UNISYS

Product Information Announcement

o New Release o Revision • Update o New Mail Code

Title

System 80 OS/3 Integrated Communications Access Method (ICAM) Utilities Programming Guide

This Product Information Announcement announces the release and availability of Update A to the *System 80 OS/3 Integrated Communications Access Method (ICAM) Utilities Programming Guide.*

This guide describes the utilities provided with OS/3 ICAM. ICAM is a software product that enables OS/3 users to establish, use, and maintain a communications network.

This update contains only minor technical corrections.

You can order the update only, or the complete manual with the update. To receive only the update, order UP-9748 Rev. 2-A. To receive the complete manual, order UP-9748 Rev. 2.

Additional copies of this document can be ordered through your branch representative or from Unisys Corporation, Corporate Software and Publications Operations, 13250 Haggerty Road North, Plymouth, Michigan 48170.

Announcement only:
MB00, SAB, and SAE

Announcement and attachments:
MBB1, MB20, and MBW

System: System 80
Release: 13
Date: June 1990
Part Number: UP-9748 Rev.2-A

UNISYS

PUBLICATIONS REVISION

System 80

OS/3 Integrated Communications Access Method (ICAM) Utilities Programming Guide

UP-9748 Rev. 2

This Library Memo announces the release and availability of the *System 80 OS/3 Integrated Communications Access Method (ICAM) Utilities Programming Guide*, UP-9748 Rev. 2.

This guide is a standard library item (SLI). It is part of the standard library provided automatically with the purchase of the product.

Changes to this guide for Release 13.0 include the following:

- Description of the differences in handling ICAM functions between model 7E and models 3-6 and 8-20
- Addition of journal utility job control stream example
- Removal of references to DCT 1000 printer
- Removal of sign-on procedures for remote terminals
- Changes to the DCP dump procedure

All other changes in this revision are corrections, deletions, or expanded descriptions applicable to items present in the software prior to this release.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists SAB, SAE, and MB00	Mailing Lists MBB1, MB20, and MBW (224 pages plus Memo)	Library Memo for UP-9748 Rev. 2
		RELEASE DATE: January 1990

UNISYS

System 80 OS/3 Integrated Communications Access Method (ICAM) Utilities

Programming Guide

January 1990

Printed in U S America
UP-9748 Rev. 2

Priced Item

UNISYS

System 80 OS/3 Integrated Communications Access Method (ICAM) Utilities

Programming Guide

OS/3 Release 13

June 1990

Printed in U S America
UP-9748 Rev. 2 – Update A

Priced Item

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: Update A – UP-9748 Rev. 2

Part/Section	Page Number	Update Level
Cover		Orig.
Title Page/Disclaimer		A
PSS	iii	A
About This Guide	Tab Breaker	Orig.
	v thru xi	Orig.
Contents	xiii thru xxi	Orig.
Section 1	Tab Breaker	Orig.
	1 thru 3	Orig.
Section 2	Tab Breaker	Orig.
	1 thru 11	Orig.
Section 3	Tab Breaker	Orig.
	1 thru 40	Orig.
Section 4	Tab Breaker	Orig.
	1	Orig.
Section 5	Tab Breaker	Orig.
	1 thru 37	Orig.
Section 6	Tab Breaker	Orig.
	1 thru 30	Orig.
Section 7	Tab Breaker	Orig.
	1 thru 4	Orig.
Section 8	Tab Breaker	Orig.
	1 thru 18	Orig.
Section 9	Tab Breaker	Orig.
	1 thru 5	Orig.
Section 10	Tab Breaker	Orig.
	1 thru 5	Orig.
Section 11	Tab Breaker	Orig.
	1 thru 9	Orig.
Section 12	Tab Breaker	Orig.
	1 thru 3	Orig.
	4	A
	5 thru 8	Orig.
Index	Tab Breaker	Orig.
	1 thru 4	Orig.
User Comments Form		
Back Cover		Orig.

Technical changes are denoted by a change bar (|) in the margin.

About This Guide

Purpose

This guide describes the utilities provided by the Unisys Operating System/3 (OS/3) Integrated Communications Access Method (ICAM).

Scope

ICAM is the OS/3 software product that enables you to establish, use, and maintain communications between ICAM end users. As one of a series, this manual is designed to guide you in programming and using ICAM on System 80. Depending on your need, you may wish to refer to one of the other ICAM manuals, which are listed later under "Related Product Information."

Some features described in this guide do not apply to the model 7E; others apply only to the model 7E. These exceptions are so noted in the text.

Audience

This guide is for system console operators and system programmers of System 80 OS/3 systems.

Prerequisites

Anyone using this guide should be familiar with the concepts of communications networks and system operations.

How to Use This Guide

This guide presents the detailed information you need to know in order to perform operations on an ICAM network. Before using a particular ICAM utility, read Section 1 of this guide which summarizes the utilities and their purposes. Section 1 also explains the ways in which the System 80 model 7E handles the ICAM utilities differently from the System 80 models 3-6 and 8-20.

Organization

This guide contains twelve sections and an index:

Section 1. Introduction

Summarizes the ICAM utilities and their purposes.

Section 2. ICAM Device Emulation System (IDES)

Explains how System 80 (models 3-6 and 8-20) can be used to emulate a 1004 card processor system or a DCT 2000 data communications terminal; and how all models can be used to emulate IBM 2780 or 3780 terminals.

Section 3. Remote Batch Processing (RBP)

Describes how to submit batch jobs to the system from a remote terminal and receive output at a remote terminal.

Section 4. RPG II Telecommunications

Provides an example of an ICAM network that supports RPG II.

Section 5. Journal Utility

Describes how to produce printouts of message texts and requeue messages if ICAM fails due to a disk failure.

Section 6. COBOL Message Control System (CMCS)

Discusses CMCS, which is the software interface between a COBOL 74 communications user program and ICAM.

Section 7. Dumping the Single Line Communications Adapter (SLCA)

Explains how to dump and list the contents of a single line communications adapter.

This section does not apply to the model 7E.

Section 8. ICAM Trace Facility (ITF)

Describes the use of the ICAM trace facility to get records of key ICAM operations for the ICAM edit dump.

Section 9. Device Trace Symbiont (DT)

Describes the use of the device trace symbiont to monitor communications activity.

Section 10. ICAM Edit Dump (IED)

Describes the edit dump of ICAM tables for diagnostic purposes.

Section 11. UNIX System Access Module (UNXSAM)

Describes the use of the UNIX® System Access Module, which allows you to access a UNIX operating system to run UNIX applications and shell commands from a local workstation or an OS/3 UTS 400 or SVT class terminal or workstation.

This section does not apply to the model 7E.

Section 12. DCP/Telcon Load and Dump Facilities

Describes how to downline load a distributed communications processor (DCP) connected to an OS/3 system on a universal data link control (UDLC) connection, and how to cross-channel load a DCP connected to an OS/3 system on a channel connection. Also describes how to capture a Telcon dump and write it to tape for printing on an 1100/2200 system.

Notation Conventions

The following conventions are used in this guide:

- ICAM macros and operands are shown in UPPERCASE letters.
- User-supplied variables are shown in lowercase letters in format descriptions and in the text.
- Default values are shaded.
- Optional operands are enclosed in brackets [ ].
- Braces { } enclose groups of operands from which you choose one.
- Parentheses ( ) in format descriptions indicate literal parenthesis characters that must be entered in the code.

UNIX is a registered trademark of AT&T Information Systems.

Related Product Information

The following Unisys documents may be helpful in understanding and implementing the information presented in this guide. Throughout this guide, when we refer you to another document, use the version that applies to the software level in use at your site.

OS/3 Integrated Communications Access Method (ICAM) Technical Overview **(UP-9744)**

This manual provides an overview of the facilities offered by ICAM, including the hardware supported, the types of programs supported (assembler, COBOL, and RPG II), and the services provided (polling, queuing, buffering, etc.).

Models 3-6 and 8-20 OS/3 Integrated Communications Access Method (ICAM) Operations Guide **(UP-9745)**

This guide describes how to define an ICAM network on models 3-6 and 8-20, submit it to the system generation procedure, and load and operate the resulting ICAM symbiont. Sample network definitions are provided to make it easier to define your ICAM network.

Model 7E OS/3 Integrated Communications Access Method (ICAM) Operations Guide **(7002 3908)**

This guide describes how to define an ICAM network on the model 7E, submit it to the system generation procedure, and load and operate the resulting ICAM symbiont. Sample network definitions are provided to make it easier to define your ICAM network.

OS/3 Integrated Communications Access Method (ICAM) Standard MCP Interface Programming Guide **(UP-8550)**

The standard message control program (MCP) interface is a logical interface that provides a general communications capability with message queueing and a message processing capability. It provides all of the macroinstructions, programming requirements, and terminal information you need to interface with the standard interface. You will need this user guide only if you are writing your own communications program. Programs that use the standard interface directly must be coded in basic assembly language (BAL).

OS/3 Integrated Communications Access Method (ICAM) Direct Data Interface (DDI) User Guide **(UP-8549)**

This interface commonly supports ICAM utility programs and programs written in the RPG II language. If you are using an ICAM utility only, or your program is written in RPG II, you won't need this user guide because the utility programs and the RPG II compiler automatically convert any requests by your program to the proper instructions needed to work with this interface.

DDI also enables you to write your own specialized communications program to work with it. If you do this, you must take care of your own message buffering and queuing. If you write a program to interface directly with DDI, it must be written in BAL.

OS/3 Integrated Communications Access Method (ICAM) Communications Physical Interface (CPI) Programming Guide **(UP-9746)**

This interface requires the least amount of main storage, but it also provides a minimum amount of support. If you use it, you must have considerable knowledge of data communications because your program must initialize the hardware, format all output messages using the appropriate protocol, perform any required translations, acknowledge and process all input messages, and perform all error detection and recovery procedures. In addition, your program must be written in BAL.

This guide is not applicable to the model 7E.

OS/3 Integrated Communications Access Method (ICAM) Programming Reference Manual **(UP-9749)**

This manual summarizes the information found in the other ICAM manuals. It also provides information on the message processing procedure specification (MPPS), which enables you to write message processing routines and include them in your ICAM network. Except for MPPS, no introductory information or examples are given; however, it is a useful document when you are familiar with ICAM and you need a quick reference to macroinstructions, formats, and tables.

OS/3 Job Control Language Programming Guide **(UP-9986)**

The job control language (JCL) is the part of the operating system that manages system resources, prepares programs for execution, and starts program execution. This guide presents basic job control statements, which allow novice programmers to perform simple program executions; and more complex statements, which allow experienced programmers wider and more varied control of job processing.

OS/3 Integrated Communications Access Method (ICAM) Remote Terminal Processor (RTP) User Guide **(UP-8990)**

The remote terminal processor is a data communications program that permits your System 80 processor to function as a remote job entry terminal to one or more IBM® host processors.

This guide is not applicable to the model 7E.

Using the ICAM software, the remote terminal processor enables you to:

- Send jobs to an IBM host
- Transmit and receive files on tape, punched cards, or diskette
- Send messages to the central site
- Receive output data and console messages from the IBM host

Remote terminal processor operations are directed from the OS/3 system console.

NTR Utility Programming Guide **(UP-9502)**

NTR is a system utility that allows System 80 to operate as a remote job entry/batch terminal to a Unisys Series 1100/2200 system via ICAM. NTR permits operation of reader, printer, and punch device-dependent files. It also supports user own-code tasks to process device-independent files (tape, disk, paper tape, and so on).

This guide is not applicable to the model 7E.

OS/3 System Messages Reference Manual **(UP-8076)**

Describes the system messages displayed by the System 80. The message description is composed of the remedial action or response required as applicable.

OS/3 Installation Guide **(UP-8839)**

Provides the procedures necessary to install, tailor, and maintain OS/3 software in a System 80 models 3-6 and 8-20 environment.

Model 7E Installation Guide **(7002 3858)**

Provides the procedures necessary to install, tailor, and maintain OS/3 software in a System 80 model 7E environment.

IBM is a registered trademark of International Business Machines Corporation.

***OS/3 Operations Guide* (UP-8859)**

Describes the System 80 models 3-6 and 8-20 hardware configuration and covers all commands and procedures used within the OS/3 environment.

***Model 7E Operations Guide* (7002 3866)**

Describes the System 80 model 7E hardware configuration and covers all commands and procedures used within the OS/3 environment.

***RPG II Programming Reference Manual* (UP-8044)**

Provides a quick reference for applications programmers familiar with OS/3 RPG II.

***RPG II Programming Guide* (UP-8067)**

Describes the RPG II programming language. Includes information on general concepts and program logic, file organization, record and data formats, RPG II specifications forms, telecommunications, and how to compile, link, and execute a user program.

***1974 American Standard COBOL Programming Reference Manual* (UP-8613)**

Describes 1974 ANS COBOL for the applications programmer. Covers the COBOL character set, data types, character strings, and the statements permitted in each COBOL division. Also describes statements pertaining to table handling, file processing, sort/merge, interprogram communication, and communications.

***System 80 Models 8-20 Processor Complex Controllers Programming Reference Manual, Volume 2* (UP-9732)**

Contains hardware design, operations, and programming information to assist in programming peripheral devices and communications attachments integrated with System 80 models 8, 10, 15, and 20 facilities.

***Distributed Communications Processor Operating System (DCP/OS) Operations Reference Manual* (UP-11541)**

Provides information on the basic DCP/OS operations. These include booting and dumping the DCP; building, debugging, and editing programs; and executing utility programs.

***Telcon Dump Analysis Guide* (UP-10301)**

Provides information on Telcon system problem analysis. Includes information on how to run the TELFOR and DCPDUMP programs.

Contents

Section 6. COBOL Message Control System (CMCS)

Section 7. Dumping the Single-Line Communications Adapter

Section 8. ICAM Trace Facility (ITF)

Section 9. Device Trace Symbiont (DT)

Section 10. ICAM Edit Dump (IED)

Section 11. UNIX System Access Module (UNXSAM)

Figures

Tables

Section 1
Introduction

1.1. ICAM Utilities Summary

The Unisys Operating System/3 (OS/3) Integrated Communications Access Method (ICAM) provides utility programs that perform specialized functions. These ICAM utilities:

- Enable a System 80 to function as a terminal to another computer system
- Let you submit batch jobs from remote stations
- Provide printed reports of system activity
- Provide communications capability for RPG II and COBOL programs
- Aid in diagnosing operational problems

The following list summarizes the ICAM utilities described in this guide:

- IDES - ICAM Device Emulation System

  Enables a System 80 (models 3-6 and 8-20) to emulate a 1004 card processor system or a DCT 2000 terminal. Enables all models to emulate an IBM 2780 terminal, or an IBM 3780 terminal

- RBP - Remote Batch Processing

  Allows you to submit batch jobs from a remote station and receive output

- Journal Utility

  Provides printed reports of system activity and input and output messages

- CMCS - COBOL Message Control System

  Provides a software interface between a COBOL 74 program and ICAM, thus providing a communications capability

- Single Line Communications Adapter Dump Routine (models 3-6 and 8-20 only)

  Enables the system console operator to dump and print the random access memory (RAM) of the single line communications adapter (SLCA) (model 7E uses a COMMDLP, not SLCA.)

- ITF - ICAM Trace Facility

  Helps locate the cause of ICAM operational problems

- DT - Device Trace Symbiont

  Monitors communications and workstation activity

- IED - ICAM Edit Dump

  Provides an edited dump of selected ICAM tables for diagnostic purposes

- UNXSAM -UNIX System Access Module (models 3-6 and 8-20 only)

  Allows you to access a UNIX operating system from a local workstation or OS/3 UTS or SVT class terminal or workstation (UNXSAM is not applicable for model 7E)

- DCP/Telcon Load Facility

  Allows you to downline or cross-channel load a DCP connected to an OS/3 system

- DCP/Telcon Dump Facility

  Allows you to upline dump a DCP connected to an OS/3 system

1.2. Differences between Model 7E and Models 3-6 and 8-20

The model 7E handles various ICAM functions in ways that are different from the other System 80 models. Most notably, model 7E uses a COMMDLP as its communications connector instead of an SLCA. Also, model 7E does not support locally connected card readers and punches; thus, those parts of the text that refer to these apply only to models 3-6 and 8-20.

References that apply only to models 3-6 and 8-20 and not to model 7E are:

- 1004 Card Processor
- DCT 2000
- UNXSAM (TTY emulation)
- NTR
- Circuit Switched PDN (X.21)
- Workstations
- 9200/9300 Processors
- LINKPAK keyword on the LINE statement
- LBL keyword on the LINE statement
- DIALER keyword on the LINE statement
- DCPCHNL statement

References to the MCPBUF keyword apply to model 7E only and not to other models:

References that vary for model 7E and other models are:

- ID keyword on the LINE and VLINE statements
- CACH statement

These differences are indicated at various points throughout the text.

Section 2
ICAM Device Emulation System (IDES)

2.1. Overview

The ICAM device emulation system (IDES) allows a System 80 to function as a remote batch terminal to another computer system. With IDES, you can use the System 80 (models 3-6 and 8-20) as an emulated 1004 card processing system or a DCT 2000 data communications terminal. All models can emulate IBM 2780 or 3780 data communications terminals through IDES.

The model 7E does not support card reading or punching or the DCT 2000 terminal. Where these are referred to in the text, the statements do not apply to the model 7E.

The computer system using the System 80 as a remote batch terminal is known as the remote host or simply the host. The remote host may be another System 80, a Unisys Series 1100/2200 system, an IBM 360/370 system, or any other computer system that can support the emulated remote batch terminals.

IDES operates under OS/3 as a user job and allows for multi-jobbing, provided the necessary facilities are available. IDES operations are controlled by console keyins.

To provide flexibility in the reading and punching of card images, there are two versions of IDES:

- TZ$SME

  The IDES module TZ$SME accepts EBCDIC card images.

- TZ$SMF

  The IDES module TZ$SMF accepts Fieldata card images.

This feature aids conversion of data from configurations using Fieldata punch card code.

IDES interfaces with ICAM through the direct data interface (DDI) for communications with the remote host and interfaces with OS/3 data management for reading, punching, and printing functions.

Figure 2-1 illustrates the organization of the software components in a System 80 remote batch emulation configuration.

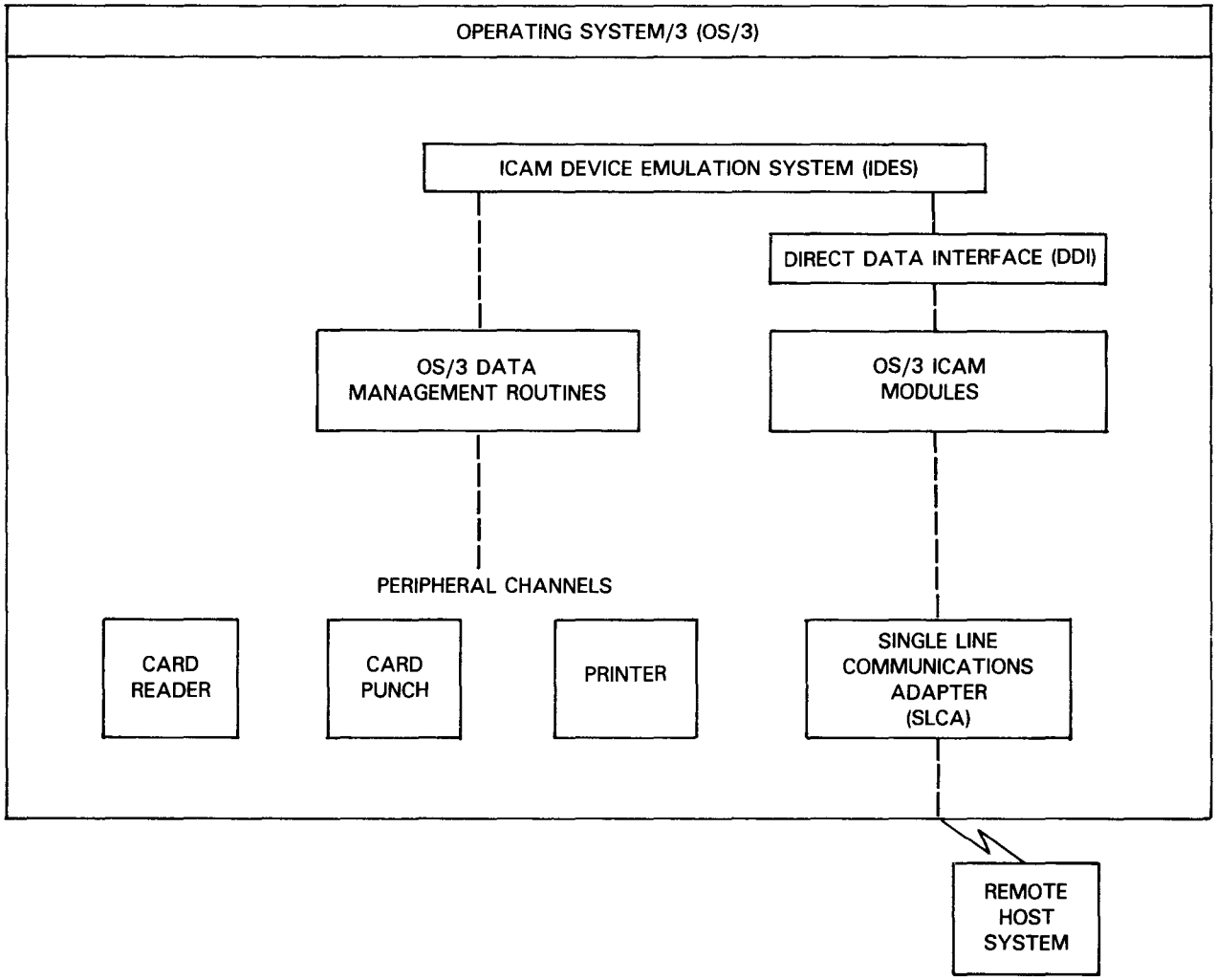


Figure 2-1. Structure and Organization of Remote Batch Emulation Configuration

Note: *The model 7E remote host connection is through a COMMDLP, not through an SLCA. Also, model 7E does not have card reader or punch.*

2.2. Creating an ICAM Symbiont to Support IDES

You must create an ICAM symbiont that includes a direct data interface (DDI) network for IDES. To generate an ICAM symbiont, you code the parameters in the COMMCT phase of system generation (SYSGEN).

Note: *The COMMCT parameters include the network definition macros described in the* Models 3-6 and 8-20 ICAM Operations Guide *(UP-9745) and in the* Model 7E ICAM Operations Guide *(70023908), and also the message control program (MCP) parameters described in the* OS/3 Installation Guide *(UP-8839) and the* Model 7E Installation Guide *(70023858).*

You can define more than one network in your ICAM symbiont by including several sets of network definition macros. You can create an ICAM with only an IDES network or with an IDES network and other types of networks. If you have multiple IDES lines, you include a separate IDES network for each line.

You submit the COMMCT parameters to the SYSGEN parameter processor (SG$PARAM) and then run the SG$COMMK job control stream to assemble and link the ICAM symbiont.

Note: *For details on this SYSGEN process, see the current version of the* OS/3 Installation Guide, *UP-8839, (models 3-6 and 8-20 users) or the* Model 7E Installation Guide, *70023858 (model 7E users).*

2.3. Defining an IDES Network

In the network definition for IDES, you use only five different macros: CCA, BUFFERS, LINE, TERM, and ENDCCA. These are described in the current version of the *ICAM Operations Guide* (UP-9745) and the *Model 7E Operations Guide* (70023866). Table 2-1 summarizes these network definition macros, specifies any differences in how they apply to IDES, and describes operands specific to IDES.

In the case of multiple IDES lines, you need a separate IDES network definition for each line. (You also need to execute a separate IDES program for each network. See 2.5.2.) The order of presentation of macros to define a network for IDES follows. Note that the CCA... ENDCCA sequence is repeated for each IDES network.

Macro	**Meaning**
CCA_1	Start of a network definition and its type
BUFFERS	Specifies resources required for network
LINE	Characteristics of the IDES line
TERM	Defines terminal emulated on this line
ENDCCA	Specifies the end of the network definition
CCA_n	Start of a subsequent network definition for IDES
.	
.	
.	
ENDCCA	

Table 2-1. ICAM Network Definition Macros for IDES

Macroinstruction	Operand	Meaning
BUFFERS	... ARP=integer	Number of activity request packets (ARPs) you want made available to ICAM to run this network
	[,STAT=YES]	Includes software to keep track of network buffer and ARP pool activity NOTE: Three commas must precede the first keyword operand (BUFFERS ,,,ARP=10,STAT=YES).
CCA	TYPE=(DDI)	Specifies the direct data interface
	[,PASSWORD=password]	1- to 8-character password to further identify this network
	[CCAID=identifier]	1- to 8-character identifier printed at top of network definition listing and summary report at the end of the listing to identify CCA code
ENDCCA		Indicates the end of a network definition
LINE	DEVICE={(1004EM) (DCT2EM [,{84 | 132}]) (BSC, line-buffer-length [,{ASCII | EBCDIC | TRANSCOD}])}	Describes the type of system being emulated. Specifies this line supports a System 80 computer emulating a 1004 card processor system Specifies this line supports a System 80 computer emulating a DCT 2000 Data Communications Terminal (84 and 132 represent line buffer lengths) Specifies this line supports binary synchronous communications. Must be specified for a System 80 to emulate an IBM 2780 or 3780 terminal. TRANSCOD is not valid for IBM 2780EM or IBM 3780EM.
	[,TYPE=([line-speed][,{UNAT | AUTO}][,SWCH][,SYNC][,CRTS][,FLDQ][,FULL])]	Specifies the characteristics of this communications line
	[,CALL=phone-number]	Telephone number used for automatic or operator manual dialing to a terminal
	[,ID=ca-port-number]	Port on communications adapter
	[,DIALER=(ca-port-number)[,eon]]	Specifies port on communications adapter to which an automatic dialer is attached
	[,RETRY=(input-number,output-number)]	Specifies number of retries you want ICAM to make before marking a terminal on this line down

continued

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

Macroinstruction	Operand	Meaning
	[,STATS=YES]	Specifies you want ICAM to accumulate terminal statistics for each terminal on this line
	[,TIMEOUT=(input-time,output-time]	Specifies amount of time in seconds you want ICAM to wait before time-out occurs
	[,XLATE=([{labeli,idi / NO / YES}][,{labelo,ido / NO / YES}])]	Specifies translation tables used for this terminal only
TERM	FEATURES={ ({2780EM / 3780EM} [,max-output-record-length] [,{max-no-records/block / MULTI}] [,{SECOND / PRIMARY}] [,TRANSPARENT] [,transparent-input-record-length] [,max-block-length] [{MODEL1 / MODEL2 / MODEL3 / MODEL4}]) / (DCT2EM [,{ATTENDED / UNATTENDED}]) / (TN4E) }	Specifies this computer emulates an IBM 2780 or 3780 terminal to allow communications between this computer and a host computer. This operand incorporates the emulation mode (slave mode) version of the BSC remote device handler. Specifies this computer emulates a DCT 2000 terminal. When emulating the DCT 2000, the remote device handler is ordinarily operated in unattended mode. However, when processing remotely to an 1100 Series system, the attended mode must be used. If attended mode is specified, the remote device handler does not wait for a poll from the host before transmitting. If the host is not ready, and no response is received from the host, the remote device handler retransmits status to your program. In unattended mode, the remote device handler waits for a poll from the host before transmitting text. Specifies this computer emulates a 1004 card processor system. This operand enables this processor to appear to be a 1004 or a 9200/9300 processor running a REM 1 package.
	[,XLATE=([{labeli,idi / NO / YES}][,{labelo,ido / NO / YES}])]	Specifies translation tables used for this terminal only
	[,AUX1=(PCH)]	Specifies an auxiliary punch device is used

Note: *Refer to the* Models 3-6 and 8-20 ICAM Operations Guide *(UP-9745) or the* Model 7E ICAM Operations Guide *(70023908) for details on each macro.*

The following sample coding is an ICAM network definition that permits System 80 to emulate a 1004 card processing system:

```
      1        10    16                                          72
1.  NET1      CCA   TYPE=(DDI),PASSWORD=DDIDRIVE
2.            BUFFERS ,,,ARP=20
3.  LNE1      LINE  DEVICE=(1004EM),TYPE=(2400,SYNC,SWCH)
4.  TRM1      TERM  FEATURES=(TN4E),AUX1=(PCH)
              ENDCCA
```

Explanation:

1. IDES network uses the DDI interface. A password (DDIDRIVE) is assigned. This password is used on the network descriptor card when the IDES network is initialized.

2. DDI interface does not require network buffers, but does require activity request packets (ARPs) for ICAM activity processing.

3. LNE1 is a System 80 emulating a 1004. It is a switched line with synchronous transmission.

4. Terminal 1 is an emulated 1004 with a site-id of 00001 and a punch auxiliary device. TN4E is the required FEATURES specification.

2.4. IDES Operation

You initiate remote batch operation on the System 80 (terminal) end of the communications line by submitting a network descriptor statement (see 2.5.1). This statement defines the communications network that is used and starts the operation. Subsequent control over remote operations is accomplished through the use of console commands that control input, output, termination, and line control functions (see 2.6).

You should be familiar with the operating discipline of the three remote terminals that can be emulated. When operating under the 1004 card processor discipline, IDES can accept input text at any time. When in DCT 2000 or 2780/3780 discipline, IDES accepts input text whenever output is not active. Conversely, when IDES is in either DCT 2000 or 2780/3780 mode, output is not sent until current input operations are finished.

Due to the nature of the 1004 card processor, input and output operations may take place at the same time; that is, reading of cards and printing can effectively be performed simultaneously although, in reality, message traffic is sent and received alternately. IDES does not support simultaneous emulation of mixed disciplines.

The 1004 card processor discipline is the same one that applies to current 9200/9300 subsystem users who operate under the REM1 (1004 card processor remote emulation on 9200/9300 subsystem) package.

2.5. Initiating IDES

You initiate IDES by entering a job control stream that executes the IDES program TZ$SME (for EBCDIC card images) or TZ$SMF (for Fieldata).

A network descriptor statement in the job control stream identifies the IDES network for this line and provides information needed to initialize the network.

2.5.1. Network Descriptor Statement

The format and parameters of the network descriptor statement are listed in Table 2-2. Note that some parameters are optional but, when specified, must be coded in the columns shown in the table. Parameters may be separated by either commas or spaces.

Table 2-2. IDES Network Descriptor Statement Format

Columns	Contents	Description
1–8	NETWORK=	Identifies card as network descriptor (Required)
9–12	network-name	4-character network name. Corresponds to the label of the CCA macroinstruction in the ICAM network definition. (Required)
14–21	password	8–character password. Corresponds to the password in the CCA macroinstruction. (Optional)
23–26	line-name	4–character line name as defined in the label field of the LINE macroinstruction. (Required)
28–31	device-type	4-character device type. Specifies mode of operation (1004, 2000, or 2780). (Required)
33–37	site-id	5-character site-id. (Required in 1004 mode only.) Specifies the remote site-id to be sent to the host to establish communications.
39–40	NO or blank	NO inhibits IDES from issuing breakpoints to the printer when end-of-file is received from the remote device handler. When left blank, breakpoints are issued.
42–43	NO or blank	NO inhibits IDES from issuing breakpoints to the card punch when end-of-file is received from the remote device handler. When left blank, breakpoints are issued.
45–46	NO or blank	NO inhibits skip to top-of-form when overflow is detected on the printer. When left blank, skip to top-of-form is issued.

Example 1

```
NETWORK=NET1,DDIDRIVE,LNE1,1004,00001,NO,NO,NO
```

Identifies the IDES network named NET1 with the password DDIDRIVE. The remote station uses line LNE1, operates in 1004 mode, and has a site-id of 00001. (These specifications correspond to the sample ICAM network definition in 2.3.) Breakpoints to the printer and card punch are inhibited, and skip to top-of-form is also inhibited.

Example 2

```
NETWORK=NET2,!!!!!!!!,LNE2,2780
```

Identifies network NET2. The remote station uses line LNE2 and operates in 2780 mode. Breakpoints and skip to top-of-form are to be issued. Note that site-id is omitted because the remote station does not operate in 1004 mode. A password also is omitted, indicating that no password is specified in the ICAM network definition. If a password is defined to ICAM, that password must be included on the network descriptor statement.

Example 3

```
NETWORK=NET3!IDESNETW!LNE3!2000!!!!!!!!!!!!!!NO
```

Identifies network NET3 with the password IDESNETW. The remote station uses LNE3 and operates in 2000 mode. Skip to top-of-form is inhibited, but breakpoints to the printer and card punch are to be issued. Note that blanks are used as separators in this example.

2.5.2. Example Job Control Stream for Initiating IDES

A typical job control stream for initiating IDES is shown in the following example:

```
1.  // JOB SLAV,,,,2
2.  // DVC 20   // LFD PRINTER
3.  // DVC 30   // LFD TY#CDIN
4.  // DVC 40   // LFD PUNCH
5.  // EXEC TZ$SME
6.  // EXEC TZ$SMF
    /$
7.  NETWORK=NET1,DDIDRIVE,LNE1,1004,00001
    /*
    /&
    //FIN
```

Explanation:

1. IDES requires two tasks to be specified on the JOB card. Note the four commas before the tasks parameter.

2. Device assignment for the printer. IDES requires the LFD-name PRINTER.

3. Device assignment for the card reader. IDES requires the LFD-name TY#CDIN.

4. Device assignment for the card punch. IDES requires the LFD-name PUNCH. (Optional)

5. Executes the IDES program that reads and punches EBCDIC card images.

6. Executes the IDES program that reads and punches Fieldata card images.

7. Network descriptor statement (see 2.5.1).

2.6. Console Control of IDES

As the IDES job begins to execute, the console operator must establish a physical connection with the remote host. If switched communications lines are used, the console operator must answer the ICAM dial message. When the communications link with the host is established, input/output transmissions can start.

