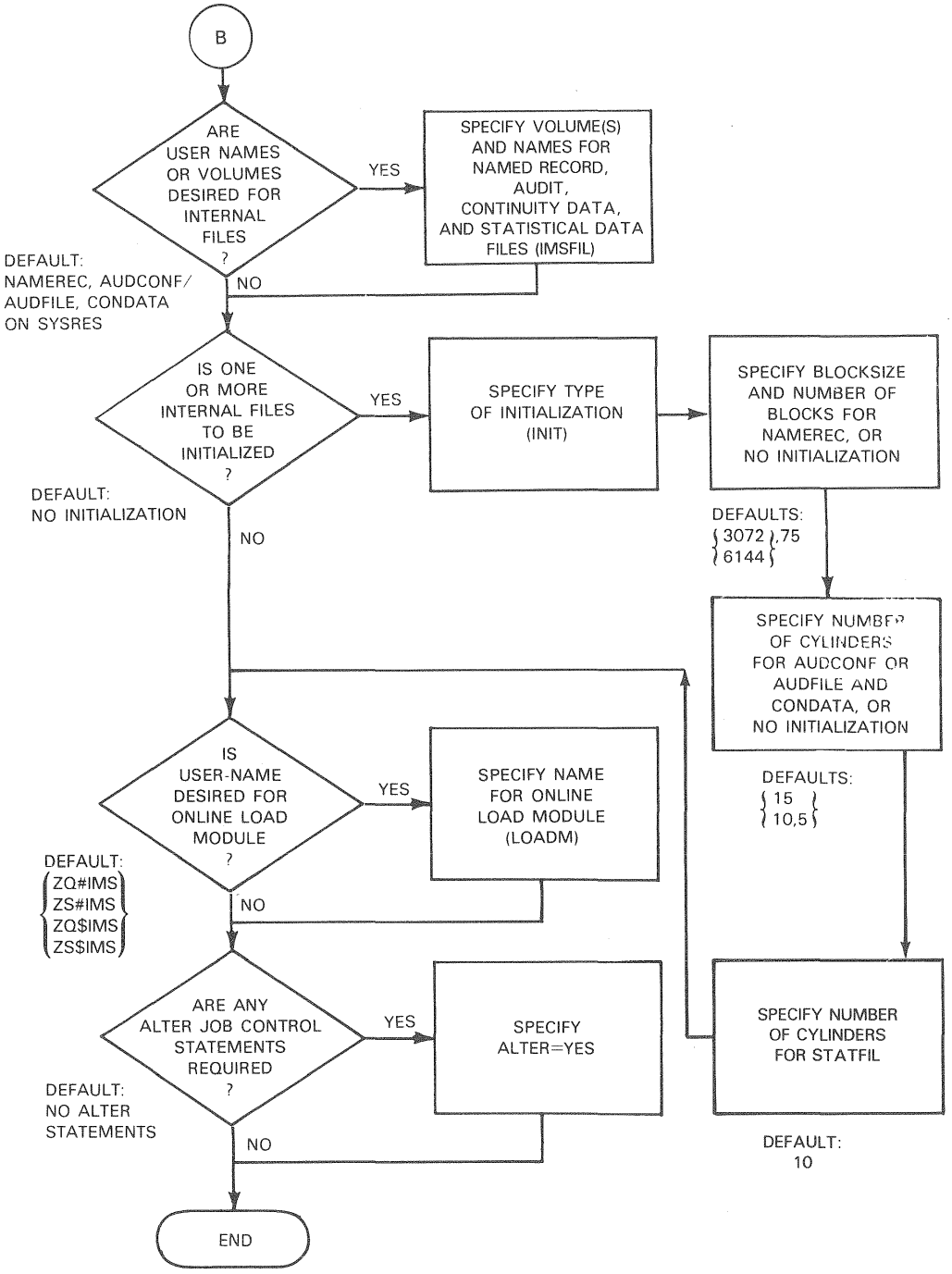


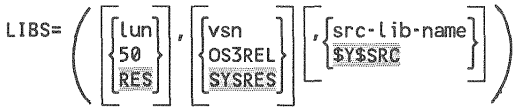
Figure 4-4. Flowchart for Selecting IMSCONF Jproc Parameters (Part 3 of 3)

The LIBS and LIBL parameters default to the YSRC and YLOD library files on SYSRES. The LIBO parameter defaults to the YOBJ library on OS3REL.

The purpose for the OS3REL default is to save file space on the SYSRES volume; once the configuration process is completed, the object modules produced are no longer needed, and it would serve no useful purpose to keep them online. For the same reason, you have the option at SYSGEN of excluding the IMS object modules and the configurator load modules from the SYSRES volume being created. If you do so, you must have the OS3REL volume online at configuration and specify (or default) OS3REL for LIBO and for the ZCNF keyword parameter. As an alternative to both SYSRES and OS3REL, you may copy these modules to a user volume.

Note: *OS/3 defines SYSRES as the volume from which the initial program load (IPL) of the supervisor has been made. If you configure IMS immediately after a SYSGEN in which you are creating a new SYSRES and the IPL was made from OS3REL, that volume is defined as SYSRES and you must identify the new SYSRES volume by a logical unit number and volume serial number. If the IPL for the SYSGEN procedure was from a previous SYSRES, you must re-IPL from the new SYSRES to ensure that current system and data management modules are included in your configuration.*

LIBS Keyword Parameter



Generates a device assignment set for the configurator source library file.

Subparameter 1:

lun
: Is any acceptable logical unit number ranging from 51 to 89, indicating a disk unit containing a volume other than SYSRES or OS3REL.

50
: Specifies the device containing the OS3REL volume.

RES
: Specifies the device containing the SYSRES volume and is the default value.

Subparameter 2:

vsn
: Specifies the volume serial number of the volume containing the source library file.

OS3REL
: Specifies that the source library file is on OS3REL. IMSCONF generates a volume serial number of RELnnn when OS3REL is specified, where nnn represents the release level.

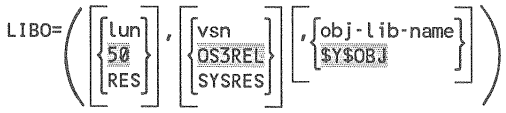
SYSRES
Indicates that the source library file resides on SYSRES and is the default assumption.

Subparameter 3:

src-lib-name
Defines a source library file other than YSRC.

YSRC
Specifies that configurator source modules are to be filed in YSRC on either SYSRES or OS3REL and is the default assumption.

LIBO Keyword Parameter



Generates a device assignment set for the configurator object library file.

Subparameter 1:

lun
Is a logical unit number, ranging from 51 to 89, indicating a disk unit containing a volume other than SYSRES or OS3REL. Use the same logical unit number specified for the source library file if both are on the same volume.

50
Specifies the device containing the OS3REL volume.

RES
Specifies the device containing the SYSRES volume.

Subparameter 2:

vsn
Specifies the volume serial number of the volume containing the object library file.

OS3REL
Indicates that the object library file is on OS3REL and is the default assumption. IMSCONF generates a volume serial number of RELnnn, where nnn represents the release level.

SYSRES
Specifies that the object library file is on SYSRES.

Subparameter 3:

obj-lib-name
Defines an object library file other than YOBJ. If you specify a user library, you must copy the IMS object modules provided on your OS/3 release volume into this library.

YOBJ
Specifies that configurator object modules are in YOBJ on either SYSRES or OS3REL and is the default assumption.

LIBL Keyword Parameter

Generates a device assignment set for the configurator load library file. This file will contain the online IMS load module.

Subparameter 1:

lun
Is a logical unit number, ranging from 51 to 89, indicating a disk unit containing a volume other than SYSRES or OS3REL. Use the same logical unit number specified for the source and/or object library file if they are on the same volume.

50
Specifies the device containing the OS3REL volume.

RES
Specifies the device containing the SYSRES volume.

Subparameter 2:

vsn
Specifies the volume serial number of the volume containing the load library file.

OS3REL
Specifies that the load library file is on OS3REL. IMSCONF generates a volume serial number of RELnnn, where nnn represents the release level.

SYSRES
Indicates that the load library file is on SYSRES and is the default assumption.

Subparameter 3:

load-lib-name
Defines the load library file other than YLOD which is to contain the online IMS load module produced by the configuration process.

YLOD
Indicates that the online IMS load module is to be filed in YLOD on either SYSRES or OS3REL and is the default assumption.

4.2.2. Defining Configurator Input (INPUT)

The input to the configurator program may be in the card reader or may be stored on disk. If the configurator input is on disk, you must specify the name of the source module by means of the INPUT keyword parameter. The module must be in the source library file defined by the LIBS keyword parameter.

INPUT Keyword Parameter

INPUT= {src-module-name / CARD}

where:

src-module-name
Defines the source module that the configurator accesses for its input. The module must be in the source library file defined by the LIBS keyword parameter.

CARD
Indicates that configurator input is in the card reader and is the default assumption.

4.2.3. Selecting Listing Options (LST)

You may use the optional LST keyword parameter to specify your choice of the various listing options available from the configuration process - but you may also use it to bypass configuration altogether and simply list the directory of the NAMEREC file to review its current state. Otherwise, the NAMEREC directory is routinely printed at the end of each configurator run, regardless of what listing options are selected.

The NAMEREC directory lists, for each configured IMS load module having records in the file, the identification of the configuration, the name of the ICAM network definition, the date of configuration, the time of the configurator run, the version number of the IMS configurator used, and an indication of the number of errors present in the file. Entries are printed in ascending order of configuration identifiers. Figure 4-5 provides an example of the NAMEREC file directory.

NAMEREC DIRECTORY

CNF-ID ①	CNF-NETW ②	CNF-DATE ③	CNF-TIME ④	CNF-VERS ⑤	CNF-ERRORS ⑥
0001	IMS5	81/12/18	04:34:08	VER8.0.08-MT	0049
0009	IMS5	81/11/07	12:08:02	VER8.0.07-MT	0039
0022	IMS1	80/02/07	01:37:23	VER7.0.07-ST	0001
0105	IMS5	81/12/18	03:35:03	VER8.0.08-ST	0011
0111	IMS6	81/09/03	07:27:26	VER8.0.06-MT	0004

Notes:

① Configurator identifier -- Decimal integers in the range 0001 - 0255 that uniquely identify an IMS load module. Specified to the configurator in the NETWORK section via the CONFID keyword parameter.

② Configured network -- A 4-character alphanumeric identification of the ICAM network definition, specified to the configurator in the NETWORK section via the NAME keyword parameter.

③ Configuration date -- Date of the configuration run.

④ Configuration time -- Time of the configurator run. Uses the 24-hour time-keeping system.

⑤ Configurator version -- The version number of the IMS configurator used.

⑥ Configuration errors -- The number of errors logged by the configurator during the run represented by this line; not necessarily the number of errors in the NAMEREC file records generated for the IMS load module identified in the first column.

Figure 4-5. Example of NAMEREC File Directory Listed by the IMS Configurator

To cause the directory to be listed alone, without going through configuration, you specify LST=(A) to the IMSCONF jproc, and you do not provide any input to the configurator. See 4.2.15 for a coding example.

Another listing option of the IMSCONF jproc, LST=(D), causes the listing of all generated data and prints the contents of each table generated by the configurator for the NAMEREC file. This output may be useful in debugging the configuration process, but it does not list the entire contents of the NAMEREC file. The records inserted in the NAMEREC file by the edit table generator or the data definition processor, and the password definition records filed by the ZP#NRU utility, for example, are not listed by this LST option, nor are they represented in the NAMEREC file directory. You can obtain a listing of all the records in the NAMEREC file only by means of the NAMEREC file utility (ZP#NRU), as described in 3.2.2.

The four keyletters C, D, O, and S are nonpositional, optional subparameters. They may be specified in any order, separated by commas. All listing options except NO must be enclosed within parentheses.

LST Keyword Parameter

```
LST= [(A)                    ]
     {([C][,D][,O][,S])      }
     [(C,D,S)                ]
     [NO                     ]
```

where:

A

Lists the directory of the NAMEREC file and bypasses the configuration process. No other IMSCONF keyword parameter except ALTER=YES may be specified, nor is any input provided for the configurator.

When the NAMEREC file contains no configurator-generated records, the following message is listed in place of the directory:

NO CONFIGURATIONS PRESENT IN THIS NAMEREC FILE

C

Lists all configurator options, plus defaults.

D

Lists generated data and prints the contents of each table generated by the configurator for the NAMEREC file (useful as a configurator debugging aid).

O

Lists all configurator output generated to the source library.

S

Lists all input parameter lines read by the configurator.

NO

Suppresses the configuration listing. The configurator nevertheless prints the following:

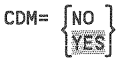
- The NAMEREC directory
- The NETWORK section, including all its keyword parameters input to the configurator
- All configurator errors (Table E-2)
- The error cross-reference listing

If the LST keyword parameter is omitted, IMSCONF assumes LST=(C,D,S).

4.2.4. Choosing DTF or CDM Mode (CDM)

With the CDM parameter, you tell IMSCONF whether to generate an IMS system in DTF mode or CDM mode. In DTF mode, IMS supports DAM, ISAM, MIRAM, and SAM disk files and SAM tape files. In CDM mode, IMS supports MIRAM disk files and CDM sequential tape files. Refer to 1.4 for more information about DTF and CDM modes.

CDM Keyword Parameter



where:

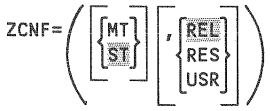
NO
: Designates DTF mode.

YES
: Designates CDM mode.

4.2.5. Configuring a Single-Thread or Multithread System (ZCNF)

The ZCNF keyword parameter determines whether this is a single-thread or multithread IMS system and identifies the library containing the configurator load modules. Single-thread and multithread IMS systems are discussed in 1.1.2.

ZCNF Keyword Parameter



Subparameter 1:

MT
: Configures a multithread IMS system.

ST
: Configures a single-thread IMS system.

Subparameter 2:

REL
: Indicates that the configurator load modules reside in YLOD on OS3REL. The configurator generates a logical unit number of 50 for this volume. Do not use a logical unit number of 50 in the LIBS, LIBO, or LIBL specifications unless those libraries reside on OS3REL.

RES
: Indicates that the configurator load modules reside in YLOD on your system resident (SYSRES) volume.

USR
: Indicates that the configurator load modules reside in a user library file. If USR is specified, the configurator load modules must be in the file specified by the LIBL keyword parameter.

4.2.6. Selecting the Control Streams to Be Generated (CNFJCS)

There may be occasions when you do not require all of the five control streams to be generated by the configuration process; for example, if you have already established a network load module and internal files for another configuration. The CNFJCS and INIT keyword parameters determine which control streams IMSCONF will generate. The CNFJCS parameter, described here, governs the generation of four of the IMSCONF job steps through the specification of positional subparameters. If CNFJCS is omitted, the four steps are all generated. The INIT parameter (4.2.9) controls the initialization run stream, which initializes (or scratches and reinitializes) the IMS internal files.

CNFJCS Keyword Parameter

```
CNFJCS=([ALL],[CCA],[CNF],[ASM],[LNK])
```

where:

ALL
: Generates the entire run stream within the IMSCONF jproc (except for file initialization).

CCA
: Generates the run stream to link the user communications control area (CCA).

CNF
: Generates the run stream to execute the configurator.

ASM
: Generates the run stream to assemble the output of the configurator.

LNK
: Generates the run stream to link the online IMS.

Consider the following examples of coding the CNJFCS keyword parameter:

```
     1         10    16                                              72
①  | CNFJCS=(ALL)
②  | CNFJCS=(,CCA,CNF)
③  | CNFJCS=(,,CNF)
④  | CNFJCS=(,,CNF,ASM,LNK)
⑤  | CNFJCS=(,,,ASM,LNK)
```

① Generates the entire IMSCONF jproc run stream except for file initialization (specified by the INIT parameter). Omitting the CNFJCS keyword parameter has the same effect. If a fatal error occurs during the configurator execution step, the job is terminated and the assembly and online module linkage steps are not performed.

② Generates the run streams to link the CCA and execute the configurator.

③ Generates only the run stream for executing the configurator.

④ Generates the run streams for executing the configurator and for assembling and linking its generated output. If a fatal error is encountered during the configurator execution step, the assembly and linkage steps are not performed.

⑤ Generates the run stream for assembling and linking the output from the configurator. This specification should not be made if the INIT keyword parameter is used.

A recommended approach to initial configuration of IMS, using the CNFJCS and INIT keyword parameters, is as follows:

1. Run the CCA linkage, file initialization, and configurator execution steps (Example 2 and the INIT parameter).

2. Check the configurator output for errors, and rerun the configurator step (Example 3) until the configurator output is acceptable.

3. Run the configurator, assembly, and online module linkage steps (Example 4), or omit the configurator step (Example 5) if the previous run was error free.

Any reconfiguration you perform with the intent to produce a new IMS load module must at least include the configuration, assembly, and linkage steps of the overall configuration process. If you intend to use the same NAMEREC file established for one or more previously configured load modules, it is recommended that you run only the configurator step, using a temporary NAMEREC file, until the output is acceptable; then execute the configurator, assembly, and linkage steps using the established NAMEREC file. This will minimize the chance of compromising an already established NAMEREC file.

4.2.7. Identifying the Communications Control Area (CCA)

The CCA keyword parameter identifies the communications control area (CCA) object module to be linked in the CCA linkage step of the configuration process. The CCA object module is created in the system generation procedure that produces the ICAM symbiont for use by your online IMS system. You must specify SAVE=YES on the CCA macro to save the object module. The CCA module resides in the YOBJ library file of the OS/3 release volume (OS3REL), or the SYSRES volume, unless you copy it to another volume. The load module created by the CCA linkage step is stored in the file containing the configurator load modules, as indicated by the ZCNF keyword parameter.

CCA Keyword Parameter

Subparameter 1:

cca-name
: Specifies the name of the CCA object module to be linked by the CCA linkage step. This name must match the label of the CCA macro in the ICAM generation and the specification of the NAME keyword parameter in the NETWORK section of configurator input. The default value is IMS4.

Subparameter 2:

REL
: Indicates that the CCA object module generated by the SYSGEN procedure resides in YOBJ on OS3REL. The configurator generates a logical unit number of 50 for this volume. (Do not use a logical unit number of 50 in the LIBS, LIBO, or LIBL specification unless those libraries reside on OS3REL.)

RES
: Indicates that the CCA object module resides in YOBJ on your system resident (SYSRES) volume.

USR
Indicates that the CCA object module is in a user library file. If USR is specified, the CCA object module must be in the file specified by the LIBO keyword parameter.

Note: *If the initial program load (IPL) has been performed from OS3REL, that volume must be defined as RES.*

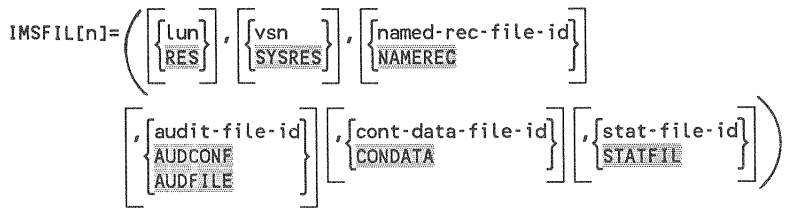
4.2.8. Assigning IMS Internal Files (IMSFIL[n])

The IMSFIL[n] keyword parameter controls the assignment of the IMS internal files required by this configuration. A named record file (NAMEREC) is always required. Additionally, a multithread configuration requires an audit file (AUDFILE) and generally requires a continuity data file (CONDATA); a single-thread configuration requires an audit and continuity data file (AUDCONF), which is the counterpart to both the AUDFILE and the CONDATA files. A STATFIL is always required for a single-thread configuration, but it is optional for multithread configuration. Specify NO for subparameter 6 of the INIT keyword parameter to exclude the STATFIL in a multithread configuration.

If all of the internal files are to reside on the same disk volume, you may use the IMSFIL format, without a subnumber (n). If these files are to reside on separate volumes, you code the keyword parameter IMSFIL1 for the named record file, IMSFIL2 for the audit file or audit and continuity data file, IMSFIL3 for the continuity data file, and IMSFIL4 for the STATFIL. In this case, subparameters 4, 5, and 6 are omitted, and the file identifier in each case is coded as subparameter 3. Examples are given following the parameter descriptions.

You may choose your own file identifiers for the IMS internal files. However, regardless of the file identifiers used, you must specify the names NAMEREC, AUDFILE, CONDATA, AUDCONF, and STATFIL on the LFD job control statements at IMS start-up time.

IMSFIL[n] Keyword Parameter



where:

n
: Indicates which of the IMS internal files is to be assigned:

  1 = named record file

  2 = audit file (multithread) or audit and continuity data file (single-thread)

  3 = continuity data file (multithread)

  4 = statistical data file

  If *n* is omitted, all the IMS internal files are to reside on the same volume.

Subparameter 1:

lun
: Is a logical unit number, ranging from 51 to 89, indicating a disk unit containing a volume other than SYSRES. Use the same logical unit number specified for LIBL if the internal files are to reside on the same volume with the online IMS load module. For diskette volumes, logical unit numbers range from 130 to 133 and 136 to 159. Refer to the *Job Control Programming Guide,* UP-9986, for appropriate logical unit numbers.

RES
: Specifies the device containing the SYSRES volume and is the default value.

Subparameter 2:

vsn
: Specifies the volume serial number of the disk volume containing the file or files referenced by this keyword parameter.

SYSRES
: Specifies that the file or files referenced by this parameter are to reside on SYSRES. This is the default assumption.

Subparameter 3:

named-rec-file-id
: Specifies a file identifier for the named record file. The default is NAMEREC.

Subparameter 4:

audit-file-id
: Specifies a file identifier for the audit file in a multithread configuration, or the audit and continuity data file in a single-thread configuration. The default for multithread is AUDFILE; for single-thread, AUDCONF. If the keyword parameter IMSFIL2 is used, the audit file identifier is specified for subparameter 3.

Subparameter 5:

cont-data-file-id
: Specifies a file identifier for the continuity data file in a multithread configuration. The default is CONDATA. If the keyword parameter IMSFIL3 is used, the continuity data file identifier is specified for subparameter 3.

Subparameter 6:

stat-file-id
: Specifies a file identifier for the statistical data file. The default is STATFIL. If the keyword parameter IMSFIL4 is used, the statistical data file identifier is specified for subparameter 3.

Following are several coding examples for the IMSFIL[n] keyword parameter:

```
      1         10    16                                                  72
①  | IMSFIL=(,,IMSREC,AUDIT1)
②  | IMSFIL=(51,DISK01)
③  | IMSFIL1=(51,DISK01,IMSREC),IMSFIL2=(52,DISK02,AUDIT1),                X
   | IMSFIL3=(52,DISK02,CONREC1)
④  | IMSFIL4=(51,DISK01,MYSTAT)
```

① Specifies the file identifiers IMSREC for the named record file and AUDIT1 for the audit file, if this is a multithread configuration, or the audit and continuity data file, if single-thread. If this is a multithread configuration, the name CONDATA will be assigned to the continuity data file and STATFIL to the statistical data file by default. In this example, all the internal files are to reside on SYSRES. Note that commas must be coded in place of the missing subparameters.

② Specifies that all IMS internal files are to reside on disk volume DISK01; the disk unit is assigned the logical unit number 51. The files are to be labeled NAMEREC, AUDFILE, CONDATA, and STATFIL if this is a multithread configuration, or NAMEREC, AUDCONF, and STATFIL, if single-thread.

③ Specifies that the named record file, IMSREC, is assigned to volume DISK01, on disk unit 51, and the audit and continuity data files, AUDIT1 and CONREC1, are assigned to volume DISK02, on disk unit 52. Note that the file identifier is coded as subparameter 3 in each case.

④ Specifies that a statistical data file, MYSTAT, is to reside on disk volume DISK01; the disk unit is assigned to logical unit number 51.

Note: *Embedded blanks are permitted in the file identifiers for the named record, audit, continuity data, and statistical data files and for the configurator library files. Do not use quotation marks in the IMSFIL[n], LIBS, LIBO, or LIBL parameter specifications. When you use embedded blanks, IMS generates an LBL job control statement in the form / / LBL 'file-identifier'. For example, the specification*

```
IMSFIL2=(55,DISK03,AUDIT FILE)
```

generates the job control statement / / LBL 'AUDIT FILE', and the specification

```
LIBS=(60,DISK01,IMS SOURCE)
```

generates the job control statement / / LBL 'IMS SOURCE'.

4.2.9. Initializing IMS Internal Files (INIT)

The INIT keyword parameter generates the control stream for the initialization step, which allocates and initializes the IMS internal files. This parameter can be used to selectively initialize, or scratch and reinitialize, the NAMEREC, AUDFILE, CONDATA, AUDCONF, and STATFIL files.

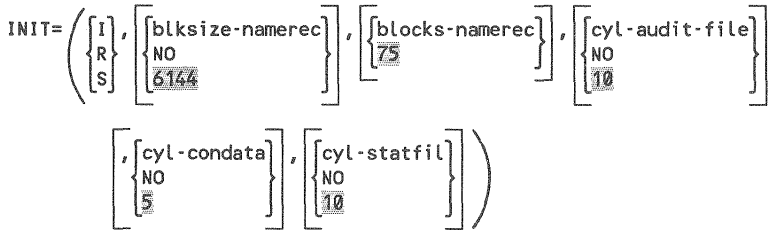
The INIT keyword parameter has two forms; one for multithread and one for single-thread. The multithread version has six subparameters and is used for the NAMEREC, AUDFILE, CONDATA, and STATFIL files. The single-thread version has five subparameters and is used for the NAMEREC, AUDCONF, and STATFIL files.

If you omit the INIT parameter, no file initialization is performed; IMS assumes that all files have previously been allocated and initialized. If only the NAMEREC file has previously been initialized, include INIT but specify NO for the second subparameter. If you reinitialize the NAMEREC file at this time (regardless of whether you specify I, R, or S), all its contents will be destroyed, including any generated data definitions, password definitions, edit tables, and configuration records from other IMS load modules.

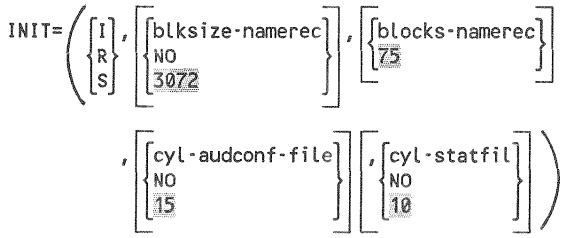
The INIT parameter is also used to specify the blocksize and number of blocks for the NAMEREC file and the number of cylinders for the audit, continuity, and statistical data files.

Default values are assumed if you specify INIT but omit any of its subparameters; you must specify NO for each file you don't want initialized. A comma must be coded in place of any omitted subparameters, except for trailing subparameters.

INIT Keyword Parameter (Multithread)



INIT Keyword Parameter (Single-thread)



Subparameter 1:

I
: Initializes and allocates file space for the files indicated in subparameters 2 through 6.

R
: Reformats a previously allocated NAMEREC file. Other internal files are assumed to have been previously allocated and initialized. When this option is specified, the total size of the NAMEREC file cannot be changed. If blocksize is altered, the number of blocks must be changed in inverse proportion. Both blocksize and number of blocks must be specified, or both omitted. If both are omitted, the previously established specifications remain in effect.

S
: Deallocates (scratches), reallocates, and reinitializes the indicated files.

Subparameter 2:

blksize-namerec
: Specifies the number of bytes in a block of the NAMEREC file. This value may range from 1024 to 12,800, but it must be a multiple of 256 and must not exceed the track size of the disk or diskette subsystem on which the file resides. If omitted, IMSCONF applies a default blocksize of 3072 for a single-thread configuration, 6144 for multithread; if R is specified for subparameter 1, the previously established blocksize remains in effect.

NO
: Specifies that a NAMEREC file is not to be allocated or initialized. The configurator assumes that a previously initialized NAMEREC file is available and can be assigned for the configuration step.

Subparameter 3:

blocks-namerec
: Specifies the number of blocks that IMSCONF is to allocate when initializing the NAMEREC file. The default value is 75 for both single-thread and multithread IMS; if R is specified for subparameter 1, the previously established number of blocks remains in effect.

  The default value of 75 blocks is suitable for an average size configuration, i.e., approximately 20 terminals, 30 files, 30 actions, 30 programs, 15 transactions. Larger configurations require more blocks, usually 100 or 150. Also, if the NAMEREC file holds more than one configuration (see the CONFID parameter, 4.3.1), the number of blocks specified should be a multiple of the number of configurations. For more information on specifying the number of blocks in the NAMEREC file when multiple configurations are used, see C.2.2.

Subparameter 4:

`cyl-audit-file (multithread) or cyl-audconf-file (single-thread)`
Specifies the number of cylinders to be allocated for the AUDCONF file in a single-thread configuration or the AUDFILE file in multithread. The default value for an AUDCONF file is 15 cylinders; the default for an AUDFILE file is 10 cylinders. You must generate an audit file whether or not this IMS system includes file updating. If file updating is not configured, you can allocate as little as 1 cylinder for this file.

`NO`
Specifies that an AUDCONF or AUDFILE file is not to be allocated or initialized. The configuration assumes that this file is available and assignable for the configuration step.

Subparameter 5:

`cyl-condata (multithread only)`
Specifies the number of cylinders to be allocated for CONDATA file in a multithread configuration. The default value supplied by IMSCONF is 5 cylinders.

`cyl-statfil (single-thread only)`
Specifies the number of cylinders to be allocated for the STATFIL. The default assumed by IMSCONF is 10 cylinders.

`NO`
Specifies that a multithread CONDATA file or a single-thread STATFIL file is not to be allocated or initialized. The multithread configurator assumes that the CONDATA file is available and assignable for the configuration step.

Subparameter 6 (multithread only):

`cyl-statfil`
Specifies the number of cylinders to be allocated for the STATFIL. The default assumed by IMSCONF is 10 cylinders.

`NO`
Specifies that a STATFIL file is not to be allocated or initialized.

Note: *You may initialize the NAMEREC file by executing the NAMEREC file utility (ZP#NRU). However, the AUDCONF, AUDFILE, CONDATA, and STATFIL files can be preformatted for online IMS use only by running the IMSCONF job control stream.*

Initializing (or reformatting) the NAMEREC file results in a new file, and after a successful configuration, only the new configuration records exist in the file. All previous nonconfiguration records (data definitions, password definitions, edit tables), as well as configuration records from any other IMS load modules, must be regenerated into this new NAMEREC file.

The following coding examples illustrate the uses of the INIT keyword parameter. Note that commas are coded in place of missing subparameters.

```
      1         10    16                                          72
①  | INIT=(I)
②  | INIT=(I,4096,100,20)
③  | INIT=(S,NO,,,NO)
④  | INIT=(R,1024,150)
⑤  | INIT=(I,,10,20,NO)
⑥  | INIT=(I,NO)
```

① Initializes all the IMS internal files. If this is a single-thread configuration, default values of 75 and blocksize of 3072 will be applied for the NAMEREC file, 15 cylinders for the AUDCONF file and 10 cylinders for the STATFIL file. If it is multithread, NAMEREC blocksize will be 6144, 10 cylinders will be allocated for the AUDFILE and STATFIL files, and 5 cylinders for the CONDATA file.

② Indicates that all files are to be initialized and specifies value for NAMEREC and AUDCONF or AUDFILE files. The STATFIL file will be assigned 10 cylinders and if this is a multithread configuration, the CONDATA file will be assigned 5 cylinders.

③ Scratches and reinitializes the AUDFILE and STATFIL files in a multithread configuration using default values of 10 cylinders. In a single-thread configuration only the AUDCONF file is scratched and reinitialized using the default value of 15 cylinders.

④ Reformats the existing NAMEREC file with 150 blocks and a blocksize of 1024.

⑤ Initializes the NAMEREC, AUDFILE and STATFIL files in a multithread configuration. The default value of 6144 is applied to the NAMEREC blocksize and 10 cylinders to the STATFIL file. In a single-thread configuration, only the NAMEREC and AUDCONF files are initialized. The default value of 3072 is applied to the NAMEREC blocksize.

⑥ Initializes all internal files except NAMEREC.

4.2.10. Naming the Online IMS Load Module (LOADM)

The LOADM keyword parameter assigns a name to the online IMS load module being configured. The load module is generated into the load library file indicated by the LIBL keyword parameter; if LIBL is omitted, it is stored in YLOD on SYSRES.

LOADM Keyword Parameter

```
LOADM=online-module-name
```

Assigns a name to the online IMS load module. The default name is:

- ZQ#IMS for a multithread DTF configuration
- ZS#IMS for a single-thread DTF configuration
- ZQ$IMS for a multithread CDM configuration
- ZS$IMS for a single-thread CDM configuration

4.2.11. Reading ALTER Job Control Statements (ALTER)

The ALTER job control statement, used to make minor changes in a load module at execution time, is described in detail in the *Job Control Programming Guide,* UP-9986. If a problem arises with one of the IMS configurator load modules, and you receive a patch to be applied to it via one or more ALTER statements from your system analyst, you must use the ALTER keyword parameter to prevent the ALTER job control statements from being processed as input for the configurator. If your system does not include a card reader, this parameter does not apply.

The cards containing the Unisys-supplied ALTER statements must be in the card reader and (if card input for the configurator is specified) in front of the configurator input cards. You need a FIN job control statement to separate the ALTER cards from the configurator input cards, which must also be followed by a FIN statement so that the CR job control statements embedded within the jproc may terminate the reading of cards for each set of data required in the generated run stream. (Subsection 4.1.1 and Figure 4-3.)

ALTER Keyword Parameter

Specifies whether ALTER job control statements are present in the card reader for making temporary changes to configurator load modules.

```
ALTER=  {NO }
        {YES}
```

where:

NO
: Specifies that no ALTER job control statements are inserted in the card reader and is the default assumption.

YES
: Specifies that ALTER job control statements are inserted in the card reader (as just described) and causes them to be read as such by the IMSCONF jproc.

  If ALTER job control statements are present in the card reader but the ALTER keyword parameter is omitted (or ALTER=NO is specified), they are not applied but rejected as invalid configurator input.

4.2.12. Assigning a Task Switching Priority (SWPRI)

The SWPRI keyword parameter allows you to assign a task switching priority to the IMS configuration job steps.

SWPRI Keyword Parameter

```
SWPRI=priority
```

where:

priority
: Is the task switching priority assigned to the IMS configuration job steps. The priorities range from 1 (highest priority) to 60 (lowest priority). If you omit the SWPRI keyword specification, the priority specified in the supervisor generation is used.

For additional information on task switching priority, see the *Job Control Programming Guide,* UP-9986.

4.2.13. Identifying Precataloged, Password-Protected Files

When using previously cataloged files with read/write password protection, you must identify any of the IMSCONF files according to the LBL format described in the *Job Control Programming Guide,* UP-9986. In this case, cataloged and password-protected files should be specified as user files by specifying USR as the second positional parameter on the CCA and ZCNF parameters of the IMSCONF jproc. You must also follow this procedure for identifying system files on SYSRES. (See the IMSCONF jproc format in 4.2.) For example:

```
LIBO=(RES,,$Y$OBJ(rpw/wpw))
LIBL=(RES,SYSRES,$Y$LOD(rpw/wpw)
IMSFIL=(RES,,NAMEREC(rpw/wpw),AUDCONF)
```

where:

`(rpw/wpw)`
: Is read password/write password.

4.2.14. Main Storage Requirements

Before running the configurator job (or filing the control stream that contains your IMSCONF jproc call in YJCS), you need to estimate the minimum main storage required for the configuration process so that it may be specified on the JOB job control statement. You are concerned here with the *min* parameter of the JOB statement. The IMS configurator is not structured to take advantage of additional main storage; to assign more via the *max* parameter would serve no useful purpose.

Your requirements depend upon the size of the configurator load module you are using, the size of the CCA object module to be linked, and the block size you have determined for the NAMEREC file. Use the following formula in estimating the minimum main storage required and round the result upward to the next higher multiple of 256 bytes before specifying your final result in hexadecimal via the *min* parameter on the JOB control statement for the IMSCONF job:

```
R = (configurator-size) + (CCA-size) + (2*(NAMEREC-blocksize)) + 3072
```

where:

`R`
: Is the intermediate result, in decimal, to be rounded upward to the next 256-byte boundary.

`configurator-size`
: Is the length, in bytes, of the configurator load module to be used. The size of the single-thread configurator module, ZS#CNF, is $92D0_{16}$ bytes, or approximately 37.6K decimal bytes; the size of the multithread configurator module, ZQ#CNF, is $A384_{16}$ bytes, or approximately 41.9K decimal bytes.

CCA-size
: Is the length, in bytes, of the CCA object module to be linked in the first step of the configuration process. The size of the CCA module is listed in the output of the SYSGEN run that produced the ICAM symbiont for use by your online IMS system. The name of the CCA module in the SYSGEN output listing is CCA$*xxxx*, where *xxxx* is the network name specified as the label on the CCA macro. For more information on CCA-size, see 4.2.7.

NAMEREC-blocksize
: Is the block size of the NAMEREC file. This should be the same figure as specified to the IMSCONF jproc via the INIT keyword parameter (4.2.9), or the BLKSZE keyword parameter of the NAMEREC file utility program (3.2.1).

The following example applies this formula to calculating main storage requirements for configuring a single-thread IMS load module.

Given:

configurator-size = 37,584 bytes

CCA-size = 8192 bytes (from SYSGEN output listing)

NAMEREC-blocksize = 3072 bytes (This happens to be the default assumption for single-thread.)

Intermediate results:

R = 37,584 + 8192 + (2*3072) + 3072 = 54,992

Round up:

54,992 + 48 = 55,040

Specification for *min* parameter on JOB card (in hexadecimal):

D700

The next example uses the same formula to illustrate calculation of main storage required for configuring a multithread IMS load module.

Given:

configurator-size = 37,584 bytes

CCA-size = 16,384 bytes (from SYSGEN output listing)

NAMEREC-blocksize = 8000 bytes (Assume that the application required a data definition record larger than the multithread default value, 6144 bytes.)

Intermediate results:

R = 37,584 + 16,384 + (2*8000) + 3072 = 73,040

Round up:

73,040 + 176 = 73,216

Specification for *min* parameter on JOB card (in hexadecimal):

11E00

4.2.15. Sample Control Streams for IMS Configuration

Control streams containing the IMSCONF jproc call are illustrated in the following coding examples:

Example 1

```
1        10    16                                                    72
-------------------------------------------------------------------------
// JOB IMSST,,D700
// IMSCONF INIT=(I)
/&
// FIN
```

Example 1 illustrates the minimum job control stream for running all five steps of the configurator process for a single-thread CDM configuration. The only keyword parameter required is INIT, which initializes the IMS internal files. Default values are applied for all other parameters and for the optional subparameters of the INIT keyword parameter.

Example 2

```
1        10    16                                                    72
-------------------------------------------------------------------------
// JOB IMS,,E000
// IMSCONF ZCNF=MT,CNFJCS=(,CCA,CNF),IMSFIL=(51,DISK08),INPUT=IMSSRC,  X
// LIBL=(51,DISK08,IMSLOD),LOADM=IMS01,CCA=(IMS1)
/&
// FIN
```

The control stream in Example 2 indicates the following:

- Only the CCA linkage and configuration job steps are to be run. (Once an acceptable listing is obtained, the assembly and linkage steps can be added.) Internal files are assumed to have been previously allocated and initialized, perhaps from another configuration.
- This is a multithread CDM configuration and the configurator load modules reside in YLOD on OS3REL.

- The NAMEREC and AUDCONF files reside on volume DISK08, on disk unit 51. The online IMS load module is to be placed on the same volume in a file named IMSLOD.
- Configurator input is in YSRC on SYSRES (LIBS default); the name of the input module is IMSSRC.
- The online load module is to be named IMS01. (This control stream will not produce a load module; however, the link stream is generated in the configuration step.)
- The name of the CCA object module is IMS1; it is stored in YOBJ on OS3REL.

Example 3

```
// JOB IMSMT,,11E00
// IMSCONF ZCNF=MT,IMSFIL2=(51,DSK100,AUDIT1),                    X
// IMSFIL3=(51,DSK100,DATA1),INIT=(S,,100,NO,NO),CDM=NO
/&
// FIN
```

Example 3 generates all five job streams for a multithread DTF configuration.

- The audit file, named AUDIT1, and the continuity data file, named DATA1, are on volume DSK100 on disk unit 51. The NAMEREC file is on SYSRES.
- The NAMEREC file is to be scratched and reinitialized. Block size is defaulted to 6144, and 100 blocks are to be allocated. The AUDIT1 and DATA1 files are assumed to have been previously allocated.
- Input is from cards.

Example 4

```
// JOB LIST
// IMSCONF LST=(A),CNFJCS=(,,CNF)
/&
// FIN
```

Example 4 does not run any of the configurator control streams but provides a listing of the NAMEREC file directory.

- No configurator input is present.
- The *min* parameter of the JOB statement is not required.
- The NAMEREC file is assumed to be on the system resident volume (SYSRES).
- If you specify the CDM=NO parameter, you must assign an ISAM NAMEREC file; otherwise, a MIRAM NAMEREC file is assumed. If the type of the NAMEREC file is not consistent with the CDM specification, a fatal error results.

- If the NAMEREC file is not on SYSRES, you must use the IMSFIL or IMSFIL1 parameter.
- The CNFJCS parameter avoids a CCA linkage.

4.3. Input to the IMS Configurator

In the third step of the configuration process performed by the IMSCONF jproc, the IMS configurator program itself is executed. The configurator generates a number of tables and writes them to the NAMEREC file for the online use of the IMS system being configured.

The configurator program is controlled by the runstreams generated by the IMSCONF jproc and by input from a card reader or from a source library on disk. After analyzing its input, the configurator collects the modules required by your operation and generates and lists error diagnostics from the configuration process. When an error-free configuration run has been made, a source module is produced that is assembled and linked in the last two steps of the configuration process.

Input to the configurator is organized in 12 logical sections. Some sections may be repeated. A prescribed sequence must be followed in submitting the various sections of configurator input.

The 12 sections of configurator input are as follows:

NETWORK Section
: Defines the ICAM network for IMS; must be the first section.

GENERAL Section
: Defines overall configuration parameters.

OPTIONS Section
: Defines optional IMS modules to be included in configuration.

TIMEOUTS Section
: Defines various timeout values.

FILE Section
: Describes each user data file to be accessed by IMS; coded once for each file.

TERMINAL Section
: Further defines terminals already included in this IMS network.

TRANSACT Section
: Supplies transaction code data to the configurator; coded once for each transaction.

ACTION Section
: Describes the actions to be used in this configuration; coded once for each action.

PROGRAM Section
: Describes the user action programs to be included in this IMS configuration; coded once for each action program.

LANGUAGE Section
: Creates a UNIQUE lexicon record in the NAMEREC file.

LOCAP Section
: Identifies a remote system to which transactions can be routed (multithread only).

DRCRDMGT Section
: Defines the user interface with defined record management.

Input for the configurator falls into two categories: repeatable and nonrepeatable sections. The nonrepeatable group consists of the NETWORK, GENERAL, OPTIONS, TIMEOUTS, and DRCRDMGT sections, and the repeatable sections are FILE, TERMINAL, TRANSACT, ACTION, PROGRAM, LANGUAGE, and LOCAP.

The NETWORK section must be the first section specified. The other nonrepeatable sections except DRCRDMGT follow the NETWORK section.

The repeatable group of sections follows the nonrepeatable group. All repeated sections must be specified consecutively; i.e., all FILE sections together, all TERMINAL sections together, etc.

Although it is a nonrepeatable section, the defined record management (DRCRDMGT) section must be the last section specified in the configuration.

Configurator sections and parameters applicable to single-thread and multithread IMS are summarized in Table 4-1. Following the table and an example of input coded for a multithread configuration, each section is discussed in detail. Appendix A presents the statement and coding conventions used for describing and presenting input to the IMS configurator.

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
NETWORK		X	X	Mandatory, nonrepeatable section. Must be the first section input
	BATCH	X	X	Specifies configuration of batch transaction processor
	CONFID	X	X	Identifies current IMS system in configurator-generated records for the NAMEREC file
	CUP	X	X	Gives LOCAP-name for this IMS program in a global network definition
	KATAKANA	X	X	Specifies Katakana support
	NAME	X	X	Specifies CCA name for ICAM network
	PASSWORD	X	X	Specifies password for ICAM network
	PRCSNUM		X	Specifies number of process file DTFs
	STATUSMG		X	Specifies globally the suppression of IMS timer-generated status messages for all terminals
	TERMS	X	X	Specifies maximum number of online terminals
	UNSOL		X	Specifies globally if all terminals are to receive unsolicited output at the end of an action or transaction

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
GENERAL		X	X	Nonrepeatable section
	AUDITNUM	X	X	Specifies number of audit records required in AUDFILE (multithread) or AUDCONF (single-thread)
	CHRS/LIN	X	X	Specifies message line length
	DDPBUF		X	Specifies size of buffer required for distributed data processing
	DDPSESS		X	Specifies number of DDP sessions that can be active at one time
	INBUFSIZ	X	X	Specifies the size of the input message staging buffer to be allocated at IMS start-up time
	LNS/MSG	X	X	Specifies number of lines per message
	MAXCONT	X	X	Specifies largest continuity data area
	TRANLEN		X	Specifies maximum number of characters in transaction code and lexicon names
	UCHAR		X	Specifies urgent priority on input messages for multithread IMS system
OPTIONS		X	X	Nonrepeatable section
	CONTOUT	X	X	Includes continuous output capability in this configuration

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
OPTIONS (cont.)	DLLOAD	X	X	Provides support for downline loading user programs to a UTS terminal
	DMS	X	X	Allows IMS access to DMS data bases
	FASTLOAD	X	X	Allows use of fast load feature
	FUPDATE	X	X	Specifies inclusion of file updating modules in IMS configuration
	INTLIST	X	X	Allows interruption of output from UNIQUE LIST command
	MSGCLR	X	X	Provides capability of clearing screen of unprotected and protected data
	MSGPOS	X	X	Provides capability of displaying messages at top or bottom of screen
	OPCOM	X	X	Provides support for console transaction processing
	RECLOCK	X		Allows record locking across actions
	RECOVERY	X	X	Specifies recovery options for this IMS configuration
	RESEND	X	X	Specifies configuration of support for ZZRSD terminal command
	RESFMT	X	X	Specifies maximum number of screen formats to remain resident between screen format services calls

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
OPTIONS (cont.)	RESMEM		X	Specifies the number of transaction buffers, the maximum number of transaction buffers to be acquired from IMS storage pool, and the maximum number of transaction buffers a transaction can acquire
	RFMTONLY		X	Specifies RESFMT keyword specification. Value is to be used to determine total number of resident formats.
	SFS	X	X	Provides support for screen format services
	SNAPED		X	Specifies whether snapshot dumps are to be printed with or without an edited directory
	STATS	X	X	Specifies recording of statistical information at shutdown time
	SUBPROG	X	X	Provides support for user-written subprograms
	TOMFILE	X	X	Specifies that terminal output messages are written to the TOMFILE for online recovery
	TOMTRCE	X	X	Specifies that terminal output messages written to the TOMFILE are also written to the trace file for use in offline recovery

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
OPTIONS (cont.)	UNIQUE	X	X	Specifies whether UNIQUE language modules are to be configured and whether the ZU#CIN interpreter module is resident
	UNSOL	X	X	Specifies whether capability for unsolicited output is configured (SEND function, SWTCH command)
TIMEOUTS		X	X	Nonrepeatable section
	ACTION	X	X	Specifies maximum time action program may have control
	INTRFQCY		X	Specifies timer island code interrupt frequency
	STATUS		X	Specifies waiting period for automatic status message at terminal in multithread IMS system
FILE		X	X	Required, repeatable section; one for each user data file accessed; immediately follows last of nonrepeatable sections except DRCRDMGT
	file-name	X	X	Positional parameter
	FILETYPE	X	X	Always required; specifies type of user data file described
	CAFILE	X	X	Specifies that the file is a common storage area file and makes the file resident in main storage

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
FILE (cont.)	CUPDATE	X	X	Specifies updating of disk images when the common storage area file is updated
	DELETP	X	X	Specifies physical deletion of MIRAM file records
	LOCK	X	X	Specifies record lock for transaction or update
	OUTPUT	X	X	Allows output processing for a dedicated sequential MIRAM file
	PKEY	X	X	Specifies the primary key of a multikeyed MIRAM file
	SEQVIEWS		X	Specifies number of sequential file views
	TRACE	X	X	Specifies whether file is to be traced for offline recovery
	(DTF or RIB keyword parameters)	X	X	Data management DTF or RIB keywords describing this data file are input immediately following the other FILE parameters
TERMINAL		X	X	Optional, repeatable section
	terminal-id	X	X	Positional parameter specifying ID of terminal described by subsequent keywords
	IMSREADY	X	X	Specifies whether this terminal is to receive the IMS READY message at start-up time

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
TERMINAL (cont.)	MASTER	X	X	Specifies this as a master terminal in the IMS network
	STATUSMG		X	Specifies whether status messages generated by IMS timer for the terminal are suppressed
	UNATTEND		X	Specifies whether the terminal is unattended
	UNSOL	X	X	Specifies whether this terminal is to receive notice of unsolicited output at the end of an action or a transaction
	URGENT		X	Specifies whether all messages input from this terminal have urgent priority
TRANSACT		X	X	Optional; may be omitted only if user has no transactions to describe, in which case the UNIQUE keyword of OPTIONS section must be specified as YES, RES, or TRAN; otherwise, must be coded once for each transaction
	trans-code	X	X	Positional parameter; required to supply transaction code to configurator
	ACTION	X	X	Specifies name of action program to be scheduled for this transaction

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
TRANSACT (cont.)	LOCAP		X	Specifies a remote system where this DDP transaction is sent for processing, using directory routing
	UNDEF	X	X	Specifies whether this transaction is to receive control when the terminal operator enters a transaction code that has not been defined to IMS
	URGENT		X	Specifies whether transactions with this code are urgent
ACTION		X	X	Optional; may be omitted only if user has no actions to describe, in which case the UNIQUE keyword of the OPTIONS section must be specified YES, RES, or TRAN; otherwise, must be coded once for each action
	program-name	X	X	Positional parameter naming the first action program to be executed for this action
	ALLRNT		X	Specifies whether all action programs for this action are reentrant
	BYPASS		X	Specifies number of times this action may be bypassed before it must be initiated

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
ACTION (cont.)	CDASIZE	X	X	Specifies size of continuity data area for this action
	DDRECORD	X	X	Specifies name of data definition record associated with defined file for this action
	DFILE	X	X	Specifies name of defined file accessed by this action; required if DDRECORD specified
	EDIT	X	X	Specifies editing requirements on input messages for this action
	EXPIRTME		X	Specifies action program time-out value
	FCCEDIT	X	X	Specifies FCC editing requirements for input messages from UTS or workstation terminals
	FILES	X	X	Names conventional files to be accessed or created by this action; not specified printer files
	INSIZE	X	X	Specifies input message length for this action
	MAXSIZE	X	X	Specifies size of largest nonresident action program for this action
	MAXUSERS		X	Specifies the maximum number of users that may process this action concurrently

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
ACTION (cont.)	OUTSIZE	X	X	Specifies length of output message for this action
	SFSINCAP		X	Specifies at normal (N) action terminations whether screen format services input capabilities are to remain or be turned off
	SHRDSIZE		X	Specifies size of volatile data area shared by COBOL action programs involved in this action; not used for BAL or RPG II action programs
	TRANSLAT	X	X	Specifies lowercase-to-uppercase translation on input messages for this action
	WORKSIZE	X	X	Specifies size of work area for this action
PROGRAM		X	X	Optional; may be omitted if user has no action programs to describe, in which case the UNIQUE keyword of the OPTIONS section must be specified YES, RES, or TRAN; otherwise, must be coded for each action program
	program-name	X	X	Positional parameter; required to name the action program described

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
PROGRAM (cont.)	ERET	X	X	Specifies whether control is returned to this action program in case of error
	RESIDE	X	X	Specifies whether this is action program is resident
	SUBPROG	X	X	Specifies whether this is a subprogram
	TYPE	X	X	Specifies whether this action program is reentrant, shared code, or serially reusable
LANGUAGE		X	X	Optional, repeatable section which allows user to define UNIQUE lexicons
	lexicon-name	X	X	Specifies lexicon and transaction name
	language-element	X	X	Specifies words, characters, and delimiters from the standard lexicon, and their replacements
LOCAP			X	Required when distributed data processing is used; defines the remote system where transactions are processed; repeatable
	locap-name		X	Specifies locap name of the remote system

continued

Table 4-1. Summary of Sections and Parameters Input to IMS Configurator (cont.)

Section	Parameters	Single-Thread IMS	Multithread IMS	Remarks
LOCAP (cont.)	RCHAR		X	Specifies a special character that represents the remote system where transactions are processed, using operator routing
DRCRDMGT		X	X	Optional, nonrepeatable; follows last of repeatable sections
	RESIDE	X		Specifies whether defined record management is resident or transient
	UPDATE	X	X	Specifies whether defined record management file updating functions are to be included in this IMS configuration

Legend

blank = Not applicable
X = Applicable

In most cases, the configurator generates default values for parameters you omit. Default values are shaded in the section formats or are noted in the parameter descriptions. Some default values are dependent on your specifications for other parameters. For instance, if you specify CONTOUT=YES in the OPTIONS section, you automatically get UNSOL=YES because the unsolicited output module is needed for continuous output. However, the printout of the input source does not show the actual default values that the configurator generates in these special cases. The listing shows the normal default value for each parameter you omit.

Figure 4-6 illustrates configurator input prepared for a multithread IMS system. In a single-thread system, the UCHAR, STATUS, and URGENT parameters do not apply.

```
NETWORK   NAME=IS80 PASSWORD=IMSPASS CONFID=1 TERMS=20
              CUP=IMS3 UNSOL=TRANS STATUSMG=YES PRCSNUM=10
GENERAL       CHRS/LIN=80  LNS/MSG=24 AUDITNUM=20 MAXCONT=1800
              TRANLEN=8
OPTIONS       UNIQUE=Y FUPDATE=Y SUBPROG=Y CONTOUT=Y UNSOL=Y SNAPED=Y
              RECOVERY=ALL TOMFILE=Y TOMTRCE=Y SFS=10 RESFMT=10
              OPCOM=MASTER RESMEM=(800,200,8) FASTLOAD=YES STATS=YES
TIMEOUTS      ACTION=180  STATUS=30
FILE      CUSTFIL FILETYPE=DMRAM LOCK=TR
              BLKSIZE=512 INDAREA=CUSTFIL INDSIZE=256
              KEYARG=CUSTFIL RECFORM=FIX PROC=KEY KEY1=(5,1)
              MODE=RAN WORK=YES RECSIZE=80 SEEKADR=CUSTFIL
              IOAREA1=CUSTFIL RETR=MOD
FILE      DUEOUT FILETYPE=ISAM LOCK=TR
              IOROUT=ADDRTR KEYLEN=17 RECFORM=FIXBLK RECSIZE=22
              IOREG=(8) PCYLOFL=10 KEYLOC=1 WORK1=DUEOUT IOAREA1=DUEOUT
              TYPEFLE=RANSEQ BLKSIZE=1514
FILE      PRODFIL FILETYPE=DMRAM LOCK=TR
              BLKSIZE=256 RECSIZE=256 RCB=NO WORK=YES
              INDAREA=PRODFIL INDSIZE=256 IOREG=(8)
              KEYARG=PRODFIL KEY1=(17,0) PROC=KEY MODE=RAN
FILE      CATALOG FILETYPE=ISAM LOCK=TR IOROUT=ADDRTR KEYLEN=17
              RECFORM=FIXBLK RECSIZE=30 IOREG=(8) PCYLOFL=10 KEYLOC=1
              TYPEFLE=RANSEQ BLKSIZE=1017 IOAREA1=CATALOG WORK1=CATALOG
FILE      DUEIN FILETYPE=ISAM LOCK=TR IOROUT=ADDTRT KEYLOC=1 KEYLEN=17
              RECFORM=FIXBLK RECSIZE=22 IOREG=(8) TYPEFLE=RANSEQ
              PCYLOFL=10 BLKSIZE=758 IOAREA1=DUEIN WORK1=DUEIN
FILE      CONTROL FILETYPE=DMRAM LOCK=TR
              BLKSIZE=512 RCB=NO WORK=YES IOREG=(8) RECSIZE=256
              MODE=RAN PROC=UNK
FILE      SAMDISK FILETYPE=SAMD BLKSIZE=1024 IOAREA1=SAMDISK
              TYPEFLE=OUTPUT WORKA=YES
FILE      VENDOR FILETYPE=DAMR LOCK=TR BLKSIZE=256
              READID=YES WRITEID=YES
TERMINAL  TRM1 MASTER=Y UNSOL=ACTION STATUSMG=NO
TERMINAL  TRM2 UNATTEND=YES
TERMINAL  TRM3 MASTER=NO UNATTEND=Y
TERMINAL  TRM4
TERMINAL  TRM5
TERMINAL  TRM6 MASTER=Y UNSOL=ACTION STATUSMG=N
TERMINAL  TRM7
TERMINAL  TRM8 UNATTEND=YES
TERMINAL  TRM9 UNATTEND=Y
TRANSACT  CUST ACTION=AP1
TRANSACT  PROD ACTION=AP2
TRANSACT  AP3
TRANSACT  AP4
TRANSACT  AP5
TRANSACT  SAMD ACTION=AP6
TRANSACT  ROLL ACTION=AP7
TRANSACT  F#01 ACTION=CNTRL
ACTION    AP1 FILES=CUSTFIL OUTSIZE=233 TRANSLAT=YES WORKSIZE=215
ACTION    AP2 FILES=PRODFIL OUTSIZE=80 CDASIZE=256 TRANSLAT=YES
              WORKSIZE=512
ACTION    AP3 OUTSIZE=80 WORKSIZE=256 TRANSLAT=YES
ACTION    AP4 OUTSIZE=80 WORKSIZE=256 TRANSLAT=NO
ACTION    AP5 FILES=CATALOG OUTSIZE=80 WORKSIZE=256 TRANSLAT=YES
ACTION    AP6 FILES=CUSTFIL,SAMDISK
              OUTSIZE=80 CDASIZE=256 WORKSIZE=256 TRANSLAT=YES
```

Figure 4-6. Sample Configurator Input (Part 1 of 2)

```
ACTION    AP7 FILES=PRODFIL,VENDOR
              OUTSIZE=80 CDASIZE=256 WORKSIZE=256 TRANSLAT=YES
ACTION    CNTRL FILES=CONTROL WORKSIZE=4000 INSIZE=1200
              OUTSIZE=8000  CDASIZE=256   EDIT=:
PROGRAM   AP1     ERET=YES TYPE=RNT
PROGRAM   AP2     ERET=YES TYPE=RNT
PROGRAM   AP3     ERET=YES
PROGRAM   AP4     ERET=YES
PROGRAM   AP5     ERET=YES TYPE=SER
PROGRAM   AP6     ERET=YES TYPE=SER
PROGRAM   AP7     ERET=YES TYPE=RNT
PROGRAM   CNTRL   ERET=YES TYPE=SER
PROGRAM   SUB     ERET=YES SUBPROG=YES
DRCRDMGT      UPDATE=YES
```

Figure 4-6. Sample Configurator Input (Part 2 of 2)

4.3.1. Identifying the ICAM Network Definition - the NETWORK Section

The NETWORK section identifies the ICAM network used by IMS, provides the configuration identifier used in the NAMEREC file, and supplies information needed to configure this IMS load module for batch transaction processing. The NETWORK section cannot be repeated and must be the first section specified.

Format

```
NETWORK [BATCH= {n   }]
        [       {NO  }]
        [       {YES }]
        CONFID=n

        [CUP=locap-label]

        [KATAKANA= {NO  }]
        [          {YES }]

        [NAME= {network-name}]
        [      {IMS4       }]

        [PASSWORD=password-name]
        [PRCSNUM=n]

        [STATUSMG= {YES }]
        [          {NO  }]
```

(continued)

```
(continued)

      [TERMS=max-no-terminals]
      [UNSOL= {TRANS  }]
              {ACTION }
```

Example

```
NETWORK  BATCH=YES  NAME=NET3  PASSWORD=SESAME
 CONFID=3  CUP=IMS9    TERMS=75
```

Specifying Batch Processing of Transactions (BATCH)

The BATCH keyword parameter specifies whether this IMS single-thread or multithread system is to be configured for batch processing of transactions. If batch processing is specified, this keyword determines the number of batch pseudoterminals that the configurator will generate and the number of input source modules that may be processed at one time. Batch processing is described in Section 6.

BATCH=n
: Specifies that the batch transaction processor is to be configured in this IMS system and stipulates the number of batch pseudoterminals the configurator is to generate. The completion, *n*, is a decimal number in the range 1-4; it represents the number of batch pseudoterminals and determines the maximum number of batch input source modules and data sets that may be submitted at one time to the batch transaction processor. Only one batch pseudoterminal can be generated in a single-thread system.

Note: *If the value specified exceeds 1 for single-thread IMS or 4 for multithread IMS, the configurator assumes a default value of 1 for single-thread and 4 for multithread. It also issues a diagnostic message in the configurator listing (Figure 4-7).*

BATCH=NO
: Specifies that the batch transaction processor is not to be configured and is the default assumption.

BATCH=YES
: Specifies that the batch transaction processor is to be configured, that one batch pseudoterminal is to be generated, and that a single source input module will be processed at a time. The single-thread user may specify BATCH=YES or BATCH=1, both specifications having the same effect.

Specifying the Configuration Identifier (CONFID)

The purpose of the CONFID keyword parameter, which is always required, is to identify all configurator-generated records and control tables in the NAMEREC file with the specific IMS load module for which they are created. One NAMEREC file may contain records from as many as 255 distinct IMS configurations; the configuration identifier, or ID, allows each online IMS load module to single out the records it needs. (Configuration IDs are specified as decimal integers in the range 1-255.) Password definition records, data definition records, and input edit tables can be used by various IMS load modules since they do not necessarily relate to a specific configuration.

In specifying the CONFID keyword parameter, you may reuse the configuration ID specified for a previously configured IMS load module that used the same NAMEREC file. When you do so, the records newly generated by the current configurator run replace those already so identified in the NAMEREC file, and an appropriate diagnostic message is issued. (Refer to Figure 4-7.)

If you omit the CONFID keyword parameter or make an invalid specification, there is no default assumption. The IMS configurator terminates and issues the error message illustrated in Figure 4-8.

`CONFID=n`
: Specifies a 1- to 3-digit decimal integer in the range 1-255 that uniquely identifies all records generated for the NAMEREC file by the current configurator run with the IMS load module being configured.

IMS makes another use of the configuration identifier specified with the CONFID keyword -- to uniquely identify DTF and CDIB/RIB object modules generated by multiple configuration runs in the assembly step. The configuration identifier becomes part of the object module name, which takes the form:

```
ZS_I0nnn     (for single-thread DTF configurations)
ZQ_I0nnn     (for multithread DTF configurations)
ZS$I0nnn     (for single-thread CDM configurations)
ZQ$I0nnn     (for multithread CDM configurations)
```

where:

`nnn`
: Is the 3-digit configuration identifier (zero-filled, if necessary) specified with the CONFID keyword parameter for the configuration to which this DTF or CDIB/RIB object module applies.

Relating the IMS Program to a Global Network (CUP)

The CUP keyword parameter associates the online IMS program with a global network via the label of a LOCAP macro in the ICAM network definition.

`CUP=locap-label`
: Names the IMS program as given on the label of a LOCAP macro in the network definition.

Specifying Katakana Support (KATAKANA)

The KATAKANA parameter indicates whether Katakana characters are to be supported. The Japanese use Katakana to represent non-Japanese words. IMS supports the use of Katakana characters in transaction codes, lexicons created in LANGUAGE sections, defined file names, data definition records, routing characters, and urgent priority characters in input messages.

`KATAKANA=NO`
Indicates that the IMS system being configured is not to include Katakana support. This is the default assumption.

`KATAKANA=YES`
Indicates that the IMS system being configured is to include Katakana support.

Katakana support does not apply to IMS-supplied action programs (UNIQUE, ZSTAT, DLOAD, DLMSG).

Specifying the Name of the ICAM Network (NAME)

The NAME keyword parameter names the ICAM network used by IMS. The name specified must agree with either:

- the name specified via the CCAMOD keyword parameter of the COMMCT section input to the SYSGEN process when the ICAM symbiont is created; or
- the name specified as the label of the CCA macro in the COMMCT section. (Refer to Section 2.)

`NAME=network-name`
Specifies the 1- to 4-character alphanumeric name of the ICAM network. The first character must be alphabetic.

If the NAME keyword parameter is omitted, the configurator assumes that the network-name is IMS4.

Identifying the Network Password (PASSWORD)

This keyword parameter identifies the password associated with the ICAM network. If a password is assigned in the network definition (via the PASSWORD operand of the CCA macro), you must code the PASSWORD parameter in the configurator NETWORK section, and the two specifications must agree. If a password is not assigned in the network definition, this parameter may be omitted.

`PASSWORD=password-name`
Specifies the 1- to 8-character alphanumeric password assigned to the network when the ICAM load module was generated. The first character must be alphabetic.

If the PASSWORD keyword parameter is omitted, but a password was assigned to the network definition when the ICAM symbiont was created, the ICAM network will not be initialized or accessible to your IMS load module.

Specifying the Number of Process File DTFs (PRCSNUM)

The PRCSNUM keyword parameter specifies the number of process file DTFs to be generated for unsolicited output (specify UNSOL=YES in the OPTIONS section). The ICAM generation should have at least the same number of process files.

PRCSNUM=n
: Specifies the number of process file DTFs generated in IMS. The default value is 0 when UNSOL=NO, or 5 when UNSOL=YES is specified in the OPTIONS section. Maximum value is 25.

Process files must be allocated to an action in order for a transaction to send unsolicited output. Allocated process files remain unavailable to other actions until the original action terminates. (Note that a low PRCSNUM value may reduce concurrency when long-running actions produce unsolicited output). To increase process file availability, specify unsolicited output to be sent when the action is nearly complete.

Globally Suppressing IMS Timer-Generated Status Messages (STATUSMG)

The STATUSMG keyword parameter enables all IMS terminals to receive or to suppress IMS timer-generated status messages.

STATUSMG=YES
: Is the default, and enables all terminals to receive timer status messages.

STATUSMG=NO
: Suppresses timer status message reception to all terminals.

You can override the STATUSMG specification for individual terminals in the TERMINAL section. During online IMS sessions, use the ZZMSG and ZZNMG commands to allow or suppress timer status messages. Refer to the *IMS Operations Guide,* UP-12027, for a complete description of IMS terminal commands and transactions.

Limiting the Number of Online Terminals (TERMS)

The TERMS parameter sets a limit on the number of terminals that can be online at one time. It is required for both dedicated and global networks.

With a dedicated network, you must specify a value equal to or greater than the number of terminals listed in your ICAM network definition. You might want to specify more than the number of terminals in the network definition so you can regenerate ICAM (with additional terminals) without reconfiguring IMS.

With a global network, you must specify a value equal to or greater than:

- the number of static session terminals named in SESSION macros in the network definition, plus
- the number of dynamic session terminals defined to the configuration in TERMINAL sections (4.3.6), plus
- the maximum number of other dynamic session terminals allowed to be online at the same time.

`TERMS=max-no-terminals`
Stipulates the maximum number of terminals allowed to be online at one time.

If omitted, the configurator terminates with the message:

```
***TERMS MISSING OR INVALID - KEYWORD VALUE REQUIRED***
```

When you use a global network, IMS reserves a terminal slot for each static session terminal in the network definition and each dynamic session terminal that you define to the configurator in a TERMINAL section. The remaining slots -- that is, the difference between the number you specify with the TERMS parameter and the number of reserved terminal slots -- are available for the remaining dynamic session terminals.

For example, if you have 75 terminals in your network, of which 25 are static session terminals and 10 are defined in TERMINAL sections, 35 terminal slots are reserved at all times. If you specify TERMS=50, 15 terminal slots are available for the 40 other dynamic session terminals in your network.

Also, when you start up IMS with STARTUP=WARM or STARTUP=COLD, any dynamic session terminal that was in the middle of an updating transaction when IMS last terminated has a slot reserved for it. The terminal can sign on ($$SON) and issue the DLMSG transaction code to retrieve the output message of its last completed transaction. After the terminal signs off ($$SOFF), the terminal slot is released.

In the previous example, suppose two of the dynamic session terminals not listed in TERMINAL sections are in the middle of updating sessions when IMS terminates, and you start up IMS again with a warm or cold start. IMS reserves slots for the two terminals, leaving only 13 slots for the other 38 terminals in the network.

Globally Specifying Message Notification after Each Action (UNSOL)

Unless otherwise specified, switched unsolicited output messages are queued for a terminal during the transaction process. After the transaction is complete, the terminal operator receives an unsolicited output indicator. If you want all terminals to receive messages at the end of each action, use the UNSOL keyword parameter. This specification creates the default condition for all terminals.

UNSOL=ACTION
Globally specifies all terminals to receive switched, unsolicited output messages on completion of each action.

UNSOL=TRANS
Globally specifies all terminals to receive switched, unsolicited output messages on completion of a transaction. TRANS is the default.

To override this specification for individual terminals, use the TERMINAL section. When UNSOL = ACTION is specified, terminal operators receive a queued message via the unsolicited output indicator. The terminal operator can accept the queued output or transmit another input message. If an input message is transmitted, the input will be processed while the switched messages remain queued.

Sample Configurator Output for the NETWORK Section

Consider the portion of a configurator listing illustrated in Figure 4-7 and assume that the first line represents input for the NETWORK section in a single-thread configuration.

The specification of the BATCH keyword (BATCH=2) exceeds the maximum for single-thread; this causes the error message on the second line to be issued and the default value (BATCH=1) to be assumed, as shown on the bottom line.

The configuration identifier specified (CONFID=7) evidently is a reuse of the ID specified for a previously configured IMS load module using the same NAMEREC file. The third, fourth, and fifth lines of the listing represent the diagnostic message issued to notify you of the replacement of the NAMEREC file records from the earlier run by those generated in this run. See also 4.2.3 and Figure 4-5.

```
   00001    NETWORK   NAME=IMS1  PASSWORD=IMSNET01 BATCH=2 CONFID=7 TERMS=20
***TOO MANY BATCH TERMINALS - DEFAULT VALUE ASSUMED***
THIS CONFIGURATION REPLACES THE FOLLOWING ONE:
  CNF-ID  CNF-NETW  CNF-DATE  CNF-TIME    CNF-VERS    CNF-ERRORS
   0007     IMS1   01/11/86  14:41:58    VER10.0.08-ST   0005
SECTION NETWORK  INPUT AND DEFAULT VALUES
NAME     = IMS1      INPUT
PASSWORD = IMSNET01  INPUT
CONFID   = 7         INPUT
CUP      = IMS3      INPUT
TERMS    = 20        INPUT
BATCH    = 1         DEFAULT
NETWORK
```

Figure 4-7. Configurator Listing for BATCH Keyword Specification Error and Specification of Previously Used Configuration Identifier (NETWORK Section)

Figure 4-8, a portion of another configurator listing, illustrates the output made when the CONFID keyword parameter is not specified. The first line represents the NETWORK input, the second shows the error message issued by the configurator, and the following lines represent a tabulation of the values input or assumed by default for the section.

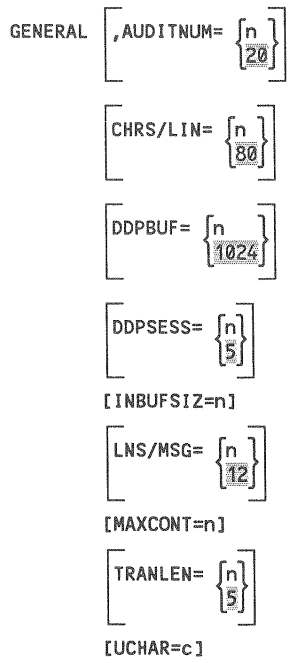
```
  00001    NETWORK   NAME=IMS1  PASSWORD=IMSNET01 BATCH=2 TERMS=20
***NO CONFID SPECIFIED - CONFIGURATION TERMINATED***
SECTION NETWORK  INPUT AND DEFAULT VALUES
NAME     = IMS1      INPUT
PASSWORD = IMSNET01  INPUT
CONFID   = ***ERROR  DEFAULT
BATCH    = 2         INPUT
TERMS    = 20        INPUT
NETWORK
```

Figure 4-8. Configurator Listing for Omission of CONFID Keyword Parameter (NETWORK Section)

4.3.2. Specifying Overall Configuration Parameters - the GENERAL Section

The GENERAL section defines certain general characteristics of this IMS configuration. This section is required and nonrepeatable.

Format

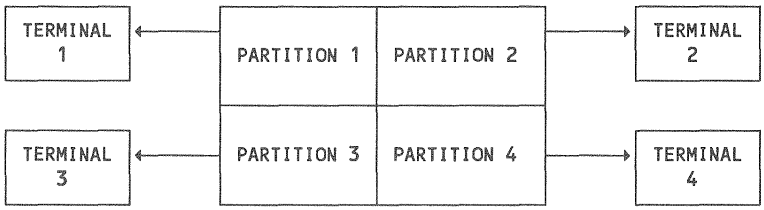


Example

```
GENERAL CHRS/LIN=80 LNS/MSG=16 MAXCONT=400
```

Designating Number of Audit Records (AUDITNUM)

This keyword specifies the number of audit records required per terminal in the audit file (the AUDFILE for multithread IMS; the AUDCONF file for single-thread). The audit file, as shown in the following diagram, is a partitioned system access technique (SAT) file, containing as many partitions as there are terminals in your network. Each partition is allocated to one of the terminals and is formatted to contain the number of records specified by the AUDITNUM keyword parameter. The records written to the file are the before-images of records to be updated.



At rollback points and transaction termination, the current record pointer for the active partition is reset to point to the first record of the partition. If a transaction generates more before-images than specified by the AUDITNUM keyword before writing a rollback point or before it terminates, it is canceled and the record pointer for the active partition is reset.

AUDITNUM=n
: Specifies maximum number of records required to contain all before-images written from one terminal to the audit file between two rollback points or between initiation of one transaction and its termination (whichever occurs first). The default assumption is 20 records for either single-thread or multithread IMS. If file updating is not configured, specify AUDITNUM=1. AUDITNUM value x (TERMS value + BATCH value) cannot exceed 16,777,215.

For details on the partitioned SAT file, refer to the *Supervisor Macroinstructions Programming Reference Manual,* UP-8832.

Specifying Message Line Length (CHRS/LIN)

The CHRS/LIN keyword parameter determines the number of characters per line in the standard IMS message.

CHRS/LIN=n
: Specifies the standard number of characters per line for your IMS applications. The default value is 80.

Specifying DDP Buffer Size (DDPBUF)

The DDPBUF keyword parameter specifies the size of the buffer required for distributed data processing (multithread IMS only).

DDPBUF=n
: Specifies the length of the DDP buffer. The default assumption is 1024.

You should calculate the buffer size as follows:

Length of 80 to 90 percent of your messages (including DICE) + 16 bytes for ICAM header + 70 bytes for DDP protocol

The maximum buffer size you can specify is 4352 bytes. If the buffer size you specify is not large enough to handle all messages, you must specify FEATURES=(SEGMENTS) on the CCA macro in your network definition.

Specifying Number of DDP Sessions (DDPSESS)

The DDPSESS keyword parameter specifies the maximum number of distributed data processing sessions that can be active in this system simultaneously (multithread IMS only). It is the total of:

- transactions this system sends to a remote system for processing, plus
- transactions sent to this system by another for processing

`DDPSESS=n`
Specifies the maximum number of DDP sessions that can be active at one time. The default is 5. Specify a value between 2 and 25.

Specifying Input Message Buffer Size (INBUFSIZ)

The INBUFSIZ parameter specifies the size of the input message staging area that IMS allocates at start-up time. You calculate this specification differently for single-thread and multithread IMS, and online IMS startup processing generates different default values for single-thread and multithread systems when you omit this parameter. The reason for the different calculation is that the input message staging area requires a single buffer in a single-thread system and multiple buffers in a multithread system. In either case, the INBUFSIZ value should be a multiple of 8.

`INBUFSIZ=n`
Specifies the length of the input message staging area.

Single-thread

In a single-thread system, the default value is 2048. The maximum value you can specify is 32,760.

When you include the INBUFSIZ parameter in a single-thread IMS system that runs with transient ICAM, the specification must be equal to or greater than the size of the ICAM buffer.

If you have action programs that use delayed internal succession or output-for-input queueing, your input message staging buffer must be large enough to contain the largest output message generated by these programs. Use the following formula to calculate the INBUFSIZ specification:

```
(CHRS/LIN+4) * (LNS/MSG) + 24 + (5 * (number of FCCs))
```

where the number of field control characters (FCCs) is the maximum number of FCCs that appear in a message. Ignore the 5 * (number of FCCs) portion of the formula if your network does not include FCC terminals. The result of the entire expression is double-word aligned.

When you use screen format services, your input message staging buffer must be double the output area length needed to build your largest screen format. To determine the INBUFSIZ, use the formula described under the OUTSIZE parameter for screen format support (4.3.8) and double the value obtained. Use this formula regardless of whether your screen formats are built in dynamic main storage or in the output message area.

Multithread

When you omit the INBUFSIZ parameter in multithread IMS, online IMS startup processing calculates the size from Step 1 or from both of the following steps:

Step 1: For up to 30 terminals. The calculated value ranges from a minimum of 8192 to a maximum of 16384.

$$\frac{(\text{standard message size}) * (\text{number of terminals; up to 30})}{2}$$

$$= \text{Size of Step 1 of } \begin{cases} n \text{ rounded up to 256 byte aligned} \\ \text{or} \\ 8192 \text{ minimum} \\ \text{or} \\ 16384 \text{ maximum} \end{cases}$$

Step 2: For more than 30 terminals. The calculated value does not exceed a maximum of 32512.

$$\left(\left(\frac{\text{number of terminals} - 30}{5}\right) * 1024\right) + \text{Size of Step 1}$$

$$= \text{IMS calculation of INBUFSIZ, 256 byte aligned}$$

where:

$$\text{standard message size} = (\text{CHRS/LIN}+4) * (\text{LNS/MSG}) + 12$$

The default message value generated by online IMS start-up processing is usually adequate. However, depending on the number of transactions you have and the sizes of your messages, you may find that a smaller or larger INBUFSIZ specification is optimum for your system. An inadequate input message staging area size can degrade performance and cause deadlock.

Specifying the Number of Lines per Message (LNS/MSG)

The LNS/MSG keyword parameter determines the number of lines in the standard IMS message.

`LNS/MSG=n`
Specifies the standard number of lines per message for your IMS applications. The default value is 12.

Defining Largest Continuity Data Area (MAXCONT)

The MAXCONT keyword parameter defines the size of the largest continuity data area in this IMS configuration.

`MAXCONT=n`
Specifies the number of bytes in the largest continuity data area for any action. This value should be a multiple of 256. The minimum value you should specify is 512 if UNIQUE and DLLOAD are not configured, 1024 if only DLLOAD is configured, and 2048 if UNIQUE is configured.

If MAXCONT is omitted, all action programs (including UNIQUE) are scanned, and the size of the largest continuity data area is used. If UNIQUE is included and the largest CDASIZE specified for a user action program is less than 2048, IMS assumes MAXCONT=2048 and issues a diagnostic message.

Note: *MAXCONT is used to determine the logical block size for the continuity data file (CONDATA) in a multithread configuration or the continuity data partition of the AUDCONF file in single-thread. If you specify both MAXCONT=0 and UNIQUE=NO, the continuity data file or partition is not allocated. If UNIQUE capability is included, the UNIQUE CDA requirements override a MAXCONT=0 specification.*

Specifying Maximum Length of Transaction Code Names (TRANLEN)

The TRANLEN keyword parameter specifies the maximum number of characters allowed in transaction code and lexicon names (multithread IMS only).

TRANLEN=n
: Specifies the maximum length for transaction codes and lexicon names. You must specify a value for *n* in the range 5-8. The default is 5.

If the transaction code specified in the TRANSACT section is longer than the value specified for *n* (or the default value), IMS truncates the code to *n* characters (or 5).

Specifying Urgent Priority on Input Messages (UCHAR)

The UCHAR keyword parameter specifies which character designates urgent priority when used as the first character of an input message. It is not available to single-thread users.

UCHAR=c
: Designates the character that, when included as the first character of an input message, notifies IMS that the message has urgent priority. If omitted, the default is a blank; no urgent priority character is recognized. We recommend that you use a special character so that all alphanumerics can be used as the first character of normal messages.

  If KATAKANA = YES is specified in the NETWORK section, this character can be a Katakana character.

In a multithread IMS system, an input message can be given urgent priority in any of the following ways:

- The first character of the message has been designated as the urgent priority character.
- The terminal at which the message is entered has been designated as an urgent priority terminal (4.3.6).
- The transaction code of the message has been designated as an urgent priority transaction code (4.3.7).

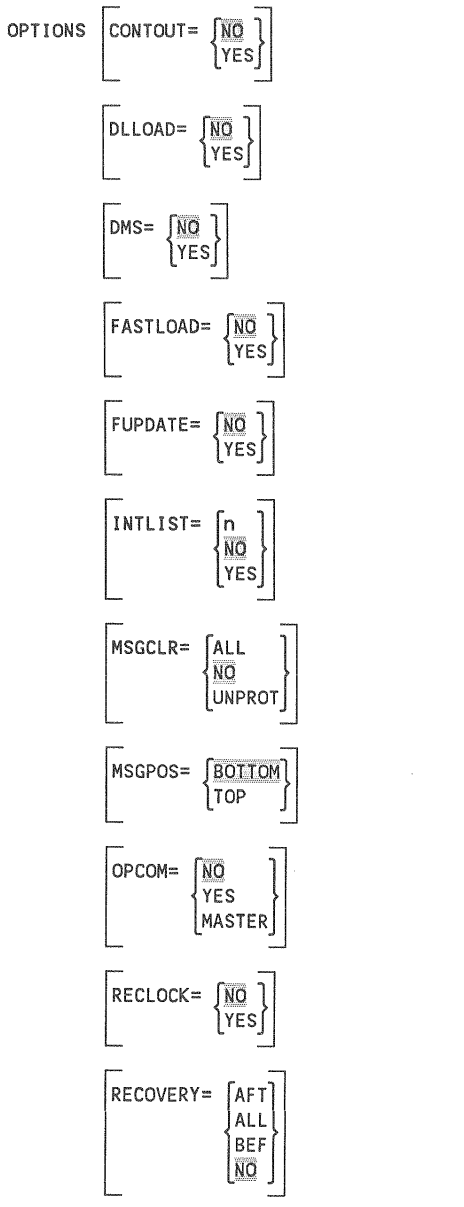
IMS gives these threads/ terminals/actions/functions a higher priority during scheduling and execution. Consider this when doing any of the following in order to avoid deadlocks or time-out cancellations when these features are utilized:

- Designing content of action programs
- Choosing options for holding/releasing of locks
- Choosing expiration or loop values

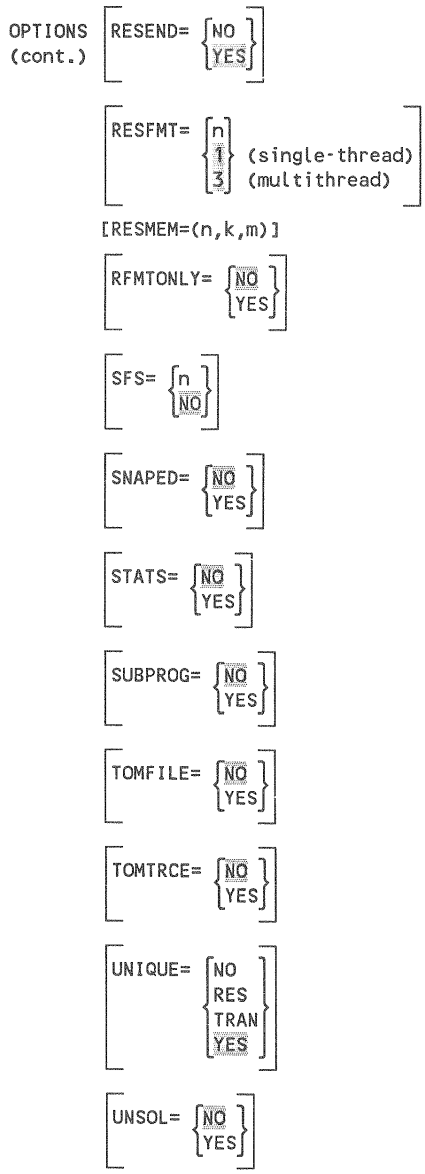
4.3.3. Including Optional IMS Modules - the OPTIONS Section

The OPTIONS section of the configurator input specifies whether certain optional IMS modules are to be included in this configuration. The OPTIONS section is optional and nonrepeatable.

Format



(continued)



Example

```
OPTIONS UNIQUE=TRAN RECOVERY=ALL FUPDATE=YES
 TOMFILE=YES CONTOUT=YES SNAPED=YES
```

Including Continuous Output Capability (CONTOUT)

The CONTOUT keyword parameter determines whether the capability for continuous output is to be included in this configuration. It is available for single-thread and multithread IMS. You must configure continuous output in order to:

- Print continuous output transactions, either at the originating terminal or at a different terminal
- Address output to an auxiliary device
- Disconnect a single-station dial-in line following delivery of an output message
- Downline load a user program to a UTS terminal

If you specify CONTOUT=YES, IMS also includes the capability for unsolicited output in your configuration, whether or not you also specify UNSOL=YES, because continuous output functions require the availability of the unsolicited output module. Continuous output capability is automatically included if you specify downline loading by means of the DLLOAD parameter.

`CONTOUT=NO`
Specifies that continuous output capability is not to be included in this configuration and is the default assumption unless downline loading capability is configured (DLLOAD=YES). If specified along with DLLOAD=YES, CONTOUT=NO is ignored.

`CONTOUT=YES`
Specifies that continuous output capability is to be included in this configuration.

Providing Support for Downline Loading (DLLOAD)

The DLLOAD keyword parameter determines whether capability for downline loading of user programs to a UTS 40 or UTS 400 terminal is to be configured. You must specify DLLOAD=YES whether you want to downline load your UTS programs by means of the IMS-provided transaction DLOAD or your own downline load action program. Downline loading requires continuous output capability; therefore, the configurator automatically generates CONTOUT=YES and UNSOL=YES when you include downline loading.

`DLLOAD=NO`
Indicates that downline loading capability is not to be included in this configuration.

`DLLOAD=YES`
Specifies that downline loading capability is to be included.

Allowing IMS/DMS Interface (DMS)

The DMS keyword specifies IMS access to DMS data bases from user action programs or UNIQUE.

DMS=YES
: Allows IMS access to DMS data bases.

DMS=NO
: Does not allow IMS/DMS interface.

Using the Fast Load Feature (FASTLOAD)

The FASTLOAD parameter is available in single-thread and multithread IMS. It allows IMS to create its own load file (LDPFILE) during online processing by copying the action program load modules from the load library.

The first time a transaction calls on an action program, IMS copies the action program into the LDPFILE. After that, the action program is loaded from the LDPFILE.

The fast load feature improves performance for applications with large action programs or extensive action program loading. It increases the size of online IMS by about 6000 bytes.

To use the fast load feature, you must place all action programs in a separate load library in unblocked format and assign this library at IMS start-up with the LFD-name LOAD. Do not place action programs in the system load library, YLOD. You must allocate the LDPFILE as a system access technique (SAT) file and assign it at start-up. The LDPFILE should be 20 percent larger than the combined size of all action programs.

FASTLOAD=NO
: Specifies that the fast load feature is not included in this configuration.

FASTLOAD=YES
: Includes the fast load feature.

Note: *When using the fast load feature, you must place IMS action programs, including the online recovery program, ZS#ROL (single-thread) or ZU#ROL (multithread), in the LOAD action program library.*

Allowing Updating of Files (FUPDATE)

This keyword parameter specifies whether file updating is to be performed in this IMS configuration. If UNIQUE is to be used for updating files, you must specify FUPDATE=YES. If you specify FUPDATE=YES, IMS writes before-images to the audit file of all records of a file for which you specify LOCK=TR in the FILE section.

FUPDATE=NO
: Specifies that file updating may not be performed; update modules are not included in this configuration.

FUPDATE=YES
: Specifies that file updating may be performed; necessary modules are included by the configurator.

Specifying Interruption of LIST Output (INTLIST)

The INTLIST keyword parameter increases throughput by automatically interrupting the generation of output from the UNIQUE LIST command so that alternate actions may be scheduled while a LIST command is in effect. It is most useful when many conditional LIST commands are to be performed and is used only with UNIQUE.

If INTLIST=YES or INTLIST=*n* is specified, LIST processing is interrupted automatically with either the first screenful of output or when the number of file accesses exceeds 200 or your number specification (*n*).

When the INTLIST access number is reached, LIST processing is interrupted and the accumulated results are output to the terminal. The terminal operator can:

- Cancel the current transaction
- Enter another command
- Continue LIST generation by transmitting the MORE command, repeating this process until conditional LIST requirements are satisfied or until a new transaction is entered

The number of accesses is checked repeatedly for the length of the file.

INTLIST=n
: Specifies that LIST generation at the terminal is interrupted automatically on the first screenful of output or when this number of file accesses is reached, whichever occurs first. The maximum value of *n* is 32,000.

INTLIST=NO
: Indicates that the interrupted LIST is not to be performed and is the default assumption.

INTLIST=YES
Specifies that list generation is to be interrupted automatically when the first screenful of output has been generated during execution of the LIST command, or when 200 file accesses have occurred, whichever takes place first. The results are output to the terminal.

Clearing the Screen (MSGCLR)

The MSGCLR keyword parameter specifies that IMS is to clear the screen before displaying a message. You can clear:

- Both protected and unprotected data
- Unprotected data only
- Only the lines required to display the message

MSGCLR=UNPROT
Specifies that IMS is to clear the screen of unprotected data and return the cursor to the home position.

MSGCLR=ALL
Specifies that IMS is to clear the screen of both protected and unprotected data and return the cursor to the home position.

MSGCLR=NO
Specifies that IMS is to clear only the lines required for the message. This is the default assumption.

Positioning Messages on the Screen (MSGPOS)

The MSGPOS keyword parameter specifies whether IMS status, error, and informational messages are to be displayed at the top or bottom of the screen. In either case, the message will occupy however many lines it needs to be displayed in its entirety. By using MSGPOS, you can prevent user data from being overwritten by IMS messages.

MSGPOS=BOTTOM
Specifies that the message is to be displayed at the bottom of the screen. This is the default assumption.

MSGPOS=TOP
Specifies that the message is to be displayed at the top of the screen.

Providing Support for Console Transaction Processing (OPCOM)

The OPCOM keyword parameter enables single-thread and multithread IMS users to enter terminal commands and transactions from the system console or master workstation. If you do not designate a master terminal (by including the MASTER=YES parameter in a TERMINAL section), IMS considers the system console or master workstation to be your master terminal. In this case, support for console transaction processing is automatically included, and the configurator ignores the OPCOM parameter.