Unsolicited console commands and console messages allow the operator to control the sequence of events in the operation of the remote batch environment. The console keyins and applicable modes are listed in Table 2-3 and described in 2.6.1, 2.6.2, and 2.6.3. Special forms messages are covered in 2.6.4.

2.6.1. Input Control Console Keyins

In the DCT 2000 and 2780/3780 disciplines, the ABTI keyin causes the remote device handler (RDH) to send an end of transmission (EOT) message to the host. In the 1004 discipline, input is aborted by using the ABPR or ABPU keyins. These keyins cause the RDHs to send the abort print and abort punch functions to the host. Transmission of the print or punch files being received by the System 80 is aborted at the discretion of the host.

When IDES is operating in the 1004 mode, the HALT keyin causes the RDH to send a halt message to the host. Following that, IDES displays a message on the console and operations are suspended until an operator response reinitiates activities.

Table 2-3. IDES Console Keyins

Keyin	Meaning	Mode		
		1004	2000	2780
ABPR	Abort printer	X		
ABPU	Abort punch	X		
ABTI	Abort input being received from host		X	X
ABTO	Abort output being sent to host	X	X	X
HALT	Temporarily halt communications	X		
OFLN	Terminate normally	X	X	X
SEND	Send card deck to host	X	X	X
TERM	Terminate	X	X	X
TRSP	Send transparent data			X

NOTE:

The absence of an X in the mode column for a specific keyin indicates that the keyin, if made while running in that mode, will be ignored.

2.6.2. Output Control Console Keyins

The IDES keyins to initiate and control output operations are:

- ABTO

  Causes the RDH to send an EOT or equivalent function to the host. The RDH returns a special status to IDES and output operations cease.

- HALT (1004 mode only)

  Causes the RDH to send a halt message to the host. Following that, IDES displays a message on the console and operations are suspended until an operator response reinitiates activities.

- SEND

  Initiates action to transmit a card deck to the host. In DCT 2000 or 2780/3780 modes, if input is active, the SEND keyin does not take effect until the current input is finished. In 1004 mode, however, input and output may be active immediately. Reading of a card deck is terminated by a 7-8 punch in columns 1 and 2; a job stream is terminated by putting "" in columns 1 and 2.

- TRSP (2780/3780 mode only)

  Initiates basically the same action as the SEND keyin. The TRSP keyin is used to initiate the transmission of a job stream to the host in transparent mode.

2.6.3. Termination Console Keyins

Keyins used to terminate IDES operations are:

- OFLN

  This keyin causes a normal termination of operations. If operating in 1004 mode, the termination does not take effect until all input and output operations cease.

- TERM

  If abnormal conditions dictate the need for a termination of operations, the TERM keyin is used to cause an abnormal termination.

2.6.4. Special Forms Message

When running with a spooling supervisor, IDES recognizes special forms messages being sent from the host system and displays an appropriate console message.

2.7. Console Messages

During IDES operation, both action and information messages are displayed. If an action message is displayed, operator intervention is required.

Note: *See the* System Messages Reference Manual *(UP-8076) for a listing of all IDES messages.*

Special attention should be given to IDES console messages. When errors occur and a message is displayed, it may be necessary to retransmit or partially retransmit the card deck (if any) currently being sent to the host. Card or print images should be checked closely by the system operator to ensure integrity.

Section 3
Remote Batch Processing (RBP)

3.1. Overview

Remote batch processing (RBP) software allows you to submit jobs to a System 80 from a remote terminal and receive output at remote terminals. The remote batch terminals are called remote stations.

The following remote stations are supported for remote batch processing:

- DCT 2000 (Not applicable to the model 7E)
- 1004 (RMS1) (Not applicable to the model 7E)
- 9200/9300 (REM1) (Not applicable to the model 7E)
- UDS 2000 (batch mode binary synchronous device emulating an IBM 2780/3741)
- IBM 2780/3780 (ASCII and EBCDIC nontransparent modes)
- UNIX system

To use remote batch processing, you generate an ICAM symbiont that includes a remote batch processing (RBP) network. The network definition for RBP contains special macros that identify valid remote users to the system. The ICAM/RBP symbiont can be loaded by the system console operator or automatically by OS/3 when spooled output is available for transmission to a remote station.

You submit jobs from a remote station by entering special RBP commands to start a work session, followed by one or more job control streams. You can direct output to a central installation printer, to yourself, or to an alternate user at another remote station. You can also submit jobs from the central computer and direct output to a remote station. To direct the job's output to a remote printer or punch, you include a DST job control statement in the job control stream.

In addition to submitting jobs and receiving output, RBP commands allow you to:

- Control the operating environment of the remote station
- Control the input to the system (you can cancel a card reader jam and continue processing without restarting the job)

- Monitor job status and receive automatic notification of job completion
- Specify that job output be returned either immediately or on command
- Stop job output to the remote station and either cancel or restart it on command
- Send messages to another remote station or to the central site operator

The system console operator communicates with RBP through console commands. These commands allow the operator to:

- Shut down the remote batch processing network
- Restart batch processing
- Read batch jobs from the central card reader on behalf of a remote station

Though card readers are not applicable on the model 7E, controls can be read via a console key-in.

3.2. System Generation Requirements

You must create an ICAM symbiont that includes a remote batch processing network to process commands and handle job streams submitted for a remote station. To generate the ICAM symbiont, you code the parameters in the COMMCT phase of system generation (SYSGEN).

Note: *The COMMCT parameters include the network definition macros described in the* Models 3-6 and 8-20 ICAM Operations Guide *(UP-9745) and in the* Model 7E ICAM Operations Guide *(70023908), and also the message control program (MCP) parameters described in the* OS/3 Installation Guide *(UP-8839).*

You can define more than one network in your ICAM symbiont by including several sets of network definition macros. You can create an ICAM with only an RBP network or with an RBP network and other type of networks.

You submit the COMMCT parameters to the SYSGEN parameter processor (SG$PARAM) and then run the SG$COMMK job control stream to assemble and link the ICAM symbiont.

Note: *For details on the SYSGEN process, see the* OS/3 Installation Guide *(UP-8839).*

Remote batch processing requires a supervisor generated with the SPOOLING=REMOTE or SPOOLING=DDP parameter in the SUPGEN phase. This parameter generates all spool file subdirectories required by RBP.

You should also specify the SPOOLICAM parameter in the SUPGEN section to identify the ICAM symbiont to be loaded automatically when spooled output becomes available for transmission to a remote station (if another ICAM is not already loaded).

3.3. Defining an RBP Network

When you define a remote batch processing network, you use nine macros. Five of these (BUFFERS, CCA, LINE, TERM, and ENDCCA) are regular ICAM network definition macros described in the *Models 3-6 and 8-20 ICAM Operations Guide* (UP-9745) and the *Model 7E ICAM Operations Guide* (70023908). Four other macros described in this section (BFILES, RBEGIN, RNAME, and REND) are special for RBP. You specify them just before ENDCCA.

Following is the order of presentation required for an RBP network definition and the purpose of each macro:

Macro	Meaning
CCA	Start of a network definition and its type (RBP1)
BUFFERS	Specifies resources required for the network
$LINE_1$	Characteristics of this line
TERM	Defines the remote batch device or terminal on this line
.	
:	
$LINE_n$	Characteristics of last line in this network
TERM	Defines the remote batch device or terminal on this line
BFILES	Extends the number of main storage staging areas for high traffic situations
RBEGIN	Begins a userid table for RBP
$RNAME_1$	Specifies the userid, account number, and optional password for each valid user
.	
:	
$RNAME_n$	
REND	Specifies the end of the userid table
ENDCCA	Specifies the end of this network

3.3.1. Standard Network Definition Macros for RBP

Table 3-1 summarizes the five standard ICAM network definition macros used in an RBP network. Note that the CCA macro specifies the TYPE operand as TYPE=RBP1 and the BUFFERS macro describes special considerations for the number of network buffers you need for each line. These are the only special considerations for defining an RBP network.

Table 3-1. ICAM Network Definition Macros for RBP

Macroinstruction	Operand	Meaning
BUFFERS	num,size	Number of network buffers and size of each in 4-byte words NOTE: Specify at least 7/line if input and output traffic flow at the same time; at least 6/line if input and output traffic alternate. Size must be in multiples of 64 words.
	,,ARP=integer	Number of activity request packets (ARPs) you want made available to ICAM to run this network
	[,STAT=YES]	Includes software to keep track of network buffer and ARP pool activity
	[,THOLD=n]	Includes software for threshold processing of activity request packets
CCA	TYPE=(RBP1)	RBP1 means RBP software support.
	[,PASSWORD=password]	1- to 8-character password to further identify this network
	[,CCAID=identifier]	1- to 8-character identifier you want printed on top of network definition listing and summary report at the end of listing to identify CCA code
ENDCCA		Indicates end of a network definition
LINE	DEVICE={(DCT1000,BATCH) DCT2000 [,{64 / 132}] (1004) (BSC,{line-buffer-length / 216}[,{ASCII / EBCDIC / TRANSCOD}]) (9200) (9300)}	Describes type of terminal (remote station) used on this line
	[,TYPE=([line-speed] [,{UNAT / AUTO}] [,SWCH] [,SYNC])]	Specifies characteristics of this communications line
	[,CALL=phone-number]	Telephone number used for automatic or operator manual dialing to a remote batch device or terminal
	[,ID={ca-port-number / slca-number}]	Port on communications adapter or single line communications adapter that this line uses
	[,DIALER=(ca-port-number)[,eon]]	Specifies port on communications adapter to which an automatic dialer is attached.
	[,LBL=line-buffer-length]	Line buffer length in 4-byte words

continued

Table 3-1. ICAM Network Definition Macros for RBP (cont.)

Macroinstruction	Operand	Meaning
	[,RETRY=(input-number,output-number)]	Specifies number of retries you want ICAM to make before assigning a down status to the remote batch device or terminal on this line
	[,STATS=YES]	Specifies that you want ICAM to accumulate statistics for the remote batch device or terminal on this line
	[,TIMEOUT=(input-time,output-time)]	Specifies amount of time in seconds you want ICAM to wait before time-out occurs
LINE (cont)	[,XLATE=([{labeli,idi / NO / YES}][,{labelo,ido / NO / YES}])]	Specifies translation tables you want ICAM to use
TERM	FEATURES=((1004 [,,{DLT3 / DLT1}]) (DCT1000) (DCT2000[,128] [,SBLK]) 9200 [,{96 / 120 / 132}] [,,{DLT3 / DLT1}] 9300 [,,{DLT3 / DLT1}] ({3780 / 2780 / BSC} [,max-output-record-length] [,{max-no-records/block / MULTI}] [{SECOND / PRIMARY}] [,max-block-length] [,MODEL1 / MODEL2 / MODEL3 / MODEL4]))	Specifies type of remote station and its characteristics
	[AUXn=({PTP / PTR / RDR / PCH / PRNTR})]	Specifies peripheral equipment attached to remote batch device or terminal; i.e., paper tape punch (PTP), paper tape reader (PTR), card reader (RDR), card punch (PCH), or printer (PRNTR)
	[,XLATE=([{labeli, idi / NO / YES}][,{labelo,ido / NO / YES}])]	Specifies translation tables to be used for this remote batch device or terminal only
	[,ANSWER=(int,C,text)]	Specifies a site identification code (int is number of characters in text)
	[,INRECSZ={80 / 96 / 128}]	Specifies input record size (characters) for this remote batch terminal

Note: *Refer to the* Models 3-6 and 8-20 ICAM Operations Guide *(UP-9745) or the* Model 7E ICAM Operations Guide *(70023908) for details of each macro.*

The following network definition example uses a DCT 2000 with an auxiliary device card punch. This allows job output to be received as punched cards. An explanation follows the example.

```
      1         10      16                                          72
1.  | NET2      CCA     TYPE=(RBP1)
    |                   BUFFERS  8,64,1,ARP=8
2.  | L01       LINE    DEVICE=(DCT2000),TYPE=(2400,SYNC),ID=8
3.  | T001      TERM    FEATURES=(DCT2000),AUX1=(PCH)
4.  |           RBEGIN
5.  |           RNAME   PHILA,6923
    |           RNAME   PHILA,1120
    |           RNAME   PHILA,7520,ENG
6.  |           REND
    |           ENDCCA
```

Explanation:

1. Identifies a fully resident remote batch processing network.
2. Specifies the input/output line handler to support a DCT 2000 in synchronous transmission on a dedicated line attached to port 8 of the single line communications adapter.
3. Specifies the input/output terminal as a DCT 2000 with auxiliary punch.
4. Specifies the beginning of authorized userid information (see 3.3.2).
5. Defines each valid remote batch processing user (see 3.3.2).
6. Specifies the end of userid information (see 3.3.2).

3.3.2. Special Network Definition Macros for RBP

In addition to the normal set of network definition macros, remote batch processing uses four special macros:

- Buffer message extension (BFILES)
- Begin userid table (RBEGIN)
- Identify a user (RNAME)
- End userid table (REND)

Buffer Message Extension (BFILES)

The BFILES macro extends the number of main storage staging areas used by remote batch processing for queueing messages and output files. This macro is required for networks in which high traffic situations may arise. The high traffic situations generally arise whenever status messages or output files are generated or requested faster than they can be sent over communications lines and printed or punched.

Format

LABEL	ΔOPERATIONΔ	OPERAND
	BFILES	RBPA=m,STAT=n

Operands

RBPA=m
: Specifies the total number of special remote batch processing activity request packets to be generated. Default is 10 times the number of remote stations defined in the network.

  These packets are used to schedule RBP activities. For each remote station, one activity request packet is needed for input, one for output, and one for the current activity. In addition, if output becomes available by means of requests or job termination, one activity request packet is needed for each job or job step that has output associated with it.

  If at any time this special remote batch processing activity request packet pool is depleted, the remote batch processing task is cancelled.

STAT=n
: Specifies the total number of status message buffers to be generated. Default is 10 times the number of terminals.

Begin Userid Table (RBEGIN)

The RBEGIN macro specifies the beginning of the userid table for remote batch processing.

Format

LABEL	ΔOPERATIONΔ	OPERAND
[symbol]	RBEGIN	

Identify a User (RNAME)

The RNAME macro specifies the name, account, and optional password for valid remote batch processing users. These specifications are for ICAM use only. They have no relationship to userid, account, and password specified at local or remote workstation logon.

Format

LABEL	ΔOPERATIONΔ	OPERAND
	RNAME	userid,acct[,password]

Operands

userid
: Is a 6-character identifier of a valid remote batch processing user.

acct
: Is a 4-character account number. Multiple accounts for a userid may be generated by submitting multiple RNAME macros with the same userid.

password
: Is an 8-character password that may be assigned in conjunction with a userid and account number.

End Userid Table (REND)

The REND macro specifies the end of the userid table for remote batch processing.

Format

LABEL	ΔOPERATIONΔ	OPERAND
	REND	

3.4. Remote Batch Processing Commands

Remote batch processing commands allow you to submit remote batch jobs, receive output, and communicate with the central processing system from a remote station. The commands are interspersed between job entries in the remote input stream. They are processed exclusively by the remote batch processing symbiont and are not passed to the OS/3 job control or spooling functions.

Each remote batch processing command is submitted by means of a punched card and is identified by a slash (/) in column 1 followed immediately by the command starting in column 2. One or more spaces must separate the command from the parameters.

The remote batch processing commands are:

```
RSTART
RSTOP
RLOGON
RLOGOFF
ROUT
RMSG
RSTATUS
```

To submit jobs and data from a remote station, you first attach the remote station to the system with an RSTART command, log on with an RLOGON command, and then begin entering job control streams. After the last job stream, you can enter the LOGOFF command or wait for your job's immediate output before logging off. Another user can then log on and begin entering jobs. The RSTOP command ends operation at the remote station.

If the output from your jobs is not printed before you log off, you can retrieve it later by using the ROUT command. You must also use this command to retrieve output from jobs submitted at the central processing system or at another remote station.

The RMSG command lets you send a message to the system console operator or to another remote station. The RSTATUS command lets you determine the status of jobs you submitted or for which you are to receive output.

The system console operator can also use the remote batch processing command to read jobs and data from the central card reader on behalf of a remote station and to communicate with remote stations. To use the RBP commands, the console operator first issues the RD console command to the ICAM symbiont. The RD command is described in 3.9.3.

3.4.1. Activate a Remote Station (RSTART)

The RSTART command logically attaches a remote station to the remote batch processing system. Once attached, the remote station can monitor the system for output directed to it and gain access to the central system by logging on. Also, the RSTART command identifies the remote station.

The RSTART command must be the first statement you submit at an inactive remote station. If you want to resume remote batch processing activity after your remote station has been logically detached from the system (after an RSTOP command or a system failure), you must resubmit the RSTART command.

Format

```
/RSTARTΔtermid
```

Parameter

`termid`
Specifies the remote station name as defined by the label of the TERM macro in the ICAM remote batch processing network definition. It consists of 1 to 4 alphanumeric characters, the first of which must be alphabetic. The remote batch processing software requires unique remote station names. This command is rejected if the system already has an attached remote station with this name.

3.4.2. Deactivate a Remote Station (RSTOP)

The RSTOP command allows you to logically detach a remote station from the system. Prior to detachment, all queued messages directed to the remote station are transmitted. No job output is returned to the remote station after the RSTOP command is processed. The last message transmitted indicates that the remote station is logically detached. No further communication occurs until the remote station resumes remote batch processing activity with an RSTART command. If the line connection is via the switched (dialed) network, the remote station is disconnected.

Format

```
/RSTOP
```

3.4.3. Begin Remote Batch Processing (RLOGON)

The RLOGON command identifies you to the remote batch processing system. The system validates the userid, account number, and password in the RLOGON command and, if accepted, grants access to the system. At this point, you can begin requesting the use of remote batch processing facilities.

During the processing of a valid RLOGON command, the spool file is automatically searched for any immediate output belonging to this userid. This output, including remote log, print, and punch files, is sent to the remote station without the need to input a ROUT command. If you do not want this automatic process, specify the RESTART=NO parameter.

The RLOGON command remains in effect until another RLOGON, RLOGOFF, or RSTOP command is issued.

You cannot log on at more than one remote station at a time. If you want to change remote stations, log off at your current remote station before logging on to a new remote station.

Format

```
/RLOGONΔuserid,acct[,password] [, {RESTART=YES} ]
                               [   {RESTART=NO } ]
```

Parameters

userid
: Specifies the 6-character identification of the user.

acct
: Specifies the 4-character account number.

password
: Gives the 8-character password as specified by the RNAME macro in the network definition.

RESTART=YES
: Indicates that the spool files should be automatically searched for any immediate output belonging to this userid.

RESTART=NO
: Specifies that the spool files should not be automatically searched. To obtain any immediate output already queued, you must issue a ROUT command. (You usually do this if you desire to begin output on a certain page or card.)

3.4.4. End Remote Batch Processing (RLOGOFF)

The RLOGOFF command terminates a remote batch processing job. It must always be used in conjunction with an RLOGON command when a remote station job is terminated. If you wish to change remote stations, you must log off at the current remote station before logging on at a new one.

Format

```
/RLOGOFF
```

3.4.5. Retrieve or Cancel a Deferred Job (ROUT)

The ROUT command allows you to retrieve deferred job output for which you are a valid recipient or to delete the job from the system. Output is resumed with the record that was being output when interrupted or at a specific page or card number. If the job is not complete when the command is processed, a message is returned indicating this and the command must be resubmitted after the job is completed.

The remote batch processing system returns an invalid request response if you are not a valid recipient or if the job is not in the system. You are automatically the recipient of the job output log for each job you submit; however, to be a valid recipient of print or punch output, your userid must appear on the appropriate DST job control statement.

You can delete only print/punch files for which you are a valid recipient. The remote batch processing system returns an invalid request if you attempt to delete files that do not belong to you. Files may be deleted at any time after they are completed.

Format

```
/ROUTΔ { J=jobname } [, { BEGIN [,P=±n][,C=±n] } ]
       { U=ALL     }   { CONTINUE              }
       { U=userid  }   { DELETE                }
```

Parameters

J=jobname
Indicates that the request is for the specific job named in the parameter.

U=ALL
Indicates that the request is for all output in the system for which you are a valid recipient. You receive all available output of jobs submitted by you and of jobs submitted by other users which name you as a valid recipient. If the userid ALL is in the RBP system, you receive all output for which you are a valid recipient.

U=userid
Indicates that the request pertains only to the jobs submitted by the named user. If you give your own userid, you receive or delete all output from jobs you submitted. If you give another userid, you receive output from all jobs submitted by the named user that designate you as a valid recipient.

BEGIN
Specifies that job output is to begin with the first record in the output file, whether or not output was previously discontinued.

BEGIN [,P=±n][,C=±n]
Specifies where printing or punching is to start when output resumes after being discontinued. This option is valid only with /ROUT J=jobname

where:

P=+n or P=n
Specifies the actual page number where printing of the suspended print file should resume.

P=-n

Specifies that printing should resume *n* pages previous to where the printing terminated.

Note: *When the terminated print file is the job log, printing resumes at the start of the log regardless of the P=±parameter.*

C=+n or C=n

Specifies the actual card number where punching of the suspended punch file should resume.

C=-n

Specifies that punching should resume *n* cards previous to where the punching terminated.

Any print or punch files for the specified job name that were not output before termination will begin printing at the beginning of the file when the command is executed.

CONTINUE

Specifies that job output is to resume with the record that was being written when the output was discontinued. If the job output is not in a discontinued state, output begins with the first record in the output file. CONTINUE is the default specification.

When this option is used after output files have terminated abnormally, the following actions occur:

- Files that were terminated abnormally resume output as follows:
  - Print files resume printing at the page following the last complete page indicated as having been successfully output.
  - Punch files resume punching with the card following the last card indicated as having been successfully output.

  When using buffered terminals, this may sometimes result in duplicate pages or cards when restarting due to the buffers being able to contain multiple pages or cards. However, there will not be any pages or cards lost.
- Files that were not output prior to termination, begin output at the beginning of the file.

DELETE

When used with J=jobname, specifies that the named job is to be deleted from the system. The job must belong to the user who is currently logged on at the remote station.

When used with U=userid, specifies that all current jobs submitted by the named user are to be deleted from the system. The only valid user is the one who is currently logged on at the remote station. The system returns a message specifying the name of each deleted job.

DELETE is not used with U=ALL.

3.4.6. Send a Message to Another User (RMSG)

This command allows you to send a message to the central processor console operator or to another remote station. You may direct messages to other remote batch processing users by means of their user identifier (userid), or to a remote station identified by a terminal identifier (termid).

Messages are delivered to active users only. An active user is one who is currently logged on. If you send a message to both a userid and a termid, remote batch processing attempts to send the message to the location defined by the userid first. If the userid is not active, remote batch processing attempts to send the message to the location defined by termid. If this location is not active, the message is rejected. Note that 2-line messages requiring two status message buffers are generated and sent to the specified destination. (Refer to the BFILES macro for more information).

Format

```
/RMSGΔM=C'text',U=userid,T=termid
```

Parameters

`M=C'text'`
Specifies the message text to be sent. The text must be framed by apostrophes and is limited to a maximum of 39 printable characters and blanks. Embedded apostrophes are not permitted. When a second apostrophe is detected, scan for text is halted.

`U=userid`
Specifies the user that is to receive the message. The message is rejected if the user is not active. A userid of OS3CTR directs the message to the central operator console.

`T=termid`
Specifies the remote station that is to receive the message. The message is rejected if the remote station is not currently attached to the system. The message is sent to the central computer console if you specify the name of the remote batch processing network. (The name of the network is the label of the CCA macro in the ICAM RBP network definition.)

3.4.7. Determine Job Status (RSTATUS)

This command allows you to determine the status of one or more jobs submitted from remote stations. Remote batch processing returns the status of only those jobs current in the system for which the requestor is a valid recipient. You can request status for a specific job, for jobs submitted by a specific user, or for those jobs submitted from your remote station that are currently in the system.

You receive a response for each job that satisfies the command. Each response contains the job name, submitting userid, submitting termid, destination userid, and one of the following indications:

- IN RUN PROCESSOR
- ON JOB QUEUE
- JOB IS EXECUTING
- JOB TERMINATED NORMALLY
- JOB TERMINATED ABNORMALLY
- JOB PROCESSED AND BEING RETAINED
- JOB NOT IN SYSTEM OR ALREADY PROCESSED
- OUTPUT AVAILABLE

Note: *Each response is a 2-line message and requires two status message buffers. (See BFILES macro earlier in this section.)*

Format

```
/RSTATUSΔ [ J=jobname ]
          { U=userid  }
          [ T=termid  ]
```

Parameters

J=jobname
: Specifies the job name for which status is requested. If you are not a valid recipient, the request is denied.

U=userid
: Specifies a userid for which job status is requested. If you specify a userid that relates to submitted jobs, you receive status for all jobs that you submitted under that userid. If you specify a userid that relates to a job recipient, you receive status for all jobs that pertain to that destination only.

This is often the same userid as the job submitter. For example, if you submit two jobs, one with a destination of USR1 and the other with a destination of USR3 (and both jobs are currently in the system), the status you receive will depend on the userid you submit. That is, the submitter userid receives status for both jobs, the userid for USR1 receives status for the job to be sent to USR1, and the userid for the job whose destination is USR3 receives status only for the job to be sent to USR3.

`T=termid`

Specifies this request is for all current jobs in the system that were submitted from this remote station. A report is made for all jobs in this category regardless of the submitting user.

3.5. Examples of the Use of RBP Commands at a Remote Station

Example 1

```
1. /RSTART  T001
2. /RLOGON  USERA,6944
3. // JOBA
   // DVC 20
   // DST USERA,OS3CTR
   // LFD PRNTR
       .
       .   Job control statements
       .
   /&
   // FIN
4. /RMSG    M=C'USERA SENDING TO CENTRAL',U=OS3CTR
5. /RLOGOFF
```

Explanation:

1. Remote station T001 is attached via the RSTART command.
2. USERA, account number 6944, is logged on at T001.
3. Job A is submitted by USERA from T001. The DST job control statement makes the print file available to the central printer (OS3CTR) and to USERA.
4. USERA sends a message to the central site.
5. End the work session.

Example 2

```
1.  /RSTART  RBP1
2.  /RLOGON  USERA,7476,RESTART=NO
3.  // JOB A
    // DVC 20
    // DST USERA,OS3CTR,USERB
    // LFD PRNTR
       .
       .   Job control statements
       .
    /&
4.  // JOB B
    // DVC 20 // LFD PRNTR
       .
       .   Job control statements
       .
    /&
    // FIN
5.  /RMSG M=C'USERA SUBMITTED JOB FOR USERB',U=USERB
6.
7.  /RLOGOFF
8.  /RLOGON  USERC,3853
```

Job control stream 1

Job control stream 2

Explanation:

1. Remote station RBP1 is attached via the RSTART command.

2. USERA, account number 7476, is logged on at RBP1. The user indicates that he does not want to receive output that has already been spooled for him.

3. JOB A, with at least one print file, is submitted by USERA from RBP1. The DST statement makes the print file available to USERA, USERB, and the central printer (OS3CTR).

4. JOB B, with at least one print file, is submitted by USERA from RBP1. The print file has no destinations specified, so it is sent only to the central printer.

5. USERA sends a message to USERB telling USERB he has submitted a job with USERB as a destination. USERB only receives this message if he is currently logged on.

6. USERA waits for his immediate output. At the completion of JOB A

   a. The log is printed back at RBP1 (USERA is still logged on).

b. The print file is printed at RBP1 and at the central printer.

c. The print file is available for USERB to request it.

At the completion of JOB B

a. The log is printed back at RBP1.

b. The print file is printed at the central printer.

7. USERA logs off.

8. USERC, account 3853, logs on at RBP1. He may now begin his run work session.

Example 3

```
1. /RSTART  RBP2                                  ┐
2. /RLOGON  USERB,1146,PASSWORD                   ├  Job control stream 1
3. /RSTATUS U=USERA                               ┘
4.                                                ┐
5. /ROUT    J=A                                   ├  Job control stream 2
6.                                                ┘
7. /RMSG    M=C'RBP2 SIGNING OFF',U=OS3CTR        ┐
8. /RLOGOFF                                       ├  Job control stream 3
9. /RSTOP                                         ┘
```

Explanation:

1. Remote station RBP2 is attached via the RSTART command.

2. USERB, account number 1146 password PASSWORD, logs on at RBP2.

3. USERB requests the status of all jobs submitted by USERA that have specified USERB as a destination.

4. Depending upon the response to the RSTATUS, USERB may reissue the RSTATUS (if job is executing) or may issue a ROUT command (if job has terminated).

5. USERB requests any output destined for him from job A (he knows the job name to request as a result of the response to the RSTATUS command).

6. USERB waits for the completion of the printing of the output.

7. USERB sends a message to the central console operator (OS3CTR) indicating that he is signing off.

8. USERB logs off. At this time another user could log on.

9. Remote station is detached from system via the RSTOP command. If the terminal is on a switched line, the line is disconnected.

3.6. Remote Station Logical States

All remote stations in the RBP system are considered to be in one of three mutually exclusive logical states: inactive, active, or processing. On the basis of the state of a remote station, the central system determines what communication is permitted. A remote station switches from one state to another by submitting one of the applicable remote station commands. The state of each remote station is independent of the state of any other remote station.

- Inactive Remote Station

  An inactive remote station is logically detached from the RBP system. All remote stations are in an inactive state at central system start-up and are placed in an inactive state when the system closes down. A remote station in an active or processing state becomes inactive by submitting an RSTOP command. The central system does not initiate any transmission to an inactive remote station. When a valid RSTART command is submitted, the remote station changes from an inactive state to an active state. If any other input is received from an inactive remote station, it is rejected.

- Active Remote Station

  Active remote stations are logically attached to the central system. Remote stations enter the active state when the central system receives an RSTART command from an inactive remote station or an RLOGOFF command from a processing remote station. The central system can initiate transmissions to an active remote station. These transmissions consist of messages queued for the remote station (RMSG command with T=termid parameter). Normal status messages and job output are not sent unless a user is logged on. RBP is conditioned to receive only an RLOGON or an RSTOP command from an active remote station. All other input commands are rejected.

- Processing Remote Station

  A processing remote station is one at which a user is logged on to the system. An active remote station enters the processing state when RBP receives a valid RLOGON command. All job entries and remote batch processing can be submitted at a processing remote station. Although one particular user is logged on at a remote station, the jobs submitted during his processing session can be directed to any of the valid system users. In addition, RBP sends output to a processing remote station if it is not currently submitting input. RBP remote station users should either use the RSTOP remote station command to disconnect the communications line or guarantee that there is no user currently logged on when the phone connection is broken.

3.7. Central Processor Device Logical States

In a remote batch processing system, the central processor card reader, line printer, and operator console are considered logical remote stations. Unlike stations located at remote sites, the central devices are always considered to be in a processing state with a fixed *userid* of OS3CTR and a fixed *termid* that is the remote batch processing network name specified at ICAM generation (the label of the CCA macro).

These IDs are valid operands for the RMSG and RSTATUS commands. Card reader input is initiated by an RD console command (see 3.9.3). Line printer output is controlled by the central spoolout mechanism.

3.8. Loading and Initializing the ICAM Symbiont

Because OS/3 supports only one active ICAM symbiont at a time, it is reasonable to have more than one ICAM symbiont available, depending on the needs of your system. You might have:

- A symbiont with a remote batch processing network only
- A symbiont with multiple networks

  For example, you may have a symbiont that contains a remote batch processing network plus additional networks to serve the needs of other programs. This arrangement allows remote batch processing to run concurrently with other communications programs. These other programs may use a global network or other dedicated networks, but they all have network definitions as part of the same ICAM symbiont.

- A symbiont with no remote batch processing network

  You run this symbiont when you do not need remote batch processing, but have other communications needs. A symbiont of this type must be terminated and the appropriate one loaded before batch processing can run.

If the remote batch processing ICAM symbiont is not loaded when needed, loading may be done automatically for output by OS/3 or manually by the console operator. If it is done by the operator, it is usually by voice communications with an operator at the remote batch terminal site, or according to a prearranged schedule. The console operator loads the ICAM symbiont by entering the 2-character symbiont name at the console.

If you want OS/3 to automatically load the symbiont containing the remote batch processor, you must specify the SPOOLICAM system generation parameter when you generate your system. This parameter defines the name of the ICAM symbiont you load when a program with remote batch output terminates and the needed ICAM symbiont is not already loaded.

If no symbiont is currently loaded, the symbiont named in the SPOOLICAM parameter is loaded. If there is a symbiont already loaded (and it is not the one specified), output is held until the currently loaded symbiont terminates. The operator is advised of this situation with this message:

```
REQUEST FOR SYMBIONT xx IGNORED yy ALREADY LOADED
```

It is now up to the operator to terminate the current symbiont and load the symbiont for remote batch processing when practicable. Alternately, you can use the SPOOLICAM parameter to specify that the operator must be notified to load the symbiont manually.

Notes:

1. *See the* OS/3 Installation Guide, *UP-8839, (models 3-6 and 8-20 users) or the* Model 7E Installation Guide, *70023858, (model 7E users) for details on the SPOOLICAM parameter.*

2. *OS/3 also provides an operator command (SETIC) to permit you to directly change the name of the symbiont specified in the SPOOLICAM parameter. It is described in the* OS/3 Operations Guide, *UP-8859, (models 3-6 and 8-20 users) and in the* Model 7E Operations Guide, *70023866, (model 7E users).*

After the ICAM symbiont is loaded, initialization begins. When initialization is complete, the ICAM READY message appears at the central processor operator console. At this time, remote batch processing initialization begins; that is, an automatic network request is issued and all lines are allocated.