OPCOM=NO
: Specifies that console support is not included in this IMS configuration and is the default assumption.

OPCOM=YES
: Specifies that console transaction processing is supported.

OPCOM=MASTER
: Designates the console as a master terminal. This enables you to use IMS master terminal commands and IMS transaction codes from the console. As many IMS terminals as desired can also be designated as master terminals.

An IMS session may have a master workstation associated with it either by job control or by entering the IMS job from a workstation. If you configure the console as the master terminal or specify OPCOM=YES, you can enter unsolicited keyins from either the system console or the job's master workstation. Switched messages, however, are always sent to the master workstation (if present) and are sent to the console only if the job has no master workstation or if the workstation is logged off. For more information on console transaction processing, refer to the *IMS Operations Guide*, UP-12027.

Locking Records (RECLOCK)

The RECLOCK keyword parameter provides the ability to carry record locks across actions in single-thread IMS. This feature is automatically available under multithread IMS; the RECLOCK parameter is not valid for multithread.

RECLOCK=NO
: Specifies no record locking across actions.

  When you use RECLOCK=NO with the LOCK=TR parameter in the FILE section, the N and O options on the lock-rollback indicator of the program information block are available; the R and H options are not available. Online recovery is performed.

  With the LOCK=UP parameter, there is no online recovery and no use of lock-rollback indicators for the file.

RECLOCK=YES
Allows record locking across actions in single-thread. Record locking across actions is always available in multithread.

When you use RECLOCK=YES with the LOCK=TR parameter in the FILE section, all lock-rollback indicators are available and online recovery is performed for the file.

With the LOCK=UP parameter, there is no online recovery and no use of lock-rollback indicators for the file.

Specifying Recovery Options (RECOVERY)

The RECOVERY keyword parameter determines whether the images of updated records in DAM, ISAM, and MIRAM files are to be written online to the trace file for later use in offline recovery and what images are to be written.

RECOVERY=AFT
Specifies that after-images of all updated records (those added or deleted, as well as those changed) are to be written to the trace file.

RECOVERY=ALL
Specifies that before- and after-images of all records being updated (changed, added, or deleted) are to be written to the trace file.

RECOVERY=BEF
Specifies that before-images of all records being updated (changed, added, or deleted) are to be written to the trace file.

RECOVERY=NO
Specifies that no images are to be written to the trace file and is the default assumption.

Disallowing Use of ZZRSD Terminal Command (RESEND)

The RESEND keyword parameter provides or disallows the capability for your terminal operators to cause the most recent output response to be retransmitted by entering the ZZRSD command. Disallowing the use of the ZZRSD command (RESEND=NO) permits better utilization of existing ICAM buffers. Those that would normally be tied up holding output messages for retransmittal are instead released promptly to ICAM. In addition, since there is no mechanism for releasing invalid data held in message queues, use the RESEND=YES keyword with caution. If you forego the use of the ZZRSD command, you may be able to generate ICAM with fewer buffers. The ZZRSD command has no role in continuous output, and it is not supported for the console.

RESEND=NO
Specifies that the capability of using the ZZRSD terminal command is not to be included in this configuration. If the ZZRSD command is entered at a terminal, the response "NO MESSAGE IN QUEUE" is displayed.

RESEND=YES
Specifies that the capability of using the ZZRSD terminal command is to be included in this configuration and is the default assumption. Does not increase main storage requirements.

Specifying Number of Resident Screen Formats (RESFMT)

The RESFMT keyword parameter specifies the maximum number of screen formats per terminal that will remain resident between successive calls to IMS for screen format services (SFS).

RESFMT=n
Specifies the maximum number of screen formats that will remain resident per terminal between SFS calls. The default assumption is 1 for single-thread IMS and 3 for multithread IMS. The value specified must take into consideration the SFS keyword value specified. The product of the SFS and RESFMT values (SFS=*n* * RESFMKT=*n*) must not exceed 255. (See RFMTONLY keyword.) The RESFMT parameter is valid only when SFS=*n* is specified. Refer to the *Screen Format Services Technical Overview,* UP-9977.

Reserving Main Storage for a Transaction Buffer Pool (RESMEM)

The RESMEM keyword parameter enables you to reserve a transaction buffer pool at system generation. The transaction buffer pool is a block of main storage that can be used by action program transactions that need additional storage. The transaction buffer pool can also be used to pass data from one action to another within a transaction. Using the transaction buffer pool to pass data is more efficient than using the CDA for the same purpose. With CDA, the data is written to and read from disk. With transaction buffers, the data is stored in and retrieved from memory. For information on how to use transaction buffers to pass data, refer to the appropriate manual: *IMS Action Programming in RPG II Programming Guide,* UP-9206, and *IMS COBOL/Assembler Action Programs Programming Guide,* UP-9207.

The transaction buffer pool consists of transaction buffers that are each 4096 bytes. With RESMEM, you specify the size of the transaction buffer pool reserved during system configuration. You may reserve a transaction buffer pool of up to 32767 transaction buffers. You may also reserve up to 32767 additional transaction buffers from main storage that will become available after the initial transaction buffer pool is depleted.

Transaction buffers are acquired by the action program for a particular transaction, used by that transaction, and then returned to the transaction buffer pool. Transaction buffers can be returned to the buffer pool before the transaction terminates; otherwise, the transaction buffers are automatically returned to the pool when the transaction terminates.

An action program issues function calls to GETMEM and RELMEM to acquire and release transaction buffers. An action program may issue up to three calls to acquire transaction buffers for a single transaction. Transaction buffers are always assigned to transactions in multiples of 4096 contiguous bytes. A single transaction can acquire up to 16 transaction buffers. For detailed information on using transaction buffers in RPG II, COBOL, and BAL, refer to the appropriate manual: *IMS Action Programming in RPG II Programming Guide,* UP-9206, and *IMS COBOL/Assembler Action Programs Programming Guide,* UP-9207.

You can use the // PARAM RESMEM statement in IMS startup to override the RESMEM specifications set during IMS configuration. The // PARAM RESMEM statement can be used only if RESMEM is defined, so the ZM#MEM module is included in your configuration load module. The ZM#MEM module maintains the transaction buffer pool. You may wish to define all RESMEM values as zero during configuration and then describe the size of the transaction buffer pool using the // PARAM RESMEM statement. Refer to section 5 of this manual for a description of the // PARAM RESMEM statement.

You cannot recover the contents of a transaction buffer after a RELMEM call, IMS termination, or abnormal system termination.

The RESMEM keyword parameter format is:

```
RESMEM=(n,k,m)
```

where:

n
: Describes the number of transaction buffers you wish to reserve during startup. The transaction buffers reserved during startup form a transaction buffer pool. RESMEM=*n* is a numeric value from 0 to 32767. You must specify *n*. (Each transaction buffer is 4096 bytes.)

k
: Is the maximum number of transaction buffers that can be acquired from IMS main memory after the transaction buffer pool is depleted. RESMEM=*k* may not exceed 32767. The default for *k* is zero (0).

m
: Is the maximum number of transaction buffers that a transaction may acquire. RESMEM=*m* may not exceed 16. The default for *m* is 4.

Set Absolute Maximum Number of Resident Screen Formats (RFMTONLY)

The RFMTONLY keyword parameter is used to indicate that the RESFMT keyword value or the product of the SFS and RESFMT keyword parameter values is to be used as the absolute maximum number of resident screen formats. This value will be used as the number of resident screen formats, which may not exceed 255.

RFMTONLY=NO

The number of resident screen formats will be the product of the SFS and RESFMT keyword specifications, which is not to exceed 255.

RFMTONLY=YES

The number of resident screen formats will be as specified by the RESFMT keyword value.

Note: *If the RFMTONLY keyword specification is* NO *and the product of keywords SFS and RESFMT exceeds 255, the RFMTONLY specification will be changed to* YES *and the RESFMT value will be set to 255. Refer to the* Screen Format Services Technical Overview, *UP-9977.*

Providing Support for Screen Formatting (SFS)

The SFS parameter allows your action programs to call on IMS to construct screen formats via the OS/3 screen format coordinator. Screen format services require a control system generated in CDM mode or mixed mode. However, you can use screen format services with an IMS system configured in either CDM or DTF mode. With the SFS parameter, you designate the number of terminals that can use screen formatting simultaneously. Refer to the *Screen Format Services Technical Overview,* UP-9977.

You must allow one slot for each terminal that:

- Is waiting for formatted screen output
- Has received formatted screen output and is waiting for the operator to enter input (if the screen contains input fields)
- Has entered input and is waiting for either an error screen or verification that the input was accepted

SFS=n

Allows action programs to use screen format services. The *n* specifies the maximum number of terminals that will use screen formatting simultaneously. Do not specify more than 255.

SFS=NO

Specifies that screen format services are not included in this configuration.

Specifying an Edited Directory for Snapshot Dumps (SNAPED)

The SNAPED keyword parameter specifies whether an edited directory of key IMS information is to be printed with snapshot dumps. This option is available only for multithread IMS and increases the size of the online load module by 3600 bytes. In single-thread IMS, termination snapshot dumps routinely include an edited directory; CALL SNAP dumps do not include an edited directory.

SNAPED=NO
: Specifies that snapshot dumps are not to include an edited directory.

SNAPED=YES
: Specifies that an edited directory is to be printed preceding snapshot dumps.

Recording Statistical Information at Shutdown (STATS)

The STATS keyword parameter provides for writing of statistical information in the statistics file (STATFIL) at IMS shutdown. When you specify STATS=YES, IMS automatically schedules the ZSTAT transaction at shutdown time. ZSTAT records data on the activity of files, programs, transactions, and terminals during a session.

STATS=NO
: Specifies that data is not written into the statistics file at shutdown. It is the default assumption.

STATS=YES
: Specifies that data is written into the statistical file at shutdown.

Note: *STATS=YES is not valid if you specify NO for subparameter 6 of the INIT keyword. The INIT keyword specification takes precedence.*

Including Support for User-Written Subprograms (SUBPROG)

The SUBPROG keyword parameter specifies that resident user-written subprograms may be called by COBOL or BAL action programs. To use this feature, you must also indicate the name of each subprogram on the program-name parameter of a PROGRAM section and specify SUBPROG=YES in the same section.

SUBPROG=NO
: Specifies that user-written subprograms are not to be called by COBOL or BAL action programs.

SUBPROG=YES
: Specifies that COBOL or BAL action programs may call resident user-written subprograms.

Generating a Partition for Terminal Output Messages (TOMFILE)

The TOMFILE keyword parameter specifies that IMS is to generate a partition, called the TOMFILE, in the AUDCONF file (single-thread) or the CONDATA file (multithread) to hold terminal output messages for use by your terminal operators during online recovery and at cold or warm start-up. You must specify TOMFILE=YES to use the warm start feature in a single-thread configuration; it is optional in a multithread configuration.

During online operations, IMS writes to the TOMFILE partition the output message generated for each terminal at rollback points and at transaction termination, retaining only one message per terminal. The terminal operator can display the most recent message by entering the transaction code DLMSG. These output messages are also recorded in the trace file for use in offline recovery if you specify the TOMTRCE configuration parameter.

You can optionally specify TOMFILE=YES to use a warm start-up for a multithread configuration. When TOMFILE=YES is not specified, the transaction code DLMSG is not valid for the IMS session and is not appended to the IMS READY message. In addition, output messages are not written to the TOMFILE.

`TOMFILE=NO`
Specifies that a TOMFILE partition is not to be generated and is the default assumption.

`TOMFILE=YES`
Specifies that IMS is to generate a TOMFILE partition in the AUDCONF or CONDATA file at configuration time and to write to this partition, during online operations, the output message generated for each terminal at each rollback point and at transaction termination.

Specifying Terminal Output Message Tracing (TOMTRCE)

When you include the TOMFILE keyword parameter, online IMS writes terminal output messages to the TOMFILE partition of the AUDCONF or CONDATA file for use during online recovery. When you also specify TOMTRCE=YES, these messages are also written to the trace file so that they are available for reconstructing the TOMFILE partition, using the ZC#TRC offline recovery program. If output messages are to be traced, the configurator examines OUTSIZE specifications as well as data record sizes in computing the block size of the trace file.

`TOMTRCE=NO`
Specifies that output messages are not to be traced and is the default assumption.

`TOMTRCE=YES`
Specifies that terminal output messages written to the TOMFILE partition are also written to the trace file.

Configuring UNIQUE Capability (UNIQUE)

You use the UNIQUE keyword parameter to specify whether or not the uniform inquiry update element (UNIQUE) is to be used in this IMS configuration and whether the UNIQUE command interpreter action program (ZU#CIN) is to be resident. Refer to the MAXCONT keyword parameter of the GENERAL section.

`UNIQUE=NO`
Specifies that the UNIQUE capability is not to be included in this configuration.

`UNIQUE=RES or YES`
Specifies that the UNIQUE command interpreter action program (ZU#CIN) is resident (that is, permanently loaded at IMS start-up time). This specification adds approximately 1700 bytes to your main storage requirements.

`UNIQUE=TRAN`
Specifies that all UNIQUE action programs are to be loaded as required.

If omitted, IMS assumes UNIQUE=YES.

Providing Capability for Unsolicited Output (UNSOL)

The UNSOL keyword parameter specifies whether the capability for unsolicited output is to be included in your IMS configuration.

Unsolicited output involves generating multiple messages or sending messages to terminals other than the originating terminal. Unsolicited output capability is automatically included if continuous output is configured by means of the CONTOUT keyword parameter or if downline loading is configured with the DLLOAD parameter.

UNSOL=YES (or CONTOUT=YES) must be specified if:

- A user-written action program issues the SEND function
- The ZZTST master terminal command is used to test whether a network terminal is able to receive output
- The SWTCH transaction is used for terminal-to-terminal communication

In order to use unsolicited output, you must generate an ICAM module that supports unsolicited output. See 2.3.2 for specific requirements.

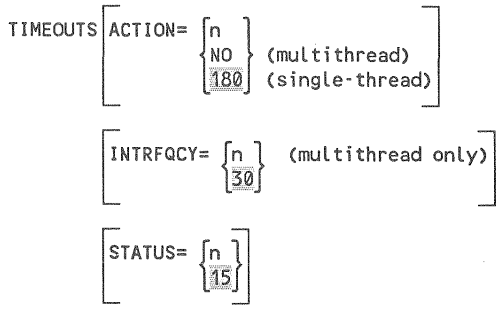
UNSOL=NO
: Specifies that no capability for unsolicited output is to be included in this configuration. When specified along with CONTOUT=YES or DLLOAD=YES, UNSOL=NO is ignored; no diagnostic message is issued.

UNSOL=YES
: Specifies that unsolicited output is to be included in this configuration. This is the default assumption when you specify CONTOUT=YES or DLLOAD=YES.

4.3.4. Specifying Timeout Values - the TIMEOUTS Section

The TIMEOUTS section stipulates the various timeout values to be used in this IMS configuration. The TIMEOUTS section is optional and nonrepeatable.

Format



Example

```
TIMEOUTS ACTION=180 INTRFQCY=10 STATUS=30
```

Specifying Action Program Elapsed Time (ACTION)

The ACTION keyword parameter specifies the maximum number of seconds that an action program is allowed to have control before timing out. This specification has a different meaning in a single-thread than in a multithread IMS system. In single-thread, it is the total execution time for an action program; in multithread, it is the amount of time an action program may retain control between calls to IMS.

ACTION=n
: Specifies the action timeout value in seconds.

Single-thread

In a single-thread system, *n* is the maximum number of seconds that an action program is allowed to execute. Maximum timeout value is 900 seconds. The minimum and default value is 180 seconds.

IMS uses a different timeout value to detect a program loop. You do not specify this value -- it is always 60 seconds. When an action program fails to issue a function call within 60 seconds after getting control, the program is canceled.

IMS uses a third timeout value at shutdown time. Ongoing transactions are allowed to continue processing for 180 seconds after the ZZSHD master terminal command is issued. You can override this value by specifying the shutdown timeout on the ZZSHD terminal command.

Multithread

In a multithread system, *n* (0 to 32,767) is the global action timeout value that IMS uses to detect an action program loop or hang condition. When an action program fails to terminate with E, D, or N in the termination indicator within *n* seconds after getting control, and optional status message goes out on the terminal, indicating an action timeout is imminent:

```
CANCEL PENDING      >ZZRDY *TS
```

From this point, the user has a specific time period in which to transmit the appended terminal command ZZRDY *TS. This time period varies; it is either three times the STATUS value in the TIMEOUTS section or 90 seconds, whichever is greater.

If the user responds to the CANCEL PENDING message within the allotted time, the action timer is reset and the transaction is allowed to continue.

If the ZZRDY *TS command is not permitted from this terminal, the CANCEL PENDING message is suppressed and the transaction is immediately marked for cancellation.

The minimum action timeout value (which is also the default) is 0 seconds. This indicates no timeouts are desired; it is equivalent to specifying NO. Values from 1 to 32,767 seconds activate the action timer for that duration. Since this value represents an elapsed time among all jobs within the system, it should be set to a value that represents the situation when the system is under it heaviest load and IMS is executing a worst-case job priority selection.

To set the timers on an individual action basis, refer to the EXPIRTME parameter in the ACTION section (4.3.8).

If no timeouts are selected on a global or individual basis, IMS will timeout only programs looping within themselves. The default time allotted is three times the value set for INTRFQCY. If the allotted time expires, the transaction is immediately marked for cancellation. The master terminal command (ZZHLD TIMER) and the terminal command (ZZRDY *TS) have no effect in this situation. If more time is required for programs set with unlimited timeouts, refer to the CHTBL transaction code for master terminals in the *IMS Operations Guide,* UP-12027.

If extra time is needed for all action timers, a global time extension is available and can be dynamically changed online. This extension adds a specific time value to the EXPIRTME value for all actions. It takes effect only at the start of an action and is only effective for those actions that have an EXPIRTME value greater than 0.

The extension time is recommended when the load on the system temporarily increases and the normal time allocated for actions to complete is not sufficient to cover periods of peak system load. When the load subsides, the extension time can be reset to 0. To change the extension time, refer to the CHTBL transaction code for master terminals in the *IMS Operations Guide,* UP-12027.

Notes:

1. *Since IMS uses the system clock to determine the current time, the console operator's SET CLOCK (SE CL) command should only be used to set the clock to the current time of day. An incorrect setting can cause undesirable action timer results.*

2. *The PAUSE command for IMS jobs should be used only for emergencies. This and other delays due to I/O devices require operator intervention, and they are counted toward the action program's elapsed wall clock time. Use the ZZHLD TIMER and ZZRDY TIMER commands in these situations to prevent premature action timeouts.*

3. *Using a supervisor generated with the DAYCHANGE=NO option with IMS is restricted. Failure to adhere to this restriction may cause premature action timer expiration and unpredictable results to offline file recovery.*

4. *No transaction will be canceled for a timeout violation until any I/O subtask executing in its behalf is completed. This includes accessing files, programs, and screen formats; waiting for output messages and the expiration of any SETIME WAIT period.*

Specifying Timer Interrupt Frequency Rate (INTRFQCY)

The timer interrupt frequency rate parameter specifies the timing interval (in seconds) to elapse before the IMS timer island code routine is given control. Once the timer island code gains control, the entry of any ZZHLD or ZZRDY command is determined and processed or each action is evaluated for exceeding its allotted time.

During shutdown, the timer interrupt interval is changed to 1 minute and all ongoing transactions are allowed to continue processing after the ZZSHD master terminal command is issued. The shutdown timeout value is 5 minutes. You can override this timeout value by specifying a 1-minute multiplier value on the ZZSHD terminal command.

INTRFQCY=
: Specifies the timer interrupt frequency rate in seconds (10 to 32,767). The default value is 30 seconds. For more accurate detection of action program expiration, this value should not exceed the lowest value specified for any one action and should be evenly divisible into all individual action timeout values.

Note: *The timer interrupt frequency rate is based on elapsed time, not processor time. To change the INTRFQCY value online, refer to the CHTBL transaction code for master terminals in the* IMS Operations Guide, *UP-12027.*

Specifying Elapsed Time between Input and Status (STATUS)

Online IMS transmits automatic status messages to notify a terminal operator of the status of the last input message. The STATUS keyword parameter determines how many seconds elapse after an input message has been received by IMS before the status of the message is sent and the interval at which the message is resent. Automatic status messages are not sent to a terminal executing an output-for-input queueing or continuous output action. In addition, status messages are not sent to the console when it is used for transaction processing, but ZZCNC can be keyed in at the console as soon as the first interval has ended. This keyword is valid only for a multithread IMS system.

STATUS=n
: Specifies the status message interval in seconds (minimum 15 seconds). The default value is 15 seconds.

Note: *The status message interval is based on elapsed time, not processor time.*

4.3.5. Describing Each User Data File - the FILE Section

You must code a FILE section to describe each of the ISAM, MIRAM, DAM, or SAM data files your action programs access - whether these are accessed directly or indirectly via the medium of the defined file. User-specified printer files are used only in multithread configurations and are coded in the FILE section following all other user defined files. You therefore provide a FILE section for:

- Each user data file named via a FILES keyword parameter in an ACTION section
- Each data file that contributes a primary or supplementary part to the defined records of a defined file or subfile specified by a DFILE keyword parameter
- Each user-specified printer file

Do not code a file section for DMS data base files.

A maximum of 240 files may be defined in multithread IMS, but only 123 in single-thread.

Format

```
FILE    filename

        FILETYPE= [DAMR  ]
                  [DMRAM ]
                  {ISAM  }
                  {SAMD  }
                  {SAMT  }
                  [TMRAM ]
                  [PRNT  ]

        [CAFILE= [NO ]]                          ]
                 {YES}                           ]
                                                 ]
        [CUPDATE= {NO }]                         ]
                  {YES}                          ]
                                                 ]
        [DELETEP= [NO ]]                         ]
                  {YES}                          ]
                                                 ]
        [LOCK= [IR]]                             ]
               {UP}                              ]
                                                 } Invalid when FILETYPE=PRNT
        [OUTPUT= [NO ]]                          ]
                 {YES}                           ]
                                                 ]
        [PKEY= [n]]                              ]
               {1}                               ]
                                                 ]
        [SEQVIEWS=(n[,blksize])]                 ]
                                                 ]
        [TRACE= [NO ]]                           ]
                {YES}                            ]

           .
           .
           .
        valid DTF or RIB keyword parameters for the
        specified file
           .
           .
           .
```

Example 1 (DTF mode)

```
FILE  CUSTFIL FILETYPE=ISAM LOCK=TR
              IOROUT=ADDRTR KEYLEN=5 RECFORM=FIXBLK RECSIZE=80 IOREG=(8)
              PCYLOFL=10 KEYLOC=1 WORK1=CUSTFIL
              IOAREA=CUSTFIL TYPEFLE=RANSEQ BLKSIZE=1022
FILE  DUEIN FILETYPE=ISAM LOCK=TR CAFILE=YES CUPDATE=YES
              IOROUT=ADDRTR KEYLEN=17 RECFORM=FIXBLK RECSIZE=22 IOREG=(8)
              PCYLOFL=10 KEYLOC=1 WORK1=DUEIN
              IOAREA1=DUEIN BLKSIZE=758 TYPEFLE=RANSEQ
FILE  SAMDISK FILETYPE=SAMD
              BLKSIZE=1024 IOAREA1=SAMDISK TYPEFLE=OUTPUT WORKA=YES
FILE  VENDOR  FILETYPE=DAMR LOCK=TR
              BLKSIZE=256 KEYLEN0= READID=YES WRITEID=YES
FILE  PRTINV  FILETYPE=PRNT
              BLKSIZE 120 PRINTOV PRTINV
```

Example 2 (CDM mode)

```
FILE  CUSTFIL FILETYPE=DMRAM INDAREA=CUSTFIL INDSIZE=256 KEYARG=CUSTFIL
              KEY1=(8,1)
FILE  EMPLOY FILETYPE=TMRAM
FILE  PRTNM  FILETYPE=PRNT
             BLKSIZE=120 PRINTOV REPORT
```

Naming the User Data File (filename)

The *filename* positional parameter is always required; it provides the logical name of a user data file to be accessed.

If the user data file contributes to a defined file or subfile, *filename* must be the same as the file name specified in the FD entry in the data division of your data definition. It is not the same as the *filename* response to the DFILE keyword parameter in the ACTION section.

On the other hand, if the user data file does not contribute to a defined file but is accessed directly by your action program, then *filename* must be the same as one of the file names specified via the FILES keyword parameter in an ACTION section and must be the logical name by which your action program refers to the file.

`filename`
Specifies a 1- to 7-character alphanumeric name whose first character is alphabetic or one of the following characters: $, #, @, or ?.

Note: *Each file name must be unique. Also, file names that are the same except for one additional letter at the end of one are not considered different files. For example, if you use INV and INVH for file names, you get an assembly error in the configuration.*

Identifying the Type of Data File (FILETYPE)

With the FILETYPE keyword parameter, which is always required, you must specify the type of user data file this section is describing.

`FILETYPE=DAMR`
Specifies a DAM file, organized by relative record address.

`FILETYPE=DMRAM`
Specifies a MIRAM file (or an IRAM file accessed via MIRAM). IMS supports multikey MIRAM files, but only the primary key may be used to update the file. (See Table 4-7.) Any key may be used to access the file.

`FILETYPE=ISAM`
Specifies an ISAM file.

`FILETYPE=SAMD`
Specifies a SAM file on disk.

`FILETYPE=SAMT`
Specifies a magnetic tape SAM file in a DTF system.

`FILETYPE=TMRAM`
Specifies a sequential magnetic tape file in a CDM system.

`FILETYPE=PRNT`
Specifies a user-specified printer file

Note: *Specify user-specified printer files after all other user-defined files.*

Specifying and Updating Common Storage Area Files (CAFILE and CUPDATE)

The common storage area file is a main storage resident, variable-block or fixed-block indexed file (ISAM or MIRAM) accessed in indexed random mode. It is loaded into main storage during the IMS start-up procedure and can be used to store vital information that is frequently accessed. During online processing, you may issue the GET, GETUP, and PUT function calls to a resident common area file; however, only the primary key is accessible. During shutdown, the disk file is updated using the data in main storage.

The use of common storage area files with defined record management or UNIQUE is not supported.

`CAFILE=NO`
Specifies that this is not a common area file.

`CAFILE=YES`
Specifies that this ISAM or MIRAM file is loaded into main storage common area at start-up.

To update common area files on disk, the CUPDATE parameter is used, together with the CAFILE parameter.

CUPDATE=NO
Specifies no disk updates for this common area file after each access to the resident file. When both the CAFILE=YES and CUPDATE=NO parameters are configured, the common storage area file is written back to the disk once, at shutdown.

CUPDATE=YES
Specifies that disk updates are made for the common area file after each access to the resident file. In single-thread IMS, specify CUPDATE=YES for common storage area files. Otherwise, data corruption may occur when recording statistical data at IMS shutdown.

Allowing Physical Deletion of MIRAM File Records (DELETP)

The DELETP parameter determines whether records in a MIRAM file can be physically deleted. This option is not available for IRAM files defined as MIRAM files.

DELETP=NO
Specifies that records cannot be physically deleted. (They can still be logically deleted.)

DELETP=YES
Specifies that records can be physically deleted. Notice that physically deleted records are still reflected in the data record count in the VTOC.

The default values are:

- YES, arbitrarily set for a multikey file
- NO, for a single key file

Selecting Type of File Record Lock (LOCK)

The LOCK parameter determines the type of record lock that IMS imposes on DAM, ISAM, and MIRAM files during updating transactions.

When you select lock for transaction (LOCK=TR) or omit this parameter:

- IMS locks records being updated until the end of the action.
- In single-thread IMS, when you specify RECLOCK=YES in the OPTIONS section, you can use the R and H lock rollback indicators in the program information block to hold record locks across actions.
- Updates are always audited (before-image recorded in the audit file) and are rolled back if the transaction abnormally terminates.

When you select lock for update (LOCK=UP):

- IMS locks records only for the duration of an update.
- If an update is incomplete at the end of the action, IMS holds the record lock into the next action (except in single-thread IMS with RECLOCK=NO). You can release the lock with an R, N, or O lock rollback indicator or UNLOCK function call.
- Updates are not audited and are not rolled back if the transaction terminates abnormally.

If dynamic main storage is used to output a formatted message in a multithread environment, ZZRSD does not resend the last formatted message to the terminal. Instead, ZZRSD sends a NO MSG IN QUEUE message.

The type of record lock does not affect tracing of updates (using the trace file) for offline recovery. All updated records are traced (if RECOVERY is specified in the OPTIONS section), whether or not locking is included.

LOCK=TR
: Specifies lock for transaction.

LOCK=UP
: Specifies lock for update. This specification makes the file nonauditable and its blocksize is not considered when audit file blocksize is determined.

Allowing Output Processing for a Dedicated Sequential File (OUTPUT)

A dedicated sequential MIRAM file may be used for input or output processing, but not both. With the OUTPUT keyword parameter, you tell the configuration whether this is an input or output file. Sequential MIRAM files are not audited or traced, regardless of the OUTPUT specification, and are therefore not recoverable by IMS file recovery routines. This parameter does not apply to any other file type.

OUTPUT=NO
: Specifies that this sequential MIRAM file is used for input processing.

OUTPUT=YES
: Specifies that this sequential MIRAM file is used for output processing.

Specifying the Primary Key of a Multikey File (PKEY)

The PKEY parameter specifies the primary key of a multikey MIRAM file.

PKEY=n
: Specifies which of the keys specified by KEY*n* parameters is the primary key of the file. The default value is 1.

If you specify a value for *n* greater than any specified in the data management keyword KEY*n*, IMS creates dummy keys to make up the difference. If, for example, you specify PKEY=5 with only KEY1 and KEY2, IMS creates keys 3, 4, and 5. You should reconfigure IMS with correct KEY*n* and PKEY specifications. Otherwise, you will get an error when you attempt to access the file by its primary key. Duplicates and changes are not allowed for the primary key but are allowed for alternate keys. See KEY*n* keyword specification in Table 4-6 or 4-7.

Specifying Multiple Sequential Views (SEQVIEWS)

In order to utilize the multiple sequential views facility, the following keyword has been provided:

SEQVIEWS=(n[,blksize])
: Specifies the number of additional views (*n*)of a CDM MIRAM file that are desired for undedicated sequential use. The value of *n* must be in the range from 1 to 35. You can also specify the BLKSIZE value (*blksize*) to be used for each of the secondary views. If this optional parameter is omitted, the default value is the BLKSIZE value of the primary view.

The SEQVIEWS keyword is recommended whenever the primary view of the file is declared for random mode (MODE=RAN) and is partially or entirely used for undedicated sequential processing. This keyword is allowed only when FILETYPE=DMRAM and MODE=RAN. It is ignored for common storage area files (CAFILE=YES) and single-thread IMS.

For optimum performance in accessing the file, it is recommended that you specify ACCESS=SRDO for the primary view of the file if the file is used only for reading, and ACCESS=EXCR for all other files. The secondary views are internally generated with ACCESS=SRDO and =SRD, respectively. This ensures optimum performance and file sharing compatibility.

Specifying ACCESS=EXC, =SADD, =UCP, or =SRDF is not allowed on the primary view of the file. Any of these specifications could cause file sharing incompatibility and data management errors, and could even destroy the integrity of the file.

Since generation of secondary file structures is controlled by the configurator, no secondary files need be declared by the user.

Note: *If the total number of SEQVIEWS specified for all user files exceeds the maximum allowed by the assembler, an A054 error could occur during an IMS configuration job step. Refer to the* System Messages Reference Manual, *UP-8076, for more details. (It is unlikely that the assembler limit will be exceeded in practical configurations.)*

Suppressing File Tracing (TRACE)

If file recovery is configured (through the RECOVERY parameter in the OPTIONS section), you can select which files are to be traced by coding the TRACE parameter. Tracing only essential files improves processing performance. TRACE=NO should be specified for files that are not to be updated. Tracing is performed for DAM, ISAM, and MIRAM files only.

TRACE=NO
: Specifies that this file is not to be traced and therefore its characteristics, e.g., BLKSIZE or RECSIZE, are not used in determining the blocksize of the trace file (TRCFILE).

TRACE=YES
: Specifies that this file is to be traced. This is the default option if file recovery is configured.

Providing DTF or RIB Keywords to the Configurator

The configurator generates a data management DTF or RIB macroinstruction for each of your data files. You provide the information the configurator needs to build the DTF or RIB by coding data management keyword parameters in the FILE section after the other FILE section parameters.

Tables 4-2 through 4-10 list the DTF and RIB keywords that you may include in the FILE section for each of the file access methods IMS supports. You use DTF keywords for an IMS system in DTF mode; RIB keywords for an IMS system in CDM mode. For most parameters, the configurator generates a default value or uses the default value supplied by data management. You should code these parameters only if you require a different value. Keywords shown in the tables without default values are optional.

For detailed information on the uses and meanings of the RIB keywords, see the *Consolidated Data Management Programming Guide;* UP-9978, and *Consolidated Data Management Macroinstructions Programming Guide,* UP-9979. The configurator accepts only the data management keywords and values listed in the tables and only in the forms shown in the table. The shortened alternative forms and additional values documented in the data management guides may not be used in IMS.

If the following MNOTE or PNOTE STATEMENT errors occur during assembly of a DTF, they can be ignored:

```
DTFIS  *,ERROR SPECIFICATION MISSING

DTFMI  P,WORK AND I/O REG NOT ALLOWED,
       I/O REG IGNORED
```

Table 4-2. DTF Configurator Keywords for an ISAM File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
ACCESS=[EXC / EXCR / SRD / SRDO]	X		
BLKSIZE=[n / 2048]	X		If omitted, IMS issues a warning diagnostic; this may be ignored.
INDAREA=filename			If included, filename must match the filename positional parameter in this FILE section.
INDSIZE=n			
IOAREA1=filename	X		If included, filename must match the filename positional parameter in this FILE section.
IOREG=(8)	X		
IOROUT=[ADD / ADDRTR / RETRVE]	X		Specify IOROUT=RETRVE for read-only files. If omitted, IMS issues a warning diagnostic; this may be ignored. Always specify IOROUT=ADDTR when FILETYPE=ISAM. Otherwise, any attempt to modify or add data to an ISAM file with IOROUT=RETRVE specified causes damage to the IMS control tables and terminates IMS.
KEYARG=filename	X		If included, filename must match the filename positional parameter in this FILE section.
KEYLEN=[n / 3]	X		
KEYLOC=n PCYLOFL=nn		X	If omitted, this keyword takes the data management effective value.
RECFORM=[FIXBLK / VARBLK]		X	
RECSIZE=[n / 1]	X		

continued

Table 4-2. DTF Configurator Keywords for an ISAM File (cont.)

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
TYPEFILE=[RANDOM / RANSEQ / SEQNTL]		X	Specify TYPEFLE=RANSEQ for UNIQUE and random processing.
UPDATE=[NO / YES]		X	Specify UPDATE=NO for read-only files. If the data management keyword IOROUT=RETRVE is specified, UPDATE=NO is generated by the configurator. Otherwise, data management generates the default, UPDATE=YES.
VERIFY=YES			If omitted, this keyword takes the data management effective value.
WORKS=NO			If omitted, the keyword takes the data management effective value.
WORK1=filename			If included, filename must match the filename positional parameter in this FILE section.

Table 4-3. DTF Configurator Keywords for a DAM File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
ACCESS=[EXEC / EXCR / SRD / SRDO]	X		
BLKSIZE=[n / 2048]	X		If omitted, IMS issues a warning diagnostic; this may be ignored.
IOAREA1=filename	X		If included, filename must match the filename positional parameter in this FILE section.
READID=YES	X		If omitted, IMS issues a warning diagnostic; this may be ignored.
RECFORM=[FIXUNB / VARUNB]		X	
RELATIVE=R	X		

continued

Table 4-3. DTF Configurator Keywords for a DAM File (cont.)

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
SEEKADR=filename	X		If included, filename must match the filename positional parameter in this FILE section.
TYPEFLE={INPUT / OUTPUT}		X	
VERIFY=YES			If omitted, this keyword takes the data management effective value.
WRITEID=YES	X		If omitted, IMS issues a warning diagnostic; this may be ignored.

Table 4-4. DTF Configurator Keywords for a SAM Disk File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
ACCESS={EXEC / EXCR / SRD / SRDO}	X		
BLKSIZE={n / 2048}	X		If omitted, IMS issues a warning diagnostic; this may be ignored.
EOFADDR=filename	X		Used for input files only.
IOAREA1=filename	X		If included, filename must match the filename positional parameter in this FILE section.
IOREG=(8)	X		Used for input files only.
OPTIONS=YES			Used for input files only.
RECFORM={FIXUNB / VARUNB}		X	
TYPEFLE={INPUT / OUTPUT}		X	
VERIFY=YES			If omitted, this keyword takes the data management effective value.
WORKA=YES	X		Used for output files only.

Table 4-5. DTF Configurator Keywords for a SAM Magnetic Tape File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
BLKSIZE=[n / 2048]	X		If omitted, IMS issues a warning diagnostic; this may be ignored.
CLRW=[NORWD / RWD]			If omitted, IMS issues a warning diagnostic; this may be ignored.
EOFADDR=filename	X		Used for input files only.
FILABL=[NO / STD]		X	
IOAREA1=filename	X		If included, filename must match the filename positional parameter in this FILE section.
IOREG=(8)	X		Used for input files only.
OPTION=YES			Used for input files only.
RECFORM=[FIXUNB / VARUNB]		X	
TPMARK=NO			If omitted, this keyword takes the data management effective value.
TYPEFLE=[INPUT / OUTPUT]		X	
WORKA=YES	X		Used for output files only.

Table 4-6. DTF Configurator Keywords for a MIRAM File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
ACCESS=[EXC / EXCR / SADD / SRD / SRDO]	X		SADD is not supported for multithread IMS. If specified for single-thread, IMS cannot be responsible for the integrity of the file. SADD should be specified only if recovery is not configured and LOCK=UP is specified in the FILE section.
AUTOIO=YES	X		
BLKSIZE=[n / 256]	X		
ERROR=filename	X		
EOFADDR=filename	X		
INDAREA=filename	X		Configurator generates default when PROC=KEY. Do not specify for unkeyed files.
INDSIZE=[n / 256]	X		Must be consistent with load time value. Configurator generates default when PROC=KEY. Do not specify for unkeyed files.
IOAREA1=filename	X		
IOREG=(8)	X		
KEYARG=filename	X		Configurator generates default when PROC=KEY. Do not specify for unkeyed files.
KEYn= size,loc[,[NDUP / DUP]] [,[NCHG / CHG]]	X		Must be consistent with load time values. Configurator generates defaults when PROC=KEY. For a multikey file, KEYn may be any of the keys. IMS accesses the file using the KEYn specification. Subparameters DUP and CHG are not allowed when KEYn is the primary key (4.3.5). The configurator defaults the primary key to NDUP and NCHG.
MODE=[RAN / SEQ]		X	Specify MODE=RAN for random and nondedicated sequential processing. If MODE=SEQ, action programs may issue only GET or PUT functions, depending on OUTPUT keyword specification (4.3.5).

continued

Table 4-6. DTF Configurator Keywords for a MIRAM File (cont.)

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
NWAIT=YES	X		
OPTION=NO			
PROC={KEY / UNK}	X		Specify PROC=KEY for keyed files.
RCB=NO			Specify RCB=NO for an IRAM file.
RECFORM={FIX / VAR}		X	
RECSIZE={n / 256}	X		
RETR=MOD	X		
SEEKADR=filename	X		
VERIFY=YES			If omitted, this keyword takes the data management effective value.
VMNT=ONE			Used for dedicated sequential processing only.
WORK=YES	X		

Table 4-7. DTF Configurator Keywords for a Printer File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
BLKSIZE=121	X		
CTLCHR=DI	X		
ERROR=filename	X		Address of routine to process special spool processing errors. Configurator specifies this address.
IOAREA1=filename	X		If included, the configurator overrides the user specifications with the internally generated I/O area name.
OPTION=YES		X	
PRINTOV={SKIP / filename}	X		Use filename to be notified of forms overflow condition. Otherwise, printer is advanced to home paper position without notification of forms overflow condition.
RECFORM=UNDEF	X		
RECSIZE=(8)	X		

Table 4-8. RIB Configurator Keywords for a MIRAM File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
ACCESS=[EXC / EXCR / SADD / SRD / SRDO / UCP]	X		SADD and UCP are not supported for multithread IMS. If specified for single-thread, IMS cannot be responsible for the integrity of the file. SADD should be specified only if recovery is not configured and LOCK=UP is specified in the FILE section. Whenever UCP is specified (for single-thread or multithread IMS), it is the user's responsibility for the integrity of the file if the file is to be shared between IMS and another job In addition, IMS is not responsible if any file discrepancies occur during online or offline recovery.
AUTOIO=YES	X		
BLKSIZE= n / 256	X		If omitted, IMS issues a warning diagnostic; this may be ignored.
INDAREA=filename	X		Configurator generates default when PROC=KEY. Do not specify for unkeyed files.
INDSIZE=[n / 256]	X		Must be consistent with load time value. Configurator generates default when PROC=KEY. Do not specify for unkeyed files.
IOAREA1=filename	X		
IOREG=(8)	X		
KEYARG=filename	X		Configurator generates default when PROC=KEY. Do not specify for unkeyed files.
KEYn= size,loc[,NDUP / DUP][,NCHG / CHG]	X		Must be consistent with load time values. Configurator generates defaults when PROC=KEY. For a multikey file, KEYn may be any of the keys. IMS accesses the file using the KEYn specification. Subparameters DUP and CHG are not allowed when KEYn is the primary key (4.3.5). The configurator defaults the primary key to NDUP and NCHG.

continued

Table 4-8. RIB Configurator Keywords for a MIRAM File (cont.)

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
MODE=[RAN / SEQ]		X	Specify MODE=RAN for random and nondedicated sequential processing. If MODE=SEQ, action programs may issue only GET or PUT functions, depending on OUTPUT keyword specification (4.3.5).
NWAIT=YES	X		
OPTION=[NO / YES]		X	
PROC=[KEY / UNK]		X	Specify PROC=KEY for keyed files.
RCB=[NO / YES]			Specify RCB=NO for an IRAM file.
RECFORM=[FIXUNB / VARUNB]			
RECSIZE=[n / 256]	X		
RETR=MOD	X		
SEEKADR=filename	X		
VERIFY=[NO / YES]		X	
VMNT=ONE			Used for dedicated sequential processing only.
WORK=YES	X		

Table 4-9. RIB Configurator Keywords for a Sequential MIRAM Magnetic Tape File

Keyword	Configurator Generates Default	Data Management Generates Default
AUTOIO=YES	X	
BLKSIZE=[n / 256]	X	
CLRW=[NO / YES / UNLOAD]		X
FILABL=[NO / STD]		X
IOAREA1=filename	X	
OPTION=[NO / YES]		X
RECFORM=[FIXUNB / VARUNB]		
TPMARK=[NO / YES]		X
TYPEFLE=[INPUT / OUTPUT]	X	

Table 4-10. RIB Configurator Keywords for a Printer File

Keyword	Configurator Generates Default	Data Management Generates Default	Remarks
BLKSIZE=121	X		
CONTROL=CTL	X		
IOAREA1=filename	X		If included, the configurator overrides the user specifications with internally generated I/O area name.
OPTION=YES		X	
PRINTOV={SKIP / REPORT}	X		Use report to be notified of forms overflow condition. Otherwise, paper is advanced to home paper position without notification of forms overflow condition.
RECFORM=UNDEF	X		
RECSIZE=(8)	X		
WORK=NO	X		

4.3.6. Describing Terminals - the TERMINAL Section

The TERMINAL section defines the master terminal and other attributes of terminals in the IMS network. All terminals listed must be defined in the ICAM network definition, but you need not include a TERMINAL section for every terminal.

Format

```
TERMINAL terminal-id
         [IMSREADY= {NO }]
                    {YES}
         [MASTER= {NO }]
                  {YES}
         [STATUSMG= {YES}]
                    {NO }
         [UNATTEND= {NO }]
                    {YES}
         [UNSOL= {ACTION}]
                 {TRANS }
         [URGENT= {NO }]
                  {YES}
```

Example

```
TERMINAL T001  MASTER=YES
TERMINAL T003  URGENT=YES
TERMINAL T004  IMSREADY=NO UNSOL=ACTION
TERMINAL T005  UNATTEND=YES
TERMINAL T006  STATUSMG=NO
```

Note: *When you start up IMS with a global ICAM network, IMS reserves terminal slots for all terminals that you identify in TERMINAL sections with a terminal-id, whether or not you include any optional parameters. (See TERMS parameter, 4.3.1.)*

IMS recognizes terminals defined as screen bypass terminals in the ICAM generation and suppresses unsolicited output messages and IMS status messages to a screen bypass terminal. For further information on defining terminals, refer to TERM macro and FEATURES keyword subparameter specification SBT in the ICAM Operations Guide, *UP-9745, and* ICAM Programming Reference Manual, *UP-9749.*

Identifying the Terminal (terminal-id)

`terminal-id`
Is the 1- to 4-character alphanumeric identification you assigned to this terminal in your ICAM network definition and must always be specified. Its first character must be alphabetic or one of the characters: $, #, @, ?. (See Section 2 for ICAM considerations.)

Suppressing the IMS READY Message (IMSREADY)

The IMSREADY keyword parameter may be used to suppress output of the IMS READY message to selected terminals at IMS start-up time. This parameter does not apply to dynamic terminals in a global network.

`IMSREADY=NO`
Specifies that the IMS READY message is not to be sent to this terminal.

`IMSREADY=YES`
Indicates that this terminal is to receive the IMS READY message and is the default assumption.

You should not specify IMSREADY=NO for local workstations that are:

- Used in a dedicated ICAM environment
- Statically assigned in a global ICAM environment

Defining a Master Terminal (MASTER)

The MASTER keyword parameter identifies a terminal in the ICAM network as a master terminal. The system console can serve as the master terminal, in which case you omit this parameter or specify OPCOM=MASTER in the OPTIONS section.

You designate one terminal as a master terminal for a single-thread configuration, but you can specify more than one for a multithread configuration. If you designate only one master terminal and it becomes inoperative, use the ZZMCH terminal command to designate another terminal as the master terminal during the online IMS session. If any master terminal becomes inoperative, you may use the ZZMCH command from any non-master terminal to make it the master terminal.

`MASTER=NO`
Specifies this is not a master terminal and is the default assumption.

`MASTER=YES`
Specifies this is a master terminal. If MASTER=YES is not specified for any regular terminal, the system console is the master terminal.

For a complete description of the commands and transactions you can enter at the master terminal, refer to the *IMS Operations Guide,* UP-12027.

Suppressing the IMS Timer-Generated Status Messages (STATUSMG)

Use the STATUSMG keyword parameter to indicate if the terminal will or will not receive IMS timer-generated status messages. The STATUSMG specification entered in the NETWORK section is the default.

`STATUSMG=YES`
Specifies that this terminal will receive timer status messages. YES overrides the NETWORK section specification for this terminal only.

`STATUSMG=NO`
Specifies suppression of timer status messages to this terminal. NO overrides the NETWORK section specification for this terminal only.

You may also use the ZZMSG and ZZNMG terminal commands to allow or suppress timer status messages during an online IMS session. For a complete description of terminal commands and transactions, refer to the *IMS Operations Guide,* UP-12027.

Defining an Unattended Terminal (UNATTEND)

The UNATTEND keyword parameter indicates if this terminal is attended or unattended.

`UNATTEND=NO`
Specifies that this is an attended terminal, where normal input and output processing occurs. NO is the default for this parameter.

`UNATTEND=YES`
Specifies that this is an unattended terminal. Therefore, unsolicited (CALL SEND function) output messages and SWTCH transaction output messages are sent to this terminal without the unsolicited output indicator (MSGW). Operator intervention is not needed at a terminal designated by UNATTEND=YES.

Specifying Message Notification after Each Action (UNSOL)

Normally, switched unsolicited output messages are queued for a terminal until the transaction in progress is completed; the operator is then notified by means of the unsolicited output indicator. You can use the UNSOL keyword parameter to specify that the operator be notified of such messages at the end of each action instead. The default is the NETWORK section UNSOL specification.

UNSOL=ACTION
: Specifies that the terminal operator is to be notified of switched unsolicited output messages at the end of each action. ACTION overrides the specification in the NETWORK section for this terminal only.

UNSOL=TRANS
: Specifies that the terminal operator is to be notified only at completion of a transaction. TRANS overrides the specification in the NETWORK section for this terminal only.

If UNSOL=ACTION is specified, the terminal operator (once notified of queued messages via the unsolicited output indicator) can accept the queued output or transmit an input message. If the terminal operator transmits another input message, the input is processed and the switched messages remain queued.

Specifying Terminal Message Priority (URGENT)

The URGENT keyword parameter determines whether all input messages from this terminal are designated as urgent in priority. This parameter applies to multithread IMS only.

URGENT=NO
: Specifies that messages from this terminal have normal priority. This is the default assumption.

URGENT=YES
: Specifies that messages from this terminal are urgent.

4.3.7. Specifying Transaction Codes - the TRANSACT Section

The TRANSACT section supplies information about your transactions to the configurator. This section must be specified for each transaction code.

Format

```
TRANSACT  trans-code
          [ACTION=program-name]
          [LOCAP=locap-name]
          [UNDEF= {NO }]
                  {YES}
          [URGENT= {NO }]
                   {YES}
```

Example

```
TRANSACT CHG ACTION=CHANGE
TRANSACT DUMP URGENT=YES
TRANSACT SHOW
TRANSACT TOTAL
TRANSACT F#01 ACTION=UPDATE
TRANSACT ERRMSG UNDEF=YES
```

The DUMP, SHOW, TOTAL and ERRMSG transactions are started by action programs having the same names. All transaction codes except DUMP have routine priority. Transaction F#01 is initiated by function key F1. The ERRMSG transaction handles undefined transaction codes.

Naming the Transaction (trans-code)

The *trans-code* positional parameter assigns a transaction code to the transaction and must be the first parameter specified in the TRANSACT section. This transaction code is keyed in by the terminal operator to initiate the transaction. A transaction may also be initiated at the terminal by a function key (F1 through F33), in which case the *trans-code* parameter specifies the function key. See Appendix A for reserved (IMS internal) transaction codes.

`trans-code`
: Names the transaction and consists of one to eight alphanumeric characters (multithread IMS) or one to five alphanumeric characters (single-thread). The length of the transaction code in multithread IMS is limited to the TRANLEN parameter specification in the GENERAL section. If the transaction code is longer than the maximum allowed, IMS truncates the code.

The first character of the transaction code must be alphabetic or one of the characters: $, #, @, ?. Duplicate transaction codes are not allowed. If the transaction is to be initiated by a function key, the trans-code parameter has the format:

F#nn

where:

nn
Is the number of the function key (01 through 33).

If KATAKANA=YES is specified in the NETWORK section, the transaction code can consist of Katakana characters.

Assigning the First Action Program for This Transaction (ACTION)

The ACTION keyword parameter identifies the first action program to be scheduled for this transaction. If this is a remote transaction, omit the ACTION parameter.

ACTION=program-name
Specifies a 1- to 6-alphanumeric-character action program name; the first character must be alphabetic. This name must be the same as the *program-name* specified in the corresponding ACTION section.

If omitted for a local transaction, the first action program to be scheduled is assumed to have the same name as the transaction; that is, ACTION=*trans-code* is assumed.

Specifying a Remote System where the Transaction Is Processed (LOCAP)

The LOCAP keyword parameter specifies that this is a remote transaction and identifies the remote system where the transaction is processed. You use this parameter for directory routing of transactions in a distributed data processing environment. It is available only in multithread IMS.

LOCAP=locap-name
Specifies the local application that processes the transaction at a remote system. The *locap-name* must be a valid name specified in a CUP keyword parameter of the NETWORK section for the remote IMS system configuration. It cannot be a name specified in the CUP keyword parameter of this configuration. If LOCAP is specified, you should omit the ACTION parameter.

Indicating a Default Transaction Code (UNDEF)

The UNDEF keyword parameter specifies whether this transaction is to be used as the default transaction for this configuration. When a terminal operator enters a transaction code that has not been defined to IMS, IMS calls upon this transaction to process it, rather then sending an "invalid transaction code" message to the originating terminal.

UNDEF=NO
Specifies this transaction is not the default for undefined transaction codes.

UNDEF=YES
Specifies this transaction is to be used by default to handle undefined transaction codes.

IMS allows you to specify no more than one transaction as undefined. If more than one TRANSACT section specifies UNDEF=YES, only the first is accepted and all others are ignored.

Specifying Transaction Priority (URGENT)

The URGENT keyword parameter determines whether all transactions with this transaction code are to be given urgent priority. This parameter applies to multithread IMS only.

URGENT=NO
Specifies that transactions with this code are not necessarily urgent. This is the default assumption.

URGENT=YES
Specifies that all transactions with this code are urgent.

4.3.8. Describing the Options for Each Action - the ACTION Section

In one or more ACTION sections, you describe each of the actions in your application to the configurator. This section must be coded once for each action.

Format

```
ACTION    program-name
          [ALLRNT= {NO  }]
                   {YES }
          [BYPASS= {n}]
                   {5}
          [CDASIZE=n]
          [DDRECORD=record-name]
          [DFILE=filename]
          [EDIT= {c         }]
                 {FCCDICE   }
                 {NONE      }
                 {tablename }
          [EXPIRTME= {n  } (multithread only)]
                     {NO }
          [FCCEDIT= {NO }]
                    {YES}
           FILES=filename-1[,...,filename-n]
          [INSIZE= {n   }]
                   {STAN}
          [MAXSIZE=n]
          [MAXUSERS= {n}]
                     {3}
          [OUTSIZE= {n   }]
                    {STAN}
          [SFSINCAP= {NO }]
                     {YES}
          [SHRDSIZE=n]
          [TRANSLAT= {NO }]
                     {YES}
          [WORKSIZE=n]
```

Example

```
ACTION.UPDATE MAXSIZE=1000 OUTSIZE=204
 EDIT=% FILES=MAINFIL,PAYROLL,ACCTREC,ACCTPAY
 FILES=STATS,EMPLOYS EXPIRTME=120
ACTION DUMP OUTSIZE=84 MAXSIZE=250 FILES=ALL
ACTION LIST MAXSIZE=2000 OUTSIZE=1156 FILES=MAINFIL
```

Three actions are described: UPDATE, DUMP, and LIST. You can split multiple responses to a parameter between cards by repeating the keyword, as shown on the second and third lines of coding.

Naming the First Action Program for This Action (program-name)

`program-name`
Is the name of the first action program to be executed for this action and must be specified. The name consists of one to six alphanumeric characters; the first character must be alphabetic or one of the characters: $, #, @, ?. The same *program-name* must not be specified for more than one action. This name must be the same as a *program-name* in a PROGRAM section.

Specifying Whether All Action Programs Are Reentrant (ALLRNT)

The ALLRNT keyword parameter specifies whether all programs for this action are reentrant; it is available only for a multithread IMS system.

`ALLRNT=NO`
Specifies that not all programs are reentrant.

`ALLRNT=YES`
Specifies that all programs are reentrant.

Note: *IMS can use main storage more effectively if all action programs for an action are defined as reentrant.*

Limiting Waiting Time for an Action (BYPASS)

The BYPASS keyword parameter specifies the maximum number of times this action may be bypassed for initiation before being unconditionally initiated. It is available only for a multithread IMS system.

`BYPASS=n`
Specifies the bypass limit. If BYPASS=0 is specified, the action is never bypassed.

If omitted, a bypass limit of 5 is assumed.

Specifying Length of the Continuity Data Area (CDASIZE)

The CDASIZE keyword parameter specifies the length of the continuity data area for this action. If the action accesses a DAM file and uses a CDA as a record area, the length must be large enough to accommodate the DAM file record (which is a 256-byte multiple).

If this action accesses a DMS data base and you include the USING ACTIVATION RECORD clause in the schema section of your action program, you must allocate a continuity data area or work area large enough to contain the DMCA or subschema records (or both). Your program determines where these records are placed. Refer to the *IMS/DMS Interface Programming Guide,* UP-8748.

`CDASIZE=n`
: Specifies the length, in bytes, of the continuity data area. This value must not exceed the specification for the MAXCONT parameter in the GENERAL section, nor can it exceed the track length for the type of disk being used.

If omitted and you specify UNIQUE=RES, TRAN, or YES in the OPTIONS section, CDASIZE=2048 is assumed. If both parameters are omitted, no continuity data area is allocated.

Naming the Data Definition Record (DDRECORD)

The DDRECORD keyword parameter names the data definition record associated with the defined file for this action. When the action is scheduled, IMS places this name in the DATA-DEF-REC-NAME field of the program information block (PIB), unless a record name has been left in that field by the preceding action.

The record name specified with this keyword is the same as the *defined-file-name* in the defined file definition -- it is not the *defined-record-name* specified with the DEFINED RECORD statement.

`DDRECORD=record-name`
: Specifies the data definition record and consists of one to seven alphanumeric characters. The first character must be alphabetic.

If KATAKANA=YES is specified in the NETWORK section, the record name may consist of Katakana characters.

If omitted and the DFILE parameter is specified, the name of the data definition record is assumed to be the same as the defined file name. If both parameters are omitted, no data definition record is assumed to exist for this action, unless a record name has been placed (or left) in the DATA-DEF-REC field of the PIB by the preceding action.

Naming the Defined File for This Action (DFILE)

The DFILE keyword parameter identifies the defined file or subfile to be accessed by this action. It must be specified if the DDRECORD parameter is specified. When the action is scheduled, IMS places this name in the DEFINED-FILE-NAME field of the PIB, unless a file name has been left in that field by the preceding action.

The file name specified with this keyword is the same as the *file-name* used in your action program's function calls to defined record management, and is the *defined-file-name* supplied via the DEFINED FILE statement in your input to the data definition processor.

The response to the DFILE keyword parameter must be different from the name of any user data file named in a FILE section or via a FILES keyword parameter in any ACTION section in this configuration.

`DFILE=filename`
: Specifies the defined file and consists of one to seven alphanumeric characters; the first character must be alphabetic.

If KATAKANA=YES is specified in the NETWORK section, the file name may consist of Katakana characters.

If omitted, no defined file is accessible by this action, unless a file name has been placed (or left) in the DEFINED-FILE-NAME field of the PIB by the preceding action.

Specifying Editing Requirements for Input Messages (EDIT)

The EDIT keyword parameter specifies the editing requirements for input messages to this action.

When a terminal operator enters an input message, ICAM creates DICE sequences from the input terminal cursor movements. If you specify EDIT=*tablename*, EDIT=FCCDICE, or EDIT=*c*, or if you omit this parameter, IMS removes these DICE sequences from the message text before delivering the message to the action program. DICE sequences for carriage return are replaced by blanks (X'40'), and other DICE sequences are simply deleted and not replaced.

If you specify EDIT=NONE, IMS does not remove DICE sequences from input messages, and your action program must be prepared to process them.

If you specify EDIT=*tablename*, the capability for expanded input editing is also included in your online IMS load module. In addition to removing DICE sequences, IMS edits input messages according to the edit table generated by the ZH#EDT utility for this action and written to the NAMEREC file. Generating edit tables is described in the IMS action programming manuals.

EDIT=c
Specifies a character to be used as the field separator in input messages to this action. IMS removes DICE sequences and the specified character. If the specified character is a blank (or if the keyword is omitted), IMS retains a single blank but suppresses repeated blanks. If repeated blanks are to remain in the message, the terminal operator should enclose the message with apostrophes.

EDIT=FCCDICE
Specifies that IMS is to remove FCC and DICE sequences from input messages, replace DICE carriage return sequences with blanks (one blank for each carriage return), and retain repeated blanks. This specification is used mainly with RPG II action programs.

EDIT=NONE
Specifies that no editing is to be performed for input messages; DICE and FCC sequences are not removed from the input message text. Do not use this specification for RPG II action programs.

EDIT=tablename
Identifies the 2- to 7-alphanumeric-character name of the edit table to be used with this action.

If this keyword is omitted, a blank character is assumed as the field separator in input messages. IMS retains a single blank but suppresses repeated blanks. IMS removes DICE sequences from the input message text but replaces carriage return DICE sequences with blanks (one blank for each carriage return).

Note: *Leading blanks are removed from any input message entered at the console. Take this into consideration for any action being processed in behalf of the console when you specify EDIT=NONE.*

Specifying Individual Action Program Elapsed Time (EXPIRTME)

In a multithread system, the EXPIRTME parameter specifies the timeout value that IMS uses to detect a loop or hang condition for this action program. If this parameter is omitted, IMS uses the global action program timeout value from ACTION in the TIMEOUTS section.

EXPIRTME= {n / NO}
Specifies the individual action program timeout value (*n*). The value of *n* must be in the range from 0 to 32,767. If no timeouts are desired, specify 0 or NO.

In order to dynamically change the EXPIRTME value online, refer to the CHTBL transaction code for master terminals in the *IMS Operations Guide,* UP-12027.

Note: *All internal IMS action programs are generated with an unlimited timeout specification. However, you can also dynamically change their timeout values online.*

Specifying FCC Editing Requirements for Input Messages (FCCEDIT)

Input messages from a UTS or workstation terminal often contain field control character (FCC) sequences. Your action programs may or may not be prepared to process these sequences. The FCCEDIT parameter determines whether or not IMS removes these FCC sequences from the text of a message before delivering the message to an action program.

When you specify EDIT=*c*, EDIT=FCCDICE, or EDIT=*tablename* or omit the EDIT parameter, IMS removes DICE sequences from the text of input messages. The FCCEDIT parameter lets you choose whether or not you also want FCC sequences removed from the message text. If you omit FCCEDIT, IMS assumes FCCEDIT=YES.

When you specify EDIT=NONE, IMS does no input message editing at all. FCC sequences are not removed from input messages regardless of your FCCEDIT specification.

`FCCEDIT=NO`
Specifies that IMS is not to remove FCC sequences from the input message to this action.

`FCCEDIT=YES`
Specifies that IMS is to remove FCC sequences from the input message to this action.

Naming the Data Files Accessed Directly by This Action (FILES)

With the FILES keyword parameter, you name each individual ISAM, MIRAM, DAM, or SAM file to be accessed directly by this action. Do not include the name of any user data file that contributes to a defined file for this action.

The FILES keyword parameter allows multiple responses; you may specify several file names but must separate them with commas.

Each user data file specified with the FILES keyword parameter must also be described to the configurator with its own FILE section.

`FILES=file-name-1[,...,filename-n]`
Specifies the user data files to be accessed by this action directly. File names must be one to seven alphanumeric characters; the first character must be alphabetic.

If the FILES keyword parameter is omitted, no user data files may be accessed directly by this action, which is limited, therefore, to access of a defined file.

Note: *Do not assign user-specified printer files to the action.*

Specifying Input Message Area Length (INSIZE)

The INSIZE keyword parameter specifies the input message area (IMA) length for messages to be handled by this action. If the actual message length received exceeds the specified length, the IMA is dynamically expanded to accommodate the message. However, the IMA is not dynamically expanded to handle the additional length required by screen format services (SFS). If the IMA is too small for the screen format services, your transaction aborts and error code SFS14 is displayed. This transaction abort message indicates the action program name and the required INSIZE value. When you reconfigure IMS, enter the new INSIZE value plus 16_{10} in the ACTION section.

When the SFS14 transaction abort message is displayed at the console, IMS simulates the following change table (CHTBL) transaction:

```
CHTBL ACT (action name) IMAL=(decimal value + 16₁₀)
```

The new IMAL/INSIZE remains in effect until the current IMS session is shut down. The new IMAL/INSIZE also remains in effect in all subsequent IMS sessions as long as you use // PARAM RESTART to restart a session. If you use // PARAM START (the default) to start a session, the SFS14 transaction abort message is displayed and the IMS CHTBL process begins again. Using the change table each time IMS is brought up will avoid the SFS14 transaction abort message. This can be done until a reconfiguration of IMS has been completed.

If the INSIZE keyword is not specified, an IMA is allocated to accommodate the length of each actual message read.

If an action uses expanded input editing (EDIT=*tablename*), you must include the INSIZE parameter and specify a length at least as large as the length of the message defined in the edit table. To determine the length needed, add the values specified for all the LEN parameters in the edit table generation. (See the IMS action programming manuals.) You can also specify INSIZE=STAN if the standard message length is as large as or larger than the length the edit table requires.

The INSIZE specification is meaningful only when it exceeds the actual message length.

`INSIZE=n`
: Indicates the length, in bytes, of the input message area. The value *n* must include the input message length and accommodate any DICE sequences, FCCs, or control characters in the message text. The specified length must not be greater than the lengths specified by CDASIZE in this section or INBUFSIZE (in the GENERAL section).

`INSIZE=STAN`
: Allocates an area large enough to accommodate the standard message length. The standard message length is established by the CHRS/LIN and LNS/MSG parameters in the GENERAL section.

Notes:

1. *If you configure the console for transaction processing, the maximum length of an input message is 60 characters, not including the input message header.*

2. *The INSIZE specification is meaningful only when it exceeds the actual message length.*

Specifying Length of Largest Action Program (MAXSIZE)

This parameter specifies the length of the largest nonresident action program that may be scheduled to satisfy this action.

MAXSIZE=n
: Specifies the length, in decimal bytes, of the largest nonresident action program for this action.

If MAXSIZE=0 is specified, IMS assumes that all action programs for this action are resident programs. If the MAXSIZE keyword is omitted, IMS assumes that the first program to receive control for this action is the largest nonresident program, or that there is only one program for this action.

You must include the MAXSIZE keyword parameter if you intend to use the action with immediate internal or delayed internal succession under single-thread IMS. For multithread, MAXSIZE need be specified only if the action uses immediate internal succession. Failure to adhere to this guideline causes the online error message MAXSIZE TOO SMALL to be issued when an attempt is made to execute this action.

If the fastloader is used, add another 4096 bytes to the MAXSIZE value to account for RLD data.

Specifying Number of Concurrent Users (MAXUSERS)

The MAXUSERS keyword parameter specifies the maximum number of concurrent users allowed for this action; it is applicable only to multithread IMS configurations.

MAXUSERS=n
: Specifies the maximum number of concurrent users. The default value is 3.

This value puts an upper limit on the number of users that can be active at one time for this action. Thus, MAXUSERS should be set to a value such that, when this limit is reached, it is pointless to schedule any more users for this action.

Specifying Output Message Area Length (OUTSIZE)

The OUTSIZE keyword parameter specifies the output message area (OMA) length for this action.

OUTSIZE=n
: Specifies the maximum length, in bytes, of the OMA for this action. The maximum length, *n*, consists of the length of your output message text (including all DICE sequences, FCCs, and control characters it contains). When you use immediate or delayed internal succession, the length must not be greater than the value specified in the INSIZE parameter.

OUTSIZE=STAN
: Allocates an area large enough to accommodate the standard message length. The standard message length is based upon the CHRS/LIN and LNS/MSG parameters you specify in the GENERAL section.

  When you specify OUTSIZE=STAN and the action is entered from a terminal with a larger screen size than the OUTSIZE configured, IMS allocates an output message area large enough to accommodate the actual screen size.

  If you use the OUTSIZE=*n* format, the way you calculate the *n* value depends on whether or not you call on screen format services in this action and whether you use the dynamic main storage feature.

  If this action calls on screen format services and the screen format is built in the output message area, the output message area must be large enough to contain the screen format buffer constructed by the screen format coordinator. This buffer contains the variable output data, display constants, and device control characters. Refer to the *Screen Format Services Technical Overview*, UP-9977.

If you use the dynamic main storage feature, you do not have to use the formula for screen format services because IMS builds the screen format in a dynamic buffer outside the program area. (You indicate dynamic main storage in your action programs that use screen formats.)

If you do not use dynamic main storage, you must specify a message area large enough to contain the screen format buffer. To determine the OUTSIZE needed for a screen format, use the following formula:

```
format-length * format-width + (9 * control-char) + 200
```

where:

format-length
: Is the number of lines in the format, determined by cursor placement at format generation time.

format-width
Is the number of characters per line of the display terminal at which the screen format was generated.

control-char
Is the number of control character sequences. To determine this number, add the number of variable fields, the number of display constants, the number of new lines, and the number of special characters (such as ! and .) used in variable fields.

When the action program requests a screen format and the OUTSIZE is not large enough to contain the format buffer, IMS returns an error code in the program information block. IMS also places the actual length needed for the format buffer into the text-length field of the OMA header.

If this action does not use a screen format, use the following formula, rounding the result up to the next double-word boundary:

```
(CHRS/LIN+4) * (LNS/MSG) + 24 + (5 * (number of FCCs))
```

where:

CHRS/LIN
Is the value specified for message line length.

4
Represents the length of a DICE sequence for each message line.

LNS/MSG
Is the value specified for number of lines per message.

24
Represents the length of the 16-byte OMA header plus an allowance of eight bytes for two additional DICE sequences (for example, to initiate, clear the screen, or reposition the cursor).

5 * (number of FCCs)
This value can be ignored if your network does not include UTS or workstation terminals. The number of field control characters (FCCs) should equal the greatest number of FCCs that will appear in the output message.

At run time, your action program may specify a shorter output message length by moving an appropriate value into the text length field of the OMA header before transmitting each message. This value must include four bytes for the 2-byte text length field itself and the 2-byte auxiliary-id field that precedes your message text; it must also allow for the DICE sequences and any other control characters embedded in your text. The resulting output message length may never exceed the OMA area length specified by the OUTSIZE parameter.

If the OUTSIZE keyword parameter is omitted, no area is allocated for output messages.

Note: *If you configure the console for transaction processing, the maximum length of an output message is 120 characters, not including header or length field.*

Specifying SFS Input Capabilities after Action Termination (SFSINCAP)

The SFSINCAP keyword parameter specifies that after normal action termination (TERMINATION-INDICATOR set to N), screen format services input capabilities are to either remain in effect or be disabled for the terminal that controlled this SFS action. (Refer to the *IMS COBOL/Assembler Action Programs Programming Guide,* UP-9207, for a description of TERMINATION-INDICATOR values.) The SFSINCAP keyword has no effect for a non-SFS action.

`SFSINCAP=NO`
SFS input capabilities are to be disabled after normal action terminations.

`SFSINCAP=YES`
SFS input capabilities are to remain in effect after normal action terminations.

Specifying Length of COBOL Shared Code Data Area (SHRDSIZE)

This keyword parameter specifies the length of the shared code work area for a COBOL action program; it is usually not specified in a single-thread IMS system.

If a COBOL action program uses a volatile work area, IMS saves and restores it in the activation record for the program. If your COBOL action program is compiled under the PARAM OUT=(M) option of the extended COBOL compiler or the IMSCOD=YES option of the 1974 American Standard COBOL or COBOL 85 compiler, the printer displays the length of the shared code volatile data area, in decimal bytes, immediately prior to the COBOL COMPILATION COMPLETE message. The length of the volatile work area, if any, increases the length of the activation record and must be considered in calculating the latter. Refer to Appendix B.

`SHRDSIZE=n`
Specifies the length, in bytes, of a shared code, volatile data area in a COBOL action program.

If omitted, no shared code volatile data work area is allocated.

Specifying Lowercase-to-Uppercase Translation (TRANSLAT)

The TRANSLAT keyword parameter specifies whether lowercase-to-uppercase translation is to be performed on input messages for this action. It affects only the lowercase alphabetic characters *a, b, c,...,z,* which are translated to uppercase characters. All other characters in the input message are unchanged.

TRANSLAT=NO
Specifies no translation and is the default assumption.

TRANSLAT=YES
Specifies lowercase-to-uppercase translation.

Note: *If KATAKANA=YES is specified in the NETWORK section and input is from a Katakana terminal, no lowercase-to-uppercase translation takes place, even if TRANSLATE=YES is specified.*

Specifying Length of the Work Area (WORKSIZE)

The WORKSIZE keyword parameter specifies the length of the work area for this action. This length must be a 256-byte multiple if the action accesses a DAM file and uses the work area as a record area, or if the action accesses data files stored on a sectorized disk.

If this action accesses a DMS data base and you include the USING ACTIVATION RECORD clause in the schema section of your action program or data definition, you must allocate a continuity data area or work area large enough to contain the DMCA or subschema records (or both). Your program determines where these records are placed. Refer to the *IMS/DMS Interface Programming Guide,* UP-8748.

If this action contains 1974 American Standard COBOL or COBOL 85 programs compiled with IMSCOD=REN, add to the work area length the reentrancy control lengths reported in the compilation summary listings. WORKSIZE must be large enough to accommodate the maximum program data area requirements plus the sum of all concurrently active object program reentrancy control areas. This WORKSIZE rule applies for all 1974 American Standard COBOL or COBOL 85 programs compiled with IMSCOD=REN, regardless of how the program is configured to IMS as TYPE=SER or TYPE=SHR. Failure to specify a large enough WORKSIZE can result in destruction of your program data area and abnormal termination of the action program.

WORKSIZE=n
Specifies the length, in bytes, of the work area and must not exceed 32,767. In a single-thread system, *n* must be a multiple of eight bytes.

If omitted, no work area is allocated.

4.3.9. Describing Each Action Program - the PROGRAM Section

The PROGRAM section describes the user action programs and subprograms. A PROGRAM section must be included for each action program and subprogram.

Format

```
PROGRAM    program-name
           [ERET= {NO }]
                  {YES}
           [RESIDE= {NO }]
                    {YES}
           [SUBPROG= {NO }]
                     {YES}
           [TYPE= [RNT]
                  {SER}
                  [SHR]
```

Example

```
PROGRAM BASIC RESIDE=YES TYPE=RNT ERET=YES
PROGRAM COMPUT TYPE=SER SUBPROG=YES
PROGRAM EDIT TYPE=RNT RESIDE=YES
```

Naming the Action Program (program-name)

program-name
: Is the name of the program being described and consists of one to six alphanumeric characters. The first character must be alphabetic or one of the characters: $, #, @, ?.

Specifying Return of Control (ERET)

The ERET keyword parameter specifies whether control should be returned to this program if one of its requests to IMS is unsuccessful.

ERET=NO
: Indicates that control is not to be returned and the transaction is aborted if an error occurs.

ERET=YES
: Specifies that control is be returned to the user action program if an error occurs and status code 3 or 4 (invalid request or I/O error) is set.

If omitted or NO is specified, an unsuccessful request results in abnormal termination of the transaction. The ERET keyword has no effect for status codes 0, 1, or 2 in the program information block (PIB).

Specifying Action Program Residence (RESIDE)

The RESIDE keyword parameter specifies whether this program is resident (that is, permanently loaded into the main storage pool area at initialization time) or transient (loaded only as required by transaction processing).

RESIDE=NO
: Indicates that the program is transient and is the default assumption.

RESIDE=YES
: Indicates that the program is resident.

Identifying a Subprogram (SUBPROG)

The SUBPROG keyword parameter specifies whether the program being described is a subprogram. Since only resident subprograms are permitted, the configurator automatically includes the RESIDE=YES parameter when SUBPROG=YES is specified.

SUBPROG=NO
: Indicates that this is not a subprogram and is the default assumption.

SUBPROG=YES
: Specifies that this is a resident subprogram.

Specifying Action Program Reentrance/Reusability (TYPE)

The TYPE keyword parameter specifies whether the program is reentrant, serially reusable, or shared code. A COBOL action program may be reentrant or shared code. COBOL subprograms may be configured as serially reusable or reentrant; they cannot be shared.

TYPE=RNT
: Indicates that the program is reentrant.

TYPE=SER
: Indicates that the program is serially reusable and is the default assumption.

TYPE=SHR
: Specifies that the program is sharable. If specified for a single-thread configuration, the program is treated as serially reusable.

4.3.10. Replacing UNIQUE Language Elements - the LANGUAGE Section

The LANGUAGE section is a repeatable section that allows you to create a UNIQUE language set, called a lexicon. The new lexicon partially or completely replaces the standard UNIQUE lexicon. Thus, you can modify as many or as few UNIQUE language elements as you wish to suit your own application. For example, you may wish to create a French language set by replacing all the standard UNIQUE language elements with their French equivalents, or you may wish to replace certain elements in the standard UNIQUE lexicon with abbreviated forms.

The standard UNIQUE lexicon is listed in Appendix D. When you do not modify a standard language element by including it in a new lexicon, your terminal operators must enter that element as it appears in the appendix.

If you specify UNIQUE=YES or UNIQUE=RES in the OPTIONS section and do not specify a LANGUAGE section, the configurator generates a default lexicon record (see Appendix D) and the OPEN transaction code. If you specify a LANGUAGE section, the default lexicon specifications are replaced by the LANGUAGE section specifications and the default transaction code of OPEN is replaced with a transaction code generated from the lexicon name.

If you specify UNIQUE=NO in the OPTIONS section, the configurator does not generate a default lexicon record. Any LANGUAGE section input is ignored.

Format

```
LANGUAGE lexicon-name
    language-element=replacement
                .
                .
                .
    language-element-n=replacement-n
```

Example

```
LANGUAGE OUVREZ OPEN=OUVREZ MORE=PLUS NEXT=PROCHAIN
CHANGE=CHANGER SHOW=MONTRER
```

Five standard UNIQUE language elements are assigned new values: OPEN, MORE, NEXT, CHANGE, and SHOW.

Specifying the Lexicon Name (lexicon-name)

This positional parameter specifies the name of the lexicon and the transaction code to be created.

`lexicon-name`
Specifies the name of the UNIQUE lexicon record and the transaction code. For single-thread IMS, it consists of one to five alphanumeric characters, the first of which must be alphabetic or one of the characters: $, #, @, ?. For multithread, it consists of one to seven (see TRANLEN keyword specification in GENERAL section) alphanumeric characters, the first of which must be alphabetic.

Specifying UNIQUE Language Element to Be Replaced (language-element=replacement)

The language element parameter specifies the UNIQUE language element to be replaced and its replacement. The replacement is the term that a terminal operator enters in place of the standard language element or that UNIQUE displays in place of the standard language element.

`language-element=replacement`
language-element is the word to be replaced and must be one of those listed in Appendix D. The following special characters cannot be replaced: ;, :, ', $, (, +, -, *, /, **, and). The replacement must not exceed the maximum length listed beside the original. It must not contain embedded spaces. Punctuation symbols, special characters, and arithmetic operators must not be used as replacement values. If KATAKANA=YES is specified in the NETWORK section, the replacement may consist of Katakana characters.

Notes:

1. *Don't use any of the internal IMS transaction codes and terminal commands (A.2) as language element replacements for the UNIQUE commands OPEN, CLOSE, DISPLAY, LIST, MORE, DETAIL, NEXT, CHANGE, ADD, DELETE, ASSIGN, SHOW, CANCEL, and OK (Appendix D). In addition, don't use the same language element replacement for more than one of the above UNIQUE language element commands. Unpredictable results may occur if these rules are not followed.*

2. *The configurator parameters applicable to the LANGUAGE section do not apply to messages generated by IMS-supplied action programs (UNIQUE, ZSTAT, DLOAD, DLMSG).*

4.3.11. Identifying Remote Systems where Transactions Can Be Processed - the LOCAP Section

The LOCAP section is required when distributed data processing is used. It identifies a transaction system (IMS) on a remote node where DDP transactions can be processed or that can route DDP transactions to this IMS system for processing. Transactions can be sent to a remote node system by directory routing, program routing, or operator routing. LOCAP is a repeatable section.

Format

```
LOCAP    locap-name    RCHAR=routing-character
```

Example

```
LOCAP PRG1 RCHAR=;
```

Note: *To use distributed data processing with IMS, you must have the IMS transaction facility in your OS/3 software, define a global ICAM network definition that supports distributed data processing (2.3.2), configure multithread IMS, and include at least one LOCAP section in the configuration.*

Identifying the Remote System (locap-name)

The *locap-name* parameter specifies the name of a remote system to which transactions can be routed or which routes transactions to this IMS system. This name must match the label of a LOCAP macro in your ICAM network definition.

`locap-name`
Specifies the name of a remote IMS system. The name consists of one to four alphanumeric characters, the first of which must be alphabetic or one of the characters: $, #, @, ?. The *locap-name* specified cannot be the same as the name specified in the CUP keyword parameter of the NETWORK section.

Specifying a Routing Character (RCHAR)

The RCHAR keyword parameter specifies a routing character that is used in operator routing to identify the remote system. The terminal operator prefixes a transaction code with this character and a period. Each remote system must have its own routing character, which must be different from the priority character (if any) specified in the UCHAR parameter of the GENERAL section.

`RCHAR=routing-character`
Specifies a routing character to identify the remote system in operator-routed transactions. The routing character can be any alphabetic, numeric, or special character except the blank. If KATAKANA=YES is specified in the NETWORK section, you can designate a Katakana character. Do not use the same routing character for more than one remote system.

When a terminal operator wants to route a transaction to the remote system identified by the *locap-name* parameter, he enters the routing character, followed by a period. Then he enters the transaction code and message. For example, to route a UNIQUE transaction to a remote system for which the routing character ; is specified, the terminal operator enters:

```
;.OPEN INVFILE
```

You can also define routing characters in PARAM statements at IMS start-up, whether or not you include the RCHAR parameter in the LOCAP section. A routing character defined at start-up overrides the RCHAR specification for that IMS session only.

4.3.12. Defining the Record Management Interface - the DRCRDMGT Section

The configurator automatically includes defined record management capability in your system when you include UNIQUE in your configuration or when you include the DFILE parameter in any ACTION section. You use the DRCRDMGT section when you want to make defined record management resident in a single-thread IMS system. In a multithread system, defined record management (if included) is always resident. You also use the DRCRDMGT section when you want to use defined record management file updating functions with your own action programs. When you configure UNIQUE and FUPDATE=YES in the OPTIONS section, the configurator automatically includes defined record management file updating functions.

Format

```
DRCRDMGT   [RESIDE= {NO }]
           [        {YES}]

           [UPDATE= {NO }]
           [        {YES}]
```

Example

```
DRCRDMGT UPDATE=YES  RESIDE=YES
```

Specifying Residence for Defined Record Management (RESIDE)

In a multithread IMS system, defined record management (if included) is always resident. In a single-thread system, you can make defined record management resident by coding the RESIDE keyword parameter. A resident defined record management improves IMS performance but increases main storage requirements by 2154 bytes.

RESIDE=NO
: Specifies that the transient defined record management request processor module is to be included in this single-thread IMS configuration.

RESIDE=YES
: Specifies that the resident defined record management request processor module is to be included in this single-thread IMS configuration.

Specifying Defined Record Management Updating Functions (UPDATE)

The UPDATE keyword parameter determines whether or not updating functions are to be used with defined record management. If you use UNIQUE, the updating functions are the UNIQUE commands ADD, DELETE, and CHANGE. If you interface directly with defined record management in your own action programs, the updating functions are GETUP, PUT, INSERT, and DELETE.

UPDATE=NO
: Indicates that updating capability is not to be included and is the default assumption unless UNIQUE is configured.

UPDATE=YES
: Specifies that updating capability is to be included. Updating capability is automatically included if UNIQUE is configured and FUPDATE=YES is specified in the OPTIONS section.

4.4. Configurator Error Processing

The configurator generates errors at configuration time. It flags each error detected in the printed configuration listing on the line after the statement in error. The format for error listing is:

```
***ERROR error-code message-text
```

Table E-2 lists the error codes, the diagnostic messages associated with each, the message description, and the configurator action.

The configurator sets the UPSI byte to hex 80 (*hexadecimal*) for fatal errors and hex 40 for warning errors. Fatal errors are I/O errors (documented in the *System Messages Reference Manual*, UP-8076, and initialization errors (error codes 0101, 0102, 0103, and 0104 in Table E-2). All other errors in the table are warning errors.

The IMSCONF jproc checks the UPSI byte and skips the assembly and online module linkage steps when the setting is hex 80. It does not check for a setting of hex 40. You can test for the hex 40 setting by including a // SKIP statement in the job control stream at configuration. Refer to the *Job Control Programming Guide,* UP-9986, or *Programming Reference Manual*, UP-9984, for a description of the SKIP job control statement, and the *System Service Programs (SSP) Operating Guide,* UP-8841, or *Programming Reference Manual*, UP-8842, for more information about the UPSI byte.

If the configurator cannot load the CCA module because inadequate main storage was allocated or because of an error in the CCA linkage, the message

```
CCA LOADING ERROR - ERROR CODE nn
```

is displayed on the system console. The configurator continues executing, but it assumes a network containing only one terminal. The UPSI byte is set to hex 80 and the assembly and linkage steps are omitted.

See the *System Messages Reference Manual,* UP-8076, for an explanation of the loading error code.

In the configurator listing, the message

```
ERRORS ENCOUNTERED - 000n
```

appears at the end of the CCA linkage. If *n* is a number other than 0, this is evidence of a good linkage. If *n* is 0, this means that a bad CCA linkage is generated as a result of the following:

- The CCA object module does not exist in the library file specified by either the LIBO or CCA keywords in the IMSCONF jproc.
- The resulting linkage editor listing shows that the load module size is zero.

When this occurs, retry the entire configuration after correcting the error.

Section 5
Initiating and Terminating IMS

5.1. General

To initiate an online IMS session, you must first load ICAM and then execute the IMS load module created by the configuration process described in Section 4. If you are using a global network, you load ICAM, then execute the global user service task (GUST) before executing IMS. If you want to perform batch transaction processing in offline mode, an active communications network is not required and you need not load ICAM before executing IMS. (You must still generate an ICAM network, however, in order to configure your IMS load module.) Batch transaction processing is described in Section 6.

The online IMS session is terminated by a command from the master terminal, ZZSHD, which causes an orderly shutdown, or ZZHLT, used for emergency termination.

5.2. Establishing the Communications Environment

Before executing online IMS, you must load the ICAM symbiont that interfaces with your online IMS load module. If you use a global network, you establish the communications environment in two steps:

1. Load the ICAM symbiont.

2. Execute the global user service task (GUST).

5.2.1. Loading ICAM

ICAM is loaded from the system console by keying in the operator command:

```
Cn or Mn
```

where:

`Cn or Mn`
: Is the name specified on the MCPNAME parameter in the COMMCT phase of SYSGEN.

You can also load ICAM from a job control stream with the statement:

```
// CC [Cn]
      [Mn]
```

This allows you to load ICAM from a workstation instead of asking the console operator to do it for you, and it also allows you to load ICAM as part of your IMS execution job or as part of a job that executes the global user service task.

You can use the CC job control statement in two ways:

1. In a job control stream that loads only ICAM:

```
// JOB jobname
// CC [Cn]
      [Mn]
/&
```

2. In a job control stream that executes IMS, or if you are using a global network, in a job control stream that executes the global user service task. The CC job control statement immediately follows the JOB statement.

When ICAM has been successfully loaded, the message ICAM READY is displayed on the system console.

Note: *When you load ICAM from a workstation, ICAM messages are still directed to the operator console.*

5.2.2. Executing the Global User Service Task

If you are using a global network, you must execute the global user service task program, ML$$GI, after loading ICAM. The example job control stream in Figure 5-1 executes ML$$GI.

```
// JOB GUST,,3000
// DVC 20  // LFD PRNTR
// DVC 50  // VOL DISK01  // LBL TCIDTF  // LFD TCIDTF,,INIT
// DVC 50  // VOL DISK01  // LBL ICAM.QUEUE1  // LFD DQFILE1,,INIT
// DVC 50  // VOL DISK01  // LBL ICAM.QUEUE2  // LFD DQFILE2,,INIT
// OPTION SYSDUMP
// EXEC ML$$GI
/&
// FIN
```

Figure 5-1. Sample Job Control Stream for Executing GUST

On the JOB statement, GUST requires a minimum main storage allocation of 3000 hexadecimal bytes. The device assignments for disk buffer and disk queueing files must be included in this job stream instead of the IMS execution job stream.

When GUST is executed, it sends messages to the system console, which the operator must respond to. These messages are documented in the *ICAM Operations Guide,* UP-9745. When the network is loaded, the following message is displayed at the system console:

```
MC#430 GUST ACTIVE FOR CCA network-name
```

When this message appears, you can execute IMS.

5.3. Executing the IMS Load Module

IMS is executed as a user job, with a standard job control stream submitted from the card reader or workstation. The name of the load module you execute is the name specified on the LOADM parameter of the IMSCONF jproc at configuration. The default name for a single-thread DTF configuration is ZS#IMS; for a multithread DTF configuration it is ZQ#IMS; for a single-thread CDM configuration, ZS$IMS; for a multithread CDM configuration, ZQ$IMS.

When the online IMS module is successfully loaded, the message IMS READY is sent to each terminal that is in a ready state (except terminals for which IMSREADY=NO was specified in the TERMINAL section).

After IMS is loaded, dynamic terminals in a global network can establish and terminate sessions by using the $$SON and $$SOFF commands. See the *IMS Operations Guide,* UP-12027. Dynamic terminals do not receive the IMS READY message. Instead, the message SESSION PATH OPEN is sent at session establishment.

Note: *You should avoid using the SET DATE console command to set the system IPL date backwards during an IMS run for these reasons:*

- *OFFLINE recovery assumes that the TRACE-DATE-TIME in the TRACE file is always in ascending sequence. The TRCFILE=CLOSE option of the ZC#TRC routine closes the file by recording the EOD pointer as soon as it detects the DATE-TIME of the current trace record lower than that of the preceding record.*

- *If the TRCFILE=CLOSE option is not used, the existence of records in the TRACE file that are not in ascending sequence may have adverse effects on normal operation of OFFLINE recovery.*

5.3.1. Parameter Statements in the Control Stream

PARAM statements in the IMS execution job control stream allow you to specify:

- Whether IMS internal files and tables are to be initialized or information retained from a previous session
- Rollback of data files at system start-up
- Closing or extending of a disk trace file from a previous session
- Routing characters that identify transactions to be processed by a particular remote system in a DDP environment
- Batch transaction processing in online or offline mode

The format of the PARAM statements for IMS execution is:

```
[// PARAM {START  }]
         {RESTART}

[// PARAM STARTUP= {CLEAN}]
                  {COLD }
                  {WARM }

[// PARAM TRCFILE= {CLOSE }]
                  {EXTEND}

[// PARAM WPROTECT= {YES}]
                   {NO }
                   {N  }

[// PARAM LOCAP=(locap-name,routing-character)]

[// PARAM DDPBUF=n]

[// PARAM DDPSESS=n]

[// PARAM DEBUG= {NO }]
                {YES}

[// PARAM RESMEM=(n,k,m)]

[// PARAM {BA   } = {NO     }]
         {BATCH}   {OFFLINE}
                   {ONLINE }

[// PARAM IN=module-name/filename]
```

PARAM statements may be specified in any order except that BATCH and IN (when used) must be the last parameter statements in the job control stream. If PARAM IN is used, it must follow PARAM BATCH.

Specifying Type of Start-up (START/RESTART, STARTUP)

The START or RESTART parameter is used for multithread IMS only. It tells IMS whether to initialize control tables in the NAMEREC file or to maintain statistical and terminal information from a previous session; also, whether to reinitialize the LDPFILE. If you omit this parameter, control tables and the LDPFILE are initialized.

`// PARAM START`
Specifies initialization or reinitialization of control tables in the NAMEREC file and of the LDPFILE (if configured) at start-up of a multithread IMS system. This parameter must be used for initial start-up, after a reconfiguration, or if the version of any action program has changed since the beginning of the last PARAM START session. Changes made with the CHTBL request should be made permanent with an IMS reconfiguration process.