If there is an error during initialization processing, or if proper spooling (input, output, and remote) is not configured in the current supervisor, error messages are displayed at the central processor operator console.

Assuming that remote batch processing initialization is completed successfully, connections to the various remote stations can now be made. (If you have a direct connect (private) line, no intervention is required.) For any line for which the computer answers the phone (UNAT), the MC#98 message is displayed at the central console. The remote station operator can dial in any time after this message is displayed. For any other switched line, the MC#03 message is displayed.

The central processor console operator must dial the appropriate remote station, then reply Y when the connection is made. The MC#03 message must be answered before ICAM/remote batch processing can process input or output for any line or service that any other operator requests.

When a connection is established, the MC#100 message is sent to the remote station. Once this message is received, you may begin sending input. The MC#100 message is the key to sending input, even for those devices that support input and output simultaneously.

Once remote batch processing begins to accept input, it continues to do so until EOT is received. When EOT is received, all status messages resulting from the input are displayed at the remote station, followed by the MC#100 message.

If output is currently being printed/punched, these messages cannot be displayed; therefore, remote batch processing does not solicit any input until output is complete and the MC#100 message can be sent to the remote station.

The remote batch processing symbiont is removed from the OS/3 system after the remote batch processing activity is complete. This is determined by voice communications between the central processor and remote station operators. Remote batch processing shutdown is described in 3.9.1. Operations may be reestablished by reloading the remote batch processing ICAM symbiont or by using the RB console command as described in 3.9.2.

3.9. How the Console Operator Communicates with RBP

The OS/3 console operator communicates with the remote batch processing ICAM symbiont by means of console commands. These commands enable the operator to specify functions he would like performed. The following describes each command. The format of a console command is:

```
00Δ { Cn } Δcc[,x]
    { Mn }
```

where:

Cn or Mn
: Is a 2-character remote batch processing ICAM symbiont identifier (C1-C9) or (M1-M9)

cc
: Is a 2-character command code

x
: Is an optional operand associated with the command

The remote batch processing command codes are:

- SH

  Specifies shutdown and removes the ICAM symbiont from the system when remote batch processing is the only active ICAM interface. The SH command idles remote batch processing when other ICAM interfaces are active in the same ICAM symbiont.

- RB

  Reactivates remote batch processing after it has been idled by an SH command.

- RD

  Starts the reading of batch jobs or data files from the central processor card reader under control of remote batch processing.

3.9.1. Shutting Down RBP (SH)

The SH (shutdown) command enables the console operator to initiate an orderly shutdown of the remote batch processing network. When the console operator issues this command, all remote stations are placed in an inactive state. Remote batch processing waits until all active spooling to remote stations is completed; then it closes the remote batch processing network and releases all remote batch processing communications lines.

If the ICAM symbiont containing the remote batch processing network supports other networks (that is, other interfaces in addition to remote batch processing), the remote batch processing network idles itself. If all networks in the symbiont become inactive (the programs using them have terminated), the ICAM symbiont terminates.

If the ICAM symbiont supports remote batch processing only and shutdown is requested, the symbiont terminates itself from the system when shutdown processing is complete. The console operator is notified of symbiont termination.

While remote batch processing shutdown is in progress, no remote batch processing commands are accepted. If the console operator attempts to enter any remote batch processing commands, the system responds with a message indicating the requested user could not be found. Following shutdown, the only console command accepted by remote batch processing is the RB command.

The complete format of the shutdown command is:

```
00Δ { Cn } ΔSH[,D]
    { Mn }
```

where:

D

Is the optional dump operand that indicates to the system that a dump is to be produced during termination of the ICAM symbiont. Note that the console operator cannot obtain a dump of an ICAM symbiont by using the OS/3 CANCEL command. This is because the CANCEL command always results in an end-of-symbiont condition.

3.9.2. Reactivating RBP (RB)

The RB (remote batch) command enables the console operator to reactivate remote batch processing from its idle state after shutdown processing is complete. The ICAM symbiont that supports the remote batch processing network must still be in the system. (See 3.9.1 for considerations governing ICAM symbiont termination resulting from shutdown processing). The RB command performs remote batch processing initialization procedures described in 3.8. Following initialization, remote batch processing is ready for processing. The complete format for the RB command is:

```
00Δ { Cn } ΔRB
    { Mn }
```

RB specifies that remote batch processing is to be activated.

3.9.3. Reading Batch Jobs or Data Files from the System Card Reader under Control of RBP (RD)

The remote batch processing RD (read) command enables the operator to read batch jobs or data files from the central card reader under control of the remote batch processing software. Normally, OS/3 provides for reading batch jobs and data files that are related only to central site processing. The RD command may be used for this purpose. However, its main purpose is to provide a mechanism for reading batch jobs and data files from the central card reader on behalf of a remote station.

When the operator issues the RD command, batch jobs and data files entered from the central card reader are interpreted as having come from a remote station. If the first statement read is RSTART, remote batch processing assumes the central site is running on behalf of the remote station specified in that statement. If that remote station is currently inactive, the command is accepted and all subsequent statements, up to RSTOP, are processed as though they were submitted from the remote station.

The RD command may also be used to initiate the printing or punching of output at a remote station without the intervention of the remote station operator. This can be done provided the communications line is connected and the remote station is currently inactive.

When the RLOGON command is processed with the RESTART=YES option, the log file is automatically searched for any immediate output for that userid. Output is initiated internally, without the requirement of a ROUT command. This procedure is especially useful following a warm restart when the remote station operator does not know when the central system has come back up and if immediate output is still to come. This process is not advisable if there is immediate output that was interrupted and is to be restarted on a specific page or card. The ROUT command may be used in conjunction with the RESTART=NO option of the RLOGON command to initiate the printing or punching of specific output at the remote station.

The RD command allows the console operator to send messages to remote stations. The console operator does this by including the proper RMSG command in the input stream, just as the operator of a remote station would do.

The console operator can also use the RD command to request status of remote batch jobs. The console operator does this by including the proper RSTATUS command in an input stream that is not prefaced by a RSTART command. The complete format of the RD command is as follows:

```
00Δ { Cn } ΔRD
    { Mn }
```

The following examples illustrate the use of the RD console command to read input streams from the central site under control of the RBP software.

Example 1

This input stream enables the console operator to submit a remote batch job from the central card reader on behalf of a remote station:

```
/RSTART  X
/RLOGON  USER#X,1234
// JOB    A
   .
   .
   .
/&
// FIN
/RLOGOFF
/RSTOP
```

Example 2

This example initiates output at the remote station following a warm restart of the OS/3 system:

```
/RSTART  Y
/RLOGON  USER#Y,1234,RESTART=YES
/RMSG    M=C'OUTPUT FOR STATION Y INITIATED',U=OS3CTR
/RLOGOFF
/RSTOP
```

Example 3

This coding initiates specific output for JOB A at remote station Y, following a warm restart of the OS/3 system:

```
/RSTART  Y
/RLOGON  USER#Y,1234,RESTART=NO
/ROUT    J=A,BEGIN,P=20
/RSTOP
```

Example 4

This example sends a message to a remote station:

```
/RMSG  M=C'SYSTEM SHUTDOWN-PLEASE LOGOFF',T=X
/RSTOP
```

Example 5

This example requests the status of all jobs currently in the system that were submitted at remote station X:

```
/RSTATUS  T=X
/RSTOP
```

3.10. Job Control Considerations for RBP

When entering remote batch jobs and data, there are several job control considerations. These are:

- DST job control statement
- Job control stream delimiters
- DATA job control statement

3.10.1. Sending Output to Remote Terminals (DST)

The DST statement gives the destination of the spooled output file and is included in the device assignment set for the print or punch file. You can send output to more than one destination by listing several destinations on one DST statement or by including several DST statements. When you omit the DST statement, job output is directed to the central printer or punch. However, the job log is printed at your site.

The format of the DST statement is:

```
//[symbol] DST dest-1[,dest-2,...,dest-16]
```

The dest parameter is 1 to 6 alphanumeric characters and must match a userid defined by an RNAME macro in the ICAM network definition. In addition to valid userids, the name OS3CTR or CENTRAL can be used to specify the printer at the central site.

The following examples send printed output to destinations A, OS3CTR, C, and D. Example 1 lists all destinations on one DST statement; example 2 uses multiple DST statements.

Example 1

```
// JOB REMOTE
     .
     .
     .
// DVC 20
// DST A,OS3CTR,C,D
// LFD PRINT
     .
     .
     .
// EXEC PROG1
/&
```

Example 2

```
// JOB REMOTE
     .
     .
     .
// DVC 20
// DST A
// DST OS3CTR
// DST C,D
// LFD PRINT
     .
     .
     .
// EXEC PROG1
/&
```

3.10.2. Job Control Stream Delimiters (/& and FIN)

The remote batch processor writes the job control stream (starting with a JOB statement and ending with a FIN statement) to a job control stream (JCS) spool file. The run processor reads the job stream from the spool file.

Each job does not have to end with a FIN statement. The /& statement signals the end of a job but does not close the spool file. The FIN statement causes RBP to close the spool file and call the run processor. When you end each job with a FIN statement, you create a separate spool file for each job, and a separate copy of the run processor is called in to read each job.

The following three examples illustrate the use of the job control stream delimiters /& and FIN.

Example 1

This example shows how you create two separate spool files. In this example, multiple copies of the run processor are in main storage concurrently.

```
// JOB A          ┐
   .              │
   .              │ JCS file created
   .              │
/&                │ Run processor called
// FIN            ┘
// JOB B          ┐
   .              │
   .              │ JCS file created
   .              │
/&                │ Run processor called
// FIN            ┘
```

Example 2

This example creates a single spool file. The /& statement signals the end of JOB C but does not release the run processor.

```
// JOB C          ┐
   .              │
   .              │
   .              │
/&                │ Single JCS file created
// JOB D          │
   .              │ Run processor only called once
   .              │
   .              │
/&                │
// FIN            ┘
```

Note that when you omit the FIN statement, the run processor can proceed to the next job (or jobs) in attempting to complete processing of embedded data.

Example 3

In this example , the /* statement, signaling the end of embedded data, is missing in JOB 1. The run processor searches all the way through JOB 2 looking for it. The result is that JOB 1 does not get scheduled and there is no trace of JOB 2. This situation results whether the jobs are submitted from the central site or the remote station.

```
// JOB 1
   .
   .
   .
/$
   .
   . <---------- Note:  /* not included (missing)
   .
/&
// JOB 2
   .
   .
   .
/&
// FIN
```

3.10.3. Entering Data from Remote Terminals (DATA)

You use the DATA job control statement to submit a data file from a remote station (or from the central site on behalf of a remote station). The DATA statement identifies the file and is followed by the data. The end of the data file is signaled by a FIN statement or another DATA statement. The remote batch processor writes the data (but not the DATA and FIN statements) to an input reader spool file. Your jobs access data from this spool file.

The format of the DATA statement for remote batch processing is:

```
// DATA FILEID=file-identifier[,RETAIN]
```

The job control stream must contain a device assignment set for a reader, (e.g., DVC 30). If you've included an LBL job control statement in the device assignment set, the file identifier specified on the DATA statement must match the LBL statement's file identifier. If there is no LBL statement, the file identifier on the DATA statement must be a concatenation of the job name and the file name from the LFD job control statement. Either way, an association is made between the file you define in your job control stream and the subfile.

If this is the control stream...

```
// JOB BALANCE
// DVC 30
// LBL SPOOL1
// LFD READ
     .
     .
     .
// EXEC PROGA
/&
// FIN
```

you code this DATA statement:

```
// DATA FILEID=SPOOL1
```

If this is the control stream...

```
// JOB BALANCE
// DVC 30
// LFD READ
     .
     .
     .
// EXEC PROGB
/&
// FIN
```

you code this DATA statement:

```
// DATA FILEID=BALANCEREAD
```

The RETAIN parameter is used to maintain the subfile after it is processed. If you specify RETAIN, only the DE SPL,RDR console command can delete the subfile. The following example shows the use of the RETAIN parameter:

```
// DATA FILEID=BALANCEREAD,RETAIN
     .   ]
     .   } Data statements
     .   ]
// FIN
```

Any parameters on the DATA statment other than FILEID and RETAIN are ignored by the remote batch processor.

Note that data file input and job input cannot be mixed. If a JOB statement is encountered while an input reader RDR spool file is open, the job input is spooled to the open file. If a DATA statement is encountered while a JCS spool file is open, the control statement may be ignored completely (if it immediately follows a /&) or written to the open file.

The following example shows the correct way to enter a job control stream and the data it requires:

```
// DATA FILEID=MYFILE,RETAIN
   .
   .                                  Data file spooled
   .
// FIN
// JOB yyyy
// DVC 30
// LBL MYFILE                         Job spooled
// LFD READ
   .                                  Run processor called
   .
   .
// EXEC PROGC
/&
// FIN
```

Note that a job may access an existing data file that has been retained; the file has been retained by specifying the RETAIN parameter on the DATA statement.

3.11. Remote Terminal Considerations

3.11.1. Special Considerations for the IBM 2780/3780

Because the IBM 2780 and 3780 hardware do not allow the overprinting of a line, a print-and-no-advance function results in a print-and-advance of one line. This means that when you send output to an IBM 2780 or 3780, you should not attempt to overprint (print-and-no-advance followed by a print-and-advance). If you do this, the second line is always printed on the line following the first line. This is a basic problem with any function that ends in no advance; for example, space-before-printing or skip-to-channel-n-before-printing. Note that output to an IBM 2780 or 3780 is printed differently than output to a central printer.

Remote batch processing does not support transparent data transmission.

3.11.2. Special Considerations for the 9300 System

- REM1 READY Message

  The REM1 READY message is printed by REM1 on the 9300 printer without any spacing before or after the print line. The OS/3 BATCH MONITORING message is preceded by a home paper and followed by 34 blank lines (so the message can be seen above the print bar at both the 9300 and 1004). Depending upon the order in which these messages are placed in the 9300 line buffer, the REM1 READY and OS/3 BATCH MONITORING messages may be printed on the same line, one on top of the other.

- @RUN Card

  The @RUN card is required by REM1 to process all input transmissions except the first. Since this card is not a remote batch processing command, one of several error messages should result if it is the first card processed by remote batch processing in any given input session. However, the remote batch processing software ignores the @RUN card if it is the first card in an input transmission and no remote batch processing status message results.

3.11.3. Special Considerations for the DCT 2000

If output is to go to the DCT 2000 punch, the BEL alarm sounds after the terminal operator has responded to the MOUNT SPECIAL CARDS message, if it is required. The terminal operator then readies the device for punching and sends the appropriate input response.

When punching is complete, the alarm does not sound to notify the terminal operator to return to print mode. If the terminal operator does not set the device back to print mode before the OS/3 BATCH MONITORING message has completed its retries, the message cannot be printed; however, input can be sent if the device is placed in transmit mode.

3.11.4. Special Considerations for the 1004

The 80-column card codes for the 1004 differ from those other card readers supported on the system. Table 3-2 lists the code differences. Remote batch processing users of the 1004 should be aware of these differences.

Table 3-2. 1004 Card Code Differences

1004 Graphics	EBCDIC Extended Hollerith	1004 (80-Column Card Code)
+	12–8–6	12
?	0–8–7	12–0
! (exclamation)	11–8–2	11–0
& (ampersand)	12	2–8
=	8–6	3–8
' (apostrophe)	8–5	4–8
: (colon)	8–2	5–8
>	0–8–6	6–8
@	8–4	7–8
)	11–8–5	12–4–8
[	12–8–7	12–5–8
<	12–8–4	12–6–8
#	8–3	12–7–8
]	8–7	11–5–8
△	12–8–2	11–7–8
≠	11–8–7	0–2–8
(	12–8–5	0–4–8
%	0–8–4	0–5–8
¤	0–8–5	0–7–8

3.11.5. Special Considerations for UNIX Systems

The only special requirement for running the UNIX system with RBP is that the 2780/3780 feature must be installed on the UNIX system. The UNIX system is then defined in ICAM as a 2780 or 3780 terminal operating on a synchronous communications line. For example:

For the UNIX system, define a BSC communications line

```
LINE DEVICE=(BSC...)...
```

and define the UNIX system as a 2780 or 3780 terminal

```
TERM FEATURES=(2780...)...
```

or:

```
TERM FEATURES=(3780...)...
```

For forms control, specify the FORMS parameter in the configuration file:

```
FORMS filename
```

The forms overflow code is 12. The home paper code is 1. The following are examples of forms files for 5 1/2-, 8-, and 11-inch forms:

	Line Number	Code
5 1/2-inch	31	12
	33	1
8-inch	44	12
	48	1
11-inch	63	12
	66	1

3.12. Remote Batch Processing Messages

All remote batch processing status messages are prefixed with an ICAM identifier (MC#*nnn*) and a time stamp (*hhmm*).

Note: *Refer to the* OS/3 System Messages Reference Manual *(UP-8076) for remote batch processing messages.*

The maximum number of status messages that remote batch processing can stage in main storage at any one time is 10 times the number of terminals. In addition, the maximum number of job complete notifications that remote batch processing can successfully handle at any one time is eight. If either of these conditions is exceeded, remote batch processing omits the status message output to the remote station or it does not process the job complete notification from the supervisor. You can extend these limits by specifying the BFILES macro in your remote batch processing/ICAM network definition.

3.13. Remote Batch Processing Error Recovery

Classes of error conditions are

- Communication line errors (recoverable/unrecoverable)
- System software faults
- Online hardware errors

The only unrecoverable line error considered by remote batch processing is a disconnect. In a disconnect, remote batch processing always issues a line release (LNEREL) followed by a line request (LNEREQ) so that the remote station operator can reestablish a connection when desired. Remote batch processing also issues this sequence when an RSTOP command is received and all status messages accumulated have been printed at the remote station.

Recoverable line errors such as software time-outs do not cause the line to be disconnected. For instance, a card jam can be recovered without resubmitting the complete input stream. If recovery action is not available at a remote station or the operator does not recover from the error in the time frame allocated, the remote device handler notifies remote batch processing with a line-down status. When the error is corrected, the remote device handler can accept input again. If a job stream or a data file is being spooled, the operator need only submit the remainder of the input stream. All subsequent input is spooled to the current file until a FIN statement is read.

Similar errors can occur during output to the remote device. If recovery action is not available at the remote station or the operator does not recover from the error in the time frame allocated, the remote device handler notifies remote batch processing with a line-down status. Remote batch processing closes the active file (it is not deleted) and starts looking for new input.

Discontinued output can be restarted from the beginning, from a specific page or card (card is supported only on models 3-6 and 8-20), or it can be continued following the last valid output record by the remote operator submitting the ROUT command.

The second and third classes of error conditions are recovered by using the restart capability in the OS/3 spooler. Any remote batch processing remote print or punch files recovered by a warm restart at initial program load (IPL) may be reinitiated by the remote station operator after he has reestablished connection and logged back on at the remote site by submitting a ROUT command. Immediate files can be automatically recovered through the RESTART feature of the RLOGON command.

The RSTART, RSTOP sequence (which may be submitted from the central processor card reader using an RD command) allows the central processor operator, following a warm restart, to initiate output that belongs to a remote userid without remote station operator intervention.

If an RSTART command is read in through the central processor card reader and the remote station (identified in parameter 1) is currently active (that is, RSTART was processed from the remote station), the RSTART command is rejected. All succeeding input is processed as coming from the central processor card reader (OS3CTR) and not a remote station.

Once a remote station operator identifies his terminal by sending the RSTART command, he can receive output by using the RESTART=YES parameter on the RLOGON command or via the ROUT command. The central processor console operator cannot initiate output for him.

3.13.1. Reader Out of Cards

If the card reader runs out of cards while a job stream or data file is being spooled, the spool file remains open until additional input is transmitted. All new input is spooled to the currently open file until a FIN statement is detected.

3.13.2. Card Reader Jam

If, during input, the card reader jams, the terminal operator should correct the problem and submit the remainder of the input. Processing is the same as that described in 3.13.1.

3.13.3. Printer Out of Forms or Paper Jam

To continue printing without sending a ROUT command, the remote station operator must halt the output device before the ICAM remote device handler returns a time-out status to remote batch processing. This is accomplished by sending a HALT keyin from the 1004 or 9300. Once the problem is corrected, printing is resumed by sending a READY from the 1004 or 9300.

If a time-out status is returned to remote batch processing, the line is marked down and the spool file closed. The line is marked up automatically when the device begins to respond to the polls, but a ROUT command is required to resume output. (See 3.4.5 for a description of the different recovery capabilities.)

3.13.4. Punch Out of Cards or Card Jam

For the 1004 or 9300, punch output error processing is the same as printer output error processing (3.13.3).

3.14. User Error Recovery

Most error recovery is automatically handled by remote batch processing error routines. However, the following error conditions require you to intervene:

- For any of the LINE DOWN or TERM DOWN messages, remote batch processing goes into an elapsed-time-place-to-go (ETPTG) sequence. In this sequence, remote batch processing issues a HOME PAPER command followed by the OS/3 BATCH MONITORING message every 60 seconds. When the output is successful and additional input can be received, a TERM MARKED UP message is sent to the system console.

- Each remote device handler has its own retry count that is performed before a terminal or remote device is set down. When remote batch processing is notified of the downed terminal, the associated output spool file is repositioned to the top of the current page before it is closed; however, the input spool file remains open unless the error was a disconnect.

  - To receive output following a downed terminal situation, the ROUT command must be issued.

  - To continue input, only the additional input is sent. The entries in the job control stream are not resubmitted.

- When a disconnect is received, a line release followed by a line request sequence is automatically executed. The operator at the control processor is notified of this condition. When the remote station operator wants to resume remote batch processing activity, the start-up sequence (RSTART, RLOGON), along with the other commands, must be issued.

3.15. Format of RBP Output at the Remote Station

3.15.1. Spacing after the Last Message

When output is complete (that is, remote batch processing status messages and remote print and punch files are processed), the OS/3 BATCH MONITORING message is sent to the remote station. Following this message, the paper is advanced so that the message can be read above the print bar at the 9300 and 1004.

This same spacing takes place when the MOUNT SPECIAL FORMS/CARDS message is printed at a remote station. The terminal operator must be able to read the message to know the type of paper or cards to be used.

3.15.2. Home Paper Considerations

Remote batch processing homes paper before it begins output (that is, before status messages are printed or before job output begins if there are no remote batch processing status messages to print). If file separators are desired, remote batch processing homes before and after it prints the file separator.

If remote batch processing does not print file separators, it does not home paper before or after printing any file except for the log, which it does home before printing. It is the responsibility of the creator of the output file to issue a top of forms command at the start of the file if he wishes to ensure output will begin at the top of the form, and file separators are not being printed.

3.15.3. Invalid Output DICE Sequences

The ICAM remote device handlers scan the output text for device independent control expression (DICE) sequences to control the line spacing. The DICE sequences consist of an X'10' followed by three space control characters. The output of a file containing X'10' characters that are not a part of a valid DICE sequence may incur line spacing problems at the remote station. The user is cautioned that such character sequences may appear graphically as blanks when printed at the central printer.

3.16. Special Input and Output Functions

Remote batch processing uses the direct data interface (DDI) for input/output operations.

All special input functions that are passed by the remote device handler back to a direct data interface user are ignored by remote batch processing.

Remote batch processing continues to accept input until the end-of-message status is received.

For print file processing, all special output functions that are passed by the remote device handler back to a direct data interface user are ignored by remote batch processing except for the abort print function. (Currently, this function is available on the 1004 only.) When an abort print is received, the current file is repositioned and then closed. An abort print function is sent to the remote device handler indicating the end of the current print output.

For punch file processing, all special output functions that are passed by the remote device handler back to the direct data interface user are ignored by remote batch processing except for the illegal truncation and abort punch functions. (Currently, these functions are available on the 1004 only.) For illegal truncation, the special function field is cleared to zero and then normal output processing continues. An abort punch function is sent to the remote device handler indicating the end of the current punch output.

For status message processing, all special output functions that are passed by the remote device handler to a direct data interface user are ignored by remote batch processing.

3.17. Restrictions That Apply to Remote Batch Processing

The following restrictions apply to remote batch processing:

- If manual dialing is configured for a remote batch processing line, the response to the dial message must be Y. If the phone call is not made, a software time-out message will appear; however, it is possible to dial and make a successful connection at a later time.

- When remote batch processing is sending output, new input is not allowed. Input should only be sent by the remote station after the receipt of the OS/3 BATCH MONITORING message. On certain devices (1004 and 9200/9300), input and output processing can occur at the same time (that is, immediate output may be sent while input is still being accepted); however, once an input EOT is received, no more input is accepted by remote batch processing until the output has completed and the OS/3 BATCH MONITORING status message is sent. To avoid this problem, do not send additional input until receiving the status messages resulting from the previous input transmission. If an error occurs, further activity can be withheld as long as necessary.

- The DUMP parameter on the SPL job control statement is not supported by remote batch processing. If you wish to have the print or punch output written to a tape and then later retrieved, you have the following two options:

  Option 1

  ```
  // DVC
  // SPL DUMP
  // LFD
  ```

  Here, the file is written to the tape at the central site. When it is retrieved, the output is directed to the proper central printer or punch unit.

  Option 2

  ```
  // DVC
  // DST OS3CTR,userid,userid,...
  // SPL DUMP
  // LFD
  ```

  Here, the file is written to the printer or punch at the remote destination specified by the userid. The file is also written to tape at the central site via the OS3CTR specification. When the file is retrieved, it is printed and punched at the central site. There is no method for placing a file on a tape and having that output go directly to a remote station when it is retrieved.

- If a nonremote batch processing ICAM symbiont is in the system when a job with remote output terminates (with a DST statement), the output (log, print, and punch files) is not processed at either the central or remote site. To avoid this situation, you should execute jobs with remote output only when one of the following is true:

  - An ICAM symbiont that supports remote batch processing is in main storage.

  - The default ICAM symbiont (specified on the supervisor SUPGEN SPOOLICAM parameter) supports remote batch processing. (The default ICAM symbiont is automatically loaded when a job with remote output terminates.)

- If a remote batch processing routine attempts to get a buffer from any buffer pool other than the status message pool (BFILES STAT) and a buffer is not available, RBP initiates a system shutdown and sends a shutdown message to the system console. Subsequent attempts to reinitialize the RBP subsystem using the ICAM RB console type-in are ignored. If RBP is the only ICAM user active at the time of buffer depletion, the entire ICAM system is canceled.

- The UP and DO ICAM system console type-ins are not supported for remote batch processing.

Section 4
RPG II Telecommunications

RPG II runs as a utility to ICAM. RPG II telecommunications programs use the direct data interface (DDI) level 3. The DDI serves only as an entry point to ICAM. The DDI macroinstructions are not available to your RPG II program. Thus, the only operation you need do with ICAM is define the communications network.

Note: *The* Models 3-6 and 8-20 ICAM Operations Guide *(UP-9745) and the* Model 7E ICAM Operations Guide *(70023908) detail how to define a network.*

The following is an example of an ICAM generation for RPG II telecommunications. Note that no network buffers are specified in the BUFFERS macro.

Example

```
1         10    16
-------------------------------------------------------------------------
COMMCT
NET3      CCA   TYPE=(DDI)
          BUFFERS 0,0,0,ARPS=15
LNE1      LINE  DEVICE=(BSC,410,ASCII),TYPE=(2400,SYNC,SWCH,UNAT),ID=08
TRM1      TERM  FEATURES=(2780,132,3,SECOND,,400,MODEL1)
          ENDCCA
          MCP   MCPNAME=C7
                CACH=(08,NET3,01)
END
// FIN
```

Note: *For additional information concerning RPG telecommunications programs, refer to the* RPG II Programming Guide *(UP-8067) or the* RPG II Programming Reference Manual *(UP-8044). These Unisys manuals illustrate the use of the control card, file description, and telecommunications specifications forms in preparing an RPG telecommunications program.*

Section 5
Journal Utility

5.1. General Description

The journal utility produces printed reports that list the text of incoming and outgoing messages, the number of messages sent or received, and the extent to which your network buffers and activity request packet pool were used. If ICAM abterms, the journal utility also has a feature that allows you to recover input or output messages if they were queued on disk. You then perform a warm restart to resume message flow.

To obtain printed reports, three control statements are available: BSTAT, SUM, and SELECT. A fourth control statement, RESTART, recovers the disk queueing file, allowing ICAM to perform a warm restart. These control statements are placed in your job control stream. You can use only one for each execution.

Obtaining these printed reports and re-creating the disk queuing file is the final half of a 2-part process. First, have ICAM create records and store them in a journal file. To create the records needed by the journal utility, you must include certain specifications in your ICAM network definition. To create one of the record types (for example, journal records), you also write a message processing routine using the ICAM message processing procedure specifications (MPPS).

ICAM writes the records you specify to the journal file during execution of your program or IMS. Afterwards, you run the journal utility to print reports or, if necessary, recover the disk queuing file.

5.2. Record Types and Required Network Definition Specifications

There are five types of journaling records you can create. You specify the types of records you want on the JRNINIT operand of the CCA macro in your ICAM network definition. Depending on which record types you want, you also specify certain other macros and operands. In addition, you must include a JRNFILE macro naming the journal file where the records are to be written.

The five record types and required specifications are

1. Journal Records

   Each of these records contains the text of a message sent or received, the time and date the journal record was created, and other related information. To incorporate, specify JOURN on the JRNINIT operand of the CCA macro.

   When you select JOURN, you also must code INPUT=mpps-name in either the TERM or LINE macro, and include the JOURN macro in your message processing routine. A journal record is written each time the JOURN macro is executed.

   Code the INPUT operand on the TERM macro to create journal records for a particular terminal. To create journal records for all terminals on a line, specify the INPUT operand on the LINE macro.

2. ODNR Records

   These records are message dequeuing notices. They are journaled when a message is dequeued for delivery to a terminal or other user program. To incorporate, specify ODNR on the JRNINIT operand of the CCA macro.

3. Line and Terminal Performance Records

   These records contain the total number of messages sent or received by each terminal or remote workstation on a line and the number of times each is polled. To incorporate, specify PERF on the JRNINIT operand of the CCA macro. When you select PERF, you also must code STATS=YES in the LINE macro.

4. Buffer Statistics Records

   These records contain the number of network buffers used and the frequency they are used. Also listed is your activity request packet size and the frequency the packets are used. To incorporate, specify STAT on the JRNINIT operand of the CCA macro. When you select STAT, you also must code STAT=YES in the BUFFERS macro.

5. Restart Records

   These records contain a copy of each message as it was queued on disk. If ICAM fails, the journal utility can reconstruct the ICAM disk message queue. You then perform a warm restart to resume message flow. To incorporate, specify RESTART on the JRNINIT operand of the CCA macro.

In addition to the specifications related to the five record types, you must include a JRNFILE macro to identify the journal file. The label of the JRNFILE macro must match the LFD name of the file assigned in the GUST job stream for global networks or in the user job stream for dedicated networks. For JOURN records, you also specify this file name as an operand on the JOURN message processing routine macro.

Note: For more information about the macros and operands required for journaling, see the Models 3-6 and 8-20 ICAM Operations Guide *(UP-9745) and the* Model 7E ICAM Operations Guide *(70023908). For details on the JOURN macro and how to write a message processing routine, see the* ICAM Programming Reference Manual *(UP-9749).*

5.3. Allocating the Journal File

When you define a journal file in your network definition, you must also allocate the file through OS/3 job control and assign the file in the job control stream each time you execute your program. (If you use a global network, you assign the file in the job control stream for GUST instead of your individual programs.) The file name on the LFD statement must match the label on the JRNFILE macro.

The journal file may be on tape or disk, depending on your TYPE operand specification on the JRNFILE macro.

To allocate a disk journal file, you include an EXT job control statement on which you identify the file as a system access technique (SAT) file and specify its size in either blocks (256 bytes in length) or cylinders:

```
// EXT ST,C,,BLK,(256,4000)

    or

// EXT ST,C,,CYL,10
```

To determine the amount of space you need for the disk journal file, multiply the average message size by the maximum number of messages you expect to transmit during the execution of your program. In the case of a global network, the file should be large enough to contain the messages for all the programs using the global network.

If you use a tape journal file, the tape must already be initialized and not have a space allocation, but you can assign as many tape volumes as you need. To assign a single-volume tape file, you include a device assignment like the following in your program's job control stream:

```
// DVC 90 // VOL TAPE01 // LFD JRNFILE
```

Note that any tape journal file definition in a jobstream LBL statement is not mandatory. If you want to specify it, the tape must be initialized with a label previously.

To assign a multivolume tape journal file, you include a device assignment similar to the following in your program's job control stream:

```
// DVC 90  // VOL TAPE01,TAPE02,TAPE03 // LBL JRNFILE,TAPE01
// LFD JRNFILE
```

5.4. Executing the Journal Utility

You execute the journal utility after ICAM shutdown. Do not run the journal utility with an active ICAM network.

The following example shows a typical job control stream used to execute the journal utility. The job stream always contains device assignments for a printer and for the journal file. The journal utility requires the LFD-name INPUT1 for the journal file. When you execute the journal utility to re-create the disk queuing file using the RESTART control statement, you also need a device assignment for the new disk queuing file. (See 5.5.4 for an example.) You also assign an output file on disk or tape when you use the SELECT control statement with the FILE parameter. (See 5.5.3.)

```
1.  // JOB jobname,,,7000
2.  // DVC 20 // LFD PRNTR
3.  // DVC 90 // VOL S10309 // LFD INPUT1                    TAPE
    // DVC 50 // VOL A12345 // LBL MYFIL // LFD INPUT1       DISK
4.  // OPTION JOBDUMP
5.  // EXEC JUST
6.  /$
7.  Δ control statement                          Preceded by at least one blank space
8.  /*
9.  /&
10. // FIN
```

Explanation:

1. Identifies the job. The first character must be alphabetic. The 7000 indicates the amount of main storage needed to run the journal utility.

2. Assigns a printer to your job. You must use PRNTR as the file name.

3. Assigns the disk or tape file that contains the journaling records. You must specify INPUT1 with the LFD statement for the journal utility to gain access to these records.

4. Provides a dump of main storage if your program abnormally terminates.

5. Tells the system to execute the journal utility.

6. Indicates that data cards follow. In the case of the journal utility, you place the control statement that corresponds with the report you want after this line.

7. Enter the control statement, skipping at least one space. You can enter only one control statement in each job control stream.

8. End of data statements.

9. End of job.

10. Terminates job stream.

5.5. Journal Utility Control Statements

The BSTAT, SUM, and SELECT control statements cause the journal utility to produce different reports. Remember, you can use only one control statement for each execution. The control statements and the reports produced are described in 5.5.1 through 5.5.3. The RESTART statement is discussed in 5.5.4.

5.5.1. Printing Buffer Statistics (BSTAT)

The BSTAT control statement produces a printed report (Figure 5-1) that shows network buffer and activity request packet utilization. For this control statement, the journal utility draws its information from the buffer statistics records (STATS) that you had ICAM write to the journal file.

Format

```
BSTAT
```

```
OS/3 ICAM          DATA REDUCTION UTILITY

                        NETWORK BUFFER UTILIZATION STATISTICS
                    FROM  01/11/80  AT  13:20:45  TO  01/11/80  AT  13:20:45

NETWORK BUFFER SIZE -          256   BYTES        TOTAL BUFFER NUMBER  -          10
MAXIMUM NUMBER AVAILABLE -      10              THRESHOLD VALUE -               1
NUMBER OF REJECTED REQUESTS -    0              NUMBER OF DEFFERRED  REQUESTS -    0
TOTAL NUMBER OF REQUESTS -      18

                    RELATIVE BUFFER NUMBER                  TIMES USED

                                 1                            18
                                 2                             0

NETWORK BUFFER UTILIZATION          NUMBER OF OCCURRENCES       PERCENTAGE OF TOTAL

      0 -  20 %                          18                        100.0
     21 -  40 %                           0                          0.0
     41 -  60 %                           0                          0.0
     61 -  80 %                           0                          0.0
     81 - 100 %                           0                          0.0

                    ACTIVITY REQUEST PACKET UTILIZATION STATISTICS
                FROM  01/11/80  AT  13:20:45  TO  01/11/80  AT  13:20:45

ACTIVITY REQUEST PACKET SIZE -  48   BYTES        TOTAL BUFFER NUMBER  -          36
MAXIMUM NUMBER AVAILABLE -      30              THRESHOLD VALUE -               5
NUMBER OF REJECTED REQUESTS -    0              NUMBER OF DEFFERRED  REQUESTS -    0
TOTAL NUMBER OF REQUESTS -      48

                    REPORT FOR BSTAT SELECTION STATEMENT WITH TIME = NET

                    RELATIVE BUFFER NUMBER                  TIMES USED

                                 1                             3
                                 2                            17
                                 3                             1
                                 4                             1
                                 5                            20
                                 6                             5
                                 7                             1
                                 8                             0

ACTIVITY REQUEST PACKET UTILIZATION  NUMBER OF OCCURRENCES  PERCENTAGE OF TOTAL

      0 -  20 %                          48                        100.0
     21 -  40 %                           0                          0.0
     41 -  60 %                           0                          0.0
     61 -  80 %                           0                          0.0
     81 - 100 %                           0                          0.0

                    REPORT FOR BSTAT SELECTION STATEMENT WITH TIME = NET
```

Figure 5-1. Typical BSTAT Report

5.5.2. Printing Terminal Statistics (SUM)

The SUM control statement produces a printed report (Figure 5-2) that shows the number of messages sent or received on each terminal or remote workstation on a line, or on a particular terminal or workstation, and the number of times each was polled. For this control statement, the journal utility draws its information from the line and terminal performance records (PERF) that you had ICAM write to the journal file.

By specifying only SUM with no parameters, you get a list of all terminals or remote workstations in your network that are polled. Since teletypewriters cannot be polled, they are shown on the report having no poll data.

Format

```
SUM [TERMS=(term-name1,term-name2,...,term-namen)]
    [,TERMD=(term-name1,term-name2,...,term-namen)]
```

Parameters

`TERMS=(term-name)`
: Gathers only the data received by the terminals or remote workstations you specify. The terminal name can be from one to four characters.

`TERMD=(term-name)`
: Gathers only the data sent by the terminals or remote workstations you specify. The terminal name can be from one to four characters.

If you do choose particular terminals, you can use a total of only 255 characters, including the SUM control statement, commas, and parentheses. You use a comma to separate terminal names.

```
OS/3 ICAM          DATA REDUCTION UTILITY
 SUM

                                MESSAGE TRAFFIC SUMMARY

                                ON: 11/27/78  AT  14:14:43

      LINE   TERM      INPUT MSG      INPUT       OUTPUT MSG      OUTPUT      POLLS      NO TRAFFIC
      NAME   NAME       COUNT      RETRANSMITS      COUNT      RETRANSMITS    SENT       RESPONSES

      LNC1   TRM1          4            0             0            0          410          402
             TRM2          0            0             0            0            0            0
             T3            0            0             3            0          407          405
             TRM4          0            0             2            0            3            2
-------------------------------------------------------------------------------------------------
             TOTAL         4            0             5            0          820          809

                          REPORT FOR SUM SELECTION STATEMENT
```

Figure 5-2. Typical SUM Report

5.5.3. Printing Selected Records (SELECT)

The SELECT control statement may produce a printed report (Figure 5-3) that shows:

- Network buffer and activity request packet utilization
- Number of messages sent or received by each terminal on a line (and the number of times each terminal was polled)
- Text of messages
- Time and date

You also can transcribe the records in the journal utility onto another file, such as a tape or disk. For this control statement, the journal utility draws its information from all record types that you had ICAM write to the journal file except restart records.

By specifying SELECT with no parameters, you obtain a printed report of four record types (JOURN, ODNR, STATS, and PERF). RESTART is excluded. Since the file was created with the record types and date/time stamps entered in each record, you also have the option of obtaining printed reports for just the records that bear a time or date of your choice. When you use the FILE parameter, the selected records are written to a disk or tape file instead of being printed.

Format

```
SELECT [FILE=filename]
       [,JEC=(record-code1,record-code2,...,record-coden)]
       [,DATE=(yyddd1-yydd2)]
       [,TIME=(hhmmss1-hhmmss2)]
```

Parameters

`FILE=filename`

Names a tape or disk file to which the selected journal records are written. The journal utility can use this file later to produce printed reports. When you use this parameter you must assign the file in your job control stream. The LFD name must match the *filename* specification on this parameter. If a disk file is designated, the file must be pre-allocated using the system access technique. If a tape file is designated, the tape should be initialized prior to use.

`JEC=(record-code)`

Prints or writes to tape or disk the specified types of record. If omitted, four record types (JOURN, ODNR, STATS, and PERF) are printed out or written. You cannot use RESTART.

DATE=(yyddd)
Prints or writes the records that bear the date you specify, or the records that fall between the range you specify by using both operands. The selection is inclusive. A hyphen separates both operands.

where:

yy
Is the last digits of the year.

ddd
Is the day of the year, from 1 to 366.

TIME=(hhmmss)
Prints or writes the records that bear the time you specify, or the records that fall between the range you specify by using both operands. The selection is inclusive. A hyphen separates both operands.

where:

hh
Is the hour, from 00 to 23.

mm
Is the minutes, from 00 to 59.

ss
Is the seconds, from 00 to 59.

Specify date and time only when printing JOURN or ODNR records. These records include date and time; STATS and PERF records do not include date and time.

Figure 5-3 is an example report that you get by specifying SELECT with no parameters. The printout includes journaling, output delivery notice, statistics, and performance records. The sequence numbers at the beginning of each record indicate the order in which the records of each type were created. For example, the first journal record or the first statistics record.

Each journal record indicates whether the message is segmented. In this report, the words COMPLETE MESSAGE mean that the message is contained in one segment. Otherwise, the segments would be introduced by the terms INITIAL SEGMENT, TEXT BYTES THIS SEGMENT, and TEXT BYTES THIS FINAL SEGMENT. The message text itself is printed on the second and third lines.

```
OS/3 ICAM            DATA REDUCTION UTILITY

 SELECT


                                        * * * * * * * * * * * * * * * * * * * * * * *

                                        *    BEGIN REPORT FOR PARAMETER SET#1     *

                                        * * * * * * * * * * * * * * * * * * * * * * *


SEQ-NO 1       JOURNAL DATE  11/27/78  TIME  14:09:33  SOURCE TRM1  DEST EMPTY COMPLETE MESSAGE;  INPUT  TEXT BYTES 113

                             TEXT IS:                          *ROUTE LENGTHV    A TRM1 0001 TRM4T3  #    BT    THIS IS
                             TEXT IS: A ONE LINE MESSAGE TO YOU    BT

SEQ-NO 1       JOURNAL DATE  11/27/78  TIME  14:09:33  SOURCE TRM1  DEST T3    COMPLETE MESSAGE; OUTPUT  TEXT BYTES 113

                             TEXT IS:                          *ROUTE LENGTHV    A TRM1 0001 TRM4T3  #    BT    THIS IS
                             TEXT IS: A ONE LINE MESSAGE TO YOU    BT

SEQ-NO 1       ODNR    DATE  11/27/78  TIME  14:09:35

SEQ-NO 1       JOURNAL DATE  11/27/78  TIME  14:09:35  SOURCE TRM1  DEST TRM4  COMPLETE MESSAGE; OUTPUT  TEXT BYTES 113

                             TEXT IS:                          *ROUTE LENGTHV    A TRM1 0001 TRM4T3  #    BT    THIS IS
                             TEXT IS: A ONE LINE MESSAGE TO YOU    BT

SEQ-NO 1       ODNR    DATE  11/27/78  TIME  14:09:36

SEQ-NO 2       JOURNAL DATE  11/27/78  TIME  14:10:20  SOURCE TRM1  DEST EMPTY COMPLETE MESSAGE;  INPUT  TEXT BYTES 109

                             TEXT IS:                          *ROUTE LENGTH4    A TRM1 0002 TRM4#    BT    THIS IS A O
                             TEXT IS:NE LINE MESSAGE TO YOU    BT

SEQ-NO 2       JOURNAL DATE  11/27/78  TIME  14:10:20  SOURCE TRM1  DEST TRM4  COMPLETE MESSAGE; OUTPUT  TEXT BYTES 109

                             TEXT IS:                          *ROUTE LENGTH4    A TRM1 0002 TRM4#    BT    THIS IS A O
                             TEXT IS:NE LINE MESSAGE TO YOU    BT

SEQ-NO 2       ODNR    DATE  11/27/78  TIME  14:10:22

SEQ-NO 3       JOURNAL DATE  11/27/78  TIME  14:12:45  SOURCE TRM1  DEST EMPTY COMPLETE MESSAGE;  INPUT  TEXT BYTES 120

                             TEXT IS:                          *DIRECT ALTO    A TRM1 0003 T3  #    BT    DIRECT THIS O
                             TEXT IS:NE LINE MESSAGE TO TERMINAL THREE    BT

SEQ-NO 3       JOURNAL DATE  11/27/78  TIME  14:12:46  SOURCE TRM1  DEST T3    COMPLETE MESSAGE; OUTPUT  TEXT BYTES 120

                             TEXT IS:                          *DIRECT ALTO    A TRM1 0003 T3  #    BT    DIRECT THIS O
                             TEXT IS:NE LINE MESSAGE TO TERMINAL THREE    BT

SEQ-NO 3       ODNR    DATE  11/27/78  TIME  14:12:47

SEQ-NO 4       JOURNAL DATE  11/27/78  TIME  14:14:07  SOURCE TRM1  DEST EMPTY COMPLETE MESSAGE;  INPUT  TEXT BYTES 120

                             TEXT IS:                          *ROUTE LENGTH2    A TRM1 0004 T3  #    BT    SEND THIS O
                             TEXT IS:NE LINE MESSAGE TO TERMINAL THREE    BT

SEQ-NO 4       JOURNAL DATE  11/27/78  TIME  14:14:07  SOURCE TRM1  DEST T3    COMPLETE MESSAGE; OUTPUT  TEXT BYTES 120

                             TEXT IS:                          *ROUTE LENGTH2    A TRM1 0004 T3  #    BT    SEND THIS O
                             TEXT IS:NE LINE MESSAGE TO TERMINAL THREE    BT

SEQ-NO 4       ODNR    DATE  11/27/78  TIME  14:14:08
```

Figure 5-3. Typical SELECT Report (Part 1 of 2)

```
OS/3 ICAM            DATA REDUCTION UTILITY

SEQ-NO 1        STATISTIC          DATE  11/27/78  TIME  14:14:43

     NETWORK BUFFER SIZE -            256    BYTES        TOTAL NUMBER OF BUFFERS -        10

     MAXIMUM NUMBER AVAILABLE -        10                 THRESHOLD VALUE -                 1

     NUMBER OF REJECTED REQUESTS -      0                 NUMBER OF DEFERRED REQUESTS -     0

                              RELATIVE BUFFER NUMBER            TIMES USED

                                          1                        4
                                          2                        1
                                          3                        3
                                          4                        0
                                          5                        0
                                          6                        0
                                          7                        0
                                          8                        0
                                          9                        0
                                         10                        0

SEQ-NO 2        STATISTIC          DATE  11/27/78  TIME  14:14:43

     ACTIVITY REQUEST PACKET SIZE -    40    BYTES        TOTAL NUMBER OF BUFFERS -        36

     MAXIMUM NUMBER AVAILABLE -        34                 THRESHOLD VALUE -                 5

     NUMBER OF REJECTED REQUESTS -      0                 NUMBER OF DEFERRED REQUESTS -     0

                              RELATIVE BUFFER NUMBER            TIMES USED

                                          1                        3
                                          2                        4
                                          3                        1
                                          4                        1
                                          5                       11
                                          6                       20
                                          7                        8
                                          8                        0
                                          9                        0
                                         10                        0
                                         11                        0
                                         12                        0
                                         13                        0
                                         14                        0
                                         15                        0
                                         16                        0
                                         17                        0
                                         18                        0
                                         19                        0
                                         20                        0
                                         21                        0
                                         22                        0
                                         23                        0
                                         24                        0
                                         25                        0
                                         26                        0
                                         27                        0
                                         28                        0
                                         29                        0
                                         30                        0
                                         31                        0
                                         32                        0
                                         33                        0
                                         34                        0
                                         35                        0
                                         36                        0

SEQ-NO 1        PERFORMANCE        DATE  11/27/78  TIME  14:14:43      LINE NAME LN01    NO. TERMINALS ON LINE 4

                TERM      INPUT MSG      INPUT      OUTPUT MSG      OUTPUT      POLLS      NO TRAFFIC
                NAME      COUNT          RETRANSMITS COUNT          RETRANSMITS SENT       RESPONSES

                TRM1      4              0           0              0           410        402
                TRM2      0              0           0              0           0          0
                T3        0              0           3              0           407        405
                TRM4      0              0           2              0           3          2


OS/3 ICAM            DATA REDUCTION UTILITY


                              END OF REPORT FOR PARAMETER SET #1
```

Figure 5-3. Typical SELECT Report (Part 2 of 2)

5.5.4. Recovering a Disk Queuing File (RESTART)

The RESTART control statement re-creates the disk queue file, thus enabling ICAM to perform a warm restart. For this control statement, the journal utility draws its information from the restart records you had ICAM write to the journal file.

Format

```
RESTART FILE=(filename1[,filename2,...filenamen])
```

Parameter

`FILE=filename`
Provides name of the disk file onto which the messages are re-created.

The RESTART control statement uses a different job control stream than that for the other control statements. A minimum of two files are required. The input file, which contains the journal restart records, can be disk or tape. The output file is always on disk because it replaces the disk queuing file. A new disk queuing file must be allocated through job control statements prior to use. No allocation is necessary if the old disk queuing file is to be restored. For example:

```
// JOB SAMPLE,,,7000
// DVC 20 // LFD PRNTR
// DVC 50 // VOL A531CM // LBL HISTORY-FILE // LFD INPUT1
// DVC 51 // VOL NEWFIL // LBL NEW-HISTORY // LFD OUTFIL
// OPTION JOBDUMP
// EXEC JUST
/$
     RESTART FILE=OUTFIL
/*
/&
// FIN
```

The line beginning with // DVC 50 defines the journal file, which requires the LFD-name INPUT1. The line beginning with // DVC 51 is the disk onto which the messages are queued. Since we give the name OUTFIL as the LFD, we must specify FILE=OUTFIL on the RESTART control statement.

The following are the typical steps you would go through to recover disk message queues if a network goes down, for example, because the disk queuing file on the volume or the disk volume itself was destroyed.

1. The operator loads the journal utility program since a disk queuing file cannot be shared between ICAM and the off-line journal utility program.

2. The journal utility rebuilds the disk queues for the down network.

3. The operator reloads ICAM.

4. The operator reloads the user program associated with the down network. In a dedicated network, the user program issues a NETREQ, using the RESTART=YES option. In a global network, the global user service task (GUST) issues a NETREQ with RESTART=YES.

5. The network request with restart option is complete and ICAM does a warm restart of the failed network.

6. Normal operation resumes.

5.6. Error Handling

If the journal utility encounters errors in the SELECT report, the following message is displayed on the system console:

```
**ICAM JOURNAL UTILITY PROGRAM HAS FOUND DATA ERRORS. SEE DATA ERROR
   COUNT AT END OF EACH REPORT THAT HAS ERRORS***
```

At the end of each report that has errors, the total number of errors is listed after the following message:

```
THE FOLLOWING NUMBER OF DATA ERROR FLAGS ARE FOUND IN THIS REPORT: nnn
```

You should send this report and a journal file dump, along with a software user report (SUR), to your Unisys system analyst.

Note: *Additional error messages, caused by invalid statements or keywords, are included in the* OS/3 System Messages Reference Manual *(UP-8076).*

5.7. Example of Dedicated Network That Incorporates Journaling

Figure 5-4 is an example of an ICAM dedicated network that incorporates journaling. The procedures to create and execute a program that uses this network and the procedures to execute the journal utility are described in 5.8.

In this dedicated network, the CCA macro specifies the type of network as a standard interface (STDMCP). Line and terminal performance records (PERF), buffer statistics records (STAT), and journal message records (JOURN) are configured.

The STAT=YES operand on the BUFFERS macro causes network buffer pool statistics and activity request packet pool statistics storage areas to be created. The STATS=YES on the LINE macro creates a 3-word statistics area for each terminal on the line within the terminal control table.

The INPUT=(MPSA,9) operand (Line 3) directs terminal input to the message processing procedure specifications (MPPS) that accumulate the actual journal records and direct the messages on to the process file (PRFA).

The JRNFILE macro (Line 6) specifies an SAT file on disk that is named as logical file JRNA.

The explanations that follow Figure 5-4 are keyed by line number to the network.

```
1.  JNET    CCA   TYPE=(STDMCP),JRNINIT=(PERF,STAT,JOURN)
2.          BUFFERS 20,100,,ARP=24,STAT=YES
    LNE1    LINE  DEVICE=(UNISCOPE),
                  TYPE=(2400,SWCH,SYNC,UNAT),
3.                INPUT=(MPSA,9),HIGH=MAIN,
4.                LOW=MAIN,MEDIUM=MAIN,STAT=YES
    TRM1    TERM  ADDR=(28,52),
                  FEATURES=(U400,1920)
5.  PRFA    PRCS  LOW=MAIN
6.  JRNA    JRNFILE TYPE=(SAT,DISC),BUFF=(3,1024,1)
7.  MPSA    MPSTART
            RECSEG
8.          JOURN JRNA
            RECHDR
9.          DIRECT P,PRFA,L
            DATSTP J
            RECEND
            CANCELM TN#MBINS
10.         JOURN JRNA
            RECPST
            SENSEG
            SENHDR
            SENEND
11.         JOURN JRNA
            SENPST
            ENDCCA
            MCP
                  MCPNAME=M9
                  MCPVOL=SYSRES
                  CACH=(08,2400,SWITCHED,SYNC)
            END
```

Figure 5-4. Dedicated Journal Network JNET

UNISCOPE is a registered trademark of Unisys Corporation.

1. Initialize performance statistics, buffer statistics, and message journal records.
2. STAT=YES required for buffer statistics.
3. Input to an MPPS required for journaling.
4. STATS=YES required for line performance statistics.
5. Process file defined to receive the messages after they have been journaled.
6. The logical journal file is defined as residing on disk.
7. Specifies the start of the message processing procedure specifications to define the journal records.
8. Journals the entire input message to logical file JRNA.
9. Directs the message on to process file PRFA for retrieval by the user program.
10. Journals the input message header prefix after any rerouting with a date stamp included.
11. Journals the output message to logical file JRNA.

5.8. Example of Journaling Application Using a Dedicated Network

If your network is a dedicated network (that is, you do not have interactive services, which requires a global network), use standard job control procedures to allocate the necessary files, assemble and link the user program source module, and execute the load module. These procedures are described in 5.8.1 through 5.8.4. Journal utility execution is described in 5.8.5.

- Allocating the files (5.8.1)

  Describes the job control to allocate files for the source program module, load module, and journal records.

- Creating a Source Module (5.8.2)

  Briefly describes the creation of the source code and what the program does.

- Assembling the Program and Creating a Load Module (5.8.3)

  Describes a job control stream to assemble and link the user program source module into an executable load module.

- Executing the Load Module (5.8.4)

  Describes the job control necessary to execute the load module.

- Executing the Journal Utility (5.8.5)

  Describes the job control necessary to execute the journal utility system program. This produces the printed report of the journal records and statistics after the user program has completed.

5.8.1. Allocating the Files

You allocate three files: one for source modules, one for load modules, and one for the journal records. This is accomplished by the following:

```
// JOB ALLOCATE
// DVC 50  // VOL PUB001  // EXT ST,C,CYL,4
// LBL ICAMFILE  // LFD OUTAPUT,,INIT
// DVC 50  // VOL PUB001  // EXT ST,C,CYL,2
// LBL PJRICM  // LFD OUTAPUT1,,INIT
// DVC 50  // VOL PUB001  // EXT ST,C,CYL,2
// LBL ICMJRN  // LFD INPUT1,,INIT
/&
```

5.8.2. Creating a Source Module

The program shown in Figure 5-5 is the same as the dedicated network user program described in detail in the *ICAM Standard MCP Interface Programming Guide* (UP-8550) with the following minor exceptions:

1. The name of the network in the NETREQ and NETREL macros is changed to request the journal network described in 5.9 and Figure 5-4.

2. In a journal network, input on a TERM macro in a network definition must be to a message processing procedure specification (MPPS) instead of a terminal queue. The MPPS directs the messages to a process file. The GETCP in the user program then retrieves the messages from the process file.

Briefly, this program operates in the following manner: Once you dial in to the telephone number assigned to port 8 (the port specified in the network for this line), ICAM sends the prompt messages that the program queued for output on line 25 and line 31 to the remote terminal screen. The program then puts up a deferred GETCP and yields to ICAM for any responses.

When the prompt is answered by the remote terminal operator, input entry is at BINGO. The program checks for a DONE message. If there isn't one, the input received by the program is then echoed back to the terminal with a THANK YOU. Thus it is a simple echo program, but sufficient to transfer messages across the line to test ICAM and to produce a number of journal records and statistics.

```
@P
   1.0000 PATJRN   START 0
   2.0000          TM#DSECT DUST
   3.0000          TM#DSECT GETPUT
   4.0000          BALR  10,0
   5.0000          USING *,10
   6.0000 *                  **********************************
   7.0000 *                  ****** COVER INPUT DTFCP ******
   8.0000 *                  **********************************
   9.0000          USING TM#PRCS,2
  10.0000          LA    2,PRFA
  11.0000 *                  **********************************
  12.0000 *                  ****** COVER OUTPUT DTFCP *****
  13.0000 *                  **********************************
  14.0000          USING TM#DEST,3
  15.0000          LA    3,OTPT
  16.0000 *                  **********************************
  17.0000 *                  ****** REQUEST A NETWORK *******
  18.0000 *                  **********************************
  19.0000 BEGIN    NETREQ JNET,ERRET=NETERR
+
  20.0000 *                  **********************************
  21.0000 *                  ********* SET UP PUTCP ********
  22.0000 *                  **********************************
  23.0000 LOOP     OI    TM#DSEG,TM#DHDR++TM#DTND
  24.0000          XC    TM#DERR,TM#DERR
  25.0000 PUTIT1   PUTCP OTPT,MSGOT1
  26.0000 *                  **********************************
  27.0000 *                  ********* SET UP PUTCP ********
  28.0000 *                  **********************************
  29.0000          OI    TM#DSEG,TM#DHDR++TM#DTND
  30.0000          XC    TM#DERR,TM#DERR
  31.0000 PUTIT2   PUTCP OTPT,MSGOT2
  32.0000 *                  **********************************
  33.0000 *                  **** SET UP DEFERRED GETCP ****
  34.0000 *                  **********************************
  35.0000          XC    TM#PERR,TM#PERR
  36.0000          MVI   TM#PIND,TM#FIRL
  37.0000          MVC   TM#PCMPA,=A(BINGO)
  38.0000          MVC   MSGIN,TXTCHARS
+
  39.0000 GETIT    GETCP PRFA,MSGIN
  40.0000 *                  **********************************
  41.0000 *                  ** YIELD WITH NOTHING TO DO ***
  42.0000 *                  **********************************
  43.0000          CYIELD
  44.0000 *                  **********************************
  45.0000 *                  **** WHAT KIND OF MESSAGE? ****
  46.0000 *                  ****    DONE OR A NAME?    ****
  47.0000 *                  **********************************
  48.0000 BINGO    CLC   TXTIN(4),=C'DONE'
  49.0000          BE    ENDJOB
  50.0000          CLC   TXTIN(4),=X'84969585'
  51.0000          BE    ENDJOB
  52.0000 *                  **********************************
  53.0000 *                  ********* SET UP PUTCP ********
  54.0000 *                  **********************************
  55.0000          MVC   TXTOUT,TXTIN
  56.0000          MVC   TXTIN,C' '
  57.0000          MVC   TXTIN+1(L'TXTIN-1),TXTIN
```

Figure 5-5. Sample User Program (PATJRN) that Uses the Dedicated Network JNET (Part 1 of 2)

```
+
   58.0000             GI     THDSEG,THDHDR++THDDTND
   59.0000 PUTIT3      PUTCP  OTPT,MSGOUT
   60.0000             B      LOOP
   61.0000 ENDJOB      NETREL JNET
   62.0000             B      DONE
   63.0000 NETERR      LNEREL LNE1
   64.0000             LNEREQ LNE1
   65.0000 *                    **********************************
   66.0000 *                    *****  YIELD TO ICAM  *********
   67.0000 *                    *****  DEFERRED GET STILL  ****
   68.0000 *                    *****  PENDING         *********
   69.0000 *                    **********************************
   70.0000             CYIELD
   71.0000 NOGET       LA     1,MSG2
   72.0000             OPR    MSG2,12
   73.0000             B      DONEJOB
   74.0000 NOPUT       LA     1,MSG3
   75.0000             OPR    MSG3,12
   76.0000             B      DONEJOB
+
   77.0000 NOBUFF      LA     1,MSG4
   78.0000             OPR    MSG4,19
   79.0000 DONEJOB     SNAP   BEGIN,TXTCHARS
   80.0000 DONE        EOJ
   81.0000 *                    **********************************
   82.0000 *                    ****      CONSTANT AREA      ****
   83.0000 *                    **********************************
   84.0000 PRFA        DTFCP  TYPE=GT,ERRET=NOGET
   85.0000 OTPT        DTFCP  TYPE=PT,ERRET=NOPUT,DEST=(T,TRM1),NOBAV=NOBUFF
   86.0000             DS     0F
   87.0000 MSG1        DC     CL26'ERROR ISSUING NETREQ MACRO'
   88.0000 MSG2        DC     CL12'ERROR ON GET'
   89.0000 MSG3        DC     CL12'ERROR ON PUT'
   90.0000 MSG4        DC     CL19'NO BUFFER AVAILABLE'
   91.0000             DS     0H
   92.0000 MSGOUT1     DC     X'0015'
   93.0000             DC     C'WELCOME TO OS/3 ICAM'
   94.0000             DC     X'0D'
   95.0000             DS     0H
+
   96.0000 MSGOUT2     DC     X'001A'
   97.0000             DC     C'TYPE IN YOUR NAME PLEASE'
   98.0000             DC     X'001E'
   99.0000             DS     0H
  100.0000 MSGIN       DC     X'0000'
  101.0000             DC     CL5' '      <-- (5 SPACES FOR SOE AND 4 DICE
  102.0000 TXTIN       DC     CL256' '        CHARACTERS WHEN DICE=ON)
  103.0000 INETX       DC     X'FFFF'
  104.0000             DS     0H
  105.0000 MSGOUT      DC     X'010C'
  106.0000             DC     X'0D'
  107.0000             DC     C'THANK YOU '
  108.0000 TXTOUT      DC     CL256' '
  109.0000 OUTETX      DC     X'0D'
  110.0000 TXTCHARS    DC     X'0105'
  111.0000             END
```

Figure 5-5. Sample User Program (PATJRN) that Uses the Dedicated Network JNET (Part 2 of 2)

5.8.3. Assembling the Program and Creating a Load Module

You can assemble and link a source module into an executable load module in many ways. One way is to create a job control stream using jprocs, as in the job stream to create the load module for the user program in Figure 5-5, and the following coding:

```
// JOB PATJRN,,AB00,,,,,H
//DYNJRN ASM IN=(PUB001,ICAMFILE)
//JRNLD LINK PATJRN,OUT=(PUB001,PJRICM)
/&
```

Be sure to include the OUT parameter specifying the volume and file names. Otherwise, the assembled module is placed in the YRUN temporary run file and is deleted when the job is completed. You must specify OUT to permanently save the load module in your load module file.

In the coding example, DYNJRN is the name assigned to the source module when it was created. However, the LINK jproc requires the name of a CSECT (the name in the label field of a START directive) as the object module input name, not necessarily the name of the module itself. (This is because a module could actually contain more than one CSECT or element.)

The source module could just as easily be given the name PATJRN when it is placed in the file. This would simplify the job control, since the two jproc statements then would be able to use the same name:

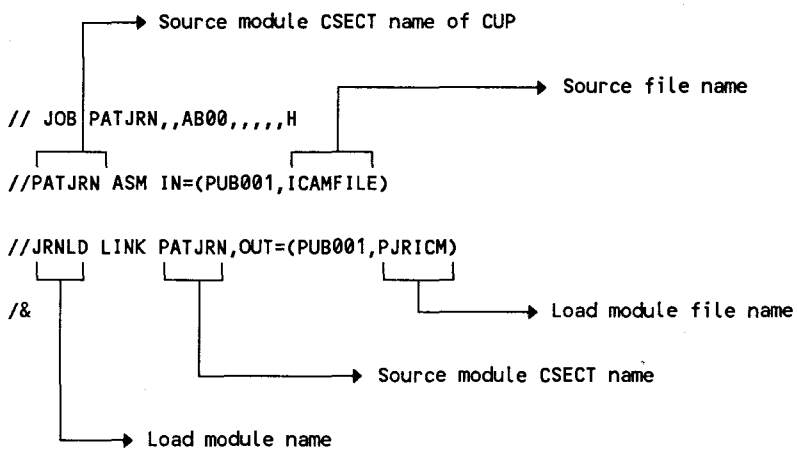
```
//PATJRN ASM IN=(PUB001,ICAMFILE)
//JRNLD LINK PATJRN OUT=(PUB001,PJRICM)
```

In the preceding example, the name on the START directive can be the same as the name of the module, since we are using only one element. The assemble (ASM) jproc statement then says assemble the source module PATJRN, which is in the file ICAMFILE on disk pack PUB001.

The LINK jproc says to link the object module CSECT PATJRN (generated by the assembler) and create a load module called JRNLD, which is to reside in a file PJRICM on disk PUB001.

You could name the load module PATJRN (in place of JRNLD) to cut down on the proliferation of names to remember, since they reside in different files (one in a source module file (ICAMFILE) and one in a load module file (PJRICM)).

This same idea could be taken one step further with the job name and the file name for the job control stream. For instance, you could write the following job control stream module and file it:



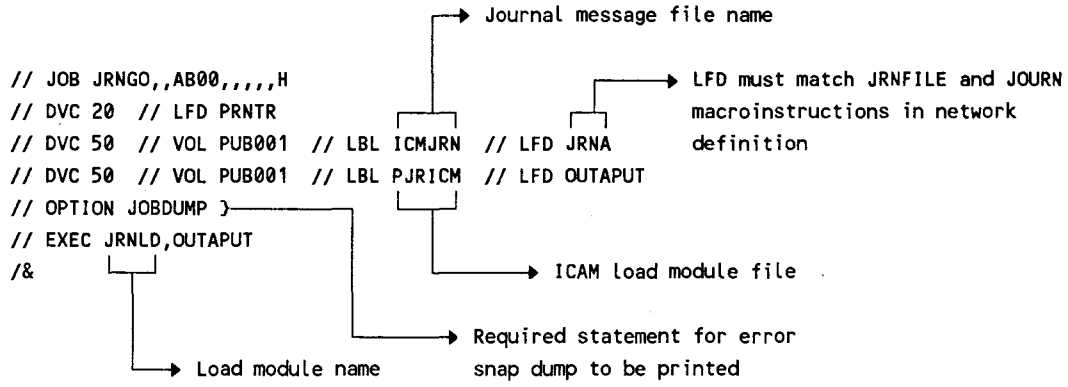
Now you could do your run where the job name, source module name, and load module name are all the same.