`// PARAM RESTART`
Specifies that statistical, terminal, program, and action information in the NAMEREC file from the previous session is to be carried forward into the new session. Statistical data generated by the ZSTAT transaction reflects multiple sessions. The LDPFILE from the previous session (if used) is reused. Use PARAM RESTART only if the previous IMS session had a normal shutdown.

The STARTUP parameter is optional for both single-thread and multithread IMS users. It specifies whether the audit file is to be initialized and whether transactions active at termination of the previous session are to be rolled back.

`// PARAM STARTUP=CLEAN`
Specifies initialization of the AUDCONF file (or AUDFILE and CONDATA files) at start-up. This option is used for the initial start-up after a successful shutdown.

`// PARAM STARTUP=COLD`
Specifies that the AUDCONF file (or AUDFILE and CONDATA files) is not to be initialized at start-up, but no file rollback is to be performed. This option is generally used after offline recovery. If you have configured a terminal output message file (TOMFILE), IMS appends the DLMSG transaction code to the IMS READY message sent to each terminal. Terminal operators can display the last valid message output to their terminals by pressing the TRANSMIT key. You must specify TOMFILE=YES in the configurator OPTIONS section if you want to use the cold start feature in single-thread IMS. It is optional in multithread IMS.

`// PARAM STARTUP=WARM`
Specifies that all files against which transactions were active at termination or system failure are to be rolled back to a consistent state. IMS appends the DLMSG transaction code to the IMS READY message sent to each terminal; terminal operators can display the last valid message output to their terminals by pressing the TRANSMIT key. You must specify TOMFILE=YES in the configurator OPTIONS section to use the warm start feature in single-thread IMS. It is optional in multithread IMS.

If no PARAM STARTUP statement is included, IMS initializes the audit file. However, if it is specified incorrectly, IMS terminates with the message:

```
INVALID PARAM CARD
```

Extending a Tape or Disk Trace File (TRCFILE)

If you want to continue the same trace file from a previous session instead of starting a new one, you must specify this in your job control stream at start-up. The way you specify extension of a trace file depends on whether that file is on magnetic tape, disk, or diskette.

You specify extension of a magnetic tape trace file with the EXTEND parameter of the LFD job control statement:

```
// LFD TRCFILE,,EXTEND
```

If your tape trace file was left open at the end of the previous session because of system failure, you must use the tape copy routine, ZC#TCP, to copy it onto another tape volume and close it properly. You then assign it in the job control stream in the same manner.

For a disk or diskette trace file, you use the PARAM TRCFILE statement to specify continuation of the same file. If the file was properly closed at the termination of the previous session, you specify PARAM TRCFILE=EXTEND. If your trace file was left open, you specify PARAM TRCFILE=CLOSE; IMS first closes the file and then extends it.

```
// PARAM TRCFILE=CLOSE
```
Specifies that IMS is to close the disk/diskette trace file before initiating online operations. All records in the trace file are validated; then the trace file is extended, starting after the last valid record. This option may be used with warm restart after a system crash if you do not use the offline recovery utility, ZC#TRC, to recover your data files.

```
// PARAM TRCFILE=EXTEND
```
Specifies that IMS is to extend the disk/diskette trace file used in the previous session, starting after the last valid record. This option is used after a normal shutdown or after offline recovery of your data files.

If the TRCFILE parameter is specified when no trace file has been assigned in the job control stream, the following message is issued:

```
OPEN ERROR ON TRCFILE
```

If a disk/diskette trace file is assigned in the job control stream and the TRCFILE parameter is omitted, IMS assumes a new trace file is being used and starts writing from the first record.

Disabling Write Protection (WPROTECT)

The WPROTECT parameter can be used only in multithread IMS systems. This parameter specifies if write protection is enabled for user action programs.

```
// PARAM WPROTECT=YES
```

Specifies that write protection is enabled during user action program execution. Write protection should always be used until a new action program is debugged.

Note: *IMS SURs must have write protect enabled (WPROTECT=YES). This will clearly demonstrate if an action program is inadvertently modifying IMS code, control structures, or main storage.*

```
// PARAM WPROTECT= {NO }
                  {N  }
```

Specifies that write protection is disabled during user action program execution. Disabling this option increases the throughput of user action programs because it eliminates the switching of storage protection keys each time an action program is given control or gives up control to IMS. The improvement depends on the number of IMS function requests within the action program and the current percentage rate of CPU time utilized by the entire system.

Use this option only in a real production environment and after your program is stable.

Specifying a Routing Character (LOCAP)

To indicate that terminal input messages prefixed by a particular routing character are to be processed by a particular remote system, you can insert a LOCAP parameter statement in the job control stream for executing IMS. The remote system must be defined in a configurator LOCAP section. A routing character specified in a PARAM statement overrides any routing character specified in the LOCAP configurator section for that remote system for this IMS session only. The PARAM LOCAP statement may be repeated for each remote system.

```
// PARAM LOCAP=(locap-name,routing-character)
```

Specifies that all terminal messages prefixed by the routing character and a period are to be sent to the remote system identified by *locap-name* for processing. A routing character can be any alphabetic, numeric, or special character except the blank. If KATAKANA=YES is specified in the NETWORK section, you can specify a Katakana character as the routing character.

When a terminal operator wants to route a transaction to the remote system identified by *locap-name*, he enters the routing character, followed by a period. Then he enters the transaction code and message. For example, to route a UNIQUE transaction to a remote system for which the routing character A is specified, the terminal operator enters:

```
A.OPEN SALES
```

Specifying Distributed Data Processing Message Length (DDPBUF)

You can override the value specified in the DDPBUF parameter of the GENERAL section of the configuration by including a PARAM DDPBUF statement. This changes the length of the DDP message buffer for the current IMS session only.

```
// PARAM DDPBUF=n
```
Specifies the DDP message length (in multiples of 256) for this IMS session. The maximum value allowed is 4352.

Specifying Number of DDP Sessions (DDPSESS)

You can decrease the value specified in the DDPSESS parameter of the GENERAL section of the configuration by including a PARAM DDPSESS statement. This changes the number of distributed data processing sessions for the current IMS session only.

```
// PARAM DDPSESS=n
```
Specifies the maximum number of DDP sessions that can be active at one time. Specify a value between 2 and 25, but not more than the value specified in the DDPSESS parameter at configuration.

Monitoring Online IMS (DEBUG)

If you have a problem with online IMS, you should include the DEBUG parameter when you start up IMS. In this way, additional IMS and terminal status data is recorded during processing and is included when you take a dump. This dump should be submitted whenever you report a problem. The DEBUG feature is available only in multithread IMS.

```
// PARAM DEBUG=NO
```
Specifies that additional IMS and terminal status information is not to be recorded. This is the default value.

```
// PARAM DEBUG=YES
```
Specifies that additional IMS and terminal status information is to be recorded.

Overriding Transaction Buffer Pool Specifications (RESMEM)

If RESMEM is defined at IMS configuration, you may use the RESMEM parameter statement in the job control stream to override the specifications set at system configuration. For a description of the RESMEM keyword parameter, refer to 4.3.3.

The PARAM RESMEM statement format is:

```
// PARAM RESMEM=(n,k,m)
```

where:

n
: Describes the number of transaction buffers you wish to reserve during start-up. The transaction buffers reserved during start-up form a transaction buffer pool. RESMEM=*n* is a numeric value from 0 to 32767. (Each transaction buffer is 4096 bytes.)

k
: Is the maximum number of transaction buffers that can be acquired from IMS main memory after the transaction buffer pool is depleted. The value for k may not exceed 32767. The default is zero (0).

m
: Is the maximum number of transaction buffers that a transaction may acquire. The value of *m* may be 1 to 16. The default is 4.

You may use this parameter statement to change any combination of parameters specified at system configuration. Note that since these parameters are positional, if you do not specify a value for *n* or *k*, you must code the commas. If you only specify *n*, coding the parentheses is optional. If you do not specify *m*, you may omit the comma after the *k*. Only the values in the // PARAM RESMEM statement override the values specified with the RESMEM keyword parameter. Note that you may specify 0 for *n* and *k*, but if you specify *m*, it must have a value of at least 1 and no more than 16. If you specify an invalid value for *m*, an error message is displayed.

Processing Batch Transactions (BATCH)

To process batch transactions in either offline or online mode, you insert a BATCH parameter statement in the job control stream for executing IMS. If your batch input is in the form of source modules filed on disk, you also insert a PARAM IN statement for each such module. The BATCH and IN parameters follow any other PARAM statements in the job control stream, and the PARAM BATCH statement precedes any PARAM IN statements. Complete details about batch transaction processing, including a sample job control stream, are in Section 6.

```
// PARAM {BA   } =NO
         {BATCH}
```

Specifies that batch transactions will not be processed in this execution of IMS.

```
// PARAM {BA   } =OFFLINE
         {BATCH}
```

Specifies that batch transactions are to be processed in offline mode, that is, without an active terminal network.

```
// PARAM {BA   } =ONLINE
         {BATCH}
```

Specifies that batch transactions are to be processed in online mode. Online mode requires an active communications network. Batch processing in online mode is controlled by the ZZBTH master terminal command (6.6.1).

Specifying an Input Module That Is on a Disk File (IN)

If you have created a file that contains your input source module for batch processing (rather than embedding the card images in the control stream), specify the file and module names in a PARAM IN statement. PARAM IN statements must follow the PARAM BATCH statement. You may include any number of PARAM IN statements (or none), and they may appear in any order.

The PARAM IN statement format is:

```
// PARAM IN=module-name/filename
```

where:

`module-name`
Specifies the name of the input source module.

`file-name`
Specifies the name of the file that contains your module.

5.3.2. Job Control Stream for IMS Execution

A standard job control stream for executing the online IMS load module is illustrated in Figure 5-2. For detailed information on the job control statements in the illustration, refer to the *Job Control Programming Guide,* UP-9986, or *Job Control Programming Reference Manual,* UP-9984.

```
1.   // JOB jobname,,min[,,tasks]
2.   // OPTION [DUMP   ]
                {JOBDUMP}
                [SYSDUMP]
3.   // DVC 20 // LFD PRNTR
4.   // DVC logical-unit-no  // VOL vol-ser-no  // LBL file-id
     // LFD data-file-name
           .
           .
           .
5.   // DVC logical-unit-no  // VOL vol-ser-no  // LBL file-id  // LFD NAMEREC
6.  [// DVC logical-unit-no  // VOL vol-ser-no  // LBL file-id  // LFD AUDFILE] (1)
7.  [// DVC logical-unit-no  // VOL vol-ser-no  // LBL file-id  // LFD CONDATA] (1)
8.   // DVC logical-unit-no  // VOL vol-ser-no  // LBL file-id // LFD STATFIL,,INIT
9.   // DVC logical-unit-no  // VOL vol-ser-no  (2)
10. [// EXT ST,C,,BLK,(block-length,no-of-blocks)]
11.  // LBL file-id  // LFD disk-buffer-file-name
     [
12.   // DVC logical-unit-no  // VOL vol-ser-no
13.   // EXT ST,C,,CYL,no-of-cylinders
14.   // LBL file-id  // LFD disk-queueing-file-name
     ]
           .
           .
           .
     [
15.   // DVC logical-unit-no  // VOL vol-ser-no-1[,...vol-ser-no-n]
16.  [// EXT ST,C,,CYL,no-of-cylinders]
17.   // LBL file-id[,file-serial-no]  // LFD TRCFILE[,,EXTEND]
     ]
     [
18.   // DVC logical-unit-no  // VOL vol-ser-no  // LBL file-id
      // LFD load-lib-name
     ]
19.  [// DVC logical-unit-no  // VOL vol-ser-no  // LBL file-id  // LFD LOAD]
     [
20.   // DVC logical-unit-no  // VOL vol-ser-no
      // EXT ST,C,,CYL,no-of-cylinders
      // LBL file-id  // LFD LDPFILE
     ]
     [
21.   // DVC logical-unit-no // VOL vol-ser-no // LBL file-id // LFD TC01FMTF
22.  [// DVC logical-unit-no // VOL vol-ser-no // LBL file-id // LFD TC02FMTF]
     ]
23.   // EXEC load-module-name[,load-lib-name],1
24.  [// PARAM parameter-statement]
           .
           .
           .
25.   /&
26.   // FIN
```

Notes:

(1) The AUDFILE and CONDATA files in a multithread IMS system are replaced by the AUDCONF file in a single-thread system.

(2) In a global network, assign ICAM files in GUST job control stream.

Figure 5-2. Job Control Stream for Executing Online IMS

In the JOB job control statement (line 1), the *min* parameter specifies the size of the job region, calculated according to the guidelines in Appendix B. The *tasks* parameter is required for multithread IMS only. It specifies the maximum number of tasks that will be active at any one time; at least 4 must be specified. For information on the optimum number of tasks to assign, see Appendix C.

An OPTION statement (line 2) must be included to enable IMS to generate a snap dump when an action program or UNIQUE abnormally terminates. Refer to the job control manuals previously cited for information on dump options. Assignment of a printer (line 4) is also required for this purpose.

On line 4 (and succeeding lines) are the device assignment sets for data files. For a multithread version, a user-specified printer data file may appear as follows:

```
// DVC 21 // LFD PRFILE
```

The NAMEREC file (line 5), and an ICAM file for message buffering (lines 9-11) are always required. The LFD name for the disk buffering file must match the label on the DISCFILE macro in the ICAM network definition (Section 2). ICAM files for disk queueing of output messages (lines 12-14) are also required if specified in the network definition. Their LFD names must also match the labels on the DISCFILE macros specifying these files. If you use a global network, you must assign the ICAM files in the GUST job stream instead of here. For information on allocating ICAM files, refer to the *ICAM Operations Guide,* UP-9745.

The AUDFILE and CONDATA files assigned on lines 6 and 7 are for a multithread system and would be replaced by the AUDCONF file in a single-thread system. The AUDFILE is always required, and the CONDATA file is required for UNIQUE, if you use a continuity data area in any of your action programs or if you have configured a TOMFILE. You do not assign the TOMFILE in the job control stream because it is a partition of the multithread CONDATA file or the single-thread AUDCONF file.

The STATFIL assigned in line 8 is required if you plan to use the ZSTAT transaction (with output to STATFIL) or if you specify STATS=YES in the OPTIONS configurator section to write statistics to STATFIL at shutdown.

The trace file assigned on lines 15 through 17 is for offline recovery and may be disk, diskette, or tape. The multiple *vol-ser-no* parameters on the VOL statement and the *file-serial-no* parameter on the LBL statement are needed only for a multivolume tape trace file. The EXTEND parameter of the LFD statement also applies only to tape and specifies that the same trace file from a previous session is to be continued. The EXT statement applies only to a disk or diskette trace file. To estimate the amount of space to allocate to this file, multiply the block size (the size of your largest applicable data record or output message area plus 104 bytes for the prefix area) by the number of updates you expect to perform during the period the trace file is to be used. If you have configured both before- and after-images, double this figure. You must also allow for rollback and termination records and (if you have configured TOMFILE tracing) an output message at each rollback point and at transaction termination.

A diskette trace file can be on multiple volumes, but all volumes must be online at once. You must have a device assignment set for each volume.

The EXT job control statement for the ICAM files (lines 10 and 13), trace file (line 16), and LDPFILE (line 20) are included only if these files were not allocated previously and must be removed from the job control stream after the first execution of online IMS.

The device assignment for a load library file (line 18) is required only if the IMS load module is not in the system load library (YLOD) on SYSRES. The *load-lib-name* parameter of the EXEC statement (line 23) must match the LFD-*name* for this file and is the library specified by the LIBL parameter of the IMSCONF jproc at configuration. The *load-module-name* on the EXEC statement is the name specified on the LOADM parameter of the IMSCONF jproc.

The LOAD and LDPFILE files assigned on lines 19 and 20 are for using the fast load feature. Action programs are stored in LOAD, and IMS copies them into the LDPFILE the first time they are called by a transaction. The LDPFILE should be 20 percent larger than the combined size of all action programs and should not be on the SYSRES volume.

You must include a device assignment set (lines 21 and 22) for each screen format file. You can have one or two files (TC01FMTF and TC02FMTF). If either of the format files cannot be opened, screen format services will be inhibited.

The third positional parameter of the EXEC statement must specify a priority of 1. Note that IMS must always be the highest priority job running. If IMS is loaded into the system while another job with a priority of 1 is executing, you should use the SWITCH operator command to lower the priority of the other job.

Typical job control streams for executing multithread and single-thread IMS load modules in a dedicated ICAM environment are illustrated in the examples that follow. A job control stream for batch transaction processing is illustrated in Section 6.

Example 1 - Executing Multithread IMS at Initial Start-up

```
// JOB IMS90,,20000,,4
// DVC 20  // LFD PRNTR
// OPTION SYSDUMP
// OPR 'IMS EXECUTION'
// DVC 51  // VOL DS1475  // LBL ZL#VEND  // LFD VENDOR
// DVC 51  // VOL DS1475  // LBL ZL#PROD  // LFD PRODFIL
// DVC 51  // VOL DS1475  // LBL ZL#CUST  // LFD CUSTFIL
// DVC 51  // VOL DS1475  // LBL ZL#DUIN  // LFD DUEIN
// DVC 51  // VOL DS1475  // LBL ZL#DUOT  // LFD DUEOUT
// DVC 52  // VOL DS1480  // LBL NAMEREC  // LFD NAMEREC
// DVC 52  // VOL DS1480  // LBL AUDFILE  // LFD AUDFILE
// DVC 52  // VOL DS1480  // LBL CONDATA  // LFD CONDATA
// DVC 52  // VOL DS1480
// EXT ST,C,,CYL,10
// LBL TRACE  // LFD TRCFILE
// DVC 52  // VOL DS1480
// EXT ST,C,,BLK,(256,600)
// LBL TCIDTF  // LFD TCIDTF
// DVC 52  // VOL DS1480
// EXT ST,C,,CYL,3
// LBL DQFILE1  // LFD DQFILE1
// DVC 52  // VOL DS1480
// EXT ST,C,,CYL,3
// LBL DQFILE2  // LFD DQFILE2
// DVC 52  // VOL DS1480
// EXT ST,C,,CYL,3
// LBL DQFILE3  // LFD DQFILE3
// DVC 53  // VOL DS1485  // LBL LOAD#LIB  // LFD LOADLIB
// EXEC ZQ#IMS,LOADLIB,1
// PARAM START
/&
// FIN
```

This example executes a multithread IMS load module for the first time. The PARAM START statement tells IMS to initialize the control tables in the NAMEREC file. The absence of a STARTUP parameter statement means that the AUDFILE also will be initialized. A new disk trace file is allocated in this job stream; the EXT statement must be removed for subsequent runs.

Three ICAM disk queueing files and an ICAM disk buffering file are allocated. Their LFD names DQFILE1, DQFILE2, and DQFILE3 match the labels of the DISCFILE macros in the network definition examples in Section 2. The EXT statements must be removed after the initial run.

The name of the multithread module being executed is ZQ#IMS, and it is loaded from the LOADLIB library file. A priority of 1 is specified on the EXEC statement.

Example 2 - Executing Single-thread IMS with Warm Restart

```
// JOB XIMS,,C000
// DVC 20  // LFD PRNTR
// OPTION DUMP
// DVC 50  // VOL DISK01  // LBL NAMEREC  // LFD NAMEREC
// DVC 50  // VOL DISK01  // LBL AUDCONF  // LFD AUDCONF
// DVC 50  // VOL DISK01  // LBL STATFIL  // LFD STATFIL
// DVC 50  // VOL DISK01  // LBL TCIDTF   // LFD TCIDTF
// DVC 90  // VOL TAPE01,TAPE02,TAPE03  // LBL TRACE,TAPE01
// LFD TRCFILE,,EXTEND
// DVC 51  // VOL DISK02  // LBL DATAFL1  // LFD DATAFL1
// DVC 51  // VOL DISK02  // LBL DATAFL2  // LFD DATAFL2
// DVC 51  // VOL DISK02  // LBL DATAFL3  // LFD DATAFL3
// DVC 51  // VOL DISK02  // LBL SCRFORMAT  // LFD TC01FMTF
// EXEC LDIMS,,1
// PARAM STARTUP=WARM
/&
// FIN
```

This example executes a single-thread IMS load module called LDIMS with a warm restart. At start-up, all transactions that were active when the previous session terminated will be rolled back.

A statistical file is assigned. This file is required if you plan to use the ZSTAT transaction or if you specify STATS=YES for automatic recording of statistics at shutdown.

A 3-volume magnetic tape trace file is assigned, and the file is to be extended from the last session. If the file was left open at the end of that session, it was copied and closed with the ZC#TCP utility before being assigned to this run. Had the trace file been on disk, the TRCFILE=CLOSE parameter would have been included in the job control stream to close and extend it. The absence of a load library file assignment and a load library name on the EXEC statement indicates that the load module is stored in YLOD on SYSRES.

Screen formats are stored in the file named SCRFORMAT.

Example 3 - Executing IMS Using a Diskette Trace file

```
// JOB SYS80,,D000
// DVC 20 // LFD PRNTR
// OPTION SYSDUMP
// DVC RES // LBL NAMEREC // LFD NAMEREC
// DVC RES // LBL AUDCONF // LFD AUDCONF
// DVC RES // LBL ICAMFILE // LFD TCIDTF
// DVC 50 // VOL DATA // LBL EMPLOYEE // LFD EMPLOY
// DVC 50 // VOL DATA // LBL PAYROLL // LFD PAYROLL
// DVC 130 // VOL TRACE1 // EXT ST,C,0,CYL,75
// LBL IMS.TRACE // LFD TRCFILE
// DVC 131 // VOL TRACE2 // EXT ST,C,0,CYL,75
// LBL IMS.TRACE // LFD TRCFILE
// DVC 132 // VOL TRACE3 // EXT ST,C,0,CYL,75
// LBL IMS.TRACE // LFD TRCFILE
// EXEC ZS$IMS,,1
// PARAM STARTUP=COLD
/&
// FIN
```

This example executes the single-thread IMS load module, ZS$IMS, with a cold start. This allows terminal operators to display the last valid output message but does not perform file rollback.

Three diskette volumes are allocated for the trace file. Because all volumes must be online at once, each volume requires a separate device assignment set. The LBL names and LFD names must be same for all volumes.

The NAMEREC, AUDCONF, and ICAM files are on SYSRES, and the IMS load module, ZS$IMS, is also stored on SYSRES. Data files are on a separate disk volume.

Example 4 - Executing IMS with DEBUG Parameter to Locate Cause of Problems

```
// JOB SOLVE,,C000,,4
// DVC 20   // LFD PRNTR
// OPTION DUMP
// DVC 50  // VOL DISK01  // LBL NAMEREC  // LFD NAMEREC
// DVC 50  // VOL DISK01  // LBL AUDFILE  // LFD AUDFILE
// DVC 50  // VOL DISK01  // LBL CONDATA  // LFD CONDATA
// DVC 50  // VOL DISK01  // LBL TCIDTF   // LFD TCIDTF
// DVC 90  // VOL TAPE01,TAPE02,TAPE03  // LBL TRACE,TAPE01
// LFD TRCFILE,,EXTEND
// DVC 51  // VOL DISK02  // LBL DATAFL1  // LFD DATAFL1
// DVC 51  // VOL DISK02  // LBL DATAFL2  // LFD DATAFL2
// DVC 51  // VOL DISK02  // LBL DATAFL3  // LFD DATAFL3
// DVC 51  // VOL DISK02  // LBL SCRFORMAT  // LFD TC01FMTF
// EXEC LDIMS,,1
// PARAM DEBUG=YES
/&
// FIN
```

This example executes an IMS load module called LDIMS. The DEBUG=YES parameter tells the system to keep track of IMS status and terminal status. When you request a dump, it will include the status information.

Example 5 - Declaring Multiple Sequential Views whenever the SEQVIEWS Keyword Is Specified

In addition to declaring the primary view of the file, it is the user's responsibility to declare all secondary views of the file in the job control stream for executing IMS.

Each secondary view defined must be declared by appending a number (1 to 9) or a letter (A to Z) onto the file name of the primary file. When declaring secondary files, you must use the numbers first (beginning with 1), then the letters (beginning with A) in ascending numeric/alphabetic sequence, until the total number of secondary views declared equals the number of sequential views specified by the SEQVIEWS keyword. You cannot skip any number or letter during the declaration process; otherwise, a mismatch results between the job control stream declarations and what the configurator generates internally.

Note: *The file name of the primary view is limited to seven characters in length.*

In the following example, PROFILE1 and PROFILE2 are secondary views to the primary file PROFILE*:

```
// DVC 50
// VOL D0001
// LBL PRODUCT
// LFD PROFILE
// DVC 50
// VOL D0001
// LBL PRODUCT
// LFD PROFILE1
// DVC 50
// LBL PRODUCT
// LFD PROFILE2
// EXEC IMSLOAD
/&
// FIN
```

The file is considered usable if at least the primary view of the file can be opened successfully. The open or close status of the file is always based on the primary view.

Note: *Since IMS reacts to the ACCESS and BLKSIZE specifications at configuration time, using the DD job control statement to override these specifications at execution time is prohibited.*

Keep in mind that each sequential view of a file requires additional memory for the following, so the job card memory allotment should be upgraded to allow for these increases:

- File control structure
- CDIB
- RIB
- I/O and index areas

5.3.3. Modifying Action Programs in the LDPFILE

You can make changes to action programs in the LDPFILE while IMS is running.

To replace an action program load module in the LDPFILE, use the ZZPCH master terminal command. The next time a transaction calls on the action program after you issue ZZPCH, IMS loads the new version from the LOAD library and copies it to LDPFILE. The ZZPCH command is described in the *IMS Operations Guide,* UP-12027.

Before issuing ZZPCH, you must replace the action program in the LOAD library by recompiling and relinking or by applying a patch (COR). Do not use ALTER job control statements. Insert the following statement in the device assignment for the LOAD file in the job control stream for the compile and link or the COR stream:

```
// DD ACCESS=EXCR
```

5.4. Terminating the IMS Session

IMS is terminated by either of two commands from the master terminal: ZZSHD (shutdown) and ZZHLT (halt). These commands may also be entered from the system console or master workstation if OPCOM=YES is specified in the configurator OPTIONS section.

- The ZZSHD master terminal command directs an orderly shutdown of the IMS session.

Format

```
ZZSHD[nn]
```

where:

nn
: Specifies the number of minutes the ongoing transactions are permitted to continue processing. The maximum allowable time is 99 minutes for both single-thread and multithread IMS. The default shutdown timeout is three minutes in single-thread IMS and five times the configured time-out value in multithread IMS. The *nn* option is not supported from the system console unless the console is the master terminal.

When ZZSHD is entered, no further transactions are accepted from any of the terminals in the IMS network. A terminal operator attempting to enter a transaction receives the message

```
SHUTDOWN IN PROGRESS
```

Ongoing transactions are permitted to continue processing until the shutdown time has elapsed. Transactions not completed after this interval are canceled.

When no transactions are active, all files are closed, communications lines are deactivated, and IMS terminates normally.

- The ZZHLT master terminal command is intended for an emergency termination of the IMS session. When ZZHLT is entered, no further transactions are permitted, any transactions in progress are immediately aborted, all files are closed, the communications lines are deactivated, and IMS terminates with a main storage dump. File recovery is generally required after this type of termination, using either the offline recovery utility ZC#TRC or the warm restart option at start-up of the next online IMS session.

  The following message is displayed on the system console:

  ```
  JOB jobname ABNORMALLY TERMINATED ERROR CODE 000
  ```

  Whenever IMS terminates abnormally and outstanding I/O requests are still in progress, the following message is sent to the console:

  ```
  POSSIBLE IMS/USER FILE CORRUPTION
  ```

  This informs the user that file recovery may be necessary in order to enable further access to IMS files.

Section 6
Batch Processing of Transactions

6.1. Purpose and Uses of Batch Transaction Processing

The batch processor is an optional component of IMS that can be added to any single-thread or multithread configuration having adequate main storage and other resources. You include batch transaction processing by specifying the BATCH parameter in the NETWORK section of the configurator. The batch processor enables you to input transactions in card format, instead of through a display terminal; its output is directed to the printer or a printer file. Batched transactions can be processed online (that is, concurrently with routine production operations involving the normal terminal communications network) or offline, when no terminal network need be active.

You can submit transactions to the batch processor either from card images filed in disk source modules or from card images included as embedded data in the job control stream at IMS start-up. In neither case do you need to allocate a card reader to the IMS job.

Some important uses of batch transaction processing are:

- To print a file at the end of a day's production. A data base administrator, for example, might need to review a file in detail after massive update activity.
- To obtain a hard-copy listing of the data contained in a defined file or subfile -- an activity that is not practical in normal operations from a display terminal
- To test new UNIQUE-based file query and update dialogs, designed for routine use at production terminals
- To test new user-written action programs that your operators initiate as transactions during normal production

Printed output listed by the batch processor reproduces all input and output messages, as well as unsolicited output, and thus provides you with a permanent hard-copy record of each transaction.

6.2. Processing and Output

Batched transactions are processed as if they originated from actual terminals; the batch processor responds to one or more pseudoterminals created by the IMS configurator and lists output on a print file that is assigned to each pseudoterminal. This output is a step-by-step record of each transaction initiated.

Immediately after each input message, the batch processor prints the output message issued by the action program - with the exception of immediate or delayed internal succession. All input messages and output messages relating to one batch pseudoterminal are listed in the same print file and are not mixed with messages from any other pseudoterminal.

For UNIQUE dialogs initiated as batch transactions, the batch transaction processor lists what is normally seen as output by the terminal operator, formatted as it would appear on the remote terminal. In this case, each input message is printed above the output message response and starts in column 1. The start-of-entry character and the cursor are not represented. Terminal diagnostic and error messages are included in the output listing.

Figure 6-1 illustrates an output listing created by the batch transaction processor for a UNIQUE dialog that opens a defined file and lists its contents. The input messages include an OPEN, a LIST, six MORE LIST commands, and a CLOSE.

The output message that follows the **IMS READY** output message contains the name of the source input module, BATCHIN, and the name of the file, IN, on which it resides (specified in the PARAM IN statement). Automatic status messages -- INPUT IN PROCESS, INPUT IN QUEUE, and ROLLBACK IN PROCESS -- are never sent to a batch pseudoterminal.

The first input message, OPEN CUSTOMR, initiates the UNIQUE transaction; its response is the OPEN COMPLETE output message, normally returned to the originating terminal. The output of the LIST command is the first screenful of data records from the defined file, CUSTOMR; the next five MORE LIST commands cause the rest of the file to be displayed. The END LIST response (above the sixth group of records) indicates that all records in the file have been displayed. The next MORE LIST command, therefore, produces the ILLEGAL MORE COMMAND message, and the CLOSE command is followed by the routine CLOSE COMPLETE output message.

For readability of the output listing, the input messages shown in Figure 6-1 are punched beginning in column 10. This is feasible because they are all UNIQUE commands, and UNIQUE allows this measure of free-form input. Normally, input messages begin in column 1.

In normal nonbatch operations, when a user-written action program terminates in delayed internal succession, the terminal operator does not receive a message. The message that would otherwise be output is queued by IMS as input to the succeeding action and not displayed. For immediate internal succession, no message is sent to the operator or is queued. Instead, IMS makes the input and output message areas in main storage available to the successor action program.

```
 ** IMS READY **


 // PARAM IN=BATCHIN/IN
 READING SOURCE MODULE BATCHIN FROM FILE IN

         OPEN CUSTOMR

OPEN     COMPLETE   78/06/07    08:17:19


         LIST

                                                                  MORE      LIST
* CUST-ID  NAME                    ADDRESS          CITY-STATE        ZIP
  BALANCE-DUE   DUE-IN-VALUE   YTD-VOLUME
* AAOAD     ANY  BUM              BOWERY           NEW YORK,N.Y.     10010
         0.00           0.00         0.00
* BR8TL     BRANNON'S  BA6        86 TUMBLE LA.    PEMBROKE, PA.     16513
         0.00           0.00         0.00
* CA11ES    CARRIAGE TAVERN       137 ELM ST       POPULAR, N.J.     08613
         0.00           0.00         0.00
* CL3MD     CLOVER LEAF            35 MEADOW DR.   PASTURE, PA.      16161
         0.00           0.00      1,896.24

         MORE LIST

                                                                  MORE      LIST
* CUST-ID  NAME                    ADDRESS          CITY-STATE        ZIP
  BALANCE-DUE   DUE-IN-VALUE   YTD-VOLUME
* CR6HA     CREST PUB             6 HIGHLAND AVE   CREST CITY, PA    16331
         0.00           0.00         0.00
* CU1BA     CUMBERLAND C1UB       111 BAY AVE.     PORTSIDE, N.J.    08131
         0.00           0.00         0.00
* DE1NS     DEW DROP INN          13 NITEFALL ST   LIGHTHOUSE, PA    16217
        76.25           0.00        76.25
*FA1LA     FARRAH'S DEN          16 LION ALLEY    HILLSIDE, PA      16314
         0.00           0.00       243.19
```

Figure 6-1. Example of Output Listed by Batch Transaction Processor (Part 1 of 3)

```
        MORE LIST

                                                                  MORE      LIST
* CUST-ID  NAME                    ADDRESS          CITY-STATE        ZIP
  BALANCE-DUE   DUE-IN-VALUE   YTD-VOLUME
* HA6ER    HANOVER HOUSE          66 FOUNTAIN RD   GRAFTON, N.J.     08124
        0.00          0.00          0.00
* JO2WC    JOCKEYS JOINT          20 WINNER CIR.   STRAITAWAY, PA    16519
        0.00          0.00          0.00
* LA7HB    LAST CHANCE SALOON     72 HOPE BLVD.    GOINGVILLE, PA    16111
        0.00          0.00          0.00
* LO2BR    LOGAN LIQUORS          21 BARREL ROAD   POTTSBURG, PA.    16420
        0.00          0.00          0.00

        MORE LIST

                                                                  MORE      LIST
* CUST-ID  NAME                    ADDRESS          CITY-STATE        ZIP
  BALANCE-DUE   DUE-IN-VALUE   YTD-VOLUME
* LO2SC    LOST CLIPPER           25 SAIL CIRCLE   HARBOR, N. J.     08304
        0.00          0.00          0.00
* PE1PS    PERRY'S PUB            162 PLANK ST.    PERRYVILLE, PA    16212
        0.00          0.00          0.00
* RE1PA    RED LANTERN            15 BACK ALLEY    LEGALTOWN, N.J.   08412
       75.35          0.00         75.350
* RI4CL    RITTER'S ROOST         46 CHICKEN LA.   BARNYARD, PA.     16013
        0.00          0.00          0.00

        MORE LIST

                                                                  MORE      LIST
* CUST-ID  NAME                    ADDRESS          CITY-STATE        ZIP
  BALANCE-DUE   DUE-IN-VALUE   YTD-VOLUME
* RO1CS    ROYAL  NIGHTCLUB       147 CASTLE ST.   BLUEBLOOD, PA.    16310
       89.60          0.00         89.60
* SH1CA    SHAMROCK PALACE        121 CLANCY AVE   IRISHTOWN, PA.    16225
        0.00          0.00          0.00
* SU5MH    SUPPER CLUB            57 MAIN HWY.     OVERTON, N.J.     08015
        0.00          0.00          0.00
* T01ER    TOWNHOUSE CAFE         19 FRENCH RD.    SPURNBURG, PA.    16611
         0.00          0.00          0.00
```

Figure 6-1. Example of Output Listed by Batch Transaction Processor (Part 2 of 3)

```
          MORE LIST

                                                                  MORE      LIST
* CUST-ID  NAME                    ADDRESS          CITY-STATE        ZIP
  BALANCE-DUE   DUE-IN-VALUE   YTD-VOLUME
* TR2HS    TRYTON TOWER           238 HIGH ST.     TINKERTOWN, PA    16663
       25.83           0.00         25.83
* WO9BL    WOODEN NICKET          93 BUFFALO LA.   MINTBURG. PA.     16621
         0.00          0.00          0.00
* YD1RA    YOUR DEMISE            100 REST AVE.    BOOTHILL, MD.     10640
         0.00          0.00          0.00
* YY0YY    HOT SHOT               JOLLY ROAD       BLUE BELL, PA.    19000
         0.00          0.00          0.00


         MORE LIST

ILLEGAL MORE    COMMAND                                          ERROR     LIST


         CLOSE CUSTOMER

CLOSE   COMPLETE   78/06/07   08:17:32

/*
```

Figure 6-1. Example of Output Listed by Batch Transaction Processor (Part 3 of 3)

In batch operations, the message is not listed as an output or an input message. With immediate or delayed internal succession, the next message the batch transaction processor lists is the output message from the succeeding action.

Unsolicited output can be generated in any online batch-initiated transaction. This is the result of issuing the SEND function call in a user-written action program or including a SWTCH transaction in your input. Unsolicited output is not supported in offline mode. In the online batch mode, unsolicited output can be routed to one or more online active terminals.

Unsolicited output destined for another batch pseudoterminal is not supported. Unsolicited output addressed to the originating pseudoterminal is listed on its own print file. An online terminal must not send unsolicited output to a batch pseudoterminal.

Transaction types processed in batch mode, online or offline, include:

- UNIQUE dialogs - Initiated by the OPEN command and containing any UNIQUE commands
- Other transactions - Initiated by a transaction code (typically, these activate user-written action programs)

- The standard terminal commands - ZZTMD and ZZNRM. (The ZZHLD, ZZRDY, ZZCNC, and ZZRSD terminal commands have no useful role in a batch environment.)

- SWTCH transaction - For terminal-to-terminal communication

Batch input messages must not include any master terminal commands because batch pseudoterminals may not be designated as master terminals. Distributed data processing is not supported in batch mode. A batch pseudoterminal cannot route transactions to a remote IMS system using operator, directory, or program routing.

6.3. Controlling Batch Transaction Processing

You set the stage for batch transaction processing when you specify batch processing in the IMS configuration. You must also make certain modifications to the IMS execution run stream.

Batch transactions can be processed in offline or online modes. Control of offline processing is essentially a matter of the order in which source input is presented in the control stream (6.5). The ZZBTH master terminal command gives the online user considerable flexibility in determining when batch processing is to begin and end and in specifying which input is to be processed and in what order (6.6).

6.3.1. Effect of IMS Configuration Options

The BATCH parameter in the NETWORK section of your input to the IMS configurator adds the batch processing modules to your IMS system.

When you use this parameter, the IMS configurator generates one or more batch pseudoterminals. If you specify BATCH=YES, one pseudoterminal is generated and assigned the terminal-id BTH1. When you specify BATCH=*n*, the configurator generates the number of pseudoterminals designated by *n*; the terminal ids are BTH1,...,BTH4.

In a single-thread IMS system, you process only one batch input source module or one embedded data set at a time, so the specifications BATCH=YES and BATCH=1 are equally valid. In a multithread IMS configuration, although you still process only one embedded data set at a time, you can process up to the maximum number (*n*) of batch input source modules simultaneously. This means that if you specified BATCH=3 to the IMS configurator, you can process up to three source modules at one time, or one embedded data set and up to two source modules. In online mode, you initiate transactions by issuing the ZZBTH master terminal command. Again, the number of transactions that can be active simultaneously is limited by the maximum specified in the BATCH parameter. (The ZZBTH command is described in 6.6.1.)

6.3.2. IMS Control Streams for Batch Processing

In coding the IMS execution run stream for batch transactions you must:

1. Assign source module input files
2. Assign printer files
3. Insert PARAM statements to control the batch processor (these must always follow any other PARAM statements present)
4. Optionally, embed sets of input source data in the control stream

Assigning Source Module Input Files

Unless all batched transactions are to be represented by data sets embedded in the control stream, you must name and create one or more source modules containing the card images of your input messages, and place these modules in a disk library file. Each input disk file contains several modules. In the control stream, you must assign these source module input files to the IMS execution job using OS/3 job control conventions. The LFD-names of these input files are used in the PARAM IN statements.

Assigning Print Files to Batch Pseudoterminals

A print file must be assigned to each batch pseudoterminal that the IMS configurator has been directed to generate by the BATCH parameter. The batch processor expects these LFD-names for the print files:

Terminal-id	Print file LFD-name
BTH1	PRNTR1
BTH2	PRNTR2
BTH3	PRNTR3
BTH4	PRNTR4

Again, the assignment of these files follows OS/3 job control conventions.

Initiating and Controlling Batch Processor (PARAM Statements)

The batch processor parameter statement (PARAM BATCH) follows all other PARAM statements (except IN) in an IMS execution run stream. It immediately follows the EXEC statement if there are no other PARAM statements. The PARAM BATCH statement is followed by optional PARAM IN statements indicating source module input files to be processed (if any). BATCH and IN parameter statements are described in 5.3.1.

Embedding Source Data in Control Stream

If you wish, you may embed sets of card images of input messages in the batch processor control stream, following OS/3 job control conventions. These can be interspersed freely among the PARAM IN statements (Figure 6-2). However, for better performance in multithread batch processing, you should group all embedded data sets last, after all your PARAM IN statements.

Sample Control Stream

Figure 6-2 illustrates a control stream for executing a multithread IMS system for online batch transaction processing. Note that the arbitrary LFD-name of the input source file (SRFILE), assigned in a DVC-LFD job control sequence, is repeated in the two PARAM IN statements later in the control stream. One PARAM IN statement calls for processing of transactions in a module of SRFILE named TEST. The other statement refers to a module in this file named PROD. This example assumes that BATCH=4 is specified in the NETWORK section input to the IMS configurator. (The number of printer files assigned to the job indicates this.)

6.4. Preparing Transaction Input for Batch Processor

Input to the batch processor is in the form of card images, either filed on disk or included as embedded data sets in the IMS execution run stream. This makes it easier for you to do batch testing, as well as standard batch production runs.

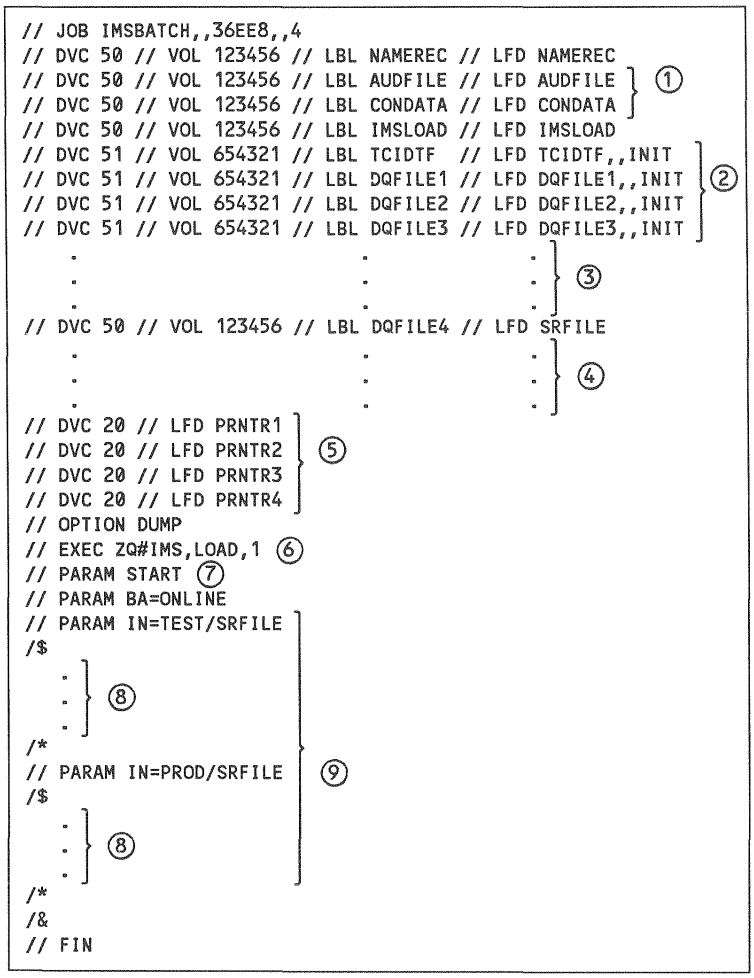


Figure 6-2. Sample IMS Execution Run Stream for Online Batch Processing in a Multithread System (Part 1 of 2)

Notes:

① A single-thread IMS system uses the AUDCONF file in lieu of these.

② Not required for offline batch mode, which does not use a terminal network.

③ Assign user data files

④ Assign source module input files referenced in ZZBTH master terminal commands

⑤ Assign printer files; one for each batch pseudoterminal created by the configurator. Equal in number to the number specified in the BATCH configurator keyword.

⑥ The program name on the EXEC statement must match the LOADM parameter specification on the IMSCONF jproc at configuration.

⑦ For offline mode, code this as // PARAM BA=OFFLINE.

⑧ Batch input cards

⑨ Param IN statements, or embedded data sets, or both, are required for offline batch mode. Data sets may be freely interspersed among PARAM IN statements. These are optional in online batch mode and when present are processed through one or more ZZBTH*(,ALL) master terminal commands. If absent in online batch mode, the *module-name,filename* form of the ZZBTH command is used to specify all input source modules to be processed.

Figure 6-2. Sample IMS Execution Run Stream for Online Batch Processing in a Multithread System (Part 2 of 2)

6.4.1. Input Message Coding

Each input message is contained on one card or as many as 18 cards; messages are grouped in the same sequence as the operator would enter them at the terminal. The batch processor takes input message text from card image columns 1 through 71. (Individual messages longer than 71 characters are continued on the next card image by coding a nonblank character in column 72; coding on the continuation card image begins in column 1.) Up to 17 continuation card images are allowed, giving a maximum length of 1346 bytes for any one message. Columns 73 through 80 are reserved for sequence numbers, which are neither required nor processed. Both input message text and device independent control expressions (DICE) are in EBCDIC code.

When coding UNIQUE dialogs for batch transaction processing, use the hard-copy format of the ADD and CHANGE commands to avoid the laborious preparation of DICE sequences needed for the display format.

When coding an input message for a user action program that does not allow free-form placement of data characters, you must code the data in the specific card columns required by the program.

The SWTCH transaction must always be coded beginning in input card column 1 because this is the position expected by the SWTCH action program.

Figure 6-3 illustrates a sample UNIQUE dialog transaction prepared as input for batch processing. It comprises 10 input messages and is followed by two SWTCH transactions for sending unsolicited output to active online terminals. These card images make an embedded data set for inclusion in the control stream. Notice that no delimiter is required to separate the UNIQUE dialog from the SWTCH transaction code that follows it (the CLOSE command serves this purpose) and that nothing is used to mark the beginning or end of the source module. Notice also the handling of continuation for the input message that contains more than 71 characters.

```
OPEN CUSTOMR
LIST
DETAIL CUSTOMR IF CUST-ID EQ 'BR8TL'
DETAIL CUSTOMR IF CUST-ID NE 'BR8TL'
MORE DETAIL
DETAIL CUSTOMR FOR CA1ES
DETAIL CUSTOMR IF NAME GT 'LOGAN LIQUORS    ';AFTER 'LOST CLIPPER'
DETAIL CUSTOMR;FROM LA7HB
MORE DETAIL
MORE LIST
CLOSE
SWTCH    TRM3;SWITCH COMMAND TSET
SWTCH    TRM2;SWITCH COMMAND TSET
```

Figure 6-3. Sample UNIQUE Dialog Transaction

6.4.2. Handling DICE Characters

Certain user-written action programs employ DICE sequences to format output messages. The characters in these sequences are written as hexadecimal symbol pairs (using EBCDIC characters). To eliminate the requirement that the hexadecimal codes be multipunched, the batch processor employs a shift character (the character, multipunch 11-7-8) to indicate that all EBCDIC characters following it are to be treated as hexadecimal digits (0-9,A-F) until another character is read. Each pair of hexadecimal digits is converted to a single byte before being passed to the batch processor. Two symbols in succession are converted to a single. For example, the following EBCDIC sequence in an input message:

¬ 10040000 ¬ YES ¬ 0571¬

is transmitted to the processor as:

1	0	0	4	0	0	0	0	E	8	C	5	E	2	0	5	7	1

Note: *The first four bytes are the DICE sequence for position control to new line on a CRT device.*

When the batch processor encounters DICE control sequences or the ESC (escape) character (27_{16}) in an output message area, it strips them before printing the message. This corresponds to the online handling of DICE codes and hardware characters, which are not displayed on the terminal. An output message area could look like this:

1	0	0	1	0	1	0	0	C	1	C	2	C	3	C	4	C	5	C	6	1	0	0	4	0	0	0	0	C	7	C	8	2	7	0	5	C	9

DICE Control Sequence | Message Text | Dice Control Sequence | Msg. text | ESC | HT | Msg. text

The print lines of this message after batch processing look like this:

First line: ABCDEF

Second line: GHI

Note the deletion of DICE control sequences and the ESC character (27_{16}) and HT (horizontal tab, 05_{16}).

6.5. Controlling Batch Processing in Offline Mode

Because IMS executes without a communications network in offline batch mode, you cannot control batch processing from the master terminal. Instead, you assign batch input source files with statements in the IMS execution run stream.

The BATCH=OFFLINE parameter statement is followed by IN parameter cards and/or embedded data sets of input card images; data sets are freely interspersed. The order in which these appear in the streams is the order in which the batches they represent are processed. The order of listed output is automatically determined by the batch processor. You can only indirectly control its order by the sequence in which you submit your input and, in a multithread IMS system, by the number of batch pseudoterminals you have generated with the BATCH configurator parameter.

6.6. Controlling Batch Processing in Online Mode

IMS requires an active communications network in the online batch processing mode. This requirement allows you to control batch processing from the master terminal, using the ZZBTH master terminal command to specify which source modules are to be processed and when.

Batch processing that is conducted in online mode during normal production hours, when the regular operating terminals are busy, downgrades performance. For this reason, it is better to run online batch processing during slow periods or before shutdown. Or you could do online batch processing during nonproductive time, perhaps configuring a special network consisting of the master terminal alone.

One factor that affects online batch performance is the number of batch modules being processed concurrently in a multithread IMS system. The smaller the number of modules, the better the performance. The maximum number of modules that can be processed concurrently is specified in the BATCH configurator parameter. However, the master terminal operator controls the number of modules that are actually submitted simultaneously to the batch processor (within the maximum limit) by his timing of the ZZBTH command.

You assign batch input source files and printer files with job control statements in the IMS execution run stream. You specify PARAM BATCH=ONLINE, followed by optional PARAM IN statements and embedded data sets. No batch processing takes place until you enter the first ZZBTH master terminal command. Note that PARAM IN statements or embedded data are appropriate only if you use the ZZBTH *[,ALL] form of the ZZBTH master terminal command to control batch processing in online mode. The *module-name, file-name* form of the ZZBTH command does not take source input from the control stream.

If you want to list the output printed by the batch processor before IMS terminates, the system console operator must call the output writer by entering PR BU; if any print files are ready, printing will begin immediately. The purpose of entering the BU (burst mode) parameter is to avoid printing the IMS job's log file first, as would otherwise happen. If you configure console support (OPCOM=YES in the OPTIONS section), you can send a message to the console operator using the SWTCH transaction.

If only one printer is available, you should configure a spooling supervisor at system installation time. (This is necessary because a nonspooling supervisor prints the output as soon as it is generated by the batch processor.)

6.6.1. ZZBTH Master Terminal Command

You enter the ZZBTH terminal command at the master terminal to initiate and control the batch transaction processor in online mode. Parameters are available to specify the source of input messages for each execution of the command, to control sequential progress through control stream input, and to terminate, suspend, or resume the batch transaction processing currently taking place.

Format

```
ZZBTH [module-name,filename]
      |*[,ALL]              |
      {CANCEL               }
      |PAUSE                |
      |RESUME               |
```

where:

`module-name`
: Is a one- to eight-character name of the source module that contains the card images of input messages to be processed by the batch transaction processor. When this *module-name, filename* form of the ZZBTH command is used, the batch processor locates the input source module in the file named by the second parameter, *filename*.

`filename`
: Is a one- to eight-character name of the file on which the foregoing source module resides. The *filename* parameter must match the *LFD-name* specified in a job control statement assigning the file to the current execution of IMS.

*
Specifies that the source of input messages for this execution of the ZZBTH command is the next PARAM IN statement or the next set of embedded data in the IMS job control stream. When this is the first ZZBTH * command for the current execution of IMS, the batch processor uses the first PARAM IN statement or embedded data set it encounters in the job stream. If this is not the first ZZBTH * command in the job stream, the batch processor uses the next PARAM IN statement or embedded data set that follows the input source used by the preceding ZZBTH * command.

ALL
Specifies that the entire run stream will be processed, starting with the next source of input messages (represented by the * parameter specified with this execution of the ZZBTH * command). If the optional ALL parameter is omitted, processing of the run stream input stops when the next PARAM IN card or set of embedded data is processed. If a subsequent source module is represented in the run stream, you must enter another ZZBTH * [ALL] command to process it.

CANCEL
Specifies that all batch transaction processing currently in progress must terminate. Processing of each transaction ceases after the current output message has been sent to the print file, and the print file is closed. File modifications made since the beginning of the current transaction are not rolled back. The batch processor substitutes a ZZCNC terminal input command for the next expected input message.

PAUSE
Specifies that all batch transaction processing currently in progress must be suspended. Processing of each transaction ceases after the current output message has been sent to the print file, but the print file is not closed.

RESUME
Specifies that all batch processing temporarily suspended by the preceding ZZBTH PAUSE command must resume.

6.6.2. Initiating Online Batch Processing

When you enter the ZZBTH command at the master terminal and the batch processor recognizes batch input, it sends the message BATCH INITIATED to the master terminal, and processing begins.

With single-thread IMS, you can enter only one ZZBTH command at a time; the transaction it requests must be completed before you enter another ZZBTH command. Conversely, in a multithread IMS system, several ZZBTH commands can be processed simultaneously, up to the maximum specified by the BATCH configurator parameter.

If the master terminal operator attempts to enter a ZZBTH command in excess of the maximum number of batch modules that can be processed concurrently (or a single-thread operator enters a second command before the processing of the current command is finished), IMS responds with a diagnostic message:

```
TOO MANY ZZBTH COMMANDS ENTERED - COMMAND IGNORED
```

6.6.3. Tracking Progress of Batch Processing

To determine when a batch run is complete or to keep track of the processing of batch modules, the master terminal operator, at any time, can enter the ZZTCT master terminal command to access the terminal control table generated for any batch pseudoterminal, and specify the batch terminal-id desired (BTH1,...,BTH4). The ZZTCT command provides the master terminal with a report of the activity outstanding at the specified terminal.

Format

```
ZZTCT terminal-id
```

where:

terminal-id
: Is the configured symbolic identification of the specified terminal.

In response to the ZZTCT terminal command, IMS sends this message to the master terminal:

```
STATUS OF terminal-id [UP ]  ;nnnn MSG;nnnn TRAN; nnnn TRM CMD;ALT=name
                      {DWN}
                      {HLD}
                      [TMD]
```

where:

nnnn
: Is the number of messages, transactions, or terminal commands.

6.6.4. Resuming Batch Processing Once Terminated

The ZZBTH CANCEL command terminates only the batch transactions currently being processed. The transaction processing that has been completed is listed. Nothing prevents you from reprocessing the same modules or processing other source modules. You can continue by issuing the appropriate ZZBTH command after having received a response to the ZZBTH CANCEL command. The response to the ZZBTH CANCEL command is this message:

```
BATCH PROCESSING CANCELLED
```

The effect of issuing a ZZBTH *[,ALL] command after you have issued a ZZBTH CANCEL command is to read the control stream at the next data set present. If you issue the ZZBTH *module-name, filename* form of the command and specify a module whose processing was interrupted by the ZZBTH CANCEL command, that module will be processed from its beginning, not from the point where processing stopped.

6.6.5. Repetitive Use of Batch Mode

There is no feature in the batch transaction processor that checks whether your files have been processed by the same input -- in fact, batch mode facilitates reprocessing with repetitive use of the same input. This enables you to use the same program at the end of each day to list the state of your files and to review the day's activity.

6.7. Continuous Output Considerations

Output delivery notification to a batch pseudoterminal is always considered successful by the batch transaction processor. A program using output-for-input queueing can be testing in batch mode, but it is not reasonable to perform continuous output by this means in a batch production run.

6.8. Batch Processor Diagnostic Messages

Besides the diagnostic and error messages IMS sends to the operator at a production terminal (which the batch processor includes in the output listing for each batch pseudoterminal), the batch processor generates a set of its own messages. It sends these to the master terminal or includes them in the output listing for the batch pseudoterminal, as is appropriate. Table E-3 lists the text of these messages, describes their causes, and indicates the corrective action you can take.

6.9. Recovery Considerations

To recover batch processing after either a cold or warm restart, you send a DLMSG input transaction (in card image form) to the batch processor. You must supply one card image for each batch terminal. For single-thread IMS, only one card is required. The printed output from the DLMSG transaction indicates how much of the input batch module was processed before the system aborted. You can then reconstruct the batch input card deck or use the librarian to edit a source library input message module.

An example of a warm restart batch input for a single-thread batch system is:

```
/$
DLMSG BTH1
/*
```

For a warm restart, the before images for the files are restored at the start of the last transaction completed.

Section 7
File Recovery

7.1. Recovery Functions

Preserving the integrity of your data base is a vital consideration when you are updating your files. IMS provides a complete file recovery system that operates in both online and offline environments to simplify the protection of your DAM, ISAM, and MIRAM files.

Online recovery is automatic when you specify file updating in your configuration (unless you configure LOCK=UP in the FILE section). Every time a transaction terminates abnormally, IMS rolls back your files to the start of the transaction or to the last rollback point specified in the action program performing the transaction. IMS also rolls back your files at system start-up when you specify a warm restart; in this case, all transactions that were active at the time of IMS termination are rolled back.

Offline recovery is performed by an IMS utility program, ZC#TRC, after termination of online processing. The offline recovery program restores files left damaged or in an inconsistent state as a result of a system failure or any other reason. A second offline utility, ZC#TCP, copies and closes a tape trace file left open by system failure. Offline recovery is available only when you specify the RECOVERY parameter in your configuration.

Both online and offline recovery functions assume that your files are not updated by non-IMS programs during the IMS session. Updating by other programs would compromise the recovery process. To avoid this possibility, you must include file-locking capability in your control system at system generation time. You can choose from several kinds of access rights by specifying the ACCESS data management parameter in the FILE section of the configuration. If you omit the ACCESS parameter, the configurator default specification prevents other programs from updating your files.

During online processing, IMS creates three internal files (if configured) that aid in recovery:

1. The audit file (AUDFILE for a multithread system, AUDCONF for single-thread), used for online recovery

2. The terminal output message file (TOMFILE), which allows terminal operators to determine how far rollback has gone

3. The trace file (TRCFILE), used for offline recovery

Table 7-1 summarizes the online and offline recovery functions, the internal files used for each, and the configuration parameters required.

Table 7-1. Recovery Options

Online/ Offline	Recovery Function	Internal Files Used	Configuration Parameters Required*
Online	Rollback: ■ Online transaction rollback ■ Warm restart	Audit file (AUDCONF for single-thread, AUDFILE for multithread)	FUPDATE
	Display last effective output message (DLMSG transaction)	Terminal output message file (TOMFILE partition of AUDCONF or CONDATA file)	FUPDATE, TOMFILE
Offline	Utility programs: ■ Offline recovery utility (ZC#TRC) ■ Tape copy routine (ZC#TCP)	Trace file (TRCFILE)	FUPDATE, RECOVERY
	Recovery TOMFILE in addition to data files	TRCFILE, TOMFILE	FUPDATE, RECOVERY TOMFILE, TOMTRCE

* All configuration parameters listed are specified in the OPTIONS sections.

7.2. Files Created for Online and Offline Recovery

7.2.1. Audit File

The multithread AUDFILE or the single-thread AUDCONF file is a partitioned system access technique (SAT) disk or diskette file containing a partition for each terminal in your network. Before-images of updated records are written to the partition for the terminal originating the transaction. Each partition must be large enough to contain all before-images recorded between two rollback points or between a rollback point and transaction termination. You specify the maximum number of records to be contained in a partition with the AUDITNUM parameter in the configurator GENERAL section. At rollback points and transaction termination, the current record pointer for the active partition is reset to point to the first record of the partition and the records from the previous transaction are overwritten. If the number of before-images exceeds the number specified by the AUDITNUM parameter, the transaction is canceled and all updates from that transaction (or since the last rollback point) are rolled back.

7.2.2. Terminal Output Message File

The terminal output message file (TOMFILE) is a partition of the single-thread AUDCONF file or the multithread CONDATA file. Each terminal is allocated its own record for output message logging. At rollback points and transaction termination, IMS records the contents of the action program's output message area in the TOMFILE, retaining only one message per terminal. The most recent output message for a terminal overlays the one before it.

The terminal operator can display the most recent output message by entering the transaction code DLMSG. The same transaction code is automatically displayed at each terminal after a cold or warm restart, and the terminal operator can retrieve the last effective output message by pressing the TRANSMIT key.

Terminal output messages are also recorded in the trace file if the TOMTRCE configurator parameter is specified. This allows the offline recovery utility to roll back the TOMFILE to reflect the state of your data files after recovery.

7.2.3. Trace File

The trace file (TRCFILE) is a disk, diskette, or magnetic tape SAT file, maintained by online IMS as a continuous, unblocked, sequential file. A tape trace file may have multiple volumes. A diskette trace file, available only in System 80, and a disk trace file may also have multiple volumes, but all volumes must be online at the same time. You can use the same trace file for more than one IMS session by specifying PARAM TRCFILE=CLOSE or TRCFILE=EXTEND for disk/diskette (5.3.1), or the EXTEND parameter of the LFD statement in the job control stream at IMS start-up for tape. The usual procedure is to start a new trace file each time you do a security dump of your data files, so that the current trace file reflects all updates entered into your files since the last security dump. When you use a diskette trace file, you usually start a new trace file for each session because space is limited to the number of diskettes you can have online at one time.

The trace file contains before-images, after-images, or both, of all updated records (depending on your RECOVERY parameter specification at configuration), plus rollback and termination records. If you specified both the TOMFILE and TOMTRCE configurator parameters, the trace file also contains output messages generated at rollback points and transaction termination.

The block size of the trace file is calculated by the configurator from the largest record length or block size specified in the FILE section, except that those files for which TRACE=NO is specified are not considered. If you specify terminal output message tracing, the configurator also checks the OUTSIZE specification in each of your ACTION sections and uses the largest of the record length or OUTSIZE specifications in your configuration.

Each block of the trace file contains a 104-byte prefix area plus a before-image, after-image, output message, rollback point, or transaction termination record. Figure 7-1 illustrates the layout of the prefix area, and Table 7-2 describes its contents.

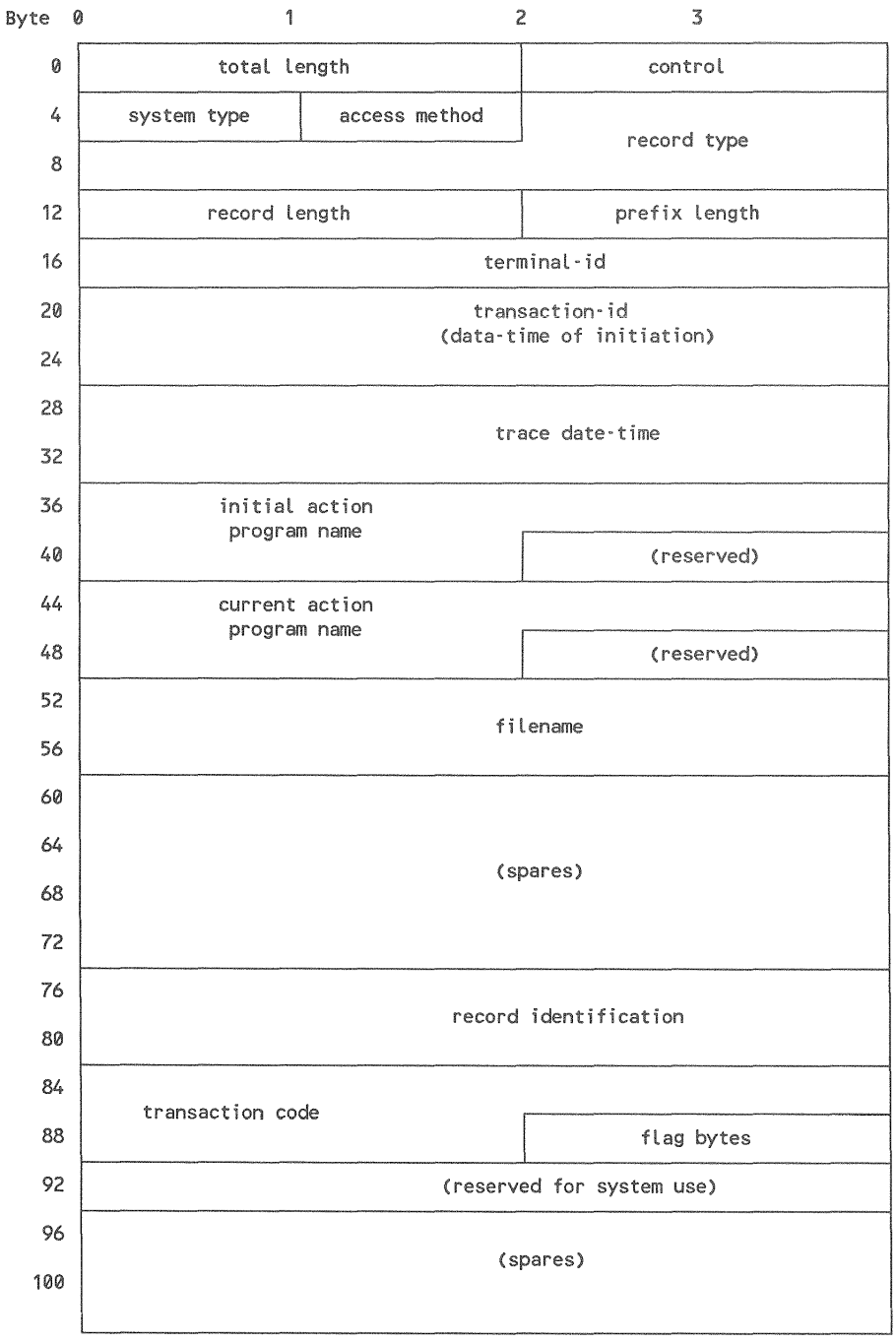


Figure 7-1. Format of Prefix Area of Records in the Trace File

Table 7-2. Contents of Prefix Area for Trace File Records

Label	Field Name	Bytes	Code	Description
ZF#VRLEN	Total length	0-1		Total length of block; 104-byte prefix plus length of record or output message
-	Control	2-3		Control bytes for OS/3 data management
ZF#STYP	System type	4	EBCDIC	Constant: I for IMS
ZF#ACM	Access method	5	Hexadecimal	D for MIRAM (DTF mode) I for ISAM M for MIRAM (CDM mode) R for DAM (relative mode)
ZF#RTYP	Record type	6-11	EBCDIC	BEFORE Before-image AFTERΔ After-image OUTPUT Output message ROLLBP Rollback point TERMIN Transaction termination record
ZF#TRL	Record length	12-13	Binary	Length of record or output message, in bytes
ZF#TPL	Prefix length	14-15		Binary 104 bytes + key length (indexed files) or 8 bytes (nonindexed files)
ZF#TTMID	Terminal-id	16-19		Configured identification of terminal originating transaction
ZF#TTRID	Transaction-id	20-27		Date-time of initiation of current transaction, in the form: yy-dd-mm-hh-mm-ss
ZF#TTIME	Trace date-time	28-35		Date-time of writing this record to the trace file, in the same format as transaction-id
ZF#NAPI	Initial action program name	36-41		Configured name of first action program called for in this transaction
-	-	42-43	-	Reserved

continued

Table 7-2. Contents of Prefix Area for Trace File Records (cont.)

Label	Field Name	Bytes	Code	Description
ZF#NAPC	Current action program name	44-49		Configured name of action program currently active
-	-	50-51	-	Reserved
ZF#TFNM	File name	52-59	EBCDIC	Configured name of the conventional file accessed by current action program, left-justified. For terminal output messages, this field contains the name TOMFILE.
ZF#ANAM	Spares	60-75	-	Unused
ZF#TKLID	Record identification	76-83		For an indexed file, bytes 76-79 contain the identification key length, and bytes 80-83 the key location, measured in hexadecimal bytes. For a nonindexed file, bytes 76-83 contain the relative record number of the record, in binary.
ZF#TCODE	Transaction code	84-89		Configured code of the current transaction, left-justified.
ZF#TFB	Flag bytes	90-91	Binary	Reserved for system use
ZF#TRCL	-	92-95	-	Reserved for system use
ZF#TSPAR	Spares	96-103	-	Unused

7.3. Online Recovery

When you are updating your data files, IMS writes a before-image of each updated record to the audit file (unless you have specified LOCK=UP in the configurator FILE section). When a transaction is abnormally terminated, or at system start-up when warm restart is specified, IMS uses these images to roll back the updated records to the beginning of the transaction or to the last rollback point. The online recovery facility is automatically included in your configured IMS system when you specify file updating (FUPDATE parameter).

If you specify TOMFILE=YES in the configurator OPTIONS section, IMS copies the contents of the action program's output message area to the terminal output message file (TOMFILE) at each rollback point and at transaction termination. When data files are rolled back, the terminal operator can determine how far rollback has gone by entering the transaction code DLMSG to display the last effective output message from the TOMFILE.

7.3.1. Online Transaction Rollback

When a transaction terminates abnormally, the rollback program rolls back data files for which you have configured LOCK=TR to the state they were in before the transaction that failed or to the last rollback point specified by your action program or UNIQUE. More information about online transaction rollback is available in the IMS action programming manuals.

7.3.2. Warm Restart

Warm restart is actually a multiple online transaction rollback. When you include the PARAM STARTUP=WARM statement in the job control stream at IMS start-up, IMS rolls back all updates performed during transactions that were active at the time of IMS termination or OS/3 system failure. Transactions are considered active whenever the last record written to the audit file for a transaction is not a termination or rollback record. To use the warm restart facility, you must specify TOMFILE=YES in a single-thread configuration; it is optional in a multithread configuration.

Note: *Warm restart is supported for indexed MIRAM files only if they were created with the data management recovery option (RECV=YES of the DD job control statement). Refer to the* Consolidated Data Management Programming Guide, *UP-9978.*

7.4. Offline Recovery

The purpose of offline recovery is to restore data files left damaged or in an inconsistent state after IMS termination or system failure. The ZC#TRC offline recovery program can perform three different types of file recovery:

1. Forward recovery (required if a portion of your data file is destroyed)
2. Backward recovery (used when you data files are accessible but in an inconsistent state)
3. Quick recovery (used when transactions are left incomplete because of system failure or emergency IMS termination)

During online operations, IMS writes before-images, after-images, or both (depending on your RECOVERY configurator parameter specification), of updated records, plus rollback points and termination records, to a tape, disk or diskette trace file. It also copies output messages generated at rollback points and transaction termination to the trace file, if you specify both TOMFILE and TOMTRCE to the configurator.

The offline recovery program uses the information in the trace file to restore your data files and to roll back the TOMFILE to reflect the state of your data base after recovery. The program can also be used to print out the contents of the trace file, giving you a listing of all the updates recorded during an IMS session.

If your trace file is on magnetic tape and is left open because of system failure, you must copy it onto another tape volume and close it properly, using the ZC#TCP tape copy routine, before it can be used for offline recovery.

7.4.1. Types of Recovery

The type of file recovery required depends on the state your data files are in at IMS termination or system failure. When any portion of your data files is destroyed, you should use the forward recovery procedure. When your files are accessible but in an invalid state, you should use backward recovery. When only the transactions that were active at the time of IMS termination need to be rolled back, you should use quick recovery. You specify the type of file recovery to the ZC#TRC utility with the RECOVERY-TYPE parameter.

Forward Recovery

Forward recovery is used for reconstruction of damaged files. Because of this, you need a copy of these files that you previously created in a security dump or dump/restore procedure. Refer to the *System Service Programs (SSP) Operating Guide,* UP-8841, for information on creating backup files.

In forward recovery, the ZC#TRC utility updates your backup files to the date-time you specify or, if you do not specify a date-time, records all completed transactions. To update your backup files, ZC#TRC uses the after-images recorded in the trace file; these after-images are recorded in the trace file only when you specify RECOVERY=AFT or RECOVERY=ALL in your configuration.

Forward recovery is a two-pass procedure. During the first pass, the program reads the trace file to determine which transactions were active against the specified user data files at the specified date-time or (if no date-time is specified) at the end of the trace file. During its second pass, the program applies to these files all after-images recorded in the trace file for completed transactions. If the ZZCLS and ZZOPN master terminal commands were issued during the IMS session for any of your files, you should include the PFILES parameter.

For the files you specify, the recovery program applies only the after-images recorded since the last valid ZZOPN master terminal command was issued. For incomplete transactions, any after-images traced between the start-of-transaction record and the last rollback point are applied. If you specify that the TOMFILE is to be recovered in addition to your data files, ZC#TRC copies to the TOMFILE the last output message recorded in the trace file for each terminal up to the recovery point.

In this type of recovery, ZC#TRC reads the trace file in the forward direction during both passes. If you have a multivolume magnetic tape trace file, you must remount the first volume at the end of the first pass.

Backward Recovery

Backward recovery should be used when your data files are accessible but, for some reason, some of the updates recorded in them are invalid. If you can pinpoint a date and time at which your files were in a valid state, you should specify the DATE-TIME parameter; only those updates recorded after that point are rolled back. When you include the PFILES parameter, only those updates recorded since the last ZZOPN command are rolled back for the files you indicate. If you omit both DATE-TIME and PFILES, all updates since the start of the trace file are rolled back.

ZC#TRC uses the before-images recorded in the trace file for backward recovery; to use this feature, you must specify RECOVERY=BEF or RECOVERY=ALL to the configurator.

ZC#TRC uses two different procedures for backward recovery, depending on whether or not you specify the DATE-TIME parameter:

- DATE-TIME specified: two-pass procedure

  During the first pass, the recovery program reads the trace file in the forward direction to find transactions active against the specified data files at the date-time indicated. In the second pass, ZC#TRC reads the trace file in the backward direction and applies all before-images recorded from the end of the trace file back to the last rollback point or start-of-transaction record before the specified date-time. For files specified on the PFILES parameter, before-images are applied back to the last ZZOPN command, if later than the date-time specified.

  When you have a multivolume magnetic tape trace file, you must mount the volumes sequentially in the forward direction for the first pass and then in the backward direction for the second pass.

- DATE-TIME not specified: single-pass procedure

  When you do not specify the date and time, ZC#TRC makes only one pass, reading the trace file in the backward direction. All before-images are applied to the specified data files back to the beginning of the trace file. For files specified on the PFILES parameter, before-images are applied back to the last ZZOPN command.

  If you have a multivolume magnetic tape trace file, volumes must be mounted in the backward direction, starting at the end of the last volume.

If you specify the DATE-TIME parameter, you can recover the TOMFILE in addition to your data files. ZC#TRC copies to the TOMFILE the output message recorded in the trace file for each terminal at the last rollback point or transaction termination before the specified date-time.

You should avoid using the SET DATE console command to set the system IPL date backward during an IMS run for the following reasons:

- Offline recovery assumes that the TRACE-DATE-TIME in the TRACE file is always in ascending sequence. The TRCFILE=CLOSE option of the ZC#TRC routine closes the file by recording the EOD pointer as soon as it detects the DATE-TIME of the current record lower than that of the preceding record.

- If the TRCFILE=CLOSE option is not used, the existence of records in the TRACE file that are not in ascending sequence may have adverse effects on normal operation of OFFLINE recovery.

Note: *When the beginning of a magnetic tape trace file volume is read during the execution of backward recovery, the following message is displayed on the system console:*

```
DM22        TRCFILE physical unit number
            HARDWARE ERROR CHECK ERROR STATUS/SEN
```

Ignore this message and respond to the next message, which tells you to mount the next volume.

Quick Recovery

Quick recovery, a two-pass procedure, may be used in the case of system failure or emergency IMS termination. When the ZZHLT master terminal command is issued, or when the system fails, the transactions that are in process cannot be completed. Quick recovery rolls back all transactions active at the time of termination. The DATE-TIME parameter is never specified.

During the first pass, the recovery program scans the trace file in the forward direction to determine which transactions were active against the specified files at termination. In the second pass, ZC#TRC reads the trace file in the backward direction and applies before-images to the specified files until a rollback point or a start-of-transaction record is reached for all the transactions active at the end of the trace file. For files specified on the PFILES parameter, recovery stops at the last ZZOPN command, if reached before a rollback point or start-of-transaction record. To use the quick recovery feature, you must specify RECOVERY=BEF or RECOVERY=ALL to the configurator.

Magnetic tape trace file volumes must be mounted sequentially in the forward direction for the first pass and then in the backward direction for the second pass.

Recovery of the TOMFILE is not necessary, because the TOMFILE already contains the output message associated with the last rollback point or transaction termination before system failure.

An alternate way of recovering your data files after system failure or emergency termination is to use the warm restart feature at initiation of your next online IMS session.

Note: *Quick recovery is supported for MIRAM files only if they were created with the data management recovery option (RECV=YES of the DD job control statement). Refer to the* Consolidated Data Management Programming Guide, *UP-9978.*

7.4.2. Running Offline Recovery

Linking the Offline Recovery Program

Before you can use the offline recovery utility, ZC#TRC, you must link the recovery object module, ZE#OLREC, with:

- The object module containing your configuration control tables and data file DTFs or CDIB/RIBs
- The appropriate data management I/O modules for
  - Your data files
  - The print file
  - The trace file, if it is on tape

The ZE#OLREC module is stored in the YOBJ library file on SYSRES or OS3REL, unless you have copied it onto a user library. Data management modules are in YOBJ on SYSRES.

The configuration object module name is in the form:

ZS$Onnn	- for single-thread DTF system
ZQ$Onnn	- for multithread DTF system
ZS$IOnnn	- for single-thread CDM system
ZQ$IOnnn	- for multithread CDM system

where *nnn* is the configuration identifier (4.3.1).

This module is stored in the library specified with the LIBO parameter of the IMSCONF jproc at configuration; the default library is YOBJ on OS3REL or SYSRES. If you use the IMSGEN option at SYSGEN and the LIBO default at configuration, the recovery and configuration object modules are on OS3REL. In this case, it may be convenient for you to link the offline recovery program immediately after you configure your IMS system, while your OS3REL volume is still online.

Standard job control streams for linking the offline recovery program ZC#TRC in a DTF system and a CDM system are shown in Figure 7-2. Note that linkage editor control statements must start in column 2 or beyond.

Lines 1 through 10 are the same in both examples in Figure 7-2. The device assignment sets on line 2 and succeeding lines identify the library files containing the recovery and configuration object modules and the output load module. These may be omitted for modules stored in system libraries on SYSRES. The PARAM statement (line 6) specifies where the linked module is to be stored; this must match the file name on the LFD statement for the output module. The WORK jproc (line 3) and the printer device assignment (line 4) are required by the linkage editor.

Lines 8 through 17 in Figure 7-2a and 8 through 12 in Figure 7-2b are linkage editor control statements. The LOADM statement (line 8) assigns the name ZC#TRC to the output module. The LINKOP statement (line 9) directs the linkage editor not to automatically include modules or phases. This statement will cause unresolved EXTRNs to be generated; these should be ignored. INCLUDE statements for the recovery and configuration modules (lines 10 and 11) are always required.

In a DTF system (Figure 7-2a), the INCLUDE statement on line 12 is for a printer I/O module, which is always required, and the INCLUDE statement on line 13 is for a SAT I/O module, required only for a tape trace file; you need not specifically include a SAT module for a disk or diskette trace file. Lines 14 through 16 include I/O modules for ISAM, DAM, and MIRAM data files. (Include the MIRAM I/O module if your files are organized as IRAM or MIRAM files.) Line 17 names ZE#OLREC as the entry point for the load module.

In a CDM system (Figure 7-2b), printer, MIRAM, and SAT I/O modules need not be specifically included. Line 12 names ZE#OLREC as the entry point for the load module.

Note: *When you link the offline recovery ZC#TRC in a CDM system, an unresolved reference occurs for module ZF#TRERR. Ignore this unresolved reference because offline recovery doesn't use ZF#TRERR and is still operable.*

```
 1.  // JOB jobname
 2. [// DVC 50  // VOL vol-ser-no  //  LBL library-name  //  LFD filename]
    .
    .
    .
 3.  // WORK1
 4.  // DVC 20  //  LFD PRNTR
 5.  // EXEC LNKEDT
 6.  // PARAM OUT=filename
 7.  /$
 8 .  LOADM ZC#TRC
 9.   LINKOP NOAUT,NOV
10.   INCLUDE ZE#OLREC,filename
11.   INCLUDE  {ZQ#I0nnn},filename
               {ZS#I0nnn}

12.   INCLUDE PR$IOE,$Y$OBJ
13.  [INCLUDE ST$1110,$Y$OBJ]
14.  [INCLUDE IS$0313,$Y$OBJ]
15.  [INCLUDE DD$D111,$Y$OBJ]
16.  [INCLUDE D3$M111,$Y$OBJ]
17.   ENTER ZE#OLREC
18.  /*
19.  /&
20.  // FIN
```

a. DTF system

```
 1.  // JOB jobname
 2. [// DVC 50 // VOL vol-ser-no // LBL library-name // LFD filename]
    .
    .
    .
 3.  // WORK1
 4.  // DVC 20 // LFD PRINTR
 5.  // EXEC LNKEDT
 6.  // PARAM OUT=filename
 7.  /$
 8.   LOADM ZC#TRC
 9.   LINKOP NOAUT,NOV
10.   INCLUDE ZE#OLREC,filename
11.   INCLUDE {ZQ$IOnnn},filename
              {ZS$IOnnn}

12.   ENTER ZE#OLREC
13.  /*
14.  /&
15.  // FIN
```

b. CDM system

Figure 7-2. Job Control Stream for Linking the Offline Recovery Program

Parameters for Offline Recovery

The type of recovery performed by the ZC#TRC utility -- forward, backward, or quick -- and the files that are to be recovered are determined by parameters that you insert in the job control stream for executing ZC#TRC. Parameters also specify whether recovery is to be delimited by a date-time, whether partial recovery is to be performed on one or more files, whether a disk trace file must be closed, and listing options.

All parameters are optional except RECOVERY-TYPE. They may be specified in any order and may begin in any column, but each must be submitted on a separate card. Parameters may not be split between cards; if all subparameters do not fit on one card, the keyword must be repeated on another card with the additional specifications.

Format

```
RECOVERY-TYPE= [BACKWARD]
               {FORWARD }
               [QUICK   ]

[DATE-TIME=yy-mm-dd/hh:mm:ss]

[FILES= [ALL                          ]]
       {filename-1,...,filename-n}
       [NONE                          ]

[PFILES=filename-1[,...,filename-n]]

[PRINT= [ALPHA]]
       {BOTH }
       [HEX  ]

[TRCFILE=CLOSE]
```

RECOVERY-TYPE Parameter

The RECOVERY-TYPE keyword parameter specifies whether forward, backward, or quick recovery is to be executed on the files named on the FILES parameter. If the FILES parameter is omitted (or FILES=NONE is specified) and the PRINT parameter is included, the RECOVERY-TYPE specification determines whether before- or after-images of records in the trace file are listed. If RECOVERY-TYPE=FORWARD is specified, after-images are printed; if RECOVERY-TYPE=BACKWARD or QUICK is specified, before-images are printed.

RECOVERY-TYPE=BACKWARD
Specifies backward recovery; the ZC#TRC utility applies before-images to the specified files.

RECOVERY-TYPE=FORWARD
Specifies forward recovery; ZC#TRC applies after-images to the specified files.

RECOVER-TYPE=QUICK
Specifies quick recovery; ZC#TRC applies before-images to the specified files for all transactions active at the time of failure.

If the RECOVERY-TYPE parameter is omitted, ZC#TRC displays the message PARAMETER ERROR on the system console.

DATE-TIME Parameter

The DATE-TIME keyword parameter specifies the point at which recovery is to terminate in each of the files rolled back or forward. All records are recovered up to the rollback point nearest to the specified date and time.

DATE-TIME=yy-mm-dd/hh:mm:ss
Specifies the date and time at which recovery is to terminate. Hyphens, the slash, and colons must be coded as shown.

yy
Specifies the year (00-99).

mm
Specifies the month of the year (01-12).

dd
Specifies the day of the month (01-31).

hh
Specifies the hour (00-23).

mm
Specifies the minute (00-59).

ss
Specifies the second (00-59).

If the DATE-TIME keyword parameter is omitted, forward recovery terminates when EOF is reached; backward recovery terminates when the beginning of the volume is reached.

Notes:

1. When you perform backward recovery using the DATE-TIME parameter, do not specify the exact date and time of a termination record. This can cause the recovered file to be in an inconsistent state. Instead, you should specify a time earlier than that of the termination record.

2. Conversely, when you perform forward recovery, you should specify a time later than that of the termination record.

FILES Parameter

The FILES keyword parameter designates which files, if any, are to be recovered. In addition to your data files, you can also specify recovery of the TOMFILE partition of the AUDCONF or CONDATA file, if you included the TOMTRCE parameter at configuration.

The filenames you specify for your data files are their LFD-names. However, you specify the file name for the TOMFILE partition, even though its LFD name is AUDCONF or CONDATA.

If you specify a data file name that you did not identify to the configurator (with the FILES keyword in the ACTION section), ZC#TRC displays the following message at the system console:

```
FILE NAME filename-n INVALID
```

Nevertheless, all files for which you have specified a valid (that is, configured) file name are recovered.

`FILES=ALL`
Specifies that all configured user data files are to be recovered. If you have configured TOMFILE tracing, the TOMFILE is also recovered.

`FILES=filename-1,...,filename-n`
Specifies one or more configured names of individual files that are to be recovered.

`FILES=NONE`
Specifies that no files are to be recovered and is the default assumption. The ZC#TRC utility lists the trace file records, with prefixes, if the PRINT keyword is specified.

PFILES Parameter

If you have a file or files for which the ZZCLS and ZZOPN master terminal commands were issued during the IMS session (and which may have been accessed by other programs), you should include the PFILES parameter to perform partial recovery of those files. The PFILES parameter directs the offline recovery program to apply to the designated files only the before- or after-images recorded in the trace file after the last valid ZZOPN command was issued. This ensures that only IMS transactions are recovered, and any updates made to the file by other programs while the file was closed to IMS are not touched.

In backward recovery, before-images are applied from the end of the trace file back to the last ZZOPN command (unless the date-time, if specified, is reached first). In forward recovery, after-images are applied from the last ZZOPN command to the date-time, if specified, or to the end of the trace file.

`PFILES=filename-1[,...,filename-n]`
Specifies that recovery of the designated files is to start or end at the last valid ZZOPN command.

PRINT Parameter

The PRINT keyword parameter has two uses. When you specify it with file recovery, you get a listing of the recovered records, with information about each record as described in its prefix area in the trace file (Figure 7-2). You can also use it without file recovery to get a complete listing of records updated during an IMS session. You do this by specifying one of the PRINT options together with FILES=NONE (or omit the FILES parameter). ZC#TRC prints the before- or after-images of all records in the trace file (depending on your RECOVERY-TYPE specification), with their prefixes.

`PRINT=ALPHA`
Specifies an alphanumeric listing.

`PRINT=BOTH`
Specifies an alphanumeric and hexadecimal listing.

`PRINT-HEX`
Specifies a listing in hexadecimal.

If the PRINT keyword parameter is omitted, no listing is provided. If you omit the PRINT keyword when FILES=NONE is indicated, explicitly or by default, no files are recovered and no records are printed.

If you specify any of the PRINT options, you must allocate a print file to the ZC#TRC job.

TRCFILE Parameter

When the object of recovery is to restore files after a system failure that has left the trace file open, this file must be closed before the ZC#TRC utility can carry out the type of recovery you have specified. The offline recovery utility itself can close the trace file only if the file is on disk or diskette; to cause it to do so, you must specify the TRCFILE keyword parameter. (To close a magnetic tape trace file left open by system failure, before submitting it to the ZC#TRC utility for use in recovery, follow the procedures detailed in 7.4.3.)

`TRCFILE=CLOSE`
Specifies that the ZC#TRC offline recovery utility is to close the disk/diskette trace file before carrying out the specified recovery operations.

Executing the Offline Recovery Utility

A standard job control stream for executing the offline recovery utility is illustrated in Figure 7-3.

```
 // JOB jobname,,min
[// OPTION DUMP]
[// DVC 20  // LFD PRNTR]
 // DVC logical-unit-no  // VOL vol-ser-no-1 [,vol-ser-no-n]
 // LBL file-id[,file-serial-no]  // LFD TRCFILE
[// DVC logical-unit-no  // VOL vol-ser-no // LBL file-id // LFD filename]
      .
      .
      .
[// DVC logical-unit-no  //  VOL vol-ser-no  //  LBL file-id  // LFD {AUDCONF}]
                                                                  {CONDATA}
[// DVC logical-unit-no  //  VOL vol-ser-no  //  LBL  file-id]
[// LFD load-lib-name]
 // EXEC ZC#TRC[,load-lib-name]
 /$
parameters
      .
      .
      .
 /*
 /&
 // FIN
```

Figure 7-3. Job Control Stream for Executing the Offline Recovery Program

You must specify the minimum main storage requirement, in hexadecimal, with the *min* parameter of the JOB statement. To compute this size, add the length of the ZC#TRC load module as listed in the output from the ZC#TRC linkage (Figure 7-1), plus four times the largest blocksize of the files to be recovered, plus 4096. The decimal result of this calculation should be rounded upward to the next 256-byte boundary and translated to its hexadecimal equivalent.

The device assignments, in the order they appear in Figure 7-3, are for:

- A printer

  Must be assigned to the job if you included the OPTION DUMP statement or the PRINT parameter.

- A tape, disk, or diskette trace file

  For a multivolume magnetic tape trace file, the VOL statement lists each volume serial number. The optional *file-serial-no* parameter of the LBL statement must match the volume serial number of the first volume in the file, whether or not that volume is the first one listed on the VOL statement.

  For a multivolume disk or diskette trace file, you must include a device assignment set for each volume, and you must define the volumes in the same sequence as they were defined at IMS start-up.

- The data files that are to be recovered

  These device assignments are omitted if you use the PRINT option without file recovery.

- The AUDCONF or CONDATA file

  Required if you are recovering the TOMFILE partition.

- The load library containing the ZC#TRC load module

  This device assignment is omitted if ZC#TRC is in the YLOD library on SYSRES. The *load-lib-name* parameter on the EXEC statement may also be omitted if ZC#TRC is in YLOD.