To run the assembly, all you need is:

```
RV PATJRN
```

5.8.4. Executing the Load Module

After you create the user program load module, you need a job control stream to execute it:



Note that the first DVC-LFD set specifies the physical file name (LBL) and the logical file name (LFD) for the journal file. The logical file name must match the label of the JRNFILE macro in the network definition.

The second DVC-LFD set specifies the user program load module physical file name and its associated logical file name.

When the program is executed, the system console displays the messages:

```
R03 RUN PROCESSOR SUCCESSFULLY PROCESSED JRNGO
JC01 JOB JRNGO EXECUTING JOBSTEP JRNLD0000 #001 12:45:02
```

At the remote terminal, you can now dial in to ICAM using the telephone number assigned to port 8, which is specified in your network definition for the line. You immediately get the prompt messages and traffic as shown in Figure 5-6.

WELCOME TO OS/3 ICAM	Welcome statement
TYPE IN YOUR NAME PLEASE	Prompt statement
JOHN Q PUBLIC	Operator response
THANK YOU JOHN Q PUBLIC	Program response
WELCOME TO OS/3 ICAM	Welcome statement
TYPE IN YOUR NAME PLEASE	Prompt statement
DONE	Operator response

NOTE: Program terminates because *DONE* is entered.

Figure 5-6. Execution Traffic of Load Module JRNLD

After you type in a number of messages, you can terminate the program. Now you can run the journal utility to obtain a printout of the journal statistics and records produced during execution of the ICAM user program.

5.8.5. Executing the Journal Utility

The journal utility program is executed by a job control stream containing an EXEC JUST statement. The DVC-LFD set specifies the file that contains the journal records created during execution of the user program load module. INPUT1 is the required logical file name for the journal file within the utility.

Note that the OPTION JOBDUMP statement is included to obtain a dump of main storage if your program terminates abnormally. The SELECT statement causes line performance statistics, buffer statistics, and journal message records to be included in the report.

5.9. Example of Global Network That Incorporates Journaling

Figure 5-7 is an example of an ICAM global network that incorporates journaling. The procedures to create and execute a program that uses this network and to execute the journal utility are described in 5.10. In this global network, we use the same journal parameters described for the dedicated network in 5.7.

```
COMMCT
JGBL     CCA   TYPE=(GBL,,OS3A),JRNINIT=(PERF,STAT,JOURN),                X
               DCA=YES,GAWAKE=YES
         BUFFERS,20,100,,ARP=50,STAT=YES,UDUCT=(20,24,2),                 X
               LINKPAK=(80,80,20)
CUP1     LOCAP TYPE=(STDMCP),LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
NODA     LOCAP TYPE=(DMI),                                               X
               IAS=(YES,OFF),MT=YES
LNE1     LINE  DEVICE=(UNISCOPE),                                        X
               TYPE=(2400,SWCH,SYNC,UNAT),                               X
               INPUT=(MPSA,9),HIGH=MAIN,                                 X
               LOW=MAIN,MEDIUM=MAIN,STATS=YES
TRM1     TERM  ADDR=(28,51),                                             X
               FEATURES=(U400,1920),TCTUPD=YES,DICE=(OFF,NEWLINE)
TRM2     TERM  ADDR=(28,52),                                             X
               FEATURES=(U400,1920),TCTUPD=YES,DICE=(OFF,NEWLINE)
PRFA     PRCS  LOW=MAIN
JRNA     JRNFILE
MPSA     MPSTART
         RECSEG
         JOURN JRNA
         RECHDR
         DIRECT P,PRFA,L
         DATSTP J
         RECEND
         CANCELM TN#MBINS
         JOURN JRNA
         RECPST
         SENSEG
         SENHDR
         SENEND
         JOURN JRNA
         SENPST
         ENDCCA
         MCP
         MCPNAME=M9
         MCPVOL=SYSRES
         CACH=(08,2400,SWITCHED)
         END
```

Figure 5-7. Global Journal Network JGBL

The CCA macro specifies a global network (TYPE=(GBL,,OS3A)). To have interactive services and user program dynamic sessions, include both the DCA=YES and GAWAKE=YES operands. The BUFFERS macro requires the UDUCT and LINKPAK operands for remote communications.

If the input records to be journaled are from a remote workstation, the DEVICE operand of the LINE macro must be specified as DEVICE=(RWS). However, only input journaling is supported for remote workstations.

We need a standard interface locap file for the user program and a demand mode interface locap file with IAS=(YES,OFF) and MT=YES operands for interactive services. The TERM macro requires a TCTUPD=YES operand so that the proper translation tables are assigned when running either the user program or interactive services.

5.10. Example of Journaling Application Using a Global Network and Interactive Services

You can use standard job control procedures as explained in 5.8 to allocate, create, assemble, and execute your program. However, if you have interactive services available in your system, you can perform these steps from your remote terminal.

The interactive services operational procedures for creating a user program, executing it, and executing the journal utility consist of the following:

- Allocating the Files (5.10.1)

  Describes how to use interactive services to allocate the necessary job control and source and load module files.

- Creating a Source Module for a Global Network (5.10.2)

  Describes how to use EDT to create the program and save it as a source module.

- Assembling the Program and Creating a Load Module (5.10.3)

  Describes how to use jprocs in a job control stream source module to assemble and link the user program source module into a load module.

- Executing the Load Module (5.10.4)

  Explains how to create a job control steam to execute the user program load module.

- Signing On to the User Program (5.10.5)

  Explains how to sign off from interactive services, and sign on to the user program, and briefly how the program works.

- Executing the Journal Utility (5.10.6)

  Explains how to create a job control stream source module to execute the journal system utility.

5.10.1. Allocating the Files

You should allocate three files: one for source program modules and job control stream modules, one for load modules, and one for the journal records. One way to do this is to sign on to interactive services and use the ALLOCATE command, such as:

```
AL ST FIL=ICAMFILE,VSN=PUB001,SI=4
AL ST FIL=PJRICM,VSN=PUB001,SI=2
AL ST FILE=ICMJRN,VSN=PUB001,SI=2
```

5.10.2. Creating a Source Module for a Global Network

You can create your source module in a number of ways. One way is to create it interactively via EDT. First, you sign on to interactive services and respond to the request for system input with the following call to EDT, and then press XMIT:

```
SYSTEM INPUT > EDT
```

A message is displayed:

```
EDT
ED000 EDITOR VERSION 13.0 READY
1.0000
```

You can now start coding your program (assuming you set your assembler program forms tabs with an @SET C=; T=10,16,42). When you are finished, write the program to the file you previously allocated as the source file:

```
@WRITE DYNJRN,ICAMFILE,PUB001
```

or:

```
@WRITE PATJRN,ICAMFILE,PUB001
```

An SOE is returned with the cursor, signifying that the file has indeed been saved. You can continue to add to the program at this point if you wish, or (if you are completely done) you may want to bring the file back up as a confidence check.

```
@READ PATJRN,ICAMFILE,PUB001
```

It is a good safety precaution to just read the file back in, where it is appended to the EDT file you just wrote. This appears as a duplicate file that can easily be erased. You wouldn't want to assume that everything went alright and erase your original program before you knew for certain that the program was written out to the disk file.

Once you are certain that the module was written to the disk file, do an @D on the duplicate file lines. For instance, suppose the program has 330 lines. When you read the file back, the duplicate file becomes lines 331 to 661. Therefore, delete those lines with @D331:661.

If (on bringing your file back up) you note some lines that need correction, make those corrections and rewrite the program back out to the file again. This time EDT will respond with:

```
IS100 DYNCUP,ICAMFILE ALREADY EXISTS; OK TO WRITE TO IT?
```

This is a prompt to alert you that you are about to write over an existing file. Since that is what you want to do in this particular case, respond with a Y and press XMIT.

EDT again returns with an SOE and the cursor at the next line number for any additional input. If you wish to exit EDT at this point, enter vH and EDT will respond with:

```
EDT NORMAL TERMINATION
```

You are now back in the system and can enter any further system commands or log off the system.

If you previously had your program on a card deck, you can spool the file with a job control stream (assuming your system has spooling capability) such as:

```
// DATA FILEID,ICAMFILE
// JOB DYNJRN
        .  ]
        .  } Source deck
        .  ]
// FIN
```

Once the file is spooled, you can copy it from the spool file to the disk file either from the operator's console/workstation or, with interactive services, from the remote terminal:

```
COP FIL=ICAMFILE,HO=N,Q=RDR TO DYNJRN,ICAMFILE,VSN=PUB001,SI=4,SAT=Y
```

Once the program is copied to the disk file, delete the JOB and FIN statements via EDT, since they were only needed for the spool file creation.

If you do not have spooling available on your system, you can enter

```
FI DYNJRN,ICAMFILE,PUB001
```

from the operator's console.

You can now read in your file on disk from the remote terminal with EDT:

```
@READ DYNJRN,ICAMFILE PUB001
```

The editor responds with the number of lines in the program (for instance, 330.0000). You can do an @P of any lines you want to look at and make changes to the program using the editor commands. To delete the JOB and FIN statements, do an @D1 and an @D330, if they are the first and last lines in the program. Then rewrite the program back out to disk with

```
@WRITE DYNJRN,ICAMFILE,BEMPAC
```

A message is then displayed:

```
IS100 DYNJRN,ICAMFILE EXISTS; OK TO WRITE TO IT?
```

Respond with a Y and press XMIT.

EDT returns an SOE at the next line number (331) to be entered. If you wish to leave EDT at this point, enter @H and EDT will respond with:

```
EDT NORMAL TERMINATION
```

Press MSG WAIT for any further system mode commands you may wish to enter.

If you want to remain in EDT and create any further elements or modules (such as the assemble and link job control stream module in the next procedure), enter @D to delete the program you previously read in since you have already written it out to the disk file. This deletes the temporary file and returns EDT to line 1, ready for a new module. If you should forget to do this, EDT assumes you want any further input appended to the line number where you were previously.

The program shown in Figure 5-8 is the same as the global network dynamic session user program described in detail in the *ICAM Standard MCP Interface Programming Guide* (UP-8550) with the following minor exceptions:

1. The name of the network in the NATTACH and NDETACH macros is changed to request the global journal network described in 5.8 and Figure 5-7.

2. In a journal network, input on a TERM macro in the network definition is to a message processing procedure (MPPS). The MPPS directs the message to a process file. The GETCP in the user program then retrieves the message from the process file. (This is accomplished on line 174 by inserting the name of the process file, PRFA, into the input DTFCP.)

```
@P
   1.0000 PATJRN   START 0
   2.0000          TM#DSECT DUST
   3.0000          TM#DSECT GAWAKE
   4.0000          TM#DSECT GETPUT
   5.0000          BALR  10,0
   6.0000          USING *,10
   7.0000 *                ********************************
   8.0000 *                ****** COVER INPUT DTFCP ******
   9.0000 *                ********************************
  10.0000          USING TM#PRCS,2
  11.0000          LA    2,DUMY
  12.0000 *                ********************************
  13.0000 *                ****** COVER OUTPUT DTFCP *****
  14.0000 *                ********************************
  15.0000          USING TM#DEST,3
  16.0000          LA    3,OTPT
  17.0000 *                ********************************
  18.0000 *                ****** REQUEST A NETWORK *******
  19.0000 *                ********************************
+
  20.0000 BEGIN    NATTACH JGBL,ERRET=NETERR,APPS=CUP1
  21.0000          GAWAKE TYPE=INPUT,ENTRY=DGENTRY,BUFADR=DATAGRAM,
 X
  22.0000                 BUFLTH=40,APPS=CUP1
  23.0000          SLL   0,24                    SHIFT OUT ALL BUT GAWAKE ERRS
  24.0000          ST    0,CHECK                 STORE IN CHECK AREA
  25.0000          CLC   CHECK,CLEAR             CHECK FOR ERRORS
  26.0000          BNE   DISPGREJ                ANY ERRORS GO DISPLAY
  27.0000          CYIELD                        NO ERRORS WAIT FOR DATAGRAM
  28.0000 *                ********************************
  29.0000 *                 DATAGRAM ENTRY TO INTERROGATE
  30.0000 *                    DYNAMIC SESSION REQUESTS
  31.0000 *                ********************************
  32.0000 DGENTRY  LA    1,DATAGRAM             MAP DATAGRAM BUFFER
  33.0000          USING DSDTGRM,1               COVER REGISTER FOR DSECT
  34.0000          CLI   DATKOFUN,DATKOPEN       IS IT AN OPEN?
  35.0000          BE    OPNDAT                  YES
  36.0000          CLI   DATKOFUN,DATKCLOS         NO-IS IT A CLOSE?
  37.0000          BE    CLOSDAT                   YES
+
  38.0000          CLI   DATKOFUN,DATKABRT         NO-IS IT AN ABORT?
  39.0000          BE    ABRTDAT                 YES
  40.0000          CLI   DATKOFUN,DATKOPAC         NO-IS IT AN OPEN ACCEPT?
  41.0000          BE    ACCPTDAT                YES
  42.0000          CLI   DATKOFUN,DATKOPRJ       NO-IS IT AN OPEN REJECT?
  43.0000          BE    REJDAT                  YES
  44.0000          CLI   DATKOFUN,DATKCLCF         NO-IS IT A CLOSE CONFIRM?
  45.0000          BE    CLCFDAT                   YES
  46.0000          B     INVALID                 NO-IT'S NOT ANY VALID
  47.0000 *                                      FUNCTION GO DISPLAY
  48.0000 *                ********************************
  49.0000 *                  DATAGRAM PROCESSING ROUTINES
  50.0000 *                ********************************
  51.0000 OPNDAT   EQU   *                       OPEN DATAGRAM
  52.0000          MVC   EUTAG,DATK1CGI          SAVE END USER NAME
  53.0000          MVC   ICSESSID,DATK1SSI       SAVE ICAM SESSION ID
  54.0000          B     OPNACC                  RESPOND WITH ACCEPT
  55.0000 *
  56.0000 CLOSDAT  EQU   *                       CLOSE DATAGRAM
```

Figure 5-8. Sample User Program that Uses the Global Network JGBL (Part 1 of 6)

```
+
  57.0000           MVC   MYSESSID,DATK2SSU          SAVE USER SESSION ID
  58.0000           B     CLCFSES                    RESPOND WITH CLOSE CONFIRM
  59.0000 *
  60.0000 ABRTDAT   EQU   *                          ABORT DATAGRAM
  61.0000           MVC   MYSESSID,DATK3SSU          SAVE USER SESSION ID
  62.0000           B     ENDJOB                       GO TO END OF SESSION
  63.0000 *
  64.0000 ACCPTDAT  EQU   *                          ACCEPT DATAGRAM
  65.0000           MVC   ICSESSID,DATK4SSI          SAVE ICAM SESSION ID
  66.0000           MVC   MYSESSID,DATK4SSU          SAVE USER SESSION ID
  67.0000           B     GOPUT                      GO TO GET/PUT ROUTINE
  68.0000 *
  69.0000 REJDAT    EQU   *                          OPEN REJECT DATAGRAM
  70.0000           MVC   MYSESSID,DATK5SSU          SAVE USER SESSION ID
  71.0000           B     DISPREJ                    MY OPEN REJECTED-GO DISPLAY
  72.0000 *
  73.0000 CLCFDAT   EQU   *                          CLOSE CONFIRM DATAGRAM
  74.0000           MVC   MYSESSID,DATK6SSU          SAVE USER SESSION ID
  75.0000           B     ENDJOB                     GO TO NORMAL CLOSE
+
  76.0000 *          ********************************
  77.0000 *               SESCON BUILD ROUTINES
  78.0000 *          ********************************
  79.0000 OPNSES    LA    1,SES1                       PROGRAM WANTS TO INIT SES
  80.0000           USING DSESCON,1                  MAP SESCON
  81.0000           MVI   SESFUNC,SESTOPEN           SET TO OPEN SESSION
  82.0000           MVI   SESTQLTH,SESS1LT           SET PROPER LENGTH
  83.0000           MVC   SESS1CI,EUTAG              INSERT END USER NAME
  84.0000           MVC   SESS1SSU,MYSESSID          INSERT MY USER ID
  85.0000           MVC   SESS1IPQ,ZEROES            CLEAR INPUT QUEUE FIELD
  86.0000           SESCON MF=(E,(1))                 ISSUE SESCON
  87.0000           SLL   0,24                       SHIFT TO KEEP ONLY SES ERRS
  88.0000           ST    0,CHECK                    STORE IN CHECK AREA
  89.0000           CLC   CHECK,CLEAR                CHECK FOR ERRORS
  90.0000           BNE   DISPOPN                    ANY ERROR GO DISPLAY
  91.0000           CYIELD                           NO ERRORS AWAIT REPLY
  92.0000 *
  93.0000 CLSSES    LA    1,SES1                     PROGRAM WANTS TO CLOSE SES
  94.0000           USING DSESCON,1                  MAP SESCON
+
  95.0000           MVI   SESFUNC,SESTCLOS           SET TO CLOSE SESSION
  96.0000           MVI   SESTQLTH,SESS2LT           SET PROPER LENGTH
  97.0000           MVC   SESS2SSI,ICSESSID          INSERT ICAM SESSION ID
  98.0000           SESCON MF=(E,(1))                 ISSUE SESCON
  99.0000           SLL   0,24                       SHIFT TO KEEP ONLY SES ERRS
 100.0000           ST    0,CHECK                    STORE IN CHECK AREA
 101.0000           CLC   CHECK,CLEAR                CHECK FOR ERRORS
 102.0000           BNE   DISPCLS                    ANY ERRORS GO DISPLAY
 103.0000           CYIELD                           NO ERRORS AWAIT CONFIRM
 104.0000 *
 105.0000 ABRTSES   LA    1,SES1                     PROGRAM WANTS TO ABORT SES
 106.0000           USING DSESCON,1                  MAP SESCON
 107.0000           MVI   SESFUNC,SESTABRT           SET TO ABORT
 108.0000           MVI   SESTQLTH,SESS3LT           SET PROPER LENGTH
 109.0000           MVC   SESS3SSI,ICSESSID          INSERT ICAM ID
 110.0000           SESCON MF=(E,(1))                 ISSUE THE FUNCTION
 111.0000           SLL   0,24                       SHIFT TO KEEP ONLY SES ERRS
 112.0000           ST    0,CHECK                    STORE IN CHECK AREA
 113.0000           CLC   CHECK,CLEAR                CHECK FOR ERRORS
+
 114.0000           BNE   DISPABT                    ANY ERRORS GO DISPLAY
 115.0000           B     DONEJOB                    GO TO SNAP AND END
 116.0000 *
```

Figure 5-8. Sample User Program that Uses the Global Network JGBL (Part 2 of 6)

```
 117.0000 OPNACC   LA     1,SES1                    ACCEPT END USER REQ TO OPEN
 118.0000          USING  DSESCON,1                 MAP DSECT
 119.0000          MVI    SESFUNC,SESTOPAC          SET TO OPEN ACCEPT
 120.0000          MVI    SESTOLTH,SESS4LT          SET PROPER LENGTH
 121.0000          MVC    SESS4SSI,ICSESSID         INSERT ICAM ID
 122.0000          MVC    SESS4SSO,MYSESSID             INSERT MY USER ID
 123.0000          SESCON MF=(E,(1))                 ISSUE SESCON
 124.0000          SLL    0,24                      SHIFT TO KEEP ONLY SES ERRS
 125.0000          ST     0,CHECK                   STORE IN CHECK AREA
 126.0000          CLC    CHECK,CLEAR               CHECK FOR ERRORS
 127.0000          BNE    DISPACC                   NO-GO DISPLAY
 128.0000          B      GOPUT                     GO TO GET/PUT ROUTINE
 129.0000 *
 130.0000 OPNREJ   LA     1,SES1                    OPEN REJECT REQUEST
 131.0000          USING  DSESCON,1                 MAP DSECT
 132.0000          MVI    SESFUNC,SESTOPRJ          SET TO OPEN REJECT
+
 133.0000          MVI    SESTOLTH,SESS5LT          SET PROPER LENGTH
 134.0000          MVC    SESS5SSI,ICSESSID          INSERT ICAM SESSION ID
 135.0000          SESCON MF=(E,(1))                     ISSUE SESCON
 136.0000          SLL    0,24                      SHIFT TO KEEP ONLY SES ERRS
 137.0000          ST     0,CHECK                   STORE IN CHECK AREA
 138.0000          CLC    CHECK,CLEAR               CHECK FOR ERRORS
 139.0000          BNE    DISPREJ                   NO-GO DISPLAY
 140.0000          CYIELD                           WAIT FOR NEW DATAGRAM
 141.0000 *
 142.0000 CLOFSES  LA     1,SES1                    CONFIRM CLOSE REQUEST
 143.0000          USING  DSESCON,1                 MAP DSECT
 144.0000          MVI    SESFUNC,SESTCLOF          SET TO CONFIRM
 145.0000          MVI    SESTOLTH,SESS6LT          SET TO PROPER LENGTH
 146.0000          MVC    SESS6SSI,ICSESSID         INSERT ICAM SESSION ID
 147.0000          SESCON MF=(E,(1))                     ISSUE SESCON
 148.0000          SLL    0,24                      SHIFT TO KEEP ONLY SES ERRS
 149.0000          ST     0,CHECK                   STORE IN CHECK AREA
 150.0000          CLC    CHECK,CLEAR               CHECK FOR ERRORS
 151.0000          BNE    DISPCLOF                  ANY ERRORS GO DISPLAY
+
 152.0000          B      CLOSE                     GO TO NORMAL END
 153.0000 *        ********************************
 154.0000 *             SET UP FIRST PUTCP
 155.0000 *        ********************************
 156.0000 GOPUT    EQU    *                         START OF GET/PUT ROUTINE
 157.0000 LOOP     OI     TM#DSEG,TM#DHDR++TM#DTND  SET FOR COMPLETE MESSAGE
 158.0000          XC     TM#DERR,TM#DERR           CLEAR ERROR BYTE AREA
 159.0000          MVC    TM#DENA,EUTAG             INSERT DESTINATION NAME
 160.0000 PUTIT1   PUTCP  OTPT,MSGOT1               ISSUE PUT
 161.0000 *        ********************************
 162.0000 *             SET UP SECOND PUTCP
 163.0000 *        ********************************
 164.0000          OI     TM#DSEG,TM#DHDR++TM#DTND
 165.0000          XC     TM#DERR,TM#DERR
 166.0000          MVC    TM#DENA,EUTAG             INSERT DESTINATION NAME
 167.0000 PUTIT2   PUTCP  OTPT,MSGOT2
 168.0000 *        ********************************
 169.0000 *        SET UP DEFERRED GETCP
 170.0000 *        ********************************
+
 171.0000          XC     TM#PERR,TM#PERR
 172.0000          MVI    TM#PIND,TM#PIRL
 173.0000          MVC    TM#PCMPA,=A(BINGO)
 174.0000          MVC    TM#PNAM,C'PRFA'            INSERT INPUT DTF NAME TO USE
```

Figure 5-8. Sample User Program that Uses the Global Network JGBL (Part 3 of 6)

```
 175.0000           MVC   MSGIN,TXTCHARS
 176.0000 GETIT     GETCP DUMY,MSGIN
 177.0000 *                 *******************************
 178.0000 *                 ** YIELD FOR ANY ACTIVITY ***
 179.0000 *                 *******************************
 180.0000           CYIELD
 181.0000 *                 *******************************
 182.0000 *                 **** WHAT KIND OF MESSAGE? ****
 183.0000 *                 ****    DONE OR A NAME?    ****
 184.0000 *                 *******************************
 185.0000 BINGO     SNAP  MSGOUT1,CHECK+3
 186.0000          CLC   TXTIN(4),=C'DONE'
 187.0000           BE    ENDJOB
 188.0000           CLC   TXTIN(4),=X'84969585'
+
 189.0000           BE    ENDJOB
 190.0000 *         *******************************
 191.0000 *                 ECHO INPUT TO OUTPUT
 192.0000 *         *******************************
 193.0000           MVC   TXTOUT,TXTIN
 194.0000           MVC   TXTIN,C' '
 195.0000           MVC   TXTIN+1(L'TXTIN-1),TXTIN
 196.0000 *         *******************************
 197.0000 *         SET UP PUTCP TO ECHO MESSAGE
 198.0000 *         *******************************
 199.0000           OI    TM#DSEG,TM#DHDR++TM#DTND
 200.0000           XC    TM#DERR,TM#DERR
 201.0000           MVC   TM#DENA,EUTAG
 202.0000 PUTIT3    PUTCP OTPT,MSGOUT
 203.0000           B     LOOP
 204.0000 *         *******************************
 205.0000 *            RELEASE THE GLOBAL NETWORK
 206.0000 *         *******************************
 207.0000 ENDJOB    NDETACH JGBL
+
 208.0000           B     DONE
 209.0000 *         *******************************
 210.0000 *           BUILD ERROR DISPLAY MESSAGES
 211.0000 *         *******************************
 212.0000 NETERR    LA    1,MSG1
 213.0000           OPR   MSG1,27
 214.0000           B     DONEJOB                    SNAP
 215.0000 NOGET     LA    1,MSG2
 216.0000           OPR   MSG2,12
 217.0000           B     DONEJOB                    SNAP
 218.0000 NOPUT     LA    1,MSG3
 219.0000           OPR   MSG3,12
 220.0000           B     DONEJOB                    SNAP
 221.0000 NOBUFF    LA    1,MSG4
 222.0000           OPR   MSG4,19
 223.0000           B     DONEJOB                    SNAP
 224.0000 DISPOPN   EQU   *
 225.0000           LA    1,MSG5
 226.0000           OPR   MSG5,18
+
 227.0000           B     DONEJOB                    SNAP
 228.0000 DISPACC   EQU   *
 229.0000           LA    1,MSG6
 230.0000           OPR   MSG6,20
 231.0000           B     DONEJOB                    SNAP
 232.0000 DISPREJ   EQU   *
 233.0000           LA    1,MSG7
 234.0000           OPR   MSG7,20
 235.0000           B     DONEJOB                    SNAP
```

Figure 5-8. Sample User Program that Uses the Global Network JGBL (Part 4 of 6)

```
 236.0000 DISPCLS  EQU   *
 237.0000          LA    1,MSG8
 238.0000          OPR   MSG8,19
 239.0000          B     DONEJOB                    SNAP
 240.0000 DISPABT  EQU   *
 241.0000          LA    1,MSG9
 242.0000          OPR   MSG9,21
 243.0000          B     DONEJOB                    SNAP
 244.0000 DISPCLCF EQU   *
 245.0000          LA    1,MSGA
+
 246.0000          OPR   MSGA,21
 247.0000          B     DONEJOB                    SNAP
 248.0000 INVALID  EQU   *
 249.0000          LA    1,MSGB
 250.0000          OPR   MSGB,25
 251.0000          B     DONEJOB                    SNAP
 252.0000 DISPGREJ EQU   *
 253.0000          LA    1,MSGE
 254.0000          OPR   MSGE,20
 255.0000          B     DONEJOB                    SNAP
 256.0000 DONEJOB  SNAP  BEGIN,CHECK+3
 257.0000 DONE     EOJ
 258.0000 *                ********************************
 259.0000 *                ****     CONSTANT AREA      ****
 260.0000 *                ********************************
 261.0000 DUMY     DTFCP TYPE=GT,ERRET=NOGET
 262.0000 OTPT     DTFCP TYPE=PT,ERRET=NOPUT,DEST=(T,DTAG),NOBAV=NOBUFF
 263.0000 SES1     SESCON FUNCT=OPEN,MID=AA01 TONAME=FILL,MF=L
 264.0000 DSESCON  SESCON MF=(D,SES)
+
 265.0000 DSDTGRM  CONTDTG MF=(D,DAT)
 266.0000          DS    0F
 267.0000 MSG1     DC    CL26 ERROR ISSUING NATTACH MACRO'
 268.0000 MSG2     DC    CL12'ERROR ON GET'
 269.0000 MSG3     DC    CL12'ERROR ON PUT'
 270.0000 MSG4     DC    CL19'NO BUFFER AVAILABLE'
 271.0000 MSG5     DC    CL18'OPEN NOT VALIDATED'
 272.0000 MSG6     DC    CL20'ACCEPT NOT VALIDATED'
 273.0000 MSG7     DC    CL20'REJECT NOT VALIDATED'
 274.0000 MSG8     DC    CL19'CLOSE NOT VALIDATED'
 275.0000 MSG9     DC    CL19'ABORT NOT VALIDATED'
 276.0000 MSGA     DC    CL21'CONFIRM NOT VALIDATED'
 277.0000 MSGB     DC    CL25'INVALID FUNCTION RECEIVED'
 278.0000 MSGE     DC    CL20'GAWAKE NOT VALIDATED'
 279.0000          DS    0H
 280.0000 MSGOT1   DC    X'0019'
 281.0000          DC    X'10030000'
 282.0000          DC    C'WELCOME TO OS/3 ICAM'
 283.0000          DC    X'0D'
+
 284.0000          DS    0H
 285.0000 MSGOT2   DC    X'001A'
 286.0000          DC    C'TYPE IN YOUR NAME PLEASE'
 287.0000          DC    X'0D1E'
 288.0000          DS    0H
 289.0000 MSGIN    DC    X'0000'
 290.0000          DC    CL7' '      <-- (7 SPACES FOR SOE AND 6 UNISCOPE SCREEN
 291.0000 TXTIN    DC    CL256' '        CONTROL CHARACTERS WHEN DICE=OFF)
 292.0000 INETX    DC    X'FFFF'
 293.0000          DS    0H
 294.0000 MSGOUT   DC    X'010C'
 295.0000          DC    X'0D'
 296.0000          DC    C'THANK YOU '
```

Figure 5-8. Sample User Program that Uses the Global Network JGBL (Part 5 of 6)

```
 297.0000 TXTOUT   DC    CL256' '
 298.0000 OUTETX   DC    X'0D'
 299.0000 TXTCHARS DC    X'0105'
 300.0000 ICSESSID DC    CL4' '                    ICAM SUPPLIED SESSION ID
 301.0000 MYSESSID DC    C'AA01'                   MY SESSION ID
 302.0000 ZEROES   DC    XL4'0'
+
 303.0000 CLEAR    DC    XL4'0'
 304.0000 EUTAG    DC    CL4' '                    NAME OF END USER IN SESSION
 305.0000          DS    0F
 306.0000 DATAGRAM DC    40' '                     DATAGRAM BUFFER AREA
 307.0000 CHECK    DC    XL4'0'                    ERROR BYTE STORAGE AREA
 308.0000 *                                        FOR GAWAKE AND SESCON
 309.0000 *                                        VALIDATION
 310.0000          END
```

Figure 5-8. Sample User Program that Uses the Global Network JGBL (Part 6 of 6)

Briefly, the program works as follows. First, dial in and sign on as explained in 5.10.6. If your sign-on is accepted, the queued message appears exactly the same as it did in the dedicated network user program in Figure 5-5. The program requests the network JGBL to send the prompt message:

```
WELCOME TO OS/3 ICAM
TYPE IN YOUR NAME PLEASE
```

It then issues a deferred GETCP request for input and yields control to ICAM. When a message is received, the program echoes it back to the sending terminal with a THANK YOU appended and repeats the prompt for more messages. This process continues until you type in a sign-off ($$SOFF), or the word DONE (in capital or lowercase letters). For example:

```
WELCOME TO OS/3 ICAM
TYPE IN YOUR NAME PLEASE
JOHN Q. PUBLIC
THANK YOU JOHN Q. PUBLIC

WELCOME TO OS/3 ICAM
TYPE IN YOUR NAME PLEASE

$$SOFF
```

Since this is a dynamic session program, we recommend that you terminate the session with a sign-off ($$SOFF). You then receive a response:

```
SESSION PATH CLOSED
```

You can now sign back on to interactive services or terminate. You terminate with a disconnect by placing the phone out of DATA mode and picking up the receiver. Ensure that a hangup took place by listening for a normal dial tone.

5.10.3. Assembling the Program and Creating a Load Module

If you have a global network with interactive services, you can sign on to interactive services after you dial in, then call on EDT to create the job control stream source module necessary to assemble the program and create the load module. It is coded exactly the same as that explained in 5.8.3. However, you can now write it directly to a file from your terminal via interactive services, and assign the module name

```
// JOB R-PATJRN,,AB000,,,,,H
//PATJRN ASM IN=(PUB001,ICAMFILE)
//JRNLD LINK PATJRN,OUT=(PUB001,PJRICM)
/&
```

Then enter:

```
@WRITE R-PATJRN,ICAMFILE,PUB001
```

and exit EDT with:

```
@H
```

Now, when you're ready to run the job, you can execute it from the remote terminal by pressing MSG WAIT or FUNC/SYSTEM MODE (according to your terminal type). Then respond to the request for input with:

```
SYSTEM INPUT > RV R-PATJRN:(ICAMFILE,PUB001)
```

If the job completes successfully, you should see the following messages on the terminal:

```
OH R03 RUN PROCESSOR SUCCESSFULLY PROCESSED R-PATJRN
4B JC01 JOB R-PATJRN EXECUTING JOB STEP ASM0000 #001 13:36:21
4C A116            NO STATEMENTS FLAGGED IN THIS ASSEMBLY
4D A118         UPSI BYTE SETTING x'00'
4D JC01 JOB R-PATJRN EXECUTING JOB STEP LINKEDT00 #002 14:33:48
4F JC02 JOB R-PATJRN TERMINATED NORMALLY               14:35:34
```

Note: *If you encounter any errors, refer to the section on assembling and creating a load module in the* ICAM Standard MCP Interface Programming Guide *(UP-8550) for a more detailed discussion of status and error messages.*

5.10.4. Executing the Load Module

After you create the user program load module, you need a job control stream module to execute it. Again, you can sign on to interactive services, call EDT, and create the module. The module is exactly the same as described in 5.8.5. However, you can now write it directly to a file from your terminal via interactive services and assign the module a name (see Figure 5-9).

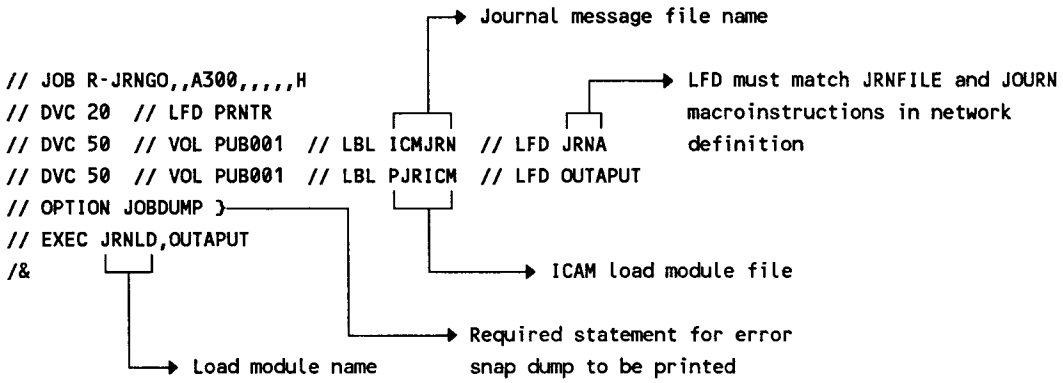


Figure 5-9. Executing the Load Module

Then enter:

```
@WRITE R-JRNGO,ICAMFILE,PUB001
```

and exit EDT with:

```
@H
```

The job control stream (JCS) module is now in your source file and you can execute it at any time by pressing MSG WAIT or FUNC/SYSTEM MODE (according to the terminal you have), and responding to the request for input with:

```
SYSTEM INPUT > RV R-JRNGO:(ICAMFILE,PUB001)
```

When the program is executed, the following messages are displayed:

```
R03 RUN PROCESSOR SUCCESSFULLY PROCESSED R-JRNGO
JC04 JOB R-JRNGO EXECUTING JOB STEP JRNLD00 #001 14:38:03
```

5.10.5. Signing on to the User Program

At this point, the ICAM user program is loaded and executing. Since it is in a CYIELD condition awaiting datagram input, it is, in effect, merely waiting for you to sign on to it. If you wish to sign on from the terminal you are presently on, you must first sign off from interactive services with a $$SOFF command.

When interactive services responds with

```
SESSION PATH CLOSED
```

all you need to do is sign on to the user program using the label of the standard interface locap macro. (This is also the label of the input DTFCP that defines this file.) For example:

```
$$SON TRM1CUP1
```

Once your sign-on is accepted, the queued message immediately appears on the screen, exactly the same as it did in the dedicated network user program described in 5.5. The program requests JGBL to send the following messages:

```
WELCOME TO OS/3 ICAM
TYPE IN YOUR NAME PLEASE
```

It then issues a deferred GETCP request for input and again yields control to ICAM. When a message is received, the program places a THANK YOU message in front of it, sends it back to the terminal that signed on, and repeats the prompt. This process continues until you type in a sign-off ($$SOFF) command, or the word DONE (in capital or lowercase letters). For example:

```
WELCOME TO OS/3 ICAM
TYPE IN YOUR NAME PLEASE
JOHN Q. PUBLIC
THANK YOU JOHN Q. PUBLIC

WELCOME TO OS/3 ICAM
TYPE IN YOUR NAME PLEASE

$$SOFF
```

Since this is a dynamic session program, we recommend that you terminate the session with a sign-off ($$SOFF). You then receive the following response:

```
SESSION PATH CLOSED
```

You can now sign back on to interactive services to run the journal system utility, or terminate. You terminate with a disconnect by placing the phone out of the DATA mode and picking up the receiver. Ensure that a hangup took place by listening for a normal dial tone.

5.10.6. Executing the Journal Utility

The journal utility system program is executed by creating a job control stream module via interactive services and EDT (see Figure 5-10). The DVC-LFD set specifies the physical journal file name (LBL) containing the journal records created by the execution of the user program load module. INPUT1 is the required logical file name. The OPTION JOBDUMP statement is required for an abnormal termination error dump. The name of the journal utility (JUST) is used on the EXEC statement. The SELECT statement causes line performance statistics, buffer statistics, and journal message records to be included. All you need to do now is press MSG WAIT and respond to the request for input with:

```
SYSTEM INPUT > RV R-JRNUT:(ICAMFILE,PUB001)
```

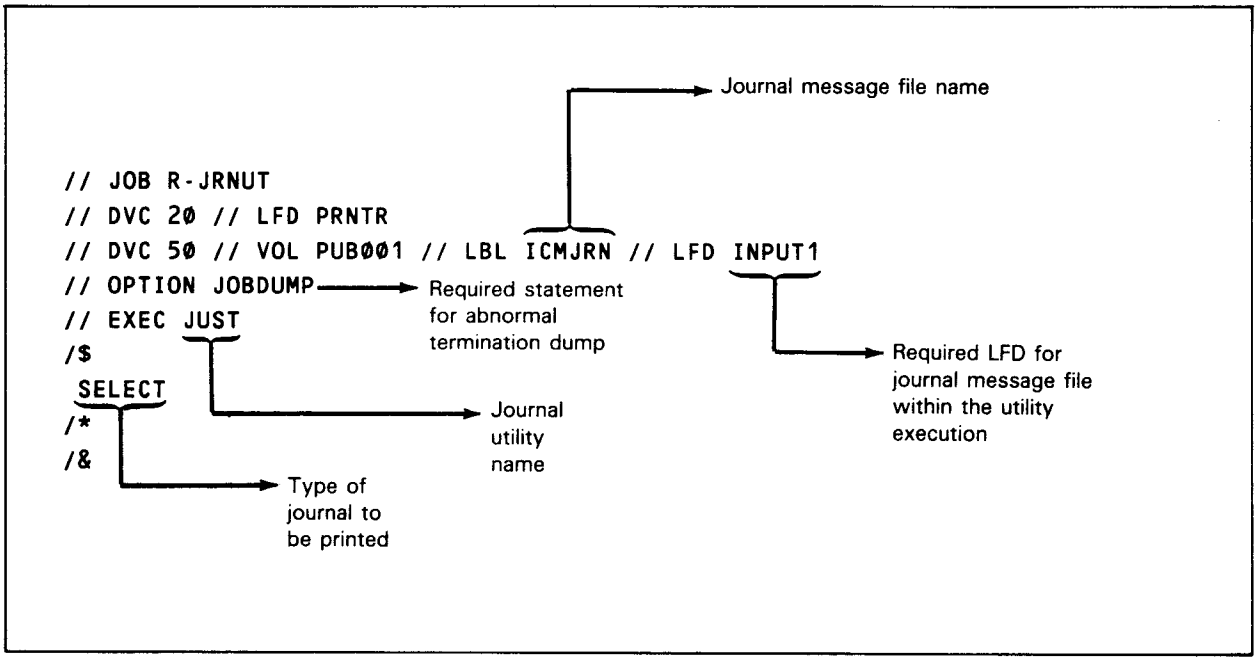


Figure 5-10. Executing the Journal Utility

Section 6
COBOL Message Control System (CMCS)

6.1. General Description

The COBOL message control system (CMCS) provides the software interface between a COBOL communications user program and ICAM. The design of CMCS is defined by the requirements of the *American National Standard COBOL, X3.23-1974* (ANSI '74). You generate a CMCS module by running the CMCS#GEN job described in 6.5.

CMCS performs the following:

- Resolves the differences between a COBOL program and ICAM terminologies
- Initiates the ICAM communications functions
- Translates ICAM error and status conditions to the ANSI '74 format

CMCS uses the standard MCP (STDMCP) interface to acquire the necessary communications features that are required by the COBOL program. Careful consideration should be given at network generation time to incorporate all the ICAM features required to support the COBOL program. A COBOL user can also use any program independent features of the standard interface, such as a message processing procedure specification (MPPS).

The COBOL program requests communications functions, such as sending an output message, via the COBOL communications verbs. Then, CMCS converts these verbs into the appropriate ICAM commands. ICAM takes care of queuing the output to the proper output queues. It also handles all input traffic, placing each input message on the correct input queue according to the specification in the network definition. When the COBOL program requests a message from an input queue, ICAM accesses the appropriate queues or hierarchical queue structure and delivers the message to CMCS and thereby to the COBOL program.

6.2. COBOL Communications Overview

An overview of the COBOL communications facilities and functions is presented in 6.2.1 through 6.2.4. A more detailed explanation of COBOL communications is presented in the *ANSI '74 COBOL Programmer Reference* (UP-8613). ANSI '74 COBOL defines the following three types of communications facilities:

- Symbolic sources
- Symbolic destinations
- Symbolic queues

Each of these facilities can have a 1- to 12-character name. The symbolic source and destination names are assigned to the source and destination of the message; for example, communications devices. The symbolic queue names are assigned to all input queues that can be configured into a hierarchical queue structure. The concept of the queue structure is explained in 6.2.2.

6.2.1. Communications Descriptors

The communications descriptors located in the COBOL program are the facility interface areas to CMCS. There are two types of communications descriptors: an input CD and an output CD.

The communications descriptors (CD) contain the symbolic names of the facilities related to a particular communications function to be performed. When a COBOL program requests an input message, it is retrieved from an input queue defined by the symbolic queue names in an input CD. Likewise, when sending messages, the message is sent to one or more symbolic destinations named in an output CD. The format of the communications descriptors is shown in Table 6-1.

6.2.2. COBOL Hierarchical Queue Structures

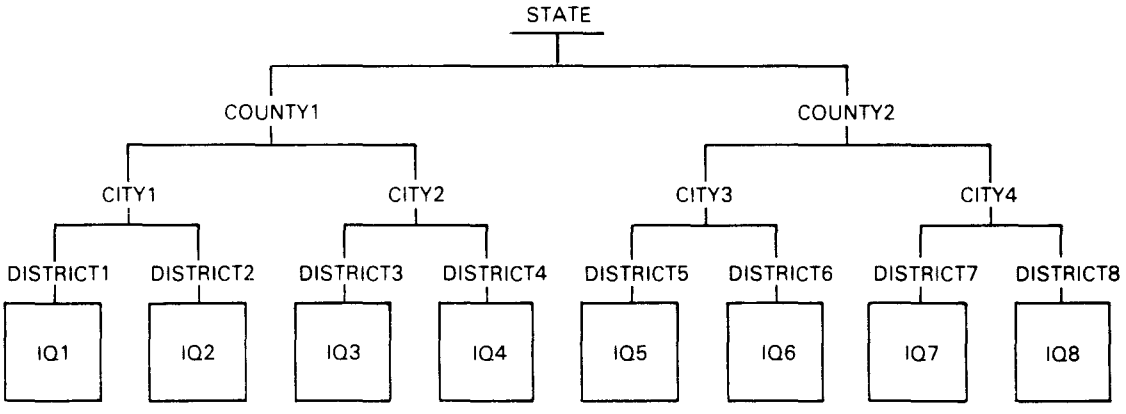
When designing a COBOL communications program, an understanding of the hierarchical queue structure is helpful. The queue structure divides the queues into a series of levels. ANSI '74 permits up to three subqueue levels. The COBOL program can then indicate the number and sequence of the input queues to be accessed via the symbolic queue names in the input communications descriptor. Whenever a message or part of a message is dequeued from an input queue, CMCS updates the input CD with the applicable symbolic queue names. Figure 6-1 is an example of a symmetrical hierarchical queue structure.

Table 6-1. Communications Descriptors

Input CD	
Name	**Size (bytes)**
Symbolic queue	12
Symbolic subqueue 1	12
Symbolic subqueue 2	12
Symbolic subqueue 3	12
Message date	6
Message time	8
Symbolic source	12
Text length	4
End key	1
Status key	2
Message count	6
Output CD	
Destination count	4
Text length	4
Status key	2
Error key no. 1	1
Symbolic destination name no. 1	12
Error key no. n	1
Symbolic destination name no. n	12

NOTES:

At compilation time, the ERROR KEY and SYMBOLIC DESTINATION names are repeated as many times as the OCCUR statement indicates. At execution time, the count inserted into the DESTINATION COUNT field determines the number of destination messages that are actually sent.



NOTES:

1. The symbolic queue name of the entire structure is STATE.
2. The symbolic subqueue-1 names (first level) are COUNTY1 and COUNTY2.
3. The symbolic subqueue-2 names (second level) are CITY1 - CITY4.
4. The symbolic subqueue-3 names (third level) are DIST1 - DIST8.
5. The input queues configured are named IQ1 - IQ8.

Figure 6-1. Typical Symmetrical Hierarchical Queue Structure

The following examples describe how the queue structure works. See Example 2 under the DB#SQT macro (6.5.2) to see how to specify this queue structure.

Example 1

Suppose your input communications descriptor contains symbolic queue name STATE and the subqueues shown in Figure 6-1. If, in your procedure division, you space-filled all the subqueues except the subqueue-1 name COUNTY1 and the subqueue-2 name CITY2, and you inserted characters representing names in those, then the input queue to process file IQ3 is accessed first. If no message is found, process file IQ4 is accessed.

Example 2

Suppose the input communications descriptor only contained the symbolic queue name STATE. All eight input queues are accessed starting with IQ1 and stopping when an available message is found.

Example 3

Suppose the input communications descriptor only contained the symbolic queue name STATE and a message is retrieved from IQ6. The symbolic queue names STATE, COUNTY2, CITY3, and DIST6 are placed in the input communications descriptor by CMCS.

The queue structure can also be asymmetrical. Figure 6-2 shows the same eight queues but in an asymmetrical order. See Example 3 under the DB#SQT macro (6.5.2) to see how to specify this queue structure.

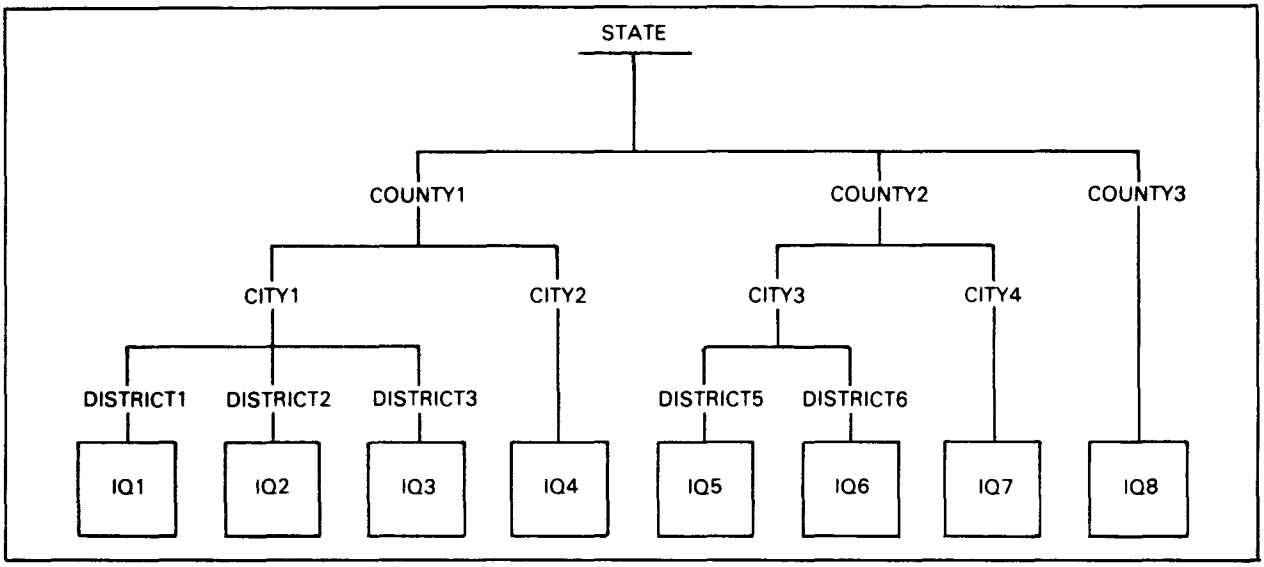


Figure 6-2. Typical Asymmetrical Hierarchical Queue Structure

6.2.3. COBOL Communications Functions

ANSI '74 has five designated verbs which perform communications functions. They are: ACCEPT, DISABLE, ENABLE, RECEIVE, and SEND. These verbs are used with both required and optional phrases that form COBOL statements. Table 6-2 lists the statements and their functions.

Table 6-2. COBOL Communications Statements/Functions

Statement	Function
ACCEPT MESSAGE COUNT	Requests a count of available input messages within a symbolic queue structure specified in an input CD. CMCS places the count in the message count field on the input CD.
DISABLE	Breaks the logical connection between a communications device and the input or output queues. There are three DISABLE variations: 1. DISABLE OUTPUT associated with an output CD and is performed for all symbolic destinations in the CD. 2. DISABLE INPUT associated with an input CD and is performed for all communications devices that route messages to input queues in the specified symbolic queue structure. 3. DISABLE INPUT TERMINAL associated with an input CD but only performed for communications devices named in the symbolic source field of input CD.
ENABLE	Establishes the logical connection between a communications device and the input/output queues. ENABLE has the same three variations as DISABLE.
RECEIVE	Requests the transfer of a message or message segment from an input queue designated by the input CD symbolic queue names to work area in the COBOL program. May optionally include a NO DATA phrase telling CMCS to return control to the COBOL program's NO DATA phrase address when no available message is found on any input queue in the specified queue structure. If NO DATA phrase is absent, control is not returned to the COBOL program until a message is available on one of the specified queues. During execution of the RECEIVE, CMCS updates the applicable fields in the input CD with: symbolic source name, text length, message time and date, and status.
SEND	Requests the transfer of message text from a COBOL program's work area to the output queues of all symbolic destinations in the named output CD. May optionally have ADVANCING phrase specifying vertical positioning information. Applies to complete message or segments.

NOTES:

1. When input or output is disabled, the COBOL program can dequeue any messages already on input queues and continue to enqueue messages on output queues.

2. Both DISABLE and ENABLE statements have related COBOL passwords. Up to 10 characters are permitted for the password.

3. In COBOL, a complete MESSAGE can be subdivided into SEGMENTS that can be subdivided into PORTIONS. The indication of the message unit being transferred is a part of the SEND statement. Message text is not transmitted to a communications device until the complete message has been enqueued on the output queues.

6.2.4. COBOL Program Execution

A COBOL communications program may be scheduled for execution in one of two methods:

1. As a result of the first input message received

2. Via job control procedures (for example, operator action)

If method 1 is used, one of the input CDs must be designated as a CD FOR INITIAL INPUT. Initially, the queue and subqueue name fields in the CD FOR INITIAL INPUT are spaced-filled. At initialization time, CMCS interrogates ICAM to determine if the COBOL program was scheduled due to the receipt of a message. If so, CMCS updates the CD with all related symbolic queue names that apply to the input queue containing the message. The message is not transferred to the COBOL program at this time, but remains enqueued until retrieved by the COBOL program by a RECEIVE statement.

If method 2 is used, the symbolic queue and subqueue name fields in the CD FOR INITIAL INPUT are not updated and remain space-filled.

6.3. ICAM Network Generation Considerations

CMCS provides support for both global and dedicated networks. The remote scheduling of a COBOL program upon receipt of the first message from a communications device is only supported in global networks. However, the remote scheduling does not include any system-to-terminal dialog until a message is sent to the initiating terminal by the COBOL program. Any delay or failure in loading or scheduling the program is not identified at the initiating terminal.

Table 6-3 presents the required ICAM network generation macros. A typical ICAM network generation is illustrated in Figure 6-3 (see 6.6.1).

Table 6-3. Required ICAM Parameters for CMCS

Macroinstruction	Parameter	Function
CCA	TYPE=(STDMCP) or TYPE=(GBL,,node)	Generates a dedicated or global standard interface network
	SAVE=YES	Causes the network to be saved in object module format for CMCS use
	FEATURES=(DATIME)	Causes the data and time stamping ICAM module to be included at ICAM generation time
	GAWAKE=YES	Used in a global network to allow automatic scheduling of the COBOL program
LOCAP (global only)	JOBNAME=jobname	Specifies the jobname of the filed control stream in YJCS scheduling the COBOL program CUP for execution
	JOBINIT=(LOAD, REPORT)	Specifies the remote scheduling of the program
TERM	DICE=ON	Specifies DICE sequence insertion on input messages supporting message segmentation
	INPUT=	Specifies the label of LOCAP, PRCS, TERM, or MPPS macroinstruction
	LOW=	Specifies a low priority output queue
DLIST	destination-name	At least two valid destination names must be specified for ICAM configuration purposes
PRCS	LOW=	Specifies a low priority input queue
SESSION	PRIMARY EU1=(name) EU2=(name)	Specifies a static session between the LOCAP of the COBOL program and the process file that is supplying messages to the LOCAP

6.4. CMCS/COBOL/ICAM Relationship

CMCS establishes the relationship between a COBOL communications program and ICAM by means of cross-reference tables and executable code created at CMCS generation time. When called by the COBOL program, CMCS translates the COBOL data in the communications descriptors and the verbs to the required ICAM format. CMCS then issues a call to ICAM requesting the communications function. After execution, ICAM returns any error and status data to CMCS, which translates this information to the required COBOL format and places it in the CD. CMCS may also update the CD with facility and message data, such as symbolic source name, message date, and so forth.

CMCS is divided into two parts: a data base section and a processing section. The data base section contains the COBOL/ICAM cross-reference tables and ICAM control tables. The processing section contains the executable CMCS code.

You use the following macros in your CMCS generation: DB#GEN, DB#SQT, DB#SNT, DB#IRT, and DB#END. The keyword parameters associated with these macros supply the required data to establish the COBOL/ICAM relationships. Table 6-4 summarizes the COBOL-related macro parameters; Table 6-5 summarizes the ICAM-related macro parameters. The macros are described in detail in 6.5.2.

Table 6-4. COBOL Program/CMCS Generation Relationship

COBOL Program	CMCS Macroinstruction
ENABLE/DISABLE password defined in working storage	DB#GEN CPASS=password
Number of output communications descriptors defined in data division	DB#GEN OUTCD=number
Symbolic name supplied with the SYMBOLIC QUEUE clause in the input CD defining the queue structure	DB#SQT SYMQ=symbolic-queue-name
Symbolic name supplied with SYMBOLIC-SUB-QUEUE-1, SYMBOLIC SUB-QUEUE-2, SYMBOLIC SUB-QUEUE-3 in the input CD	DB#SQT SUBQ1=name, SUBQ2=name, SUBQ3=name
Symbolic name supplied with either SYMBOLIC SOURCE or SYMBOLIC DESTINATION in input or output CD	DB#SNT SYMN=symbolic-name

Table 6-5. ICAM Network Definition/CMCS Generation Relationship

ICAM Macroinstruction	CMCS Macroinstruction
CCA network-name	DB#GEN NNAME=network-name
CCA PASSWORD=name	DB#GEN NPASS=name
LOCAP symbol (application-name)	DB #GEN ANAME=name
LINE PRCS=process-filename	DB#SQT and DB#IRT PFILE=filename
PRCS prcs-name	DB#SQT and DB#IRT PFILE=filename
TERM terminal-name	DB#SNT TERM=terminal-name DB#IRT TERM=terminal-name

6.4.1. CMCS Cross-Reference Tables

CMCS creates three cross-reference tables - symbolic queue table, symbolic name table, and input routing table.

- Symbolic queue table

  Each DB#SQT macro creates an entry in a symbolic queue table (SQT). The entry contains the name of the ICAM input queue (process file name) and all applicable COBOL symbolic subqueue names. There should be as many entries in an SQT as there are input queues in the symbolic queue structure. There can be multiple SQTs but the input queue can only be used once in all SQTs.

- Symbolic name table

  Each DB#SNT macro creates an entry in the symbolic name table (SNT). The entry contains the name of an ICAM source or destination (terminal name or process file name) and the applicable COBOL symbolic source or destination name. Since a process file can be a valid destination with the STDMCP interface, any process file used in this manner must have a SNT entry. There is only one SNT for each CMCS.

- Input routing table

  Each DB#IRT macro creates an entry in the input routing table (IRT). The IRT is ICAM-oriented since it defines the routing of input messages from terminals to process files. Each entry contains a process file name and index values for the terminals in the SNT that may have messages routed to the process file. The IRT is dependent on the routing performed by MPPS or LINE macros. There is only one IRT for each CMCS.

6.5. CMCS Module Generation

After you have determined the COBOL symbolic facility names and their relationships to your ICAM network logical names, you can generate a CMCS module. The CMCS generation program is initiated through a canned job control stream called CMCS#GEN. You supply information to the generation program through:

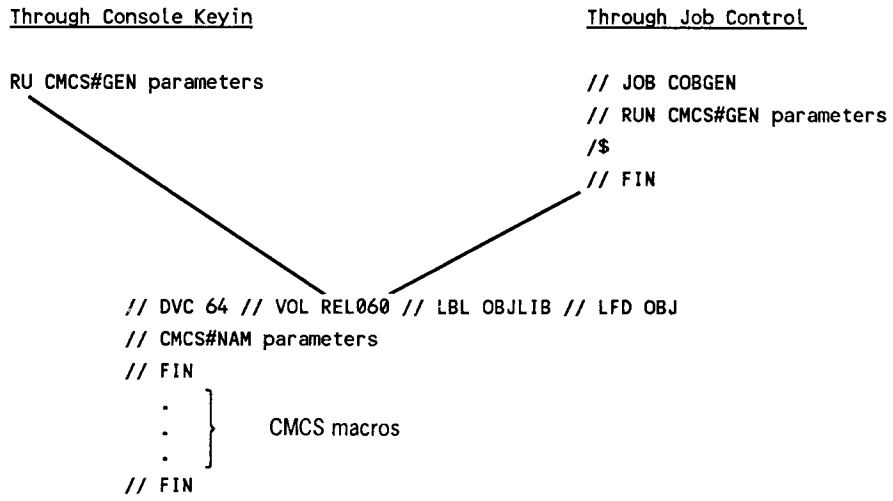
- A jproc call, CMCS#NAM

  Through parameters associated with this call, you specify the name to be assigned to the generated CMCS module, the library in which the file is to reside, and the name of the ICAM network with which it is to interface.

- The CMCS macros

  These macros and their associated parameters become, in effect, data interpreted by the CMCS#GEN program to build the appropriate module.

The CMCS generation program can be by the RU console command or through job control using a RUN job control statement.



See 6.5.3. for details for executing the CMCS generation program.

6.5.1. User-Supplied CMCS Generation Data (CMCS#NAM)

You must supply the following input data for CMCS generation:

- A name to be assigned to the CMCS object module
- The name of the related ICAM network
- A library file name for saving the CMCS object module
- A job control device assignment set for the user library
- The CMCS generation macros

The object module, network, and user library names are supplied on a single statement as keyword parameters associated with the CMCS#NAM jproc call. The jproc is part of the filed job control stream (CMCS#GEN) and places the supplied names in the appropriate job control directives. The format of the jproc call is:

```
// CMCS#NAM MOD=module-name,NET=network-name,FILE= { filename }
                                                 { $Y$OBJ   }
```

where:

CMCS#NAM
: Is the jproc name.

MOD=module-name
Is the 1- to 8-character name identifying the CMCS object module.

NET=network-name
Is the 1- to 4-character label of the CCA macro in the network definition.

FILE=filename
Is the 1- to 8-character name identifying the user library or the system object library (YOBJ).

Example

```
1. // DVC 64 // VOL REL060 // LBL MYOBJ // LFD OBJ
2. // CMCS#NAM MOD=CM5050CM,NET=CNET,FILE=OBJ
3. // FIN
```

Explanation:

1. Device assignment set for the library to contain the CMCS object module.
2. CMCS jproc denoting the module name, the network name, and the output file name where the CMCS object module is to reside.
3. Terminates card reader operation.

Notes:

1. *Omitting the network-name or library-filename or an error detected by the validation program causes the CMCS#GEN job to terminate. An appropriate error message is displayed on the system console and the output listing.*
2. *The job control device assignment set is required only if you are using your own library. The LFD name must be the same as the file name specified on the FILE parameter.*
3. *The CMCS#NAM jproc call and the JCL device assignment set make up the data deck terminated by a FIN statement. This data deck must precede the CMCS macro cards that make up the second deck.*

6.5.2. CMCS Macros

The macros described in this subsection generate various sections of the CMCS object module. The macros make up a data deck that is read by the CMCS#GEN job to create a temporary source module. The data deck is terminated by a FIN statement.

Initiating CMCS Module Generation (DB#GEN)

DB#GEN initiates the CMCS data base module generation. This macro supplies the network name and type, any passwords with an optional listing, the number of output communications descriptors (CDs), and, if a global network is used, the application name assigned to the COBOL communications program.

Format

LABEL	ΔOPERATIONΔ	OPERAND
not used	DB#GEN	NNAME=network-name [,NTYPE= { GBL / DED }] [,NPASS=network-password] ,CPASS=COBOL-user-password [,OUTCD= { 1 / n }] [,PWSUP= { YES / NO }] [,ANAME=apps-name]

Operands

NNAME=network-name
: Identifies the configured ICAM network. This name must agree with the network name in the label field on the CCA network generation macro. If omitted, an error PNOTE is displayed and the job is terminated.

NTYPE= { GBL / DED }
: Indicates whether the network is dedicated or global.

NPASS=network-password
: Identifies the network password. (Same as PASSWORD parameter in the CCA macro.) If no password is given at network generation, omit this parameter.

CPASS=COBOL-user-password
: Identifies the password specified in the COBOL program. If omitted, a PNOTE error message is printed and the generation procedure terminates.

OUTCD= { 1 / n }
: Indicates the number (in decimal) of output communications descriptors (CD) defined in the COBOL program.

PWSUP= { YES / NO }

Indicates whether to suppress the printing of all passwords in the assembly listing except for the image of the DB#GEN macro. You suppress the printing of the DB#GEN card by preceding it with a PRINT OFF assembly directive.

ANAME=apps-name

Indicates the application name associated with the COBOL program. This name is specified in the symbol field in the LOCAP macro. This parameter is required in a global network environment. If the parameter is omitted but required, the job is terminated and a diagnostic message is displayed.

Example

```
1        10      16                                                  72
---------------------------------------------------------------------
         DB#GEN NNAME=CNET,NTYPE=DED,CPASS=CM101PASS

         DB#GEN NNAME=NET1,CPASS=OKCOB1,ANAME=PROG
```

Defining Queue Structures (DB#SQT)

DB#SQT defines an entry in the queue structure and relates the COBOL and ICAM queue names. Specify one DB#SQT macro for each process file. The structure of a COBOL symbolic queue is determined by the DB#SQT. The order of the DB#SQT macros plus the symbolic subqueue names assigned to each input queue determine the access sequence and groupings within subqueue levels. Multiple queue structures are permitted but an input queue may be only used once in all queue structures specified.

Format

LABEL	ΔOPERATIONΔ	OPERAND
not used	DB#SQT	PFILE=process-filename [,SYMQ=symbolic-queue-name] [,SUBQ1=sub-queue-1-name] [,SUBQ2=sub-queue-2-name] [,SUBQ3=sub-queue-3-name]

Parameters

PFILE=process-filename

Is the name specified in the symbol field of a PRCS macro in the ICAM network definition. If omitted, an error PNOTE is displayed and no table entry is generated.

SYMQ=symbolic-queue-name
: Is a 1- to 12-character symbolic name that can be placed in the CD queue name field identifying a queue structure.

SUBQ1=sub-queue-1-name
: Is a 1- to 12-character symbolic name that can be placed in the symbolic sub-queue-1 field of a CD indicating the first subqueue level.

SUBQ2=sub-queue-2-name
: Is a 1- to 12-character symbolic name that can be placed in the symbolic sub-queue-2 field of a CD indicating the second subqueue level.

SUBQ3=sub-queue-3-name
: Is a 1- to 12-character symbolic name that can be placed in the symbolic sub-queue-3 field of a CD indicating the third subqueue level.