Typical job control streams for forward, backward, and quick recovery and for listing the contents of the trace file are shown in the following examples:

Example 1 - Forward Recovery

```
// JOB FORECOV,,C000
// DVC 90  // VOL TAPE01  //  LBL TRACE  //  LFD TRCFILE
// DVC 50  // VOL DISK01  //  LBL DATAFL1  //  LFD DATAFL1
       .
       .
       .
// DVC 51  // VOL DISK03  // LBL DATAFLX  // LFD DATAFLX
// DVC 52  // VOL IMSFIL  // LBL AUDCONF  // LFD AUDCONF
// OPTION DUMP
// DVC 20  // LFD PRNTR
// EXEC ZC#TRC
/$
RECOVERY-TYPE=FORWARD
FILES=ALL
DATE-TIME=89-09-30/14:22:00
PRINT=ALPHA
PFILES=DATAFL1,...DATAFLX
/*
/&
// FIN
```

The job stream in example 1 executes ZC#TRC for forward recovery of all configured data files and the TOMFILE. Because the PFILES and DATE-TIME parameters are both specified, recovery will start from the last valid ZZOPN command and end at the last transaction termination or rollback point before the date-time given. The PRINT parameter requests an alphabetic listing of the recovered records. The ZC#TRC program is loaded from YLOD.

Example 2 - Backward Recovery with Date-time

```
// JOB BAKRECOV,,D000
// OPTION DUMP
// DVC 20  // LFD PRNTR
// DVC 90  // VOL TAPE01,TAPE02,TAPE03
// LBL TRACE,TAPE01  // LFD TRCFILE
// DVC 50  // VOL DISK01  // LBL DATAFL1  // LFD DATAFL1
// DVC 50  // VOL DISK01  // LBL DATAFL2  // LFD DATAFL2
// DVC 51  // VOL DISK02  // LBL CONDATA  // LFD CONDATA
// DVC 52  // VOL DISK03  // LBL IMSLIB  // LFD IMSLIB
// EXEC ZC#TRC,IMSLIB
/$
RECOVERY-TYPE=BACKWARD
DATE-TIME=89-10-25/09:30:00
FILES=DATAFL1,DATAFL2,TOMFILE
PRINT=BOTH
/*
/&
// FIN
```

The job stream in example 2 executes ZC#TRC for backward recovery of DATAFL1, DATAFL2, and the TOMFILE, ending at the date-time specified. The PRINT parameter requests both an alphabetic and a hexadecimal listing. The IMSLIB parameter on the EXEC statement identifies the library from which ZC#TRC is to be loaded.

This example uses a multivolume magnetic tape trace file. Because the DATE-TIME parameter is specified, the trace file must be mounted first in the forward direction, then backward, and the VOL statement must list the volumes in the order in which they were created.

Example 3 - Backward Recovery without Date-time

```
// JOB FULRECOV,,F000
// OPTION DUMP
// DVC 20  // LFD PRNTR
// DVC 90  // VOL TAPE03,TAPE02,TAPE01
// LBL TRACE,TAPE01  // LFD TRCFILE
// DVC 50  // VOL DISK01  // DATAFL1  // FLD DATAFL1
// DVC 50  // VOL DISK01  // DATAFL2  // LFD DATAFL2
// EXEC ZC#TRC
/$
RECOVERY-TYPE=BACKWARD
FILES=DATAFL1,DATAFL2
PFILES=DATAFIL2
/*
/&
// FIN
```

The job stream in example 3 executes ZC#TRC for backward recovery of DATAFL1 and DATAFL2. The PFILES parameter specifies that DATAFL2 is to be rolled back to the last valid ZZOPN command; DATAFL1 is to be completely recovered. No listing is requested. Because the DATE-TIME parameter is not specified, the trace file is read in the backward direction only. Volumes of the multivolume magnetic tape trace file in this example must be mounted in the backward direction, starting with the last volume, and the VOL statement must list the volumes in the reverse of the order in which they were created.

Example 4 - Backward Recovery Using a Diskette Trace File

```
// JOB DISKET,,D000
// OPTION DUMP
// DVC 20 // LFD PRNTR
// DVC 50 // VOL DATA // LBL EMPLOYEE // LFD EMPLOY
// DVC 50 // VOL DATA // LBL PAYROLL // LFD PAYROLL
// DVC 130 // VOL TRACE1 // LBL IMS.TRACE // LFD TRCFILE
// DVC 131 // VOL TRACE2 // LBL IMS.TRACE // LFD TRCFILE
// DVC 132 // VOL TRACE3 // LBL IMS.TRACE // LFD TRCFILE
// EXEC ZC#TRC
/$
RECOVERY-TYPE=BACKWARD
FILES=EMPLOY,PAYROLL
/*
/&
// FIN
```

The job stream in example 4 executes ZC#TRC for backward recovery of the EMPLOY and PAYROLL files, using a multivolume diskette trace file. Unlike a multivolume tape trace file, the volumes must be defined in the order they were created, and all volumes must be online during the entire procedure.

Example 5 - Quick Recovery

```
// JOB QUICKIE,,C000
// OPTION DUMP
// DVC 20  // LFD PRNTR
// DVC 50  // VOL DISK01  // LBL TRACE  // LFD TRCFILE
// DVC 51  // VOL DISK01  // LBL DATAFL1  // LFD DATAFL1
     .
     .
     .
// DVC 52  // VOL DISK03  // LBL DATAFLX  // LFD DATAFLX
// DVC 53  // VOL DISK04  // LBL IMSLIB  // LFD IMSLIB
// EXEC ZC#TRC,IMSLIB
/$
RECOVERY-TYPE=QUICK
TRCFILE=CLOSE
FILES=ALL
PRINT=BOTH
/*
/&
// FIN
```

The job stream in example 5 executes ZC#TRC for quick recovery of all data files. The TRCFILE parameter closes the disk trace file, which was left open at the time of system failure. The PRINT parameter requests both an alphabetic and hexadecimal listing of recovered records.

Example 6 - Listing the Records in the Trace File

```
// JOB LIST,,C000
// DVC 20  // LFD PRNTR
// DVC 50  // VOL DISK01  // LBL TRACE  // LFD TRCFILE
// EXEC ZC#TRC
/$
RECOVERY-TYPE=FORWARD
PRINT=ALPHA
/*
/&
// FIN
```

The job stream in example 6 generates an alphabetic listing of records in the trace file, with no recovery of data files. Because RECOVERY-TYPE=FORWARD is specified, the records listed will be the after-images of all records updated since the beginning of the trace file.

7.4.3. Closing a Magnetic Tape Trace File

When a trace file is left open as a result of system failure, it must be properly closed before it can be used by the offline recovery or by the next online IMS session. Both the offline recovery utility and the online IMS program are capable of closing a disk or diskette trace file, but they cannot close a magnetic tape trace file.

An open tape trace file cannot be closed directly. The last volume of the file must be copied onto another tape volume that is at least as large as the original tape and then closed. The tape copy routine ZC#TCP performs this procedure. The new output file produced by ZC#TCP can then be used by the offline recovery program or, if you are using warm restart instead of offline recovery, assigned to the online IMS job.

Linking the Tape Copy Routine

The tape copy routine is provided in the form of an object module, ZE#COPY, in the YOBJ library file on SYSRES. You must execute the linkage editor to produce the load module ZC#TCP. In a DTF system (but not in a CDM system), you must include the tape sequential access method I/O module DD$T111.

A standard job control stream for linking the tape copy routine ZC#TCP is shown in Figure 7-4.

```
 1.  // JOB jobname
 2.  // DVC 20  // LFD PRNTR
 3. [// DVC 50  // VOL vol-ser-no  // LBL obj-lib-name  // LFD filename]
 4. [// DVC 51  // VOL vol-ser-no  // LBL load-lib-name  // LFD filename]
 5.  // WORK1
 6.  // EXEC LNKEDT
 7.  // PARAM OUT=filename
 8.  /$
 9.   LINKOP NOAUT,NOV
10.   LOADM ZC#TCP
11.   INCLUDE ZE#COPY,filename
12.  [INCLUDE DD#T111,$Y$OBJ]
13.   ENTER ZE#COPY
14.  /*
15.  /&
16.  // FIN
```

Figure 7-4. Job Control Stream for Linking the Tape Copy Routine

The disk device assignment sets identify the library files containing the ZE#COPY module (line 3) and the output load module (line 4). These may be omitted if the modules are stored in YOBJ and YLOD on your SYSRES volume. The PARAM statement (line 7) specifies where the output module is to be stored. The printer device assignment (line 2) and the WORK jproc (line 5) are required by the linkage editor.

Lines 9 through 13 are linkage editor control statements, which must start in column 2 or beyond. The LINKOP statement (line 9) directs the linkage editor not to automatically include modules or phases. The LOADM statement (line 10) assigns the name ZC#TCP to the output module. This statement will cause unresolved EXTRNs to be generated; these should be ignored. Lines 11 and 12 include the tape copy module and the tape SAM I/O module in the output load module. Line 13 names ZE#COPY as the entry point for the load module.

Executing the Tape Copy Routine

A standard job control stream for executing the tape copy routine is shown in Figure 7-5. The ZC#TCP program requires 8000 hexadecimal bytes of main storage, specified on the JOB card. Assignment of a printer is also required. A load library need not be assigned if ZC#TCP is in YLOD.

```
// JOB jobname,,8000
// DVC 20  // LFD PRNTR
// DVC 90  // VOL vol-ser-no  // LBL file-id  // LFD INPUT
// DVC 91  // VOL vol-ser-no  // LBL file-id  // LFD OUTPUT
[// DVC 50  // VOL vol-ser-no  // LBL file-id  // LFD load-lib-name]
// EXEC ZC#TCP[,load-lib-name]
/&
// FIN
```

Figure 7-5. Job Control Stream for Executing the Tape Copy Routine

ZC#TCP closes the new file produced by this run when it detects an error condition in the input volume. The following message is displayed on the system console:

```
DEVICE=420    STATUS=2600    SENSE=08C0    TAPE FAULT=xUxx
```

This message should be ignored.

Section 8
IMS Statistical Reporting

8.1. Statistical Reporting Functions

As an IMS administrator or analyst, you will find it useful to know what activities take place on your system and where they occur. To assist you, IMS provides a statistical file, STATFIL, for recording system activity and an offline print program to print that file. The statistical file print program is available in both single-thread and multithread IMS.

If you want IMS to create a statistical file, you must allocate and initialize this file at configuration time and assign it in the IMS start-up job control stream. Once the file has been created, you can use the ZSTAT transaction to write IMS statistical data in it. The ZSTAT transaction is documented in the *IMS Operations Guide,* UP-12027. User-written action programs can also put user-originated statistics into the file.

If you include the STATS=YES parameter in the OPTIONS section at configuration, IMS automatically schedules ZSTAT at shutdown time and records statistics for the entire IMS session.

The offline statistical file print program, ZC#ZSF, prints the data recorded by ZSTAT in a neatly formatted report. This report contains statistics on:

- Files
- Programs
- Transactions
- Terminals

Normally, statistics are accumulated from the beginning of an IMS session. When you use PARAM RESTART at IMS start-up in a multithread system, statistics are carried forward from the preceding session.

The statistical print program does not print user-generated statistics; you must write your own program to do this.

8.2. Statistical Data Recorded by ZSTAT

IMS statistics record some of the activities that take place in an IMS session. They enable you to analyze the way files, transactions, programs, and terminals are being used.

8.2.1. File Statistics

For each user file accessed during the session, the offline print program prints:

- The file name and whether it was open or closed when ZSTAT was executed
- The number of times the file was accessed during the session
- The number of times the file was updated during the session

The total number of file accesses and file updates in the session are also printed. Figure 8-1 shows an example of file statistics printed by ZC#ZSF.

```
10/05/89    15:51:02

FILE        STAT        ACCESSES    UPDATES
CUSTFIL     OPEN               0          0
AUDFILE     OPEN               0          0
PRODFIL     CLSD               0          0
STATFIL     OPEN              15         15
NAMEREC     OPEN               0          0

    TOTALS                    15         15
```

Figure 8-1. File Statistics Printed by the STATFIL Print program

8.2.2. Program Statistics

For each program executed during the session, the offline print program prints:

- The program name and whether it was up or down when ZSTAT was executed
- The program type -- serial, reentrant, or shared (multithread only)
- Whether or not the program is resident
- Whether this is a subprogram
- The number of times the program was accessed during the session
- The number of times the program was loaded during the session

The offline statistical print program also prints the total number of program accesses and program loads that took place during the session. Figure 8-2 shows an example of program statistics printed by ZC#ZSF.

```
10/05/89    15:51:09

PROGRAM     STAT        TYPE          RES    SUB    ACCESSES    LOADED
ZM#BEGOO     UP         SERIAL         N      N           1         1
ZU#OPNOO     UP         RE-ENTRANT     Y      N           0         0
ZM#FLEOO     UP         SERIAL         N      N           1         1
ZM#TER       UP         SERIAL         N      N           1         1

   TOTALS                                                 3         3
```

Figure 8-2. Program Statistics Printed by the STATFIL Print Program

8.2.3. Transaction Statistics

For every transaction that took place during the session, ZC#ZSF prints:

- The transaction name
- How many times the transaction was accessed
- The number of input and output messages it processed
- The number of file accesses it made
- The number of remote accesses by this transaction (for DDP only)
- The name of the remote system (locap-name) where the transaction is processed (DDP only)

The following totals are also printed:

- Number of transaction accesses
- Number of input and output messages
- Number of file accesses
- Number of remote accesses

Figure 8-3 shows an example of transaction statistics printed by ZC#ZSF, and Figure 8-4 illustrates transaction statistics for a DDP environment.

10/05/89	15:51:20			
TRANSACT	ACCESSES	INPUT	OUTPUT	FILE-ACC
DISP	0	0	0	0
SWTCH	0	0	0	0
ZSTAT	1	4	4	8
TOTALS	1	4	4	8

Figure 8-3. Transaction Statistics Printed by the STATFIL Print Program

10/05/89	15:51:20				
TRANSACT	ACCESSES	INPUT	OUTPUT	REM-AC	TO
DISP	0	0	0	0	
INVEN	3	4	4	2	IMSB
ZSTAT	1	4	4	0	
DISP	0	0	0	0	
TOTALS	4	8	8	2	

Figure 8-4. Transaction Statistics for DDP Users

8.2.4. Terminal Statistics

For every terminal that was active during the session, the statistical display program prints:

- The terminal-id
- The transaction code of the current transaction, if any
- Its status at the time ZSTAT was executed: up, physically down, logically down, in test mode, or in hold status
- The number of input and output messages processed
- The number of transactions processed
- The number of terminal commands processed
- The number of input and output characters processed

In addition, ZC#ZSF prints the following totals:

- Number of slots remaining (for global networks only)
- Number of input and output messages processed
- Number of transactions executed
- Number of terminal commands processed
- Number of input and output characters processed
- Length of the longest input message processed and the id of the terminal where it occurred
- Length of the longest output message processed and the id of the terminal where it occurred.

Figure 8-5 shows an example of terminal statistics printed by the STATFIL print program.

```
10/05/89    15:51:24

TERM   STAT  TRANSCODE   IN-MSG   OUT-MSG   TRANS   COMM   IN-CHAR   OUT-CHAR

TRM1     DP                  0         0       0      0         0          0

TRMC     UP    ZSTAT         2         2       1      1        39       3271

TRMA     UP                  1         1       1      0        15        250

 TOTALS                      3         3       2      1        54       3521

 MAX IN=    24(TRMC)     MAX-OUT=  1660(TRMC)
```

Figure 8-5. Terminal Statistics Printed by the STATFIL Print Program

8.3. Configuration and Start-up Requirements for Statistical Reporting

If you wish to create a statistics file, you must allocate and initialize it at configuration time. You allocate the statistics file with the IMSFIL or IMSFIL4 parameter of the IMSCONF jproc (4.2.8), and initialize it with the INIT parameter (4.2.9).

You must also assign the file in the job control stream at IMS start-up.

If you want statistical data to be recorded at shutdown time, specify the STATS parameter in the OPTIONS section of your input to the configurator (4.3.3).

8.4. Recording Statistical Data during Online Processing

There are two ways you can write records on the statistical file during online processing. One is by using the ZSTAT transaction, the other is through user-written programs that put your own statistical data in the file.

You must remember, however, that the offline print program, ZC#ZSF, prints only the data recorded by ZSTAT. It ignores statistics recorded by user-written programs.

ZSTAT generates statistics and writes them into the file every time you request it to do so. (You can also direct ZSTAT output to the master terminal or to an auxiliary device.)

ZSTAT allows you to choose which data you want to record on the statistical file. If you specify ZSTAT ALL, FILE or ZSTAT ALL, FONLY, all the data listed in 8.2 is recorded. If you enter only ZSTAT, with no parameters, you receive a menu on your screen. This enables you to specify that only the data you select is to be recorded in the file.

For a complete description of the ZSTAT transaction, refer to the *IMS Operations Guide,* UP-12027.

If you include the STATS parameter in the OPTIONS section at configuration, IMS automatically schedules ZSTAT at shutdown time and records all statistics from the IMS session in STATFIL.

8.5. Printing the Statistical File Offline (ZC#ZSF)

To print the statistical file offline, you run the ZC#ZSF program. This program retrieves only the statistics you recorded by executing the ZSTAT transaction. It ignores any statistical data placed in the file by user-written programs.

The ZC#ZSF program does not require any parameters. Figure 8-6 shows the job control stream for executing ZC#ZSF.

```
// JOB jobname
// OPTION DUMP
// DVC 20  // LFD PRNTR
// DVC logical-unit-no // VOL vol-ser-no
// LBL file-id // LFD STATFIL
// EXEC ZC#ZSF
/&
// FIN
```

Figure 8-6. Job Control Stream for Executing the Statistical File Print Program

When the program finishes execution, it displays the message:

```
END OF FILE REACHED.  ZC#ZSF  TERMINATED
```

8.6. Contents of the Statistical Data File

If you wish to write your own program for recording or printing statistical data in the STATFIL file, refer to Tables 8-1 through 8-9 for the file layout. (The data your program writes in the file is ignored by ZC#ZSF, so you must also write your own statistical print program.) The statistics file is a sequential unkeyed MIRAM file with variable-length records. The first four bytes of each record are not easily accessible when you are using COBOL, so the record length field is repeated in the data portion of the record. The STATFIL in configured with BFSZ = 1536 and RCSZ = 1102.

Table 8-1. Contents of Date/Time Stamp Record

Field Name	Bytes	Description
Length	0-1	Record length X'0016'
Record Control Block	2	Indicates record has been logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Unused	6-7	Reserved for future use
Identifier	8-9	Defines the following record: DF for file statistics DP for program statistics DT for transaction statistics DD for transaction statistics created by DDP users DC for terminal statistics
Date	10-15	In the format DDMMYY
Time	16-21	In the format HHMMSS

Table 8-2. Contents of File Statistics Record

Field Name	Bytes	Description
Length	0-1	Record length (number of files * 32) + 10
Record Control Byte	2	Indicates if record was logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Repeat Count	6-7	Number of times "file-name" through "updates" is repeated
Identifier	8-9	FI, to identify record as file statistics
File-name	10-16	File name
Unused	17	Reserved for future use
Status	18-21	OPEN Open CLSD Closed XXXX Invalid file name
Type	22-25	MIRAM MIRAM SAM SAM ISAM ISAM PRNT printer SAT SAT DAMR DAMR
Accesses	26-33	Number of times file was accessed
Updates	34-41	Number of times file was updated

Table 8-3. Contents of File Totals Record

Field Name	Bytes	Description
Length	0-1	Record length X'001A'
Record Control Byte	2	Indicates record was logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Unused	6-7	Reserved for future use
Identifier	8-9	FT File Totals Record
Total-accesses	10-25	Total number of file accesses
Total-updates	18-25	Total number of file updates

Table 8-4. Contents of Program Statistics Record

Field Name	Bytes	Description
Length	0-1	Record length (number of programs * 30) + 10
Record Control Byte	2	Indicates record was logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Repeat Count	6-7	Number of times "program-name" through "loads" is repeated
Identifier	8-9	PR identifier for program statistics record
Program-name	10-17	Program name
Status	18-19	UP Up DN Down XX Invalid program name
Type	20-21	SH Shared SE Serial RE Reentrant
Resident	22	Y Yes N NO
Subprogram	23	Y Yes N NO
Accesses	24-31	Number of times program was accessed
Loads	32-39	Number of times program was loaded

Table 8-5. Contents of Program Totals Record

Field Name	Bytes	Description
Length	0-1	Record length X'001A'
Record Control Byte	2	Indicates record was logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Unused	6-7	Reserved for future use
Identifier	8-9	PT Program Totals
Total-accesses	10-17	Number of times the programs were accessed
Total-Loads	18-25	Total number of file updates

Table 8-6. Contents of Transaction Statistics Record

Field Name	Bytes	Description
Length	0-1	Record length (number of transactions * 52) + 10
Record Control Byte	2	Indicates if record was logically deleted
Unused	3	Not used by data management
Record length	4-5	Adjusted record length (Length - 4)
Repeat count	6-7	Number of times "transaction-code" through "locap-name" is repeated
Identifier	8-9	TR Transaction Statistics Record TD Transaction Statistics Record in DDP system
Transaction Code	10-17	Transaction code
Accesses	18-25	Number of times transaction was accessed XXXXXXXX Invalid transaction code
Input-messages	26-33	Number of input messages processed by this transaction
Output-messages	34-41	Number of output messages processed by this transaction
File-accesses	42-49	Number of user file accesses made by this transaction
Remote-accesses	50-57	Number of remote accesses made by this transaction (DDP only)
Locap-name	58-61	Name of remote system (DDP users only)

Table 8-7. Contents of Transaction Totals Record

Field Name	Bytes	Description
Length	0-1	Record length X'0032'
Record Control Byte	2	Indicates if record was logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Unused	6-7	Reserved for future use
Identifier	8-9	TT Transaction Totals (IDM DDP system) TP Transaction Totals DDP system
Total-accesses	10-17	Total number of accesses
Total-input-messages	18-25	Total number of input messages
Total-output-messages	26-33	Total number of output messages
Total-file-accesses	34-41	Total number of file accesses
Total-remote accesses	42-49	Total number of remote accesses (DDP only)

Table 8-8. Contents of Terminal Statistics Record

Field Name	Bytes	Description
Length	0-1	Record length (number of programs * 52) + 10
Record Control Byte	2	Indicates if record was logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Repeat Count	6-7	Number of times "terminal-id" through "output-characters" is repeated
Identifier	8-9	TE Terminal Record
Terminal-id	10-13	Terminal identification
Status	14-15	UP Up DL Down logically DP Down physically TM Test mode HL Hold XX Invalid terminal-id
Transaction Code	16-23	Terminal input transaction code or $$SOFF (global networks only)
Input-messages	24-29	Number of input messages processed
Output-messages	30-36	Number of output messages processed
Transactions	37-41	Number of transactions processed
Commands	42-45	Number of commands processed
Input-characters	46-53	Number of input characters processed
Output-characters	54-61	Number of output characters processed

Table 8-9. Contents of Terminal Totals Record

Field Name	Bytes	Description
Length	0-1	Record length X'0050'
Record Control Byte	2	Indicates if record was logically deleted
Unused	3	Not used by data management
Record Length	4-5	Adjusted record length (Length - 4)
Unused	6-7	Reserved for future use
Identifier	8-9	TO Terminal Totals
Slots	10-13	Number of slots remaining (in global network)
Total-input-messages	14-19	Total number of input messages
Total-output-messages	20-26	Total number of output messages
Total-transactions	27-31	Total number of transactions processed
Total-commands	32-35	Total number of commands processed
Total-input-characters	36-45	Total number of input characters processed
Total-output-characters	46-55	Total number of output characters processed
Max-input-characters	56-63	Length of single largest input message
Max-input-terminal	64-67	Terminal-id at which the single largest input message occurred
Max-output-characters	68-75	Length of single largest output message
Max-output-terminal	76-79	Terminal-id at which the single largest output message occurred

Appendix A
Statement Conventions

A.1. General Rules

Throughout this document, certain conventions are observed for statement and parameter formats. General rules with examples pertaining to these conventions follow. Specific rules for coding input to the IMS configurator are given in A.2. Additional coding conventions are described in the appropriate sections of this manual; you should also refer to the *Job Control Programming Guide*, UP-9986, for coding conventions relating to job control statements and procedures (jprocs).

- Uppercase words, codes, and letters are coded exactly as shown, as are commas and equal signs. Braces { } and brackets [ ] are never coded.

- Lowercase letters and words are generic terms representing information that must be supplied by the user. Such terms may contain acronyms and hyphens for readability.

  Examples

  ```
  DDRECORD=record-name

  program-name
  ```

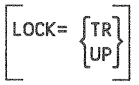
- Information presented within braces represents alternate choices, of which only one may be chosen.

  Example

  ```
  EDIT= {table-name}
        {FCCDICE  }
        {C        }
        {NONE     }
  ```

- Information presented within brackets represents optional entries that may be coded or omitted, depending on individual system requirements. Braces within brackets signify that one of the entries within braces must be chosen if the bracketed parameter is coded.

Example



- A keyword parameter consists of a word or code immediately followed by an equal sign (=), which in turn is followed immediately by a response, or specification. Keyword parameters, although they appear in the format delineations in alphabetic order, may be coded in any convenient order.

Example

```
DATE-TIME=yy-mm-dd/hh:mm:ss
```

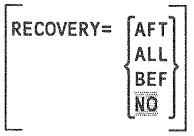
- A positional parameter is presented in lowercase letters and is not followed by an equal sign (=). Positional parameters are presented before keyword parameters in the format delineation and must be coded before any keyword parameters.

Example

```
trans-code
```

- An optional parameter with a list of optional specifications may have a default assumption supplied by IMS when the parameter is omitted by the user. When the default value assumed by IMS occurs in the format delineation, it is printed on a shaded background. If IMS performs some other default action when an optional parameter is omitted, this is explained in the parameter description.

Example



- An ellipsis (a series of three periods) occurring in a format delineation indicates the possibility of coding a variable number of repeated entries. The ellipsis itself is never coded.

Example

```
FILES=filename-1[,...,filename-n]
```

A.2. Rules for Coding Configurator Input

Rules and examples for coding input to the configurator follow. These are in addition to the general rules presented in A.1.

- Section names input to the IMS configurator are presented in the leftmost position of the format delineation, in uppercase. They are coded exactly as shown and must begin in column 1.

  Example

  ```
  NETWORK [BATCH= {n  }]
                  {NO }
                  {YES}
          CONFID=n
          [NAME=network-name]
          [PASSWORD=password-name]
  ```

- Parameters are presented to the right of the section name and are stacked vertically. A positional parameter, if it occurs, is presented before all keyword parameters; the latter are presented in alphabetic order for ease of reference in the illustration.

  Example

  ```
  TRANSACT  trans-code
            [ACTION=program-name]
            [URGENT= {NO }]
                     {YES}
  ```

- Parameters are separated from each other and from the section name by one or more spaces. Commas are not used to delimit parameters. If a positional parameter appears in the format delineation, it must always be included and must be the first parameter coded.

  Example

  ```
  TRANSACT CHG ACTION=CHANGE
  ```

- When coding multiple parameters for a section of input to the configurator, no input may be coded beyond column 71. When continuation is necessary, or when it is desirable for readability, whole parameters may be coded on continuation lines. None may be split, nor may any begin in column 1. Column 72 remains blank.

  Example

  ```
  OPTIONS  RECLOCK=YES   FUPDATE=YES
   UNSOL=YES  UNIQUE=RES
   INTLIST=NO  RECOVERY=ALL
       RESEND=YES
  ```

- Multiple specifications, or subparameters, may be coded for certain keyword parameters if so shown in the format delineation. These subparameters, when coded, are separated from each other by commas, without intervening spaces, if indicated in the format.
  Example

```
ACTION PAYROLL  FILES=MAINFLE,PAYDAY,ACTREC,EMPLYS
```

- Subparameters may span two or more lines by repeating the keyword on continuation lines with the additional specifications.

  Example

```
ACTION UPDATE MAXSIZE=1000 OUTSIZE=204
  EDIT=% FILES=MAINFIL,PAYROLL,ACCTREC,ACCTPAY
  FILES=STATS,EMPLOYS
```

- Keyword parameters, the response to which is YES or NO, may alternatively be specified in the abbreviated form Y or N.

  Examples

```
FUPDATE=YES

FUPDATE=Y
```

- With the exception of the EDIT, RCHAR, and UCHAR keywords, positional and keyword parameter specifications may use only the following characters:

  - Alphabetic: The uppercase letters A through Z
  - Numeric: 0 through 9
  - Special letters: The characters ? $ # @

  The EDIT, RCHAR, and UCHAR specifications may use any characters from this list and any additional special characters available on your terminals.

  If Katakana support is configured, Katakana characters may be used in transaction codes, UNIQUE lexicons, defined file names, data definition records, routing characters, and urgent priority characters.

- The following words are reserved and must not be used in configurator input as the names of files, transactions, actions, or programs:

  - IMS internal file names: NAMEREC, AUDFILE, AUDCONF, CONDATA, TRCFILE, TOMFILE, STATFIL, PRNTR, PRNTR1, PRNTR2, PRNTR3, PRNTR4

  - IMS transaction codes, lexicon names, and terminal commands: CHTBL, DITBL, DLMSG, DLOAD, JI, SWTCH, ZSTAT, ZZALT, ZZBTH, ZZCLS, ZZCNC, ZZCNS, ZZDEQ, ZZDWN, ZZHLD, ZZHLT, ZZMCH, ZZMSG, ZZNMG, ZZNRM, ZZOPN, ZZPCH, ZZRDY, ZZRSD, ZZSCH, ZZSHD, ZZTMD, ZZTST, ZZUP

    Note: *OPEN cannot be used as a transaction code but may be used as a lexicon name.*

  - Module names of IMS-supplied action programs

  - The alphabetic characters ZZ are not allowed in the first two spaces of an action program's input message.

Appendix B
Estimating Main Storage Requirements for IMS

B.1. General

This appendix provides guidelines for estimating the main storage requirements for executing single-thread and multithread IMS systems under OS/3. The approximate requirements documented here are subject to change as new features are added to IMS in future releases and should be expected to be revised upward. (They are accurate to within ± 10%.)

B.2. Job Region Size for Single-Thread Systems

IMS can be configured to run under OS/3 in a job region as small as 54,062 bytes. The size of the job region is specified on the JOB job control statement when executing IMS.

You have various options, when configuring IMS, that increase the size of the required job region. Figure B-1 illustrates the calculations for the major components that comprise a small system; the subsequent tables and text explain how user options can increase the size of some of those components.

Components	Bytes	
	DTF System	CDM System
Required program modules and tables	32,938	33,011
Named record file I/O area (block size of named record file)	2,560	2,560
Optional program modules. See Table B-1.	0	0
Optional tables (10 terminals). See Table B-2.	3,000	3,000
Optional files (3 user-specified ISAM DTFs or MIRAM RIBs). See Table B-2.	1,308	468
Optional resident action programs. See B.2.3.	0	0
Action program storage pool. See B.2.4.	14,000	16,000
Audit file I/O area. See B.2.5.	256	256
Trace file I/O area. See B.2.6.	0	0
Total	54,062	55,295

Figure B-1. Sample Calculation of Job Region Size Required by a Near-Minimum Single-Thread IMS System

B.2.1. Optional Program Modules

Table B-1 lists the increments in the size of a single-thread IMS load module that result from specifying appropriate configurator keywords for certain optional functions. Refer to Section 4 for configuration details.

Table B-1. Effects of Configurator Options on the Size of a Single-Thread IMS Load Module

Function	Specifications		Increment in Size (bytes)	
	Section	Keyword	DTF System	CDM System
Batch processor	NETWORK	BATCH=[1 / YES]	17,744	17,836
Global network		CUP=locap-name	1,332	1,332
Continuous output	OPTIONS	CONTOUT=YES	1,361	1,361
Downline loading		DLLOAD=YES	1,584	1,584
DMS interface		DMS=YES	1,940 (3,796 if defined record management is configured)	1,940
Fast load feature		FASTLOAD=YES	4,856	4,856
File updating		FUPDATE=YES	628	636
Interrupted LIST command		INTLIST=YES	None	None
Console transaction processing		OPCOM=YES	6,064 (1,298 if master terminal is defaulted to console)	6,064
Record locking across actions		RECLOCK=YES	1,576	1,576
Offline recovery		RECOVERY=[AFT / ALL / BEF]	1,316	1,100
Resend command		RESEND=YES	None	None
Screen format services		SFS=n	4,299	4,299
Recording statistical data at shutdown		STATS=YES	None	None
User-written subprograms		SUBPROG=YES	416	416
Audit of output messages		TOMFILE=YES	586	586
Tracing of output messages		TOMTRCE=YES	None (TOMFILE=YES must be specified.)	None (TOMFILE=YES must be specified.)
Resident UNIQUE		UNIQUE=RES	1,808	1,808
Unsolicited output		UNSOL=YES	2,337	2,337

continued

Table B-1. Effects of Configurator Options on the Size of a Single-Thread IMS Load Module (cont.)

Function	Specifications		Increment in Size (bytes)	
	Section	Keyword	DTF System	CDM System
Common storage area	FILE	CAFILE=YES	1,674	1,421
DAM files		FILETYPE=DAMR	608	Not available
MIRAM files		FILETYPE=DMRAM	2,824	3,320
ISAM files		FILETYPE=ISAM	1,528	Not available
SAM disk files		FILETYPE=SAMD	292	Not available
SAM tape files (DTF)		FILETYPE=SAMT	2,238	Not available
Sequential tape files (CDM)		FILETYPE=TMRAM	Not available	3,232
Expanded input editing	ACTION	EDIT=tablename	2,532	2,532
Resident defined record management	DRCRDMGT	RESIDE=YES	1,088	1,088
File updating through defined record management		UPDATE=YES	None	None

B.2.2. Optional Control Tables

Each type of control table generated by IMS has a characteristic size to be used in estimating your main storage requirements for single-thread IMS. Table B-2 lists these and indicates the factor by which to multiply each.

Table B-2. Single-Thread Control Table Sizes

Table Type	Table Size (Bytes)	Multiply by Number of:
Terminal control	300	Terminals (TERMS=n in NETWORK section)
Transaction control	28	Transaction codes (TRANSACT sections)
Action control	64	Actions (ACTION sections)
Program control	40	Action programs (PROGRAM sections)
DTFIS file control	436	ISAM files
DTFMT file control	284	SAM tape files
DTFSD file control	284	SAM disk files
DTFDA file control	284	DAM files
DTFMI file control	388	MIRAM files
CDIB file control	48	MIRAM disk files and sequential tape files
RIB file control	96	

B.2.3. Optional Resident Action Programs

Into your total main storage requirements you must add the size of each action program for which you have specified RESIDE=YES in the PROGRAM section of your configurator input. Further, if the action program is serially reusable (TYPE=SER), you must round this program size upward to the nearest multiple of the basic 512-, 1024-, or 2048-byte segment of main storage protected in your hardware configuration. The maximum-allowable-size action program in a single-thread configuration is 65K bytes; the maximum size for a resident subprogram is also 65K bytes.

B.2.4. Action Program Main Storage Pool Requirement

Each action has a unique main storage pool space requirement; the largest of these requirements must be determined in order to estimate the size you will require for the main storage pool in a single-thread IMS system. (Because only one action is processed at a time in single-thread mode, there is no reason for the main storage pool to exceed the largest requirement.)

Each action's main storage pool requirement is the sum of three figures:

1. the size of the largest nonresident action program to be called by the action -- specified with the MAXSIZE keyword parameter in the ACTION section (the maximum allowable size is 65K bytes), plus

2. the size of the activation record, plus

3. the size of the action's I/O area.

Figure B-2 contains a schedule for calculating the size of an action's activation record and indicates the items to be summed. The result of calculations in Part 2 of the figure is entered as the sixth item in Part 1 (if the action accesses a defined file).

The I/O area size must allow room for all of the data blocks that an action can acquire concurrently and is at least 1280 bytes if UNIQUE is specified for the action. Each data block's length is that specified via the BLKSIZE keyword parameter for its file, input in the FILE section to the configurator, and rounded up to a multiple of 256 bytes. A data block is acquired and released during the execution of a GET or INSERT file I/O function call. It is acquired by a SETL or GETUP function call and released by the corresponding ESETL, PUT, or DELETE function call.

Activation Record Item	Size (bytes)
Program information block (PIB) 144 bytes + n	See note.
Input message (actual length, excluding DICE sequences	
Output message length (value of OUTSIZE parameter)	
Continuity data area (value of CDASIZE parameter)	
Work area (value of WORKSIZE parameter)	
Defined record area (If DFILE parameter specified. From bottom line of Part 2.)	
(Sum) Total for activation record:	

Note:

n is the number of decimal bytes specified with the SHRDSIZE keyword parameter if the action is a shared code COBOL action program (4.3.8).

a. Part 1

Defined Record Area Item	Size (bytes)
Basic segment	350
2 x sum of lengths of all items in defined record	
Number of defined record items	
Length of longest identifier	
Length of longest source record key	
Length of largest source record (primary part)	
Length of largest source record (supplementary part)	
(Sum) Total for defined record area: (Enter as sixth item of Part 1)	

b. Part 2

Figure B-2. Schedule for Calculating Size of the Activation Record Required for an Action in Single-Thread IMS

B.2.5. AUDCONF Audit File I/O Area

The AUDCONF area is used to contain a single before-look. It must be large enough to contain the largest before-look for any file that can be updated. Before-look size is computed differently for indexed and nonindexed files and fixed- and variable-length records:

- ISAM - fixed-length records; indexed MIRAM

```
60 + RECSIZE value + KEYLEN value
```

- ISAM - variable-length records

```
60 + BLKSIZE value + KEYLEN value
```

- Nonindexed MIRAM

```
60 + RECSIZE value
```

- DAM

```
60 + BLKSIZE value
```

In each case, the computed size must be rounded upward to a multiple of 256.

B.2.6. TRCFILE Trace File I/O Area

The TRCFILE area is used to contain one or more before-looks and after-looks. It must be large enough to contain the largest look for any file that can be updated. Look size is computed differently for indexed and nonindexed files and for fixed- and variable-length records. This result is rounded upward to a multiple of 256:

- ISAM - fixed-length records; indexed MIRAM

```
114 + RECSIZE value + KEYLEN value
```

- ISAM - variable-length records

```
114 + BLKSIZE value + KEYLEN value
```

- Nonindexed MIRAM

```
114 + RECSIZE value
```

- DAM

```
114 + BLKSIZE value
```

B.3. Job Region Size for Multithread Systems

The job region size needed to run multithread IMS under OS/3 can vary considerably depending on the requirements of the user. Certain modules and tables are fixed and necessary for any user to run IMS. However, many of the options will increase the required size by various amounts. An action program main storage pool is always required; B.3.4 describes how to calculate this requirement. In general, the pool requirement for two or three concurrent threads is between 50K and 60K bytes.

Figure B-3 provides an example of calculations of main storage required for a multithread IMS system.

Components	Bytes	
	DTF System	CDM System
Required program modules and tables	67,992	63,272
Named record file I/O area (block size of named record file)	6,144	6,144
Optional tables (6 terminals, 2 transactions, 2 actions, 4 programs). See Table B-4.	3,304	3,304
Optional files (4 user-specified ISAM DTFs or MIRAM RIBs). See Table B-4.	1,824	624
Resident action programs. See B.3.3.	0	0
Input message buffer (1/2 (max screen x no. of terminals))	3,000	3,000
Action program storage pool (approx.) See Figure B-4.	60,000	60,000
Total (approx.)	142,264	136,344

Figure B-3. Sample Calculation of Job Region Size Required by a Multithread IMS System

B.3.1. Optional IMS Features

Table B-3 summarizes the optional IMS features that may be specified for a multithread system and indicates the sizes of the additional modules to be included. Refer to Section 4 for configuration details.

Table B-3. Effects of Configurator Options on the Size of a Multithread IMS Load Module

Function	Specifications		Increment in Size (bytes)	
	Section	Keyword	DTF System	CDM System
Batch processor	NETWORK	BATCH=1 BATCH=2 BATCH=3 BATCH=4	13,754 16,382 19,010 21,638	13,790 16,402 19,014 21,626
Global network		CUP=locap-name	1,560	1,560
Continuous output	OPTIONS	CONTOUT=YES	1,672	1,672
Downline loading		DLLOAD=YES	1,792	1,792
DMS interface		DMS=YES	3,516	3,516
Fast load feature		FASTLOAD=YES	7,584	7,584
File updating (1)		FUPDATE=YES	1,248	1,248
Console transaction processing		OPCOM=YES	5,856 (Same if master terminal is defaulted to console)	5,856
Offline recovery (2)		RECOVERY= {AFT, ALL, BEF}	1,552	1,376
Resend command		RESEND=YES	None	None
Screen format services		SFS=n	5,040	5,040
Recording statistical data at shutdown		STATS=YES	None	None
Edited snap dump		SNAPED=YES	2,880	2,880
User-written subprograms		SUBPROG=YES	752	752
Auditing of output messages		TOMFILE=YES	624	624

continued

Table B-3. Effects of Configurator Options on the Size of a Multithread IMS Load Module (cont.)

Function	Specifications		Increment in Size (bytes)	
	Section	Keyword	DTF System	CDM System
Resident UNIQUE	OPTIONS (cont.)	UNIQUE= {RES / YES}	1,896	1,896
Transient UNIQUE		UNIQUE=TRAN	412	412
Unsolicited output		UNSOL=YES	2,296	2,296
Common storage area	FILE	CAFILE=YES	1,840	1,840
ISAM files		FILETYPE=ISAM	3,040	Not available
MIRAM files		FILETYPE=DMRAM	3,546	3,890
DAM files		FILETYPE=DAMR	1,640	Not available
SAM disk files		FILETYPE=SAMD	414	Not available
SAM tape files (DTF)		FILETYPE=SAMT	1,656	Not available
Sequential tape files (CDM)		FILETYPE=TMRAM	Not available	3,938
Printer files		FILETYPE=PRNT	978	978
Continuity data area ③ (or UNIQUE used)	ACTION	CDASIZE=n	550	550
Editing of input messages		EDIT=tablename	3,184	3,184
Use of defined record management (implied with UNIQUE)	DRCRDMGT		10,432	10,432

Notes:

① An I/O area must also be allocated for the AUDFILE audit file (B.3.5).

② An I/O area must also be allocated for the TRCFILE trace file (B.3.5).

③ An I/O area must be allocated for the CONDATA continuity data file (B.3.5).

B.3.2. Optional Tables and User File DTFs

Each type of control table generated by IMS has a characteristic size to be used in estimating your main storage requirements for multithread IMS. Table B-4 lists these and indicates the factor by which to multiply each.

Table B-4. Multithread Control Table Sizes

Table Type	Table Size (Bytes)	Multiply by Number of:
Terminal control	412	Terminals (TERMS=n in NETWORK section)
Transaction control	40	Transaction codes (TRANSACT sections)
Action	104	Actions (ACTION sections)
LOCAP DOP only	20	Locaps (LOCAP section)
Program control	40	Action programs (PROGRAM sections)
DTFIS file control	456	ISAM files
DTFMT file control	284	SAM tape files
DTFSD file control	284	SAM disk files
DTFDA file control	284	DAM files
DTFMI file control	388	MIRAM files
CDIB file control	48	MIRAM disk files and sequential tape files
RIB file control	96	

B.3.3. Optional Resident Action Programs

To the total space requirement must be added the size of each action program for which you specify RESIDE=YES in the PROGRAM section of your configurator input. Furthermore, if the action program is shared code (TYPE=SHR) or serially reusable (TYPE=SER), you must round the program size upward to the nearest multiple of the basic 512-, 1024-, 2048-, or 4096-byte main storage segment given protection in your hardware configuration.

B.3.4. Main Storage Pool Calculations

In multithread IMS, main storage management uses the main storage remaining from the end of the second phase (shown in the listing for the configuration link stream) to the beginning of the resident action program area for the main storage pool. Some of the areas within the main storage pool can be shared by concurrently running threads -- others cannot.

Unsharable areas include the following:

- The thread control block
- The activation record
- Record locks

Sharable areas include:

- Loaded action programs with their program control blocks;
- Data definition records
- The lexicon area (required for UNIQUE)
- The ISAM file I/O areas

The main storage pool must be large enough to handle the requirements of all concurrently active actions. To obtain the main storage pool requirement, you must calculate the main storage needed for the largest possible thread and multiply the result by the number of threads that may be active at one time. The total main storage pool requirement is then added to the main storage needed for the IMS coding (Figure B-3) to obtain the job region size.

Figure B-4 provides a schedule for calculating the requirements of the largest possible thread.

Entry and Remarks	Size (bytes)
Thread control block	400
Size of largest nonresident program for the action (MAXSIZE parameter value, rounded to hardware block size if not reentrant (TYPE=RNT), + 64 bytes ①	
Lexicon (required only for UNIQUE: 416 bytes)	
Activation Record ②	
Program information block (PIB) 144 bytes + n ③	
Output message area (OMA) (for UNIQUE, calculate as: (CHRS/LIN+6) x (LNS/MSG)+16)	
Input message area (IMA) (length of message or INSIZE parameter value)	
Work area (WA) (for UNIQUE, use 672 bytes)	
Continuity data area (CDA) (for UNIQUE, use 2048 bytes)	
Defined record area (DRA) (required only if the action accesses a defined file)	
Basic segment	370
2 x length of all items specified	
Length of longest identifier	
Length of longest record key	
Length of longest record containing primary part	
Length of longest record containing a supplementary part	
Indexed file I/O area ② (length of I/O area + 12 bytes, for each indexed file accessed by this action)	
File locks (32 x number of indexed files accessed sequentially by this action)	
Record locks (32 bytes for each nonindexed record, 32 bytes + keylength for each indexed record updated by this action or transaction)	

Figure B-4. Schedule for Calculating Main Storage Pool Requirements for Largest Thread in a Multithread IMS System (Part 1 of 2)

Entry and Remarks	Size (bytes)
Data definition record	
36 number of items in data division	
40 x number of items in definition division	
Basic segment	250
Named record locks ④ (32 x number of locks required)	
(Sum) TOTAL MAIN STORAGE POOL FOR THREAD: ⑤	

Notes:

① If this is a UNIQUE action program, MAXSIZE=4152 bytes and TYPE=RNT.