Examples

Example 1. Basic Queue Structure:

```
1        10      16
-----------------------------------------------------------------------
         DB#SQT PFILE=IQ7,SYMQ=STATE,SUBQ1=COUNTY3
         DB#SQT PFILE=IQ8
```

Example 2. Symmetrical Hierarchical Queue Structure:

```
DB#SQT PFILE=IQ1,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY1,SUBQ3=DISTRICT1
DB#SQT PFILE=IQ2,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY1,SUBQ3=DISTRICT2
DB#SQT PFILE=IQ3,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY2,SUBQ3=DISTRICT3
DB#SQT PFILE=IQ4,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY2,SUBQ3=DISTRICT4
DB#SQT PFILE=IQ5,SYMBQ=STATE,SUBQ1=COUNTY2,SUBQ2=CITY3,SUBQ3=DISTRICT5
DB#SQT PFILE=IQ6,SYMBQ=STATE,SUBQ1=COUNTY2,SUBQ2=CITY3,SUBQ3=DISTRICT6
DB#SQT PFILE=IQ7,SYMBQ=STATE,SUBQ1=COUNTY2,SUBQ2=CITY4,SUBQ3=DISTRICT7
DB#SQT PFILE=IQ8,SYMBQ=STATE,SUBQ1=COUNTY2,SUBQ2=CITY4,SUBQ3=DISTRICT8
```

Example 3. Asymmetrical Hierarchical Structure:

```
DB#SQT PFILE=IQ1,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY1,SUBQ3=DISTRICT1
DB#SQT PFILE=IQ2,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY1,SUBQ3=DISTRICT2
DB#SQT PFILE=IQ3,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY2,SUBQ3=DISTRICT3
DB#SQT PFILE=IQ4,SYMBQ=STATE,SUBQ1=COUNTY1,SUBQ2=CITY2,SUBQ3=DISTRICT4
DB#SQT PFILE=IQ5,SYMBQ=STATE,SUBQ1=COUNTY2,SUBQ2=CITY3,SUBQ3=DISTRICT5
DB#SQT PFILE=IQ6,SYMBQ=STATE,SUBQ1=COUNTY2,SUBQ2=CITY3,SUBQ3=DISTRICT6
DB#SQT PFILE=IQ7,SYMBQ=STATE,SUBQ1=COUNTY2,SUBQ2=CITY4
DB#SQT PFILE=IQ8,SYMBQ=STATE,SUBQ1=COUNTY3
```

Defining Symbolic Names (DB#SNT)

DB#SNT establishes the relationship between the COBOL symbolic source and destination names and the ICAM terminal and process file names.

Format

LABEL	ΔOPERATIONΔ	OPERAND
not used	DB#SNT	SYMN=symbolic-name [, { PFILE=process-filename TERM=terminal-name }] [,INIT= { YES NO }]

Operands

SYMN=symbolic-name
: Is the 1- to 12-character COBOL symbolic name identifying either the source or destination terminal. This name is supplied in *either* the SYMBOLIC SOURCE or DESTINATION field in the COBOL CD.

PFILE=process-filename
: Is the 1- to 4-character name assigned in the symbol field of a PRCS macro in the network definition. (A process file can be both the source and destination for the message.)

TERM=terminal-name
: Is the 1- to 4-character name assigned in the symbol field of a TERM macro in the network definition.

If both PFILE and TERM parameters are omitted, a PNOTE is displayed and no table entry is generated.

INIT= { YES / NO }
: If YES is specified, CMCS automatically activates the source or destination terminal or the process file. If NO is specified or the parameter is omitted, the COBOL ENABLE statement is required to activate either the terminal or process file.

Example

```
1       10      16
-------------------------------------------------------------
        DB#SNT SYMN=CM101TERM,TERM=TRM1
        DB#SNT SYMN=COBTERM1,PFILE=PRF1,INIT=YES
```

Generating Input Routing Tables (DB#IRT)

DB#IRT generates an input routing table that defines the routing of messages from the terminals to a specific process file. The input routing table defines routing used with either the MPPS or direct terminal to line message processing.

Format

LABEL	ΔOPERATIONΔ	OPERAND
not used	DB#IRT	PFILE=process-filename ,TERM=(terminal-name-1, terminal-name-2,...,terminal-name-n)

Operands

PFILE=process-filename
: Is the 1- to 4-character name assigned in the symbol field of a PRCS macro in the network definition.

TERM=(terminal-name-1,terminal-name-2,...terminal-name-n)
: Identifies the terminal that may have input messages routed to the process file via the MPPS. If the MPPS is not used, all terminal names specified on the TERM macros with each line must be used.

Example

```
1       10      16
-------------------------------------------------------------
        DB#IRT PFILE=PRF1,TERM=(TRM1)
        DB#IRT PFILE=PRF2,TERM=(TRM1,TRM2)
```

Note: The DB#IRT macro must be specified after the DB#SNT macro.

Ending CMCS Module Generation (DB#END)

DB#END designates the end of the CMCS module generation and causes the executable code to be included in the CMCS module. This must be the last macro in the CMCS control stream.

Format

LABEL	ΔOPERATIONΔ	OPERAND
not used	DB#END	not used

6.5.3. Executing the CMCS Generation Job (CMCS#GEN)

You initiate the CMCS#GEN job through the following RUN job control statement:

```
// RUN CMCS#GEN,,PAR= { YES }
                      { NO  }
```

or by typing in the RUN command on the system console:

```
RU CMCS#GEN,,PAR= { YES }
                  { NO  }
```

where:

PAR=YES
: Indicates a card deck containing the CMCS#NAM jproc call, the device assignment set, and the macro deck is read from the card reader.

PAR=NO
: Indicates only the card deck containing the macros is read from the card reader. When selected, the CMCS#GEN job is terminated after the second job step.

If initiated by a console command, the CMCS generation control stream must be in the card reader when the keyin is made. If initiated through job control, the CMCS control stream follows the FIN statement of the RUN control stream, as in the following example. The device assignment set in the example identifies a user library where the CMCS object module is to be placed. This device assignment set is not required if you use the system object library YOBJ.

Example

```
1        10      16                                                    72
-------------------------------------------------------------------------
// JOB COBOLGEN,,C00,,,,,F012
// RUN CMCS#GEN,,PAR=YES
/&
// FIN
// DVC 64 // VOL REL060 // LBL OBJLIB // LFD OBJ
// CMCS#NAM MOD=CMS010CM,NET=CNET,FILE=OBJ
// FIN
         DB#GEN NNAME=CNET,NTYPE=DED,CPASS=CM501PASS
         DB#SQT PFILE=PRF1,SYMQ=CM501QUEUE
         DB#SNT SYMN=CM501TERM1,TERM=TRM1
         DB#SNT SYMN=CM501TERM2,TERM=TRM2
         DB#SNT SYMN=CM501TERM3,TERM=TRM3
         DB#IRT PFILE=PRF1,TERM=(TRM1,TRM2,TRM3)
         DB#END
// FIN
```

Note: The example shows the cards used for initiating CMCS#GEN using the RUN job control. When initiating from the system console, the first four cards (/ /JOB through / /FIN) are not required.

6.6. COBOL Communications Example Application

This procedure describes and illustrates how to develop a COBOL communications capability for an ICAM network. The steps in the procedure are presented in the order in which they must be performed (see 6.6.1 through 6.6.5):

1. Defining and generating the ICAM network
2. Generating a CMCS module
3. Compiling and linking a COBOL source program
4. Loading and executing the ICAM symbiont and GUST
5. Executing the COBOL load module

We present the procedure principally by example. We describe, in the text, only the communications features relating to COBOL and not the basic programs and job streams that are explained in other OS/3 publications.

In this example, we send a message from a terminal to the processor that ICAM places in a process file queue. The COBOL program, using the RECEIVE verb, takes the message from the process file and, using the SEND verb, places the message in the terminal output queue. ICAM then transmits the message back to the originating terminal.

In our example, we load and execute the COBOL program via console procedures. However, we have included the required parameters in the ICAM network definition (6.6.1) and in the COBOL source program (INITIAL in CD input) so that the COBOL program can be loaded and executed automatically upon receipt, at the processor, of the first UNISCOPE message after ICAM starts polling.

6.6.1. Defining and Generating the ICAM Network

The network definition for our example (Figure 6-3) is global and is named GNET. We define the network before defining CMCS because data from the network definition is needed for the CMCS module. Also, the ICAM network must be generated before the CMCS module because the CMCS module generation procedure requires access to the assembled network definition for validation. (SAVE=YES in the CCA macro causes the assembled module to be saved for later access during CMCS module generation.)

```
COMMCT
GNET     CCA    TYPE=(GBL,,S),GAWAKE=YES,SAVE=YES,                    X
                FEATURES=(OPCOM,SEGMENTS,DATIME),CCAID=GNETGEN

         BUFFERS  100,64,15,ARP=45

COBL     LOCAP  TYPE=(STD*CP),JOBINIT=(LOAD,REPORT),JOBNAME=GBLCMCS

LNE1     LINE   DEVICE=(UNISCOPE),TYPE=(9600,SYNC),LBL=256,ID=08

TRM1     TERM   ADDR=(28,51),FEATURES=(U400,1920),DICE=ON,           X
                INPUT=(PRC1),LOW=MAIN

TRM2     TERM   ADDR=(28,52),FEATURES=(U400,1920),DICE=ON,           X
                INPUT=(PRC2),LOW=MAIN

TRM3     TERM   ADDR=(28,53),FEATURES=(U400,1920),DICE=ON,           X
                INPUT=(PRC3),LOW=MAIN

PRC1     PRCS   LOW=MAIN

PRC2     PRCS   LOW=MAIN

PRC3     PRCS   LOW=MAIN
```

Figure 6-3. ICAM Generation for COBOL Communications (Part 1 of 2)

```
LIST     DLIST TRM1,TRM2

         SESSION PRIMARY,EU1=(COBL),EU2=(PRC1)

         SESSION PRIMARY,EU1=(COBL),EU2=(PRC2)

         SESSION PRIMARY,EU1=(COBL),EU2=(PRC3)

         SESSION EU1=(COBL),EU2=(TRM1)

         SESSION EU1=(COBL),EU2=(TRM2)

         SESSION EU1=(COBL),EU2=(TRM3)

         ENDCCA

         MCP
                MCPNAME=M1
                CACH=(08,9600,SYNC)
END
```

Figure 6-3. ICAM Generation for COBOL Communications (Part 2 of 2)

We use the SESSION macro in our network definition to specify the relationships among the end users - the three terminals (TRM1, TRM2, and TRM3), three process files (PRC1, PRC2, and PRC3), and the locap file (COBL). At run time, the CMCS module automatically executes the NATTACH macro to attach the COBOL load module to the COBL locap file and thus gain access to the process files and terminals. The DLIST macro is included because the CMCS module constructs and uses user-defined DLISTS at run time.

There are two ways you can run a COBOL program. You can load and execute it manually via job control, or have it done automatically when the first message is transmitted from the terminal to the ICAM process file. Although we load and execute the COBOL program manually (6.6.5), we included the following operands that support automatic loading and execution:

```
CCA      GAWAKE=YES
LOCAP    JOBINIT=(LOAD,REPORT),JOBNAME=GBLCMCS
SESSION PRIMARY
```

Note: For the instructions on how to generate an ICAM symbiont, see the ICAM Operations Guide *(UP-9745).*

6.6.2. Generating a CMCS Module

The job stream shown in Figure 6-4 assembles, validates, and saves the CMCS run-time routine used later when linking COBOL load module GALEN2. The PAR=YES option in the RUN control statement indicates that the CMCS generation job stream is made up of two parameter sets. The first set is the CMCS#NAM jproc call and the device assignment set; the second set contains the module generation macros.

```
// JOB CMCSGEN,,C000,,,,,F114
// RUN CMCS#GEN,,PAR=YES
/&
// FIN
// DVC RES  // LBL SYSOBJ  // LFD OBJ
// CMCS#NAM MOD=GAL2CMCS,NET=GNET,FILE=OBJ
// FIN
        DB#GEN NNAME=GNET,NTYPE=GBL,CPASS=07,OUTCD=1,PWSUP=NO,          X
               ANAME=COBL
        DB#SQT PFILE=PRC1,SYMQ=LNE1,SUBQ 1=TRM1
        DB#SQT PFILE=PRC2,SYMQ=LNE1,SUBQ 1=TRM2
        DB#SQT PFILE=PRC3,SYMQ=LNE1,SUBQ 1=TRM3
        DB#SNT SYMN=TRM1,INIT=YES,TERM=TRM1
        DB#SNT SYMN=TRM2,INIT=YES,TERM=TRM2
        DB#SNT SYMN=TRM3,INIT=YES,TERM=TRM3
        DB#IRT PFILE=PRC1,TERM=TRM1
        DB#IRT PFILE=PRC2,TERM=TRM2
        DB#IRT PFILE=PRC3,TERM=TRM3
        DB#END
// FIN
```

Figure 6-4. CMCS Module Generation Job Stream

The following statements make up the control stream that generates the CMCS module:

1. CMCS#NAM Jproc

   Specifies the CMCS module name (GAL2CMCS), the network name (GNET, from the network definition), and the output file where the CMCS module (OBJ) will be placed after it is assembled and validated.

2. DB#GEN Macro

   Specifies the ICAM network name (GNET), the network type (GBL), the COBOL password (007), the number of output communications descriptors (1, as defined in the COBOL program), and the locap name (COBL, from the network definition). Because we specify PWSUP=NO, we don't suppress the printing of passwords in the assembly listing.

3. DB#SQT Macro

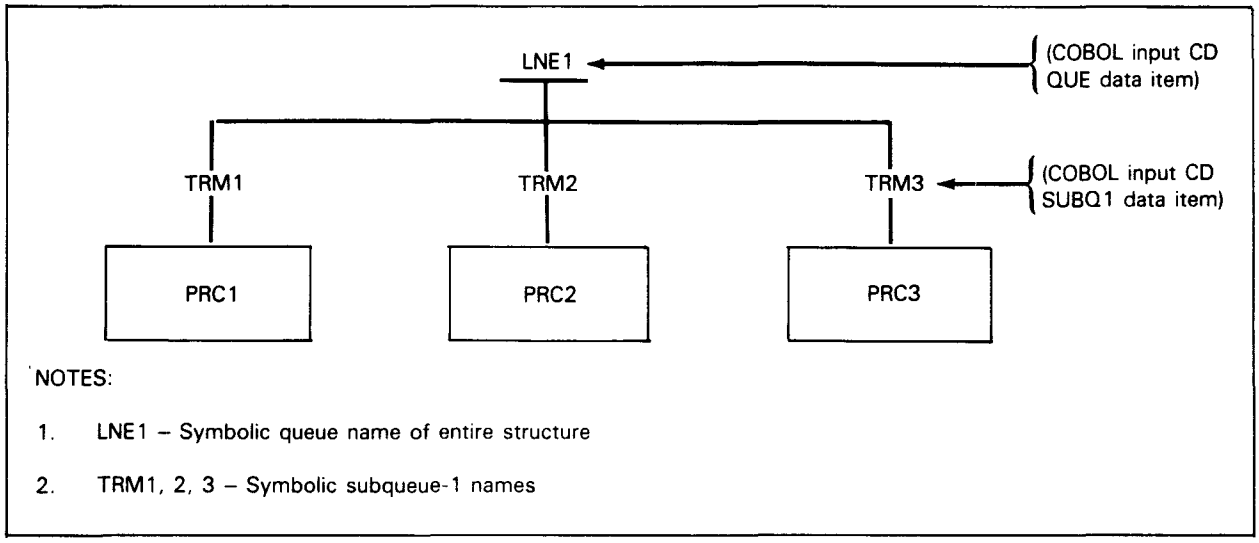
   Defines the COBOL input queue (tree) structure. As shown in Figure 6-5a, the tree structure relates, symbolically, the line, terminals, and process files whose physical relationship is shown in Figure 6-5b. Thus, the tree structure shows that line LNE1 services the three terminals: TRM1, TRM2, and TRM3. The tree structure also shows that messages from TRM1 are queued to process file PRC1, messages from TRM2 are queued to PRC2, and messages from TRM3 are queued to PRC3.

   The names LNE1, TRM1, TRM2, and TRM3 in the COBOL queue structure and in this macro are symbolic names. They need not match those used in the network definition. However, in the COBOL source program (Figure 6-6), we use the ICAM facility names as the symbolic names. Then the COBOL program names and CMCS generation names for ICAM facilities can be directly related to the ICAM facility names used in the network definition.

   Note: *The COBOL source program (Figure 6-6) is a UNISCOPE terminal test program. The program name is ECHO, and the load module name is GALEN2 (Figure 6-7). In the communications section, both INPUT-CD-1 and OUTPUT-CD-1 transmit parameters from the program to CMCS and return status from CMCS to the program. Thus, when the program prepares to execute a RECEIVE statement, it places appropriate parameters in INPUT-CD-1 and, when the program prepares to execute a SEND statement, it places appropriate parameters in OUTPUT-CD-1. When the program regains control after RECEIVE/SEND statement execution, the program interrogates the CD for the status that CMCS supplied.*

   After the symbolic names are defined for the tree structure in Figure 6-5, they are used in the parameters for the DB#SQT macros.

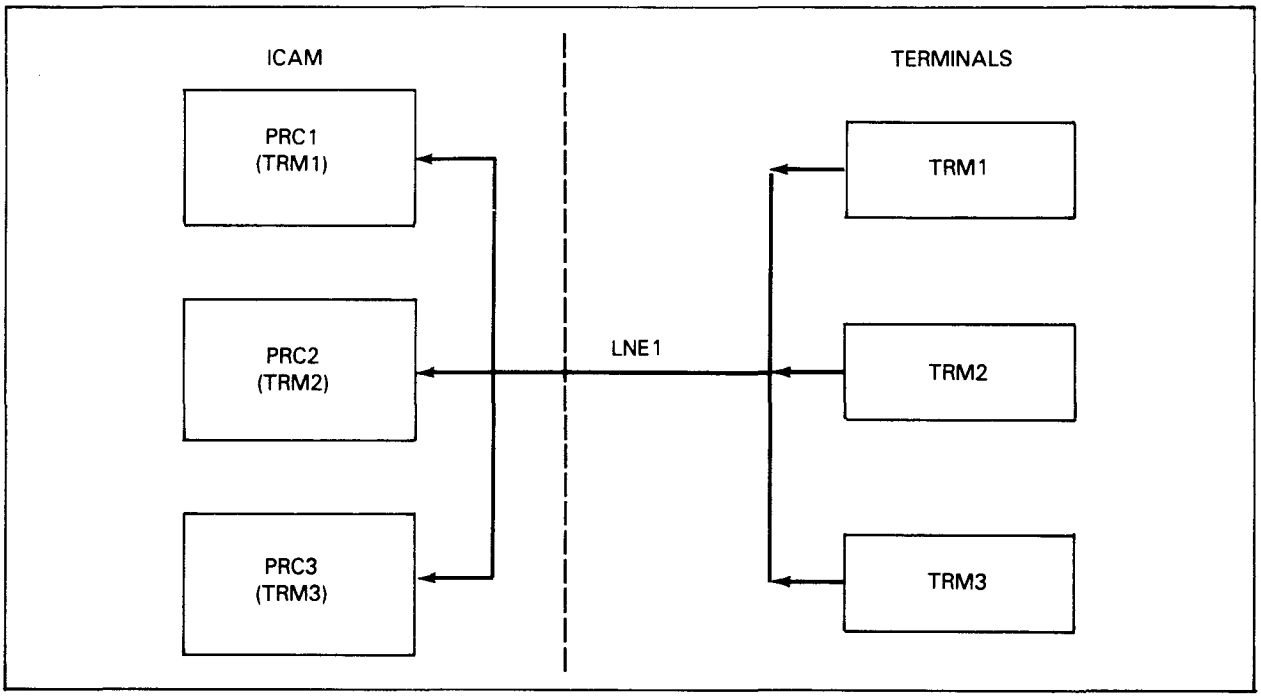
   When the COBOL program places character string LNE1 in INPUT-CD-1 data item QUE, places TRM1 in data item SUBQ1, and clears the next two fields of INPUT-CD-1 to spaces, execution of a RECEIVE statement results in retrieval of a message from process file PRC1. If TRM3 is substituted for TRM1, a message is retrieved from PRC3. If the COBOL program places LNE1 in QUE and clears the next three fields of INPUT-CD-1 to spaces, execution of a RECEIVE statement causes a message to be retrieved from PRC1, PRC2, or PRC3, in that order.



NOTES:

1. LNE1 – Symbolic queue name of entire structure
2. TRM1, 2, 3 – Symbolic subqueue-1 names

a. COBOL symbolic input queue structure



b. ICAM physical input structure

Figure 6-5. Terminal, Line, and Process File Relationship for Input

```
      COMPILED BY   UNIVAC OS/3S COBOL (ANSI-74) COMPILER   VERSION:

PROGRAM-ID: ECHO                                  SOURCE PROGRAM LISTING

  LINE NO.      SOURCE ENTRY

    00001        ************************************************************
    00002        ************************************************************
    00003         IDENTIFICATION DIVISION.
    00004        ************************************************************
    00005        ************************************************************
    00006        *
    00007         PROGRAM-ID. ECHO.
    00008        *
    00009        * COBOL COMMUNICATION FACILITY TEST PROGRAM.
    00010        * (THIS IS PART OF A LARGER UNISCOPE TEST PROGRAM.)
    00011        * FOR THIS TEST, THE CMCS GENERATION IS REQUIRED TO
    00012        * BRING UP THE OUTPUT QUEUES ENABLED, AND TO BRING
    00013        * UP THE TERMINAL INPUT LEGS ENABLED.
    00014        * THIS TEST PROGRAM USES TECHNIQUES THAT ARE NOT
    00015        * SUITABLE FOR A PRODUCTION ENVIRONMENT.
    00016        * EXAMPLE: THE USE OF DISPLAY AND ACCEPT WILL DELAY
    00017        * PROGRAM EXECUTION, AND THEREFORE DELAY TERMINAL
    00018        * MESSAGE SERVICING BY THIS PROGRAM. THIS WOULD BE
    00019        * UNACCEPTABLE IN A PRODUCTION ENVIRONMENT.
    00020        * EXAMPLE: THIS TEST PROGRAM MAKES NO ATTEMPT
    00021        * TO ANALYZE AND COPE WITH STATUS CONDITIONS RETURNED BY
    00022        * CMCS.
    00023        * EXAMPLE: SOME DATA ITEM CLEARING IS DONE FOR THE
    00024        * SAKE OF CLARITY, NOT OUT OF NECESSITY.
    00025        *
    00026         DATE-COMPILED. 81/12/08.
    00027        *
    00028        ************************************************************
    00029        ************************************************************
    00030         ENVIRONMENT DIVISION.
    00031        ************************************************************
    00032        ************************************************************
    00033        *
    00034         CONFIGURATION SECTION.
    00035        *
    00036         SOURCE-COMPUTER. UNIVAC-OS3.
    00037         OBJECT-COMPUTER. UNIVAC-OS3.
    00038        *
    00039         SPECIAL-NAMES.
    00040             SYSCONSOLE IS CONSOLE-OPERATOR,
    00041             SYSLOG IS CONSOLE.
    00042        *
    00043        ************************************************************
    00044        ************************************************************
    00045         DATA DIVISION.
    00046        ************************************************************
    00047        ************************************************************
    00048         WORKING-STORAGE SECTION.
    00049        ************************************************************
    00050        *
```

Figure 6-6. COBOL Source Program (Part 1 of 5)

```
PROGRAM-ID: ECHO                                    SOURCE PROGRAM LISTING

  LINE NO.      SOURCE ENTRY

    00051        * WE DEFINE THE ICAM NAME OF THE LINE ON WHICH
    00052        * OUR FOUR UNISCOPES ARE FOUND. REFER TO THE CMCS
    00053        * GENERATION - WE USE THE LINE NAME AS THE QUEUE
    00054        * NAME IN THE FIRST FIELD IN THE INPUT CD.
    00055         77 LINE-NAME PICTURE X(4) VALUE "LNE1".
    00056        *
    00057         77 END-KEY PICTURE X.
    00058        *
    00059        * EXAMPLE OF APPEARANCE OF THE FOLLOWING MESSAGE:
    00060        * "PARAGRAPH PARA1 SEND VERB 02 HAS STATUS KEY VALUE 10."
    00061         01 BAD-OUTPUT-STATUS-MSG.
    00062              02 NOTUSED1 PICTURE X(10) VALUE "PARAGRAPH ".
    00063              02 BAD-OUTPUT-PARANAME PICTURE X(7).
    00064              02 NOTUSED2 PICTURE X(11) VALUE " SEND VERB ".
    00065              02 BAD-OUTPUT-VERBNUM PICTURE X(2).
    00066              02 NOTUSED3 PICTURE X(22) VALUE " HAS STATUS KEY VALUE ".
    00067              02 OUTPUT-STATUS-KEY PICTURE X(2).
    00068              02 NOTUSED4 PICTURE X VALUE ".".
    00069        *
    00070        * EXAMPLE OF APPEARANCE OF THE FOLLOWING MESSAGE:
    00071        * "USED ERROR KEY VALUES ARE 0100."
    00072         01 OUTPUT-ERROR-KEYS-MSG.
    00073              02 NOTUSED5 PICTURE X(26)
    00074              VALUE "USED ERROR KEY VALUES ARE ".
    00075              02 OUT-ERROR-KEYS PICTURE X(4).
    00076              02 NOTUSED6 PICTURE X VALUE ".".
    00077        *
    00078        * EXAMPLE OF APPEARANCE OF THE FOLLOWING MESSAGE:
    00079        * "PARAGRAPH PARA1 REC VERB 02 HAS STATUS KEY VALUE 02.".
    00080         01 BAD-INPUT-STATUS-MSG.
    00081              02 NOTUSED7 PICTURE X(10) VALUE "PARAGRAPH ".
    00082              02 BAD-INPUT-PARANAME PICTURE X(7).
    00083              02 NOTUSED8 PICTURE X(10) VALUE " REC VERB ".
    00084              02 BAD-INPUT-VERBNUM PICTURE X(2).
    00085              02 NOTUSED9 PICTURE X(22) VALUE " HAS STATUS KEY VALUE ".
    00086              02 INPUT-STATUS-KEY PICTURE X(2).
    00087              02 NOTUSED10 PICTURE X VALUE ".".
    00088        *
    00089        * DEFINE START OF ENTRY CHARACTER, AND THE WORD STOP.
    00090         01 SOE-STOP.
    00091              02 SOE-CHARACTER PICTURE X VALUE ="1E".
    00092              02 STOPP PICTURE X(4) VALUE "STOP".
    00093        *
    00094         01 BIG-RECEIVE-FIELD-SEND-FIELD.
    00095        * THIS LARGE RECEIVING FIELD SIMPLIFIES OUR PROGRAM LOGIC,
    00096        * BECAUSE IT GUARANTEES THAT ONE RECEIVE STATEMENT CAN
    00097        * OBTAIN THE ENTIRE MESSAGE. AFTER THE MESSAGE HAS BEEN
    00098        * RECEIVED, THIS RECEIVING FIELD BECOMES THE SENDING FIELD,
    00099        * AND THE MESSAGE IS ECHOED BACK TO THE TERMINAL OPERATOR.
    00100              02 FIRST-TEN-CHARACTERS PICTURE X(10).
    00101              02 REDEFINITION-1 REDEFINES FIRST-TEN-CHARACTERS.
    00102                   03 FIRST-FOUR-CHARACTERS PICTURE X(4).
```

Figure 6-6. COBOL Source Program (Part 2 of 5)

```
PROGRAM-ID: ECHO                                    SOURCE PROGRAM LISTING

  LINE NO.      SOURCE ENTRY

    00103                03 NEXT-SIX-CHARACTERS PICTURE X(6).
    00104           02 REDEFINITION-2 REDEFINES FIRST-TEN-CHARACTERS.
    00105                03 FIRST-FIVE-CHARACTERS PICTURE X(5).
    00106                03 NEXT-FIVE-CHARACTERS PICTURE X(5).
    00107           02 NOTUSED31 PICTURE X(490).
    00108           02 NOTUSED32 PICTURE X(500).
    00109           02 NOTUSED33 PICTURE X(500).
    00110           02 NOTUSED34 PICTURE X(500).
    00111          *
    00112          ******************************************************
    00113           COMMUNICATION SECTION.
    00114          ******************************************************
    00115          *
    00116           CD IN-CD-1 FOR INITIAL INPUT.
    00117          *
    00118           01 INPUT-CD-1.
    00119                02 QUE PICTURE X(12).
    00120                02 SUBQ1 PICTURE X(12).
    00121                02 SUBQ2 PICTURE X(12).
    00122                02 SUBQ3 PICTURE X(12).
    00123                02 MSGDATE PICTURE 9(6).
    00124                02 MSGTIME PICTURE 9(8).
    00125                02 SYMSOURCE PICTURE X(12).
    00126                02 TEXTLENGTH PICTURE 9(4).
    00127                02 ENDKEY PICTURE X.
    00128                02 INSTATUSKEY PICTURE XX.
    00129                02 MSGCOUNT PICTURE 9(6).
    00130          *
    00131           CD OUT-CD-1; FOR OUTPUT.
    00132          *
    00133           01 OUTPUT-CD-1.
    00134                02 DESTINATIONCOUNT PICTURE 9(4).
    00135                02 OUTLENGTH PICTURE 9(4).
    00136                02 OUTSTATUSKEY PICTURE XX.
    00137                02 DESTINATIONTABLE OCCURS 4 TIMES.
    00138                     03 ERRORKEY PICTURE X.
    00139                     03 SYMBOLICDEST PICTURE X(12).
    00140          *
    00141          ******************************************************
    00142          ******************************************************
    00143           PROCEDURE DIVISION.
    00144          ******************************************************
    00145          ******************************************************
    00146          *
    00147           STARTUP1.
    00148                DISPLAY "BEGIN COBOL COMMO TEST PROGRAM." UPON CONSOLE.
    00149          *
    00150                STOP "ANSWER ICAM DIAL MSGS IF ANY. THEN ENTER GO.".
    00151          *
    00152           PARA14.
    00153          * BEGIN ECHO TEST. FROM ANY ONLINE TERMINAL THE TESTER WILL
    00154          * INPUT A MESSAGE. IF THE FIRST FOUR CHARACTERS OF THE MESSAGE
```

Figure 6-6. COBOL Source Program (Part 3 of 5)

```
PROGRAM-ID: ECHO                                SOURCE PROGRAM LISTING

 LINE NO.      SOURCE ENTRY

   00155        * ARE NOT STOP, OR IF THE FIRST FIVE CHARACTERS OF THE MESSAGE
   00156        * ARE NOT SOE STOP, THEN WE WILL RETURN THE MESSAGE TO THE
   00157        * SAME TERMINAL FROM WHICH IT ORIGINATED, AND AWAIT ANOTHER
   00158        * SUCH MESSAGE.
   00159        *
   00160             PERFORM INPUT-CD-1-SETUP-1.
   00161             MOVE LINE-NAME TO QUE.
   00162             MOVE SPACES TO BIG-RECEIVE-FIELD-SEND-FIELD.
   00163        *
   00164        * WE EXECUTE THE RECEIVE AND HANG ON IT INDEFINITELY,
   00165        * UNTIL A MESSAGE IS RECEIVED.
   00166             RECEIVE IN-CD-1 MESSAGE INTO BIG-RECEIVE-FIELD-SEND-FIELD.
   00167             IF INSTATUSKEY EQUAL ZEROS
   00168                 AND ENDKEY EQUAL 2
   00169                 GO TO PARA15;
   00170                 ELSE NEXT SENTENCE.
   00171        *
   00172        * ERROR PATH. WE GOT BAD STATUS, OR AN UNEXPECTED END KEY VALUE.
   00173        * WE NOW REPORT THE ERROR TO THE CONSOLE.
   00174             MOVE "PARA14" TO BAD-INPUT-PARANAME.
   00175             MOVE "01" TO BAD-INPUT-VERBNUM.
   00176             MOVE INSTATUSKEY TO INPUT-STATUS-KEY.
   00177             MOVE ENDKEY TO END-KEY.
   00178             PERFORM BAD-RECEIVE-NOTIFY-2.
   00179        *
   00180         PARA15.
   00181        * IF THE TERMINAL OPERATOR KEYED IN THE STOP MESSAGE, WE GO
   00182        * TO END OF PROGRAM. OTHERWISE WE RETURN THE MESSAGE
   00183        * TO HIM AT THE SAME TERMINAL.
   00184             IF FIRST-FIVE-CHARACTERS EQUAL SOE-STOP

   00185                 OR  FIRST-FOUR-CHARACTERS EQUAL STOPP
   00186                 GO TO END-OF-PROGRAM;
   00187                 ELSE NEXT SENTENCE.
   00188        *
   00189         PARA16.
   00190        * WE ECHO (RETURN) THE MESSAGE TO THE TERMINAL FROM WHENCE IT
   00191        * CAME, PRESENTING IT ON THE NEXT SUCCEEDING LINE.
   00192        *
   00193        * SET UP THE OUTPUT CD.
   00194             MOVE 1 TO DESTINATIONCOUNT.
   00195             MOVE TEXTLENGTH OF INPUT-CD-1 TO OUTLENGTH OF OUTPUT-CD-1.
   00196             MOVE ZEROS TO OUTSTATUSKEY.
   00197             MOVE ZERO TO ERRORKEY (1).
   00198             MOVE SYMSOURCE TO SYMBOLICDEST (1).
   00199        * ECHO THE MESSAGE BACK TO THE TERMINAL.
   00200             SEND OUT-CD-1 FROM BIG-RECEIVE-FIELD-SEND-FIELD WITH EMI
   00201                 AFTER ADVANCING 1 LINES.
   00202             IF OUTSTATUSKEY EQUAL ZEROS GO TO PARA14;
   00203                 ELSE NEXT SENTENCE.
   00204        * BAD STATUS PATH.
   00205             MOVE "PARA16" TO BAD-OUTPUT-PARANAME.
```

Figure 6-6. COBOL Source Program (Part 4 of 5)

```
PROGRAM-ID: ECHO                                    SOURCE PROGRAM LISTING

  LINE NO.      SOURCE ENTRY

    00206             MOVE "01" TO BAD-OUTPUT-VERBNUM.
    00207             MOVE OUTSTATUSKEY TO OUTPUT-STATUS-KEY.
    00208             MOVE SPACES TO OUT-ERROR-KEYS.
    00209             MOVE ERRORKEY (1) TO OUT-ERROR-KEYS.
    00210             PERFORM BAD-SEND-NOTIFICATION-2.
    00211         *
    00212          END-OF-PROGRAM.
    00213             STOP "END OF PROGRAM. TAKE A DUMP OR ENTER GO.".
    00214             STOP RUN.
    00215         *
    00216          INPUT-CD-1-SETUP-1.
    00217         * THIS PARAGRAPH IS PERFORMED.
    00218             MOVE SPACES TO QUE.
    00219             MOVE SPACES TO SUBQ1.
    00220             MOVE SPACES TO SUBQ2.
    00221             MOVE SPACES TO SUBQ3.
    00222             MOVE ZEROS TO MSGDATE.
    00223             MOVE ZEROS TO MSGTIME.
    00224             MOVE SPACES TO SYMSOURCE.
    00225             MOVE ZEROS TO TEXTLENGTH.
    00226             MOVE ZERO TO ENDKEY.
    00227             MOVE ZEROS TO INSTATUSKEY.
    00228             MOVE ZEROS TO MSGCOUNT.
    00229         *
    00230          BAD-RECEIVE-NOTIFY-2.
    00231         * THIS PARAGRAPH IS PERFORMED.
    00232             DISPLAY BAD-INPUT-STATUS-MSG UPON CONSOLE.
    00233             DISPLAY "AND FIRST TEN CHARS OF REC FIELD ARE: ",
    00234                 FIRST-TEN-CHARACTERS UPON CONSOLE.
    00235             DISPLAY "AND END KEY VALUE IS ", END-KEY
    00236                 UPON CONSOLE.
    00237             STOP "TAKE A DUMP.".
    00238             STOP RUN.
    00239         *
    00240          BAD-SEND-NOTIFICATION-2.
    00241         * THIS PARAGRAPH IS PERFORMED.
    00242             DISPLAY BAD-OUTPUT-STATUS-MSG UPON CONSOLE.
    00243             DISPLAY OUTPUT-ERROR-KEYS-MSG UPON CONSOLE.
    00244             STOP "TAKE A DUMP.".
    00245             STOP RUN.
```

Figure 6-6. COBOL Source Program (Part 5 of 5)

4. DB#SNT Macro

   Specifies the ICAM terminal name (TERM=) and the equivalent COBOL symbolic terminal name (SYMN=). As shown in Figure 6-5, we made the COBOL symbolic terminal names the same as the terminal names specified in the network definition. When the COBOL program in Figure 6-6 executes a RECEIVE statement, it receives a terminal message and the symbolic name of the terminal that originated the message via the SYMSOURCE data item of INPUT-CD-1. Similarly, when the COBOL program executes a SEND statement to transmit a message to a terminal, the program supplies the symbolic terminal name in the SYMBOLICDEST data item of OUTPUT-CD-1.