② Figure for each subentry must be rounded up to next multiple of 8 bytes.

③ Here, n is the number of decimal bytes specified with the SHRDSIZE keyword parameter if the action is a shared code COBOL action program.

④ If this is a UNIQUE action program, two locks are required; each action requires one lock if it accesses a defined file.

⑤ Allow 3 to 4% for main storage fragmentation.

Figure B-4. Schedule for Calculating Main Storage Pool Requirements for Largest Thread in a Multithread IMS System (Part 2 of 2)

B.3.5. AUDFILE, CONDATA, and TRCFILE I/O Areas

File I/O areas must be allocated for the audit file (AUDFILE), the continuity data file (CONDATA), and the trace file (TRCFILE) if these are used in your multithread IMS system. For the AUDFILE I/O area, use the calculations given in B.2.5 for the single-thread AUDCONF audit file. For the trace file, use the calculations given in B.2.6 for single-thread.

For the multithread CONDATA file I/O area, which must be allocated if the CDASIZE keyword is specified or if UNIQUE is used, the size is either 1280 bytes (if UNIQUE is included) or the number of bytes specified by the MAXCONT keyword parameter, whichever is larger.

B.3.6. Storage Requirements for Distributed Data Processing

The IMS transaction facility, which supports distributed data processing, is included in the job region when you specify at least one LOCAP section in the configuration. The storage requirement for the transaction facility is 30,000 bytes. In addition, there are some fixed table requirements and some variable table requirements that depend on the DDPSESS and DDPBUF parameters in the configuration or at start-up.

Figure B-5 gives the fixed and variable storage requirements and shows the storage calculation assuming the default values DDPSESS=5 and DDPBUF=1024.

Entry and Remarks	Required Size (bytes)	Sample Calculation (bytes)
Fixed code requirements	30,000	30,000
Fixed table requirements	250	250
Table and stack area requirements per session 2,000 x 5 sessions	2,000	10,000
Buffer requirements per session 2 x 1024 x 5	2 x DDPBUF specification	10,240
Total		50,490

Figure B-5. Sample Calculation of Main Storage Requirements for Distributed Data Processing

B.4. Main Storage Requirements for IMS Utility Programs

Table B-5 lists the module sizes of the IMS pre-online and offline utility programs and their main storage requirements as specified in hexadecimal on the JOB job control statement.

Table B-5. Main Storage Requirements for IMS Utility Programs

Program Name	Module Name	Module Type	Size		Min Specification on JOB Statement
			Decimal	Hexadecimal	
Data definition processor	DT3DF	Load	35,812	8BE4	C000
Edit table generator	ZH#EDT	Load	19,304	4B68	Not required unless executed from an alternate load library
NAMEREC utility	ZP#NRU	Load	34,168	8578	Not required unless executed from an alternate load library
Offline recovery utility	ZE#OLREC	Object	10,884	2A84	
	ZC#TRC	Load	Size is listed in linkage editor output (Figure 7-1)		See 7.4.2 for calculation of main storage requirement
STATFIL print program	ZC#ZSF	Load	34,160	8570	Not required unless executed from an alternate load library
Tape copy routine	ZE#COPY	Object	10,904	2A98	
	ZC#TCP	Load	Size is listed in linkage editor output (Figure 7-4)		8000

Appendix C
Operating Performance of IMS under OS/3

C.1. General

This appendix discusses a number of alternatives for ensuring the most efficient performance of IMS under OS/3 and presents general formulas useful in analyzing IMS performance.

C.2. Alternatives for Maximizing IMS Performance

Several alternative actions are available to maximize your system's performance. These have to do with:

- Improving response time through the integrated communications access method (ICAM)
- Formatting the named record (NAMEREC) file
- Making UNIQUE and user-written action programs permanently resident
- Using individualized action programs instead of UNIQUE
- Optimizing the allocation of user data files and system library files
- Scheduling batch processing and continuous output transactions for off hours
- Adjusting OS/3 system generation parameters to eliminate unwanted overhead
- Carefully selecting configuration options
- Designing action programs for better performance
- Reducing I/O in program loading

With most of these alternatives, you must consider the tradeoff between improving performance and minimizing main storage requirements.

C.2.1. Communications Factors

Communications factors that can affect system performance are: creating buffer pools, line and terminal factors, and using disk queueing.

Creating Buffer Pools

To optimize performance and main storage utilization, configure the number and size of NETWORK BUFFERS and the number of activity request packets (ARPs) as close to actual usage as possible. The NETWORK BUFFER size should be a minimum of 256 words (one word equals 4 bytes). This is a buffer size of 1024 bytes, which represents approximately a half-screen data transmission. A buffer size of 640 words is an overall optimum buffer size for most applications. Applications using full screens (i.e., screen print information or screens containing many input and/or output fields) may require a larger buffer size of 768 or 1024 words for best performance. Use the STAT=YES operand on the BUFFERS macro periodically to check usage. However, do not use the STAT operand in a production environment because it degrades performance.

Line and Terminal Considerations

You should consider the following factors when creating your network of lines and terminals and defining the LINE and TERM macros in the network definition:

- Direct connect lines have faster turnaround time than dialed lines.
- Line speed has a significant effect on response time. Use the highest line speed possible. The greater the load on the line (number of characters per time interval), the more important line speed is.
- Two-way simultaneous transmission (full duplex) lines have faster turnaround than two-way alternate transmission.
- The order in which lines and terminals are defined affects response time. List the most heavily used lines and terminals first.
- The fewer terminals per line, the better the response time. In moderate-to-heavy usage, more than 15 terminals on a line may be excessive. For polled devices, all terminals on a line should be in the same poll group.
- The choice of a polling interval affects response time.
- IMS does not use terminal statistics, so STATS=YES should not be specified on the LINE macro.
- There are instances, especially when using IMS with screen format services, when output messages appear truncated at the remote device. This occurs because the line buffer length for each line in the network definition for output messages is not large enough to contain the entire output message.

  To correct this, specify the desired line buffer length in words in the LBL parameter of the LINE macro.

Disk Queueing Considerations

You can save main storage by using disk queueing for output messages, but processing speed is slower. In an IMS system that supports unsolicited output, three queues per terminal are required. Usually, the high- and medium-level queues should be kept in main storage for processing speed, and the low-level queue should be kept on disk. Because of this requirement, unsolicited output should be configured only if needed.

Do not specify the VERIFY operand on the DISCFILE macro. This specification results in excessive disk I/O during message flow.

C.2.2. Structure of the NAMEREC File

IMS uses the facilities of OS/3 data management to load the NAMEREC file during pre-online processing with blocks that contain your edit tables, password definition records, and data definition records -- and again, during the configuration process, when certain control tables are generated by the IMSCONF jproc and the configurator and loaded into the NAMEREC file. The NAMEREC file is an ISAM file if your system operates in DTF mode; it is a MIRAM file if your system operates in CDM mode. Its record format is fixed, blocked (RECFORM=FIXBLK), and there is one record per block.

The efficiency with which IMS may use the NAMEREC file during online processing -- an important factor in your system's performance -- is affected by the structure of the file: by the size and number of the records it contains and the distribution of records in the prime data and overflow areas on disk. (For details on the layout of an ISAM file, refer to the current version of the *Consolidated Data Management Programming Guide,* UP-9978, or the *Consolidated Data Management Macroinstructions Programming Guide,* UP-9979. For a MIRAM file, see UP-9978.) If there are too many records in overflow (and the overflow records in a NAMEREC file are usually those generated by the data definition processor), performance will be degraded. Before this occurs, the NAMEREC file should be reformatted. However, it is also advisable to provide sound organization for the NAMEREC file to begin with, by carefully considering its parameters before you initialize it for its first pre-online load.

The block size you specify for the NAMEREC file is completely variable between two limits (subject only to the track size of the disk subsystem it uses), ranging between 1024 and 12,800 bytes. Since your data definition records (if you use them) will be the most numerous type of record in this file (and will be the most likely source of records in overflow), your calculation should give primary billing to their sizes.

In determining block size for the NAMEREC file, a prudent step is to estimate the size of the largest data definition record conceivable in your application and to allow a margin for the likelihood that you later may require another, larger size. The default block sizes (3072 bytes for single-thread systems, 6144 for multithread) may not leave you enough margin for the growth your own application may experience. For the formula used in calculating the size of a data definition record, refer to Figure B-4.

The configurator contributes records to the NAMEREC file. If all configuration records have the same blocksize, you may use the same NAMEREC file for all your online IMS load modules.

In planning your NAMEREC file, you should ensure that you do not underestimate the ultimate size of the file, and you should not specify that it may be extended. If automatic extension is requested and occurs, the file may be fragmented on disk. The resulting inefficiency in accessing it may well cost you more than having records in overflow.

It is important to specify enough space for a NAMEREC file (4.2.8). If the number of NAMEREC blocks specified is not adequate, one of the following conditions occurs:

- The message NAMEREC FILE EXHAUSTED appears on the console.
- The message FILE EXHAUSTED appears on the console.
- A configurator abnormal termination occurs.

The default value of 75 blocks allocated by the IMSCONF jproc is adequate for an average size configuration; however, larger configurations or multiple configurations require more blocks. For example, suppose you intend to configure three IMS load modules with the following characteristics (4.3.1):

CONFID=1	Contains approximately 20 terminals, 30 files, 30 actions, 30 programs, and 15 transactions.
CONFID=2	Contains approximately 11 terminals, 14 files, 12 actions, 17 programs, and 10 transactions.
CONFID=3	Contains approximately 9 terminals, 16 files, 13 actions, 13 programs, and 7 transactions.

Configuration 1 requires 75 blocks.

Configuration 2 requires 38 blocks.

Configuration 3 requires 38 blocks.

Approximately 151 blocks are required for three configurations.

You can obtain a substantial improvement in performance by periodically reformatting the NAMEREC file to eliminate overflow records. A 20% improvement has been achieved by logically dumping and restoring the NAMEREC file. The DATA file processing utility routine (and the UDD job control procedure) available for reformatting an ISAM file during a disk-to-disk copy or comparison run are documented in the *Data Utilities Operating Guide,* UP-8834.

You can also physically delete unwanted data definition and password records by executing the NAMEREC file utility with the DELETE function parameter, followed by the UDD job control procedure (3.2.4).

C.2.3. Permanently Resident Action Programs

You can improve performance in a single-thread IMS system by making as many action programs resident as possible, starting with the most heavily used programs. If UNIQUE is frequently used, you should consider making the command interpreter module ZU#CIN resident by specifying UNIQUE=RES in the configurator OPTIONS section.

Program residence is less important in a multithread system because of *sticking power*, an IMS feature that holds frequently referenced programs in main storage.

C.2.4. UNIQUE and Defined Record Management

UNIQUE is a general-purpose set of action programs, and you can get better throughput and response time with user-written action programs tailored to a particular application. Also, the overhead of defined record management reduces performance. However, if you do need defined record management, you may make the defined record management modules resident (RESIDE=YES in the configurator DRCRDMGT section) and thus repeatedly save DRM segment loading time.

UNIQUE is available as unblocked load modules. To take advantage of the loading performance offered by the block loader, however, block any UNIQUE modules that are larger than 3,000 bytes. Use the BLK function of the librarian to block the following UNIQUE modules:

- ZU#DSP
- ZU#LCA
- ZU#MOR
- ZU#PUP
- ZU#SUP

Keep a copy of these modules in unblocked form for patching purposes. A patched blocked load module is loaded in a degraded mode. If you must take a patch to the UNIQUE modules (or any user action program), apply it to the unblocked version, then block it again using the librarian.

C.2.5. Optimizing File Allocations

Performance is improved by observing the following rules:

- When allocating disk files, specify the C parameter on the EXT job control statement to avoid secondary extent areas.
- Build a separate IMS load library instead of using the system load library. Position the file as close as possible to the volume table of contents (VTOC) record of the disk volume.
- Position frequently accessed files, data and otherwise, around the middle cylinders of a disk to minimize head movement time.
- In general, do not place user data files on the same disk as your IMS load library -- again, to reduce head movement time. Also, use a separate disk for IMS internal files and ICAM files.
- Avoid placing all files on disk subsystems that are available to you only via a single channel or disk adapter.
- Compress load libraries and data files periodically to remove deleted modules and directory entries.
- Use resident common storage area ISAM files to save disk access.
- For files accessed sequentially most of the time, use multirecord blocks. Files usually accessed randomly should be unblocked.
- Nonindexed files are more efficient than indexed files, because references to the index area require additional accesses. However, if files must be extended, use indexed files.

C.2.6. Ensuring that the Multithread Option Is Viable

Only when sufficient main storage can be allocated to run multiple action programs concurrently, and the possibility exists for a significant degree of parallel processing, is the overhead inherent in multithread operation likely to prove supportable. See C.4 for main storage requirements for concurrent processing under multithread IMS.

C.2.7. Scheduling Batch Processing and Continuous Output Transactions

Online batch processing conducted during normal production hours when the terminal network already has heavy activity may noticeably degrade system performance. If offline batch processing is not an acceptable alternative, online batch processing is best scheduled so as to least interfere with the day's normal throughput: during slack periods, for example, or just prior to shutdown.

Extensive use of continuous output by multiple terminals in a network can affect response time at terminals doing interactive processing. Like online batch processing, continuous output should be scheduled during off hours.

C.2.8. Eliminating Unwanted System Overhead

You can eliminate unwanted overhead by eliminating certain features from your operating system. See the appropriate installation guide for details on performance tuning. Consider these SYSGEN specifications:

- ERRLOG=NO

  Eliminating error logging can reduce system I/O. You can, alternatively, include error logging in your supervisor and turn it off via console key-in when you run IMS.

- RESMOD=SM$STXIT,SM$TASK,SM$ASCKE,SM$ATCH,SM$LOD

  Making these functions resident reduces the number of I/Os. SM$LOD, SM$STXIT, and SM$ASCKE are important in both single-thread and multithread IMS. SM$TASK is used by ICAM and multithread IMS. SM$ATCH is the least important of the group. If these modules cannot be made resident, additional transient areas should be allocated.

- ONLNDIAG=NO

  Elimination of online diagnostics reduces the size of the supervisor. You can configure a second supervisor with this feature for diagnostic work.

- JOBACCT=NO

  Eliminating job accounting reduces the size of the supervisor and the number of I/Os. Job accounting may be used to examine the number of accesses to each disk drive assigned to IMS as an aid in determining file placement, then eliminated once this information is known.

- SYSLOG=NO

  Omitting the system log reduces disk space and I/Os.

- CONSOLOG=NO

  Specification of the console log has significant effect on supervisor size and system I/O. If this feature must be included, the maximum buffer size should be generated.

C.2.9. Configuration Considerations

The order in which you define the repeatable sections of configurator input affects response time. You should define the most heavily used entries first in the FILE, TERMINAL, TRANSACT, ACTION, and PROGRAM sections. This reduces table lookup time.

In addition, you can improve performance by carefully choosing or calculating certain configurator specifications:

- In the TIMEOUTS section, set both action and status timeouts as high as possible. Once your system is debugged, timeouts are seldom needed, and the higher value reduces timer interrupts.

- Suppress file tracing (TRACE parameter in the FILE section) for nonessential files, and consider recording only before- or after-images for offline recovery (RECOVERY parameter in the OPTIONS section).

- Set the AUDITNUM specification as low as possible (GENERAL section). Higher settings waste disk space and lead to excessive head movement between file partitions.

- For indexed files, include the INDAREA and INDSIZE data management keywords, and specify the largest possible value for INDSIZE. This allows part or all of the main index to reside in main storage.

C.2.10 Designing Action Programs

The way you design and code action programs has a direct bearing on performance. Here are some points to consider:

- Keep input and output messages to a minimum number of characters. Transmission time is a significant factor in response time.
- Whenever possible, terminate action programs with immediate internal succession instead of delayed internal or external succession.
- Consider using external succession instead of delayed internal succession. In single-thread IMS, delayed internal succession can prevent the processing of other messages for longer periods of time because the successor program is scheduled immediately. In multithread IMS, delayed internal succession increases response time.
- Avoid holding record locks any longer than necessary. In multithread IMS, holding record locks can lead to a deadlock situation and may inhibit the scheduling of messages. In single-thread IMS, messages may be canceled and need to be reentered at a later time.
- Avoid alternating file accesses, unless files are spread over different devices. When files are on the same device, this causes excessive head movement.

C.2.11. Reducing I/O in Action Program Loading

Action programs larger than 2560 bytes should be blocked, using the BLK control statement of the OS/3 librarian. Refer to the *System Service Programs (SSP) Operating Guide,* UP-8841. Block load modules increase the efficiency of program loading.

To avoid extra loading operations, do not use patched load modules and do not use ALTER statements in the IMS execution job control stream.

The fast load feature (4.3.3) improves performance of action program loading.

C.3. Analyzing IMS Performance

Table C-1 provides general timing formulas for estimating the response time for transactions processed by IMS. It is not possible to give precisely quantified values for each of the component timing factors because of the large variation in user environments, e.g., different transaction profiles, different processing requirements, different background workloads, and different hardware configurations. Nevertheless, analysis of these formulas may help the systems analyst determine which factors are critical in his particular IMS environment.

Table C-1. General Formulas for Analyzing IMS Performance

Time to IMS
Time to IMS = time till poll + time till IMS ready to schedule + system overhead-1 where: Time till poll = 1/2 average poll cycle + line speed + traffic on line Time till IMS ready to schedule = response time X number of previous unresponded-to inputs + ICAM handling time ICAM handling time = execution time + system overhead-1 System overhead-1 = switcher time + error log time Time in IMS = IMS schedule time + user action program time + system overhead-2 where: IMS schedule time = execution time + IMS file I/ O time + load time IMS file I/O time = channel ready time + head movement time + physical I/O time Load time = directory search time + relocation time + system overhead-3 Directory search time = (channel ready time + head movement + physical I/O time) X (number of blocks till hit) Relocation time = (channel ready time + head movement + physical I/O time) X (number of blocks in module + relocation blocks) System overhead-3 = transient time + switcher time Transient time = execution time + wait for area time + transient load time Transient load time = channel ready time + head movement + physical I/O time User action program time = execution time + I/O time + output time I/O time = execution time + user file I/O time + system overhead-4 User file I/O time = channel ready time + head movement + physical I/O time Output time = execution time + file I/O time System overhead-2 = switcher time + error log time Error log time = channel ready time + head movement time + physical I/O time
Time to Terminal
Time to terminal = time till ICAM ready + ICAM handling time + time till line available + system overhead-5 where: Time till line available = traffic on line + line speed System overhead-5 = switch time + error log time

C.4. Concurrent Processing under Multithread IMS

The following main storage areas must be available for IMS to concurrently schedule input messages under multithread IMS:

- One load area for each concurrent action, each load area being large enough to accommodate the largest program that action might need

  Note: *If an action program is serially reusable, input messages using that same action program will not be concurrent. If the action program is sharable or reentrant, only one load area is necessary for concurrency.*

- One activation record area for each concurrent action. (Activation record area size can be calculated based on user input to the configurator in the ACTION section. See 4.3.8.)

  Note: *For load areas under serially reusable programs and activation records, the size must be rounded up to a hardware protection block size and the area must be allocated on appropriate storage boundaries.*

- One I/O area for each ISAM, MIRAM, and SAM file used by any of the concurrent actions

  Note: *DAM files do not require an I/O area. For other access methods, only one I/O area is required if two or more actions are both accessing the same file.*

- One thread control block for each concurrent action

- One program control block (approximately 50 bytes) for each program in the storage pool

Other areas, such as record locks and NAMEREC records, may be needed. Because storage fragmentation takes place, the user should increase the calculated storage size by 10 to 20%.

C.4.1. Lock-for-Transaction Feature

If the IMS lock-for-transaction feature is used and retained for an entire sequence during transactions that consist of multiple input messages, concurrency is minimized. When concurrent actions access the same record and one action imposes a lock, the lock-for-transaction feature may cause space in the main storage pool to be allocated for actions that are then suspended awaiting the release of some locked record. If this occurs, main storage availability decreases, thereby decreasing concurrency possibilities.

Deadlock occurs when all transactions being processed are waiting for a resource that is allocated exclusively to some transaction and no more input messages can be scheduled because of insufficient main storage. A deadlock condition can also be caused by an action program issuing a GETUP function (read and update) to two different files while a second action program is performing the reverse (a write function). Therefore, when designing action programs, the user should consider this sequence of events. When IMS detects deadlock conditions, it cancels the transaction and rolls back the update.

C.4.2. Assigning Tasks at IMS Start-up

You must specify a minimum of four tasks on the JOB statement at start-up of a multithread IMS system. IMS attaches one I/O subtask when you specify four tasks and one for each additional task. If your application has actions that initiate many logical I/O requests (30 or more), you should assign six or seven tasks to IMS.

Appendix D
UNIQUE Language Elements

This appendix lists the language elements in the standard UNIQUE lexicon and the maximum number of characters for replacement words. You can replace any or all of the language elements in the table by including them in a LANGUAGE section of the IMS configuration (4.3.10).

Table D-1. UNIQUE Language Elements

UNIQUE Language Element	Maximum Length of Optional Replacement	UNIQUE Language Element	Maximum Length of Optional Replacement
OPEN	8	AFTER	8
CLOSE	8	IF	8
DISPLAY	8	AT	8
LIST	8	COMPLETE	8
MORE	8	END	8
DETAIL	8	NO	8
NEXT	8	PASSWORD	8
CHANGE	8	INVALID	8
ADD	8	ILLEGAL	8
DELETE	8	COMMAND	8
ASSIGN	8	MESSAGE	8
SHOW	8	RECORD	8
CANCEL	8	TOOBIG	8
OK	8	IDENT.	8
AND	4	EOM	8
OR	4	DATATYPE	8
NOT	4	I/O	8
EQ	2	ERROR	8
NE	2	BAD	8
GT	2	DATA	8
GE	2	EXISTS	8
LT	2	ARITH.	8
LE	2	EXPRESS.	8
TOTAL	8	INPUT	8
COUNT	8	WORD	8
MAX	8	LEXICON	8
MIN	8	COMPOSED	8
AVG	8	ITEM	8
FOR	8	NAME	8
FROM	8	FOUND	8

continued

Table D-1. UNIQUE Language Elements (cont.)

UNIQUE Language Element	Maximum Length of Optional Replacement	UNIQUE Language Element	Maximum Length of Optional Replacement
STATIST.	8	MANY	8
FUNCTION	8	START	8
ARGUMENT	8	RELATION	8
MISSING	8	LOGICAL	8
ALLOWED	8	OPERATOR	8
TOO	8	USAGE	8
LONG	8	PAREN.	8
EXCEEDS	8	ZERO	8
PAGE	8	NUMERIC	8
UNRECOG.	8	DIGIT	8
REQUEST	8	FEW	8
FILE	8	LESS	8
CLOSED	8	THAN	8
UNDEFINE	8	SINGLE	8
VERB	8	LOWER	8
IN	8	BOUND	8
SUCH	8	LATE	8
KNOWN	8	INCOMPL.	8
RESPONSE	8		
CANCELED	8		
DEFINED	8		
HAS	8		
CHANGED	8		
LITERAL	8		
OPERAND	8		

Appendix E
IMS Messages

The tables in this appendix list groups of IMS messages relevant to the topics discussed in this guide. They show the text of each message, along with the reason why the message was displayed and the action IMS takes or the corrective action you should take.

Table E-1. NAMEREC Utility Diagnostic Messages

Error Message	Cause	Corrective Action
1ST CARD MUST HAVE PASSWORD COLS. 1-8	Parameter PASSWORD missing in columns 1-8.	Correct and resubmit.
COL 1 IN CONTINUATION CARD MUST BE BLANK	A character appears in column 1 of a continuation card.	Correct and resubmit.
CONTINUATION EXPECTED - NOT FOUND	A character appears in of a column 1 continuation card.	Correct and resubmit.
KEYWORD PID= MISSING	Keyword parameter PID was omitted.	Include missing parameter and resubmit.
KEYWORD DDN= MISSING	Keyword parameter DDN was omitted.	Include missing parameter and resubmit.
KEYWORD FN= MISSING	Keyword parameter FN was omitted.	Include missing parameter and resubmit.
KEYWORD TID= MISSING	Keyword parameter TID was omitted.	Include missing parameter and resubmit.
PARAMETER ERROR - NAME IGNORED	The name of the specified record contains more than seven characters, spaces or commas were used incorrectly, or necessary commas were omitted.	Correct and resubmit.
PASSWORD ALREADY IN FILE	Duplicate password	None
PASSWORD ppppppp IS INCOMPLETE	A mandatory parameter omitted in definition for password ppppppp.	Include missing parameter and resubmit.

continued

Table E-1. NAMEREC Utility Diagnostic Messages (cont.)

Error Message	Cause	Corrective Action
PASSWORD rrrrrrr RESTORED TO FILE	A password previously prepared for deletion has been restored to the file.	None
ppppppp ISAM UNRECOVERABLE ERROR	An unrecoverable device error occurred while adding password definition record for password ppppppp to the named record (NAMEREC) file. The definition of this and all following passwords is not processed.	NAMEREC file is compromised and problem must be corrected. NAMEREC file must be reinitialized, and all pre-online processors must be rerun.
ppppppp OVERFLOW AREA IS FULL	Overflow area for NAMEREC file is full, preventing addition of password definition record for password ppppppp.	Reorganize NAMEREC file. NAMEREC file must be reinitialized, and all pre-online processors must be rerun.
rrrrrrr NOT FOUND	The record specified for deletion does not exist.	Check job control contents of NAMEREC file to determine if correct name was specified.
rrrrrrr ALREADY PREPARED FOR DELETION	The record specified was already marked for future deletion.	None
rrrrrrr PREPARED FOR PHYSICAL DELETION	The specified record was found and successfully marked for deletion.	None

Table E-2. Configurator Errors and Their Interpretation

Error Codes	Diagnostic Message	Message Description	Configuration Action
		Error Code 1 - Initialization Errors	
*** ERROR 01 01	NO SOURCE LINES, FILE IS EMPTY	An end-of-file was found trying to read the first input record.	No further processing is done; program is terminated.
*** ERROR 01 02	FIRST CARD IS NOT A SECTION CARD	The configurator input has to start with a section card (NETWORK if present). Section name must appear in columns 1 through 8 on the section cards. The configurator searches for the first card (record) with a nonblank character in column 1.	Disregards all input until a section card is found.
*** ERROR 01 03	NO SECTION CARDS, EOF FOUND	No cards with a nonblank in column 1 were found.	No further processing is done; program is terminated.
*** ERROR 01 04	NO SECTION CARDS AFTER NETWORK SECTION	No sections were found after the NETWORK section. This is not enough information to produce a full configuration.	No further processing is done; program is terminated.
*** ERROR 01 05	CCA LOADING ERROR TERMS=1 ASSUMED	1. CCA load module, generated in the CCA link step of IMSCONF jproc and specified by the NAME parameter in the NETWORK section, could not be found in the load library. 2. Insufficient main storage to load CCA load module. 3. CCA has zero length. 4. Locap name specified in the CUP parameter of the NETWORK section was not found in the loaded CCA.	Configuration processing continues. The number of terminals is defaulted to 1. This allows the configurator to complete and to check for other error conditions. See console error message also displayed.
		Error Code 2 - Section Errors	
*** ERROR 02 01	SECTION APPEARED BEFORE	All repeated sections must be input consecutively.	Input is ignored until another valid section is encountered.
*** ERROR 02 02	INVALID SECTION NAME	Section name input does not match a valid section name.	Input is ignored until another valid section is encountered.
*** ERROR 02 03	SECTION NAME TOO LONG	Section name found to be longer than 8 characters.	Input is ignored until another valid section is encountered.

continued

Table E-2. Configurator Errors and Their Interpretation (cont.)

Error Codes	Diagnostic Message	Message Description	Configuration Action
		Error Code 2 - Section Errors (cont.)	
*** ERROR 02 04	THIS SECTION CANNOT REPEAT	A nonrepeatable section was repeated.	Input is ignored until another valid section is encountered.
*** ERROR 02 05	TERMINAL SECTION COUNT EXCEEDS 'TERMS='	TERMS=n specification in the NETWORK section is less than the number of TERMINAL sections input to the configurator.	Input is ignored until another valid section is encountered.
*** ERROR 02 06	INVALID LOCAP NAME	The locap name is the same name specified in the CUP parameter of the NETWORK section.	Input is ignored until another valid section is encountered.
*** ERROR 02 07	LOCAP SECTION INVALID - CCA NOT GLOBAL	The ICAM CCA specified by the NAME parameter in the NETWORK section is not a global CCA.	Input is ignored until another valid section is encountered.
		Error Code 3 - Syntax Errors	
*** ERROR 03 01	PARAM WORD REQUIRED AND NOT FOUND	All repeatable sections require a parameter word immediately following the section name.	This parameter word is essential to process the section. When this error occurs, the input is ignored until another valid section is encountered
*** ERROR 03 02	KEYWORD DOES NOT END IN AN = SYMBOL	All keywords for all sections must end in equal sign with no space between the word and the equal sign.	Scanning on the input record continues until a nonblank character is encountered.
*** ERROR 03 03	INVALID KEYWORD FOR CURRENT SECTION	Keyword input (i.e., a word 8 characters or less and terminating in an equal sign) does not match any keyword for current section.	Input is ignored until the next valid keyword for this section is encountered.
*** ERROR 03 04	TOO MANY CHAR IN PWORD/KEYWORD SPEC	All positional parameters and keyword specifications must be 8 characters or less in length.	When positional parameter is too long, input is ignored until the next valid section. When keyword specification is too long, the default value for the keyword is assumed.
*** ERROR 03 05	KEYWORD INPUT MORE THAN ONCE	Self-explanatory.	The latest value for that keyword is accepted.

continued

Table E-2. Configurator Errors and Their Interpretation (cont.)

Error Codes	Diagnostic Message	Message Description	Configuration Action
Error Code 3 - Syntax Errors (cont.)			
*** ERROR 03 06	INVALID CHAR PARAM/KEYWORD SPECIFICATION	An invalid character was found in the positional parameter or keyword specifications. Refer to Assembler Programming Guide, UP-8913 for explanation of DIAGNOSTICS in assembly listings.	When invalid character is in positional parameter, input is ignored until the next valid section. When invalid character is in keyword specification, the input scan continues.
*** ERROR 03 07	PARAM WORD MUST BEGIN WITH ALPHA CHAR	First character of parameter was nonalphabetic.	Input is ignored until another valid section is encountered.
*** ERROR 03 08	PARAM WORD TRUNCATED TO MAXIMUM LENGTH	Parameter specification contains too many characters.	Value is truncated to maximum allowable length.
*** ERROR 03 09	PARAM WORD RESERVED	Reserved word was specified for transaction code. See Appendix A.	Input is ignored until another valid section is encountered.
*** ERROR 03 11	LOCAP NAME INVALID	Locap name is the same as specified in CUP keyword parameter in the NETWORK section.	Input is ignored until another valid section is encountered.
*** ERROR 03 12	DUPLICATE TRANSACTION/ LANGUAGE NAME	Transaction code/lexicon name was previously specified.	Input is ignored until another valid section is encountered.
Error Code 4 - Semantic Errors			
*** ERROR 04 01	INPUT ERROR - DEFAULT ASSUMED	Self-explanatory.	All keywords have standard defaults that are printed when the C option of the LST keyword parameter of the IMSCONF jproc is specified.
*** ERROR 04 02	MISSING REQUIRED KEYWORD FOR THIS SECTION	FILE section requires some keywords.	Standard default is assumed.
*** ERROR 04 03	KEYWORD PARAM MUST BE 'UP' OR 'TR'	Response to LOCK keyword must be either 'UP' or 'TR'.	LOCK=TR is assumed.
*** ERROR 04 04	KEYWORD PARAM MUST BE Y(ES) OR N(O)	Valid response for this keyword is Y or N, or YES or NO.	The default specification is assumed.

continued

Table E-2. Configurator Errors and Their Interpretation (cont.)

Error Codes	Diagnostic Message	Message Description	Configuration Action
Error Code 4 - Semantic Errors (cont.)			
*** ERROR 04 05	KEYWORD PARAM MUST BE A DECIMAL NUMBER	Response to this keyword must be a decimal number. One or more characters are not 0 to 9 EBCDIC characters.	An asterisk is placed in the exact location and a default of zero is assumed; e.g., an input of 12A30 with an invalid third character results in 12*30 if the C option is specified and 12030 is left to default.
*** ERROR 04 06	KEYWORD PARAM MUST BE A HEX NUMBER	Response to this keyword parameter must be a hexadecimal number (input was in EBCDIC)	Action is same as 04 05 error but valid characters are 0 through F. Error reporting and default are the same; e.g., 12A will appear 12AF*0 and default 12AF00.
*** ERROR 04 07	INCONSISTENT WITH PROC= SPECIFICATION	In the FILE section, a required keyword was omitted with PROC=KEY, or an invalid keyword was specified with PROC=UNK.	Configurator generates default when PROC=KEY, ignores invalid keyword when PROC=UNK.
*** ERROR 04 08	VALUE IS LESS THAN THE MINIMUM ALLOWED	Value specified is less than minimum allowed.	Default value for keyword is assumed.
*** ERROR 04 09	INVALID KEYWORD PARAM - DEFAULT ASSUMED	1. An invalid keyword specification was input when other keyword specifications are required. 2. Parentheses missing when keyword has subparameters. 3. Invalid FILETYPE keyword specification in relation to CDM parameter in IMSCONF jproc.	Default specification is assumed.
*** ERROR 04 10	INBUFSIZ LESS THAN 'STAN'	The INSIZE=STAN or OUTSIZE=STAN specification under the ACTION section is marked in error when the INBUFSIZ specification is less than the size of a standard input or output message.	If not corrected, an insufficient staging buffer size condition may result during online processing.
*** ERROR 04 11	VALUE GREATER THAN THE MAXIMUM ALLOWED	The value specified exceeds the maximum value allowed. The product of SFS and RESFMT keyword exceeds 255.	The value defaults to the maximum allowed for this keyword specification. RFMTONLY is set to YES and the RESFMT value is set to 255.

continued

Table E-2. Configurator Errors and Their Interpretation (cont.)

Error Codes	Diagnostic Message	Message Description	Configuration Action
		Error Code 4 - Semantic Errors (cont.)	
*** ERROR 04 12	SPECIFICATION IGNORED - SFS NOT SPECIFIED	RESFMT specification is invalid when SFS=NO.	Default is assumed.
*** ERROR 04 13	UNDEF SPECIFIED PREVIOUSLY AND ACCEPTED	UNDEF=YES is allowed for only one transaction code.	Keyword specification is ignored.
*** ERROR 04 14	KEYWORD PARAMETER TRUNCATED TO MAX LENGTH	Too many characters specified in lexicon element specification or in ACTION keyword specification.	Specification is truncated to maximum number of characters allowed.
*** ERROR 04 15	KEYWORD PARAMETER MUST EQUAL LANGUAGE ID	The OPEN lexicon element specification must equal the lexicon name.	Specification is defaulted to the lexicon name.
*** ERROR 04 16	NON CDM SUPERVISOR - DEFAULT ASSUMED	Operating system is not generated with CDM.	SFS specification is defaulted to NO.
*** ERROR 04 17	INCONSISTENT WITH IMSCONF JPROC INIT PRM	Keyword STATS=YES in configuration input OPTIONS section, but subparameter 6 of the IMSCONF keyword INIT=( ) was NO.	IMSCONF INIT=( ) keyword takes precedence over STATS specifications in the OPTIONS section.
*** ERROR 04 18	INCONSISTENT WITH IOROUT=RETRVE (SETNO)	Keyword UPDATE=YES is invalid when keyword IOROUT=RETRVE (read only) is specified.	Specification is defaulted to NO.
*** ERROR 04 19	SEQVIEWS AND ACCESS SPEC. INCONSISTENCY	ACCESS=EXC and SEQVIEWS=(n,n) is invalid.	ACCESS=EXCR specification is defaulted.
		Error Code 5 - File Errors	
*** ERROR 05 01	FILE NAME SPECIFIED IS RESERVED	File name specified is an IMS internal file. See Appendix A.	Input is ignored until the next valid section.
*** ERROR 05 02	FILETYPE KEYWORD NOT SPECIFIED	Self-explanatory.	Input is ignored until the next valid section.
*** ERROR 05 04	INVALID FILETYPE OR INCORRECT SEQUENCE	Invalid file type specification in relation to CDM specification in IMSCONF jproc.	Input is ignored until the next valid section.

continued

Table E-2. Configurator Errors and Their Interpretation (cont.)

Error Codes	Diagnostic Message	Message Description	Configuration Action
		Error Code 5 - File Errors (cont.)	
*** ERROR 05 05	INVALID FILE NAME	In the ACTION section, when parameters (i.e., file names) are input for the FILES keyword, these file names must be input consecutively and be separated by commas. For example, FILES=file1, file2, file3. This error also is caused by a file name longer than 7 characters.	Scan for next word. Drop the input record.
*** ERROR 05 06	FILE SECTION NOT INPUT FOR THIS NAME	In the ACTION section, a file name parameter specification on the FILES keyword does not match any of the file names previously input in a FILE section.	Name is dropped, but processing of file name parameters continues.
*** ERROR 05 07	LINK MODULE FOR FILE TYPE NOT FOUND	Configurator error. A module name was not found.	Module will not be included.
*** ERROR 05 08	MAX NMBR FILES EXCEEDED - NO FILE BIT	The maximum number of files allowed is 123 for single-thread and 240 for multithread.	Processing continues, but file cannot be accessed by any action program.
*** ERROR 05 09	FILE MUST PROCEED PRINTER FILE(S)	FILE sections with FILETYPE=PRNT must be input to the configurator after all nonprinter files.	FILE section is ignored and processing continues with next valid section.
		Error Code 6 - Terminal Errors	
*** ERROR 06 04	MORE THAN ONE MASTER - FIRST ONE ACCEPTED	Too many master terminals are identified.	Configurator accepts the first identified terminal as the master.
		Error Code 7 - Cross-Condition Errors	
*** ERROR 07 02	CDASIZE GT MAXCONT VALUE	CDASIZE specification for any configured ACTION section and/or configured OPTIONS section of UNIQUE or DLLOAD is greater than the MAXCONT value.	CDASIZE is set to MAXCONT value or to UNIQUE or DLLOAD CDASIZE if larger than MAXCONT value and configured.
*** ERROR 07 03	RECOVERY INVALID WHEN FUPDATE=NO	RECOVERY= was specified, but file updating was not configured.	No trace file is generated.

continued

Table E-2. Configurator Errors and Their Interpretation (cont.)

Error Codes	Diagnostic Message	Message Description	Configuration Action
Error Code 7 - Cross-Condition Errors (cont.)			
*** ERROR 07 04	AUDITNUM INVALID WHEN NO FUPDATE OR LOCK=TR	The AUDITNUM specification is invalid if FUPDATE=NO or if LOCK=TR is not specified.	The audit file is initialized, and the DTF/CDDIB is generated.
*** ERROR 07 05	ACTION SECTION OR UNIQUE=YES REQUIRED	UNIQUE=YES or TRAN is required if ACTION section is OMITTED.	None
*** ERROR 07 06	PROGRAM SECTION OR UNIQUE=YES REQUIRED	UNIQUE=YES or TRAN is required if PROGRAM section is omitted.	None
*** ERROR 07 07	TRANSACTION SECTION OR UNIQUE=YES REQUIRED	UNIQUE=YES or TRAN is required if TRANSACT section is omitted.	None
*** ERROR 07 08	NO USER FILES, FUPDATE/RECOVERY INVALID	FUPDATE and RECOVERY have been specified, but no FILE section is included for a DAM, ISAM, or MIRAM file.	FUPDATE/RECOVERY features are ignored.
*** ERROR 07 10	NO MASTER TERMINAL - USE CONSOLE	No master terminal was specified.	The system console is generated as the master terminal.
*** ERROR 07 11	OPCOM=YES IGNORED - MASTER TERM NOT SPEC	OPCOM=YES valid only when a master terminal is specified.	System console is generated as the master terminal, and OPCOM specification is defaulted to NO.
Error Code 8 - Address Resolution Errors			
*** ERROR 08 01	NO PROGRAM FOUND FOR ACTION NAME	The ACTION section must match one program name; i.e., one PROGRAM section has to be input for each ACTION section.	Relative address is not resolved.
*** ERROR 08 02	NO ACTION FOUND FOR THIS TRANSACTION SECTION	No ACTION program name was found for this TRANSACT section.	Relative address is not resolved.
*** ERROR 08 03	NO LOCAP FOUND FOR THIS TRANSACTION SECTION	No valid LOCAP section with this locap name was processed by the configurator.	None. At online IMS execution, transaction will abort.

Table E-3. Batch Transaction Processor (BTP) Diagnostic Messages

Message Text	Description or Cause	Corrective Action
BATCH INPUT MODULE READ ERROR	BTP attempts to access a nonexistent input module; or an I/O error has occurred; or the module contains no input messages.	Check that module name is correct and that input messages are indeed present. Otherwise, action as for hardware error.
INVALID BATCH PARAM CARD	The control stream contains an incorrect PARAM card.	Correct the PARAM card.
PRINTER FILE ERROR	This message is sent to the master terminal when a printer file cannot be opened or an unrecoverable hardware error occurs.	Check that the file exists and that the LFD-name is correct. Otherwise, action as for hardware error.
READING SOURCE MODULE module-name FROM FILE file-name	This message is printed upon successfully opening the source input module file.	None.
SOURCE MODULE module-name IN FILE file-name NOT FOUND	This message is printed if the batch input module is not in the library file.	Ensure that the source module exists in the file.
READ ERROR ON MODULE module-name FILE file-name	An I/O error occurs while the BTP is reading the batch input module.	As for hardware error
TOO MANY CONTINUATION CARDS	The BTP has encountered more than 17 continuation cards.	Restrict each input message to the maximum number of cards - 17
HEX CHARACTER ERROR	The BTP prints this message to the right of the input card image when an EBCDIC character other than 0-9 or A-F is coded between two input shift characters (¬).	Revise coding between shift characters (¬) to represent hexadecimal codes; see 6.4.2.
HEX CHARACTER PAIRS INCOMPLETE	The BTP prints this message to the right of the input card image when the EBCDIC characters coded between two shift characters (¬) are within the set 0-9 or A-F but are odd in number.	Revise coding between shift characters (¬) to represent hexadecimal pairs; see 6.4.2
BATCH NOT CONFIGURED - COMMAND IGNORED	Output to master terminal in response to a ZZBTH command entered when BATCH=NO is specified or the BATCH keyword is omitted from the NETWORK section of input to IMS configurator.	If batch processing of transactions is intended, specify the BATCH keyword to the IMS configurator.
BATCH=NO SPECIFIED IN PARAM CARD - COMMAND IGNORED	Output to master terminal in response to a ZZBTH command entered when batch processing is configured but the PARAM BA statement is omitted from the IMS run stream or PARAM BA=NO is specified.	Specify the intended PARAM BA statement in the IMS run stream.

continued

Table E-3. Batch Transaction Processor (BTP) Diagnostic Messages (cont.)

Message Text	Description or Cause	Corrective Action
INVALID ZZBTH PARAMETER - COMMAND IGNORED	Output to master terminal in response to a ZZBTH command entered incorrect parameters (syntax error).	Reenter ZZBTH command with with valid parameters.
TOO MANY ZZBTH COMMANDS ENTERED - COMMAND IGNORED	Output to master terminal in response to a ZZBTH command entered in command excess of the number that may be processed at one time in this configuration. In a single-thread IMS system, only one ZZBTH command may be active at one time. In multithread,the number of commands is limited by the maximum number specified with the BATCH configurator keyword.	Defer reentry of the ZZBTH until response to the ZZTCT command indicates that processing is again possible.
BATCH PROCESSING CANCELLED	Normal response to successful ZZBTH CANCEL command; only the batch processing currently in progress terminates.	None required. Batch processing may be reinitiated during this execution of IMS by entering an appropriate ZZBTH command.
ALREADY READING BATCH CONTROL STREAM DATA SET - COMMAND IGNORED	Output to master terminal in response to a ZZBTH command entered before the processing of a data set in the control stream has been completed by the current ZZBTH command. Only one data set may be processed at a time;while one is being processed, no further PARAM IN statements may be processed.	Defer reentry of the ZZBTH command until response to a ZZTCT command indicates that the current processing of a control stream data set is complete.
BATCH RESUMED	Normal response to successful ZZBTH RESUME command.	None required.
BATCH INITIATED	Normal response to successful start of batch processing with the ZZBTH command.	None required.

continued

Table E-3. Batch Transaction Processor (BTP) Diagnostic Messages (cont.)

Message Text	Description or Cause	Corrective Action
ZZBTH PARAMETER PROCESSING ERROR	Output to master terminal in response to a ZZBTH command when an input module is missing, the end of control stream input is accessed, and so forth.	Check for correct parameter information and reissue command.
BATCH PROCESSOR NOT IN PAUSE - COMMAND IGNORED	Output to master terminal in response to a ZZBTH RESUME command entered when the BTP is not in a pause state.	Do not reenter ZZBTH RESUME command until receipt of normal response to ZZBTH PAUSE command.
BATCH PROCESSOR NOT RUNNING - COMMAND IGNORED	Output to master terminal in response to a ZZBTH PAUSE or ZZBTH CANCEL command when the BTP is not running or has completed processing.	None required.
BATCH PROCESSOR IN PAUSE STATE	Normal response to successful entry of the ZZBTH PAUSE command.	None required. The only valid commands following the ZZBTH PAUSE commands are the ZZBTH RESUME or the ZZBTH CANCEL command.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

W

Z