   Because we specify INIT=YES, CMCS permits messages to flow when COBOL begins execution from the terminal to the process file, and from the terminal output queue to the terminal for messages that originate at the COBOL program.

5. DB#IRT Macro

   Specifies a process file named by a DB#SQT macro and the ICAM names of the terminals that can supply messages to it.

6. DB#END Macro

   Specifies the end of the CMCS generation macros.

6.6.3. Compiling and Linking the COBOL Program

Figure 6-7 shows the job stream to compile and link our COBOL communications program. For more information on writing COBOL compile and link job streams, see the *1974 American Standard COBOL Programming Reference Manual* (UP-8613).

```
// JOB CMPLNK,,,,,,,F114
// COBL 74L PRNTR=20,
//1 AXREF=YES,CMCS=GAL2CMCS,
//2 CMCSST=YES,MAP=YES,MXREF=YES,OBJLST=YES,PROVER=YES
/$
  .
  .   Source program
  .
/*
/$
   LINKOP OUT=$Y$LOD
   LOADM GALEN2
/*
/&
// FIN
```

Figure 6-7. COBOL Compile and Link Job Stream

6.6.4. Loading and Executing the ICAM Symbiont and GUST

Before the COBOL program is loaded and executed, the ICAM symbiont and the ICAM global user service task (GUST) must be loaded and executed. Figure 6-3 specifies the symbiont name (MCPNAME=M1). At the system console, enter

```
M1
```

to load the ICAM symbiont and place it in execution.

When the symbiont is ready for operation, the following message appears on the screen:

```
ICAM READY
```

Then load and execute GUST by entering:

```
RV GUST
```

This command calls the GUST job stream on disk.

GUST requires initializing parameters at this point. The instructions for entering the correct parameters are given in the *Models 3-6 and 8-20 ICAM Operations Guide* (UP-9745) and the *Model 7E ICAM Operations Guide* (70023908). For our network, we enter:

```
GNET,,N,ALL
```

GUST then opens the network GNET and activates all lines, and terminal polling begins.

6.6.5. Executing the COBOL Load Module

The final step in this procedure is executing the COBOL load module. We execute our load module with the following job stream:

```
// JOB RUNCMCS
// OPTION GSYSDUMP
// EXEC GALEN2
/$
/*
/&
// FIN
```

Section 7
Dumping the Single-Line Communications Adapter

The single-line communications adapter (SLCA) dump routine lets you dump and print the contents of an SLCA random access memory (RAM). This serves as an aid in diagnosing SLCA problems.

This section is not applicable for the model 7E. Instead of using SLCA, the model 7E uses COMMDLP as its communications processor.

7.1. Executing the SLCA Dump Routine

You execute the SLCA dump routine by entering the following at the system operator's console:

Format

```
RV SLCADUMP [ ,,ID= { ccnn } ] [,TYPE=t]
            [      { 0208 } ]
            [      { 1301 } ]
```

where:

`ID=ccnn`

Identifies the channel and line number of the SLCA. The default is 0208 for System 80 models 3-6 and 1301 for models 8-20. If you enter a 2-digit number, that value is accepted as the line number and a channel number of 02 (models 3-6) or 13 (models 8-20) is assumed.

Valid channel and line numbers are:

Model	Channel Number	Line Number
3-6	02	08 through 15
8-20	13 or 15	01 through 15

The line number must match the ID specification on the LINE macro in the ICAM network defintion.

Note: *Refer to the* Models 3-6 and 8-20 ICAM Operations Guide *(UP-9745).*

TYPE=t
Is a 1-digit number from 1 to 5 that identifies the type of SLCA. This specification is optional for types 1-3 and required for type 5.

Example 1

```
RV SLCADUMP
```

This example assumes that the SLCA being dumped has a channel number of 02 and line number 08 (for models 3-6) or channel number 13 and line number 01 (models 8-20). The SLCA is type 1, 2, or 3.

Example 2

```
RV SLCADUMP,,ID=1502
```

Dumps an SLCA with the channel number 15 and line number 02. The SLCA type is 1, 2, or 3.

Example 3

```
RV SLCADUMP,,ID=10,TYPE=5
```

Dumps an SLCA with the channel number 02 (assumed for models 3-6) or 13 (models 8-20) and line number 10. The SLCA is type 5.

Example 4

```
RV SLCADUMP,,,TYPE=5
```

Dumps a type 5 SLCA. Channel and line number 0208 are assumed for models 3-6 or 1301 for models 8-20. Note that three commas are required before the TYPE parameter when you omit ID.

7.2. User Considerations

Before executing the dump routine, the console operator must make certain that the communications line to which the SLCA is attached is idle. That is, your program has issued a line release or the operator has downed (DO) the line. Note, that an autodial SLCA cannot be dumped because it has no storage area. If a dump is attempted, an error message is displayed on the operator console.

7.3. Sample Listing

Figure 7-1 is a typical listing produced by the dump routine of an SLCA random access memory.

7.4. Error Messages

When the dump routine detects an error, it prints an error message on the operator console. These messages are described in the *OS/3 System Messages Reference Manual* (UP-8076).y

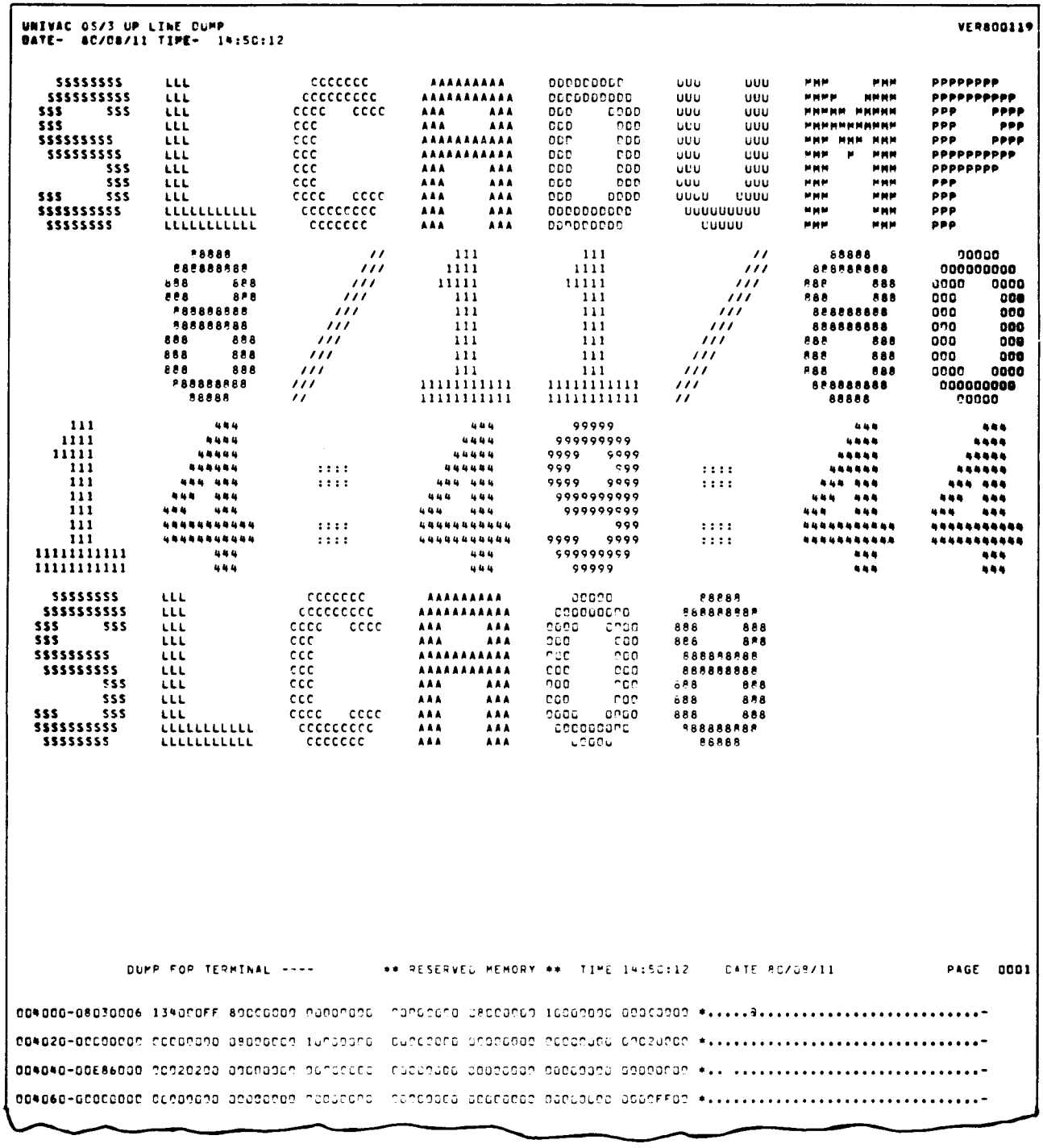


Figure 7-1. Sample Dump Listing of the SLCA

Section 8
ICAM Trace Facility (ITF)

8.1. General Description

The ICAM trace facility (ITF) is a diagnostic aid you use only when you have communications problems that require the help of Unisys personnel and resources. You load and execute the trace facility only to obtain records that you then forward to Unisys for trouble analysis.

The trace facility accumulates records of key communications operations for detailed study by Unisys to determine the cause of an abnormal condition. This section describes how to load and execute the trace facility and how to get the records needed for the analysis.

The trace facility is an OS/3 symbiont. It has no specific ICAM network definition requirements. You can load and execute it from any of the following:

- System console
- Local workstation (Not applicable to the model 7E)
- Remote workstation in system mode (Not applicable to the model 7E)
- Terminal using OS/3 Interactive Services

After you load the trace facility, you specify the segment (category) of the communications network you want traced, the number of trace events you want recorded, and other available trace options. From this, the trace facility acquires an area in main storage to accumulate the trace records for later disposition.

The ICAM trace facility system informational and error messages are listed and described in the *OS/3 System Messages Reference Manual* (UP-8076).

After you accumulate the trace records in main storage, you can write the data to a printer using the ITF snap command. When you send a dump to Unisys for analysis, specify the largest trace area (number of events) possible and try to limit your dump to just those categories where the trouble might logically be located.

You can display the events recorded by the trace facility by executing the ICAM edit dump (Section 9). Alternately, if you specify disk tracing and spooling, the disk files are automatically spooled to the printer. If you use the ICAM edit dump, you must execute it before the main storage trace area is released, whether disk tracing is active or not.

The ICAM trace facility:

- Enables and disables trace categories as specified
- Initializes and maintains trace area pointers
- Acquires, maintains, and releases main storage for recording events
- Can use two disk files to hold trace entries
- Spools data from disk files to printer
- Traces a specific line at the physical level

8.2. Loading the Trace Facility

Before loading ITF, make sure that:

- Dynamic buffer management is generated with the OS/3 Supervisor
- ICAM is loaded before the trace facility

To load the trace facility symbiont without tracing any events, type in:

```
ITF
```

The system will acknowledge the command with the following response:

```
MC#155 ITF - INITIALIZED; AWAITING COMMAND
```

If ICAM and ITF are loaded and ICAM subsequently terminates, always cancel ITF with a CANCEL command before reloading another or the same ICAM.

8.3. Executing the Trace Facility

The ICAM trace facility uses seven commands: ENABLE, DISABLE, STATUS, HELP, PAUSE, GO, and SNAP. Once the trace facility is loaded:

- ENABLE lets you specify the categories you want traced.
- DISABLE lets you turn off tracing of categories or terminate the symbiont.
- STATUS lets you find out which categories are currently active and gives current trace counts.
- HELP prints the formats of all the trace commands on the terminal and the status of the trace facility.
- PAUSE inhibits further trace entries until the GO command is received.
- SNAP enables an edited printout of trace buffers.

Each trace category monitors critical points in a major area of ICAM. These are:

- PHYSICAL - Physical input/output control system (CPIOCS)
- LOGICAL - Distributed communications architecture (DCA) structures
- NETWORK - Public data networks (PDNs)
- QUEUER - ICAM queuing
- CONTROL - ICAM activity control
- COMMDLP - Trace data provided by COMMDLP (model 7E only)

In the commands provided for executing the ICAM trace facility, the initial characters in some of the keywords are underlined. You can enter just that character to obtain the function, or you can enter the entire keyword, if you wish. However, any single message to the trace facility is limited to a maximum of 60 characters.

Both spaces and commas are used as separators, each in a different way. Use them as shown in the formats that follow.

After the ITF symbiont is loaded, you must precede each ITF command with the 00 required for OS/3 unsolicited commands and the trace facility identifier ITF as follows:

```
00ΔITFΔcommand
```

When you enter a trace facility command at the terminal, the system responds with an appropriate series of messages. Included in each response to a trace facility command (ENABLE, DISABLE, STATUS, HELP, PAUSE, GO, SNAP) are the messages MC#162 ITF and MC#163 ITF. They provide the current status of the trace facility along with any other messages that might be provided.

The MC#162 ITF message appears as follows:

```
MC#162 ITF - CATEGORY=PNLQCD  EVENTS=0250  BUFF ADDR=091218
```

where:

CATEGORY=PNLQCD
: Identifies the categories that are enabled. In this case, all six categories are enabled.

EVENTS=0250
: Specifies the number of events provided for the trace buffer. In this case, the default value of 250 is indicated.

BUFF ADDR=091218
: Indicates the start address of the ITF trace buffer in a free buffer pool.

The MC#163 ITF message appears as follows:

```
MC#163 ITF - WRAP=YES LINE=LNE2 DISK=YES VOL=RUN  SPL=YES
             WRAP CNT=0015  GAP CNT=0000
```

where:

WRAP=YES
: Indicates if new trace data is to be written to the trace buffer. In this case, new data overwrites previous trace data after the trace buffer is full.

LINE=LNE2
: Specifies the name of the communications line being traced. This name must be the same as the label of the LINE macroinstruction in the ICAM network definition. LNE2 is specified in this example.

DISK=YES
: Specifies if trace data is to be copied from the trace buffer to the ITF disk files, ITF1 and ITF2. ITF dynamically allocates these two MIRAM files on the disk pack indicated in the VOL parameter.

VOL=RUN
: Specifies the volume serial number of the disk pack where the ITF disk files are to be dynamically created. In this example, the default value of the disk pack assigned as SYSRUN (at initial program load time) is used. The VOL value displayed is RUN when the default value is used.

SPL=YES
: Specifies if trace data is to be written to the OS/3 printer spool file. In this case, trace data is copied from the ITF disk files to the OS/3 spooler.

WRAP CNT=0015
: This field indicates the number of times the trace buffer has been overwritten with new data. In this example, the trace buffer has been overwritten 15 times.

GAP CNT=0000
: This field indicates the number of times trace data may not have been written to the trace buffer. If WRAP=NO is in effect, this indicator advances after the trace buffer is full or if there is a possible loss of trace data due to trace buffer-to-disk file copying synchronization.

8.3.1. Enabling The Trace Facility (ENABLE)

You enable the trace facility and specify your parameters with the following command format:

```
00ΔITFΔTRACE=ENABLEΔ {[PHYSICAL][,LOGICAL][,NETWORK][,QUEUER][,CONTROL][,COMMDLP]Δ}
                     {all                                                        }

[EVENTS={num-of-events}]Δ[DISK={N          }]Δ[SPL={N}]Δ[LINE={line-name}]Δ[WRAP={N}]
        {250          }      {Y {,vsn}    }       {Y}      {N        }       {Y}
                             {  {RUN}    }
```

where:

TRACE=ENABLE
: Enables the ICAM trace facility for all categories specified. When specified, no individual parameter within the command can be disabled.

{[PHYSICAL][,LOGICAL][,NETWORK][,QUEUER][,CONTROL][,COMMDLP]
 ALL}

Specifies the categories for which a trace is desired. The default is all categories. If some categories have been previously enabled, the default enables the remaining disabled categories.

PHYSICAL
: Traces events pertaining to the communications physical input/output control system (CPIOCS).

LOGICAL
: Traces events pertaining to distributed communications architecture (DCA) structures.

NETWORK
: Traces events pertaining to public data networks (PDNs).

QUEUER
: Traces events pertaining to ICAM queuing.

CONTROL
: Traces events pertaining to ICAM activity control.

COMMDLP
: Traces events reported to ICAM by the COMMDLP (model 7E only).

EVENTS= {num-of-events
 250}

Specifies how many events you want traced. You can specify a value from 1 to 9999 up to the amount of main storage available. (See Table 8-1 for main storage requirements.)

If disk tracing is active, the number of events cannot be changed until the disk tracing is disabled.

DISK= {N
 Y {,vsn
 RUN}}

Specifies disk tracing (Y,*vsn*). A 6-character volume serial number (*vsn*) must be specified when you use the Y option to indicate where the disk files are to reside. (ITF dynamically creates two disk files, ITF1 and ITF2.) If you specify DISK=Y without a vsn, the disk pack that was assigned as SYSRUN at system initial program load (IPL) time is used for the ITF disk files.

You can specify disk tracing without spooling, but you cannot specify spooling without disk tracing.

SPL= {N / Y}

Specifies spooling of disk files (Y). Spooling cannot be enabled unless disk tracing is specified (DISK=Y,*vsn*).

LINE= {line-name / N}

Specifies a specific communications line to be traced. Line-name is the name specified in the ICAM network definition as the line label. If there are two network definitions in the system with the same label, only the first one will be traced. Either the PHYSICAL or the COMMDLP category must be enabled for the LINE parameter to be active. This parameter is required when enabling the COMMDLP category.

WRAP= {N / Y}

Specifies that the trace area will be overwritten after it is filled.

Notes:

1. *Parameters can be in any order following the TRACE parameter.*
2. *The DISK, SPL, LINE, and WRAP parameters can be enabled or disabled without specifying categories or events unless you want to change those categories.*

Examples

1. This is an example of a normal enable of the trace facility with all parameter defaults. The normal system responses follow:

   Trace Command:

   ```
   00 ITF TRACE=ENABLE
   ```

   System Response:

   ```
   MC#159 ITF - TRACING ENABLED
   MC#162 ITF - CATEGORY=PNLQC  EVENTS=0250  BUFF ADDR=091218
   MC#163 ITF - WRAP=YES LINE=    DISK=NO  VOL=        SPL=NO
                WRAP CNT=0000  GAP CNT=0000
   ```

2. If the trace facility is enabled as illustrated in example 1, changes can be made while the trace facility is in the enabled mode. For example, the number of events can be changed:

   Trace Command:

   ```
   00 ITF T=E E=500
   ```

System Response:

```
MC#162 ITF - CATEGORY=PNLQC  EVENTS=0500  BUFF ADDR=091218
MC#163 ITF - WRAP=YES LINE=    DISK=NO  VOL=        SPL=NO
             WRAP CNT=0015  GAP CNT=0000
```

Note: *ITF trace buffers have been overwritten 15 times as indicated by WRAP CNT=0015.*

8.3.2. Disabling the Trace Facility (DISABLE)

The DISABLE command terminates the trace facility or any of the parameters previously enabled by an ENABLE command. With this command, individual parameters cannot be changed, only disabled.

To disable the trace facility completely, specify:

```
00ΔITFΔTRACE=DISABLE
```

The normal system response is:

```
MC#154 ITF - TRACE DISABLED, ITF TERMINATED
```

If you wish to disable only certain trace categories or individual parameters while keeping the trace facility active for the remaining categories or parameters, disable just those parameters not desired.

For example, this command format disables the QUEUER trace category:

```
00 ITF TRACE=DISABLE QUEUER
```

or:

```
00 ITF T=D Q
```

The normal system response is:

```
MC#162 ITF - CATEGORY=PNLCD   EVENTS=0500  BUFF ADDR=091218
MC#163 ITF - WRAP=YES LINE=      DISK=NO  VOL=       SPL=NO
             WRAP CNT=0015  GAP CNT=0000
```

The status of the ICAM symbiont determines how you terminate the trace facility. If ICAM and ITF are loaded, terminate the trace facility with 00ΔITFΔT=D.

Notes:

1. *Before terminating the trace facility, use the ICAM edit dump (see the ITF SNAP command in Section 10) or ITF spooling (if you are using disk storage) to print the events it has traced and recorded.*

2. *All disk files created by the trace facility are dynamically allocated and erased when ITF is terminated.*

3. *Always terminate the trace facility before terminating ICAM.*

4. *If you disable disk tracing, spooling is also disabled automatically.*

8.3.3. Displaying Trace Facility Status (STATUS)

The STATUS command displays the status of the trace facility. In response to this command, the trace facility lists:

- Categories being traced
- Size of the dynamically acquired storage area (number of events it can hold)

Formats

```
00ΔITFΔTRACE=STATUS

TRACE=?
```

Examples

```
00 ITF TRACE=STATUS
00 ITF T=S
00 ITF T=?
```

System Response

The system responds to a status request by displaying ITF messages 162 and 163 in the same format as shown in 8.3.1 and 8.3.2.

8.3.4. Displaying the Command Formats (HELP)

The HELP command causes the trace facility to print all of the acceptable command formats on the console or workstation.

Format

```
00ΔITFΔTRACE=HELP
```

Examples

```
00 ITF TRACE=HELP
00 ITF T=H
```

System Response

The system responds to a help request with a display of the ITF command formats and messages 162 and 163 to provide the current status.

8.3.5. Inhibiting and Restarting the Trace Facility (PAUSE and GO)

The PAUSE command inhibits further entries into the trace until a GO command is received.

Formats

```
00ΔITFΔTRACE=PAUSE

00ΔITFΔTRACE=GO
```

Examples

```
00 ITF T=P
00 ITF T=G
```

System Response

The system responds to a PAUSE command with ITF message 279 and to a GO command with ITF message 280.

8.3.6. Displaying the Trace Buffers (SNAP)

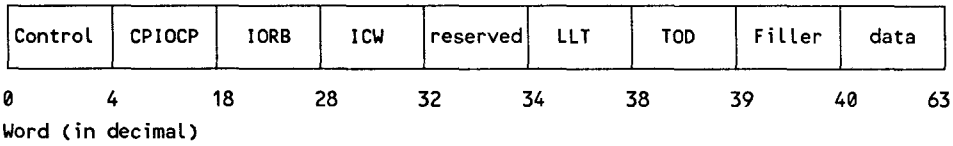
The SNAP command enables an edited printout of ITF dynamic trace buffers.

Format

```
00ΔITFΔSNAP
```

8.4. ITF Physical Entries

By analyzing the contents of the ITF physical entries, you may be able to pinpoint hardware problems. The following is the format of the physical trace entry:



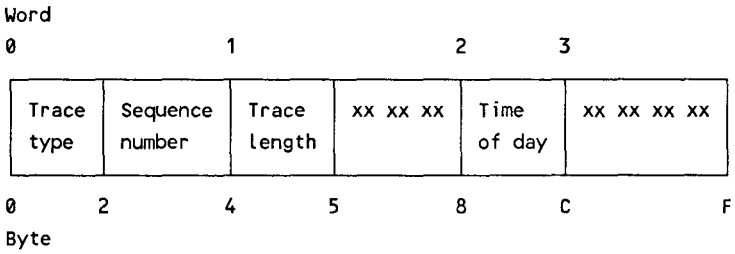
Field	Contents	Length
Control	See 8.4.1, "Field Descriptions" for details.	4 words
CPIOCP	Communications physical input/output control packet. See 8.4.1, "Field Descriptions" for details.	14 words
IORB	Input/output request block. See 8.4.1, "Field Descriptions" for details.	10 words
ICW	Input/output control word. See 8.4.1, "Field Descriptions" for details.	4 words
–	Reserved for ICAM usage	2 words
LLT	Line link table - Only the first 4 words are recorded in the trace entry. See the *ICAM Communications Physical Interface (CPI) Programming Guide* (UP-9746) for details.	4 words
TOD	Time of day (in milliseconds beyond midnight) - Current time of day stored with each physical entry. Use this time to calculate exact elapsed time values instead of the value found in the Control section.	1 word
Filler	X'AAAAAAAA' - This highlights the start of data to follow.	1 word
Data	The first 96 bytes of data transmitted/received. Data presentation is device dependent (e.g., UNISCOPE protocol is ASCII).	24 words

The total length of each physical entry traced is 64 words.

8.4.1. Field Descriptions

This subsection describes the Control, CPIOCP, IORB, and ICW fields.

Control Field



Word 0

Trace type

Value (hex)	Type
0001	Start I/O
0002	Start I/O error
0003	Normal interrupt completion
0004	Time out
0005	Error - entry before status stored
0006	Attention interrupt
0007	Error - entry has status stored
0009	Halt I/O
000A	IORB pool empty or SLCA load error
1011	Input CCRCALL issued by device handler (model 7E only)
1013	Normal input completion returned to device handler (model 7E only)
1017	Normal output completion returned to device handler (model 7E only)
1023	Error input completion returned to device handler (model 7E only)
1027	Error output completion returned to device handler (model 7E only)
00C3	COMMDLP input trace (model 7E only)
00C5	COMMDLP output trace (model 7E only)

Sequence number - Value will wrap.

Word 1

Byte 0 - Length of the entry in words. Always equals X'40'.

Word 2

Time stamp - updated once a second.

CPIOCP Field

The control packet is an activity request packet for the physical interface. It is 14 words long. The ICAM DSECT for this structure is TN#PARP.

Byte	0	1	2	3	Word
0	I R L / Activity control que index	Activity control chain address or CPIOCS work field			0
4	Completion address				1
8	User flag	LOCAP address			2
C	Logical status (primary)	Logical status (detail)	Device status	Channel status	3
10	Hardware sense byte 0	Hardware sense byte 1	Residual byte count		4
14	Hardware command code	Buffer address			5
18	Time allocation (seconds)		Buffer length		6
1C	Logical command func.	CPIOCP chain address			7
20	Control flags	Line control table address			8
24	Channel number	SLCA number	Port number		9
28					A
2C					B
30	Sense bytes 0 -3				C
34	Sense bytes 4 -7				D

Tables 8-1, 8-2, and 8-3 describe the values for the device status, channel status, and hardware command codefields. For details concerning the rest if the fields within a CPIOCP packet, refer to the *ICAM CPI Programming Guide* (UP-9746).

Table 8-1. Device Status Descriptions (CPIOCP)

Value	Meaning
2^7	Attention
2^6	Status modifier
2^5	Control unit end
2^4	Busy
2^3	Channel end
2^2	Device end
2^1	Unit check
2^0	Unit exception

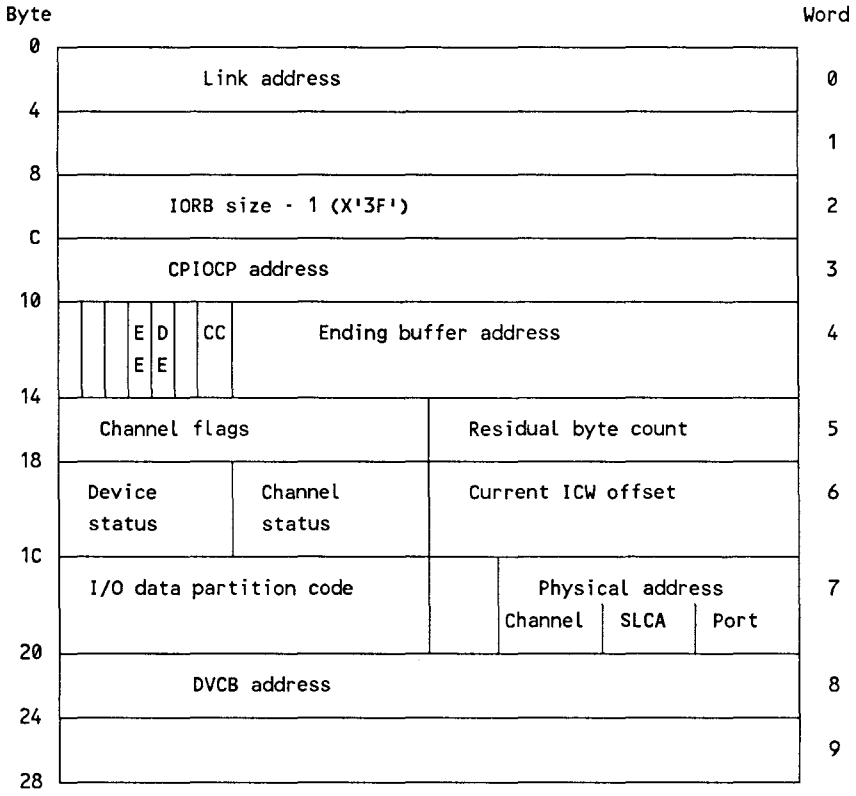
Table 8-2. Channel Status Descriptions

Value	Meaning
2^7	Unassigned
2^6	Incorrect length
2^5	Program check
2^4	Invalid address
2^3	Channel data check
2^2	Interface control check
2^1	Channel control check
2^0	Buffer terminate

Table 8-3. Hardware Command Code Descriptions

Value	Meaning
01	Enable data output
02	Enable data input
09	Send idle
11	Send space
13	Disconnect
15	Load port control word
16	Read port control word
19	Load control character detect table 1
1A	Read control character detect table 1
1D	Load control interpretation table 1
1E	Read control interpretation table 1
23	Turnoff
26	Look for sync
27	Modem test
29	Send idle
2A	New sync
2B	Line adapter test
2D	Send mark
2E	Clear active
2F	Line adapter clear
33	Disconnect
37	Enable data set ready
3B	Set full duplex
3F	Set busy
59	Load control character detect table 2
5A	Read control character detect table 2
95	Load control bytes 3 and 4
D5	Load control byte 4

IORB Field



The physical device address is a 12-bit field divided into three 4-bit elements: the channel address, the SLCA address, and the SLCA port address.

Tables 8-4 and 8-5 describe the channel flags and device status fields. For further information rearding the IORB, refer to Volume 2 of the *System 80 Models 8-20 Processor Complex Controllers Programming Reference Manual* (UP-9732).

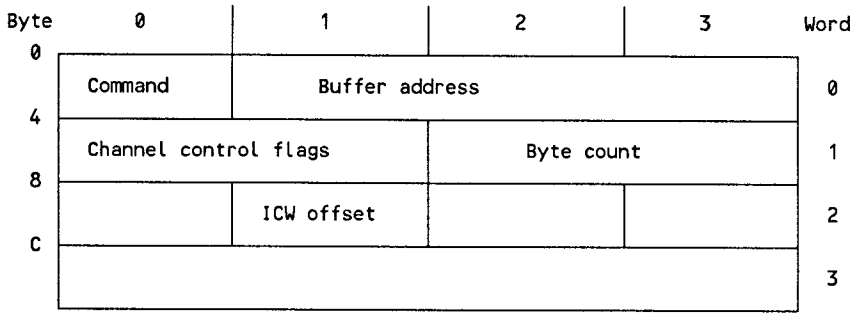
Table 8-4. Channel Flags

Value	Meaning
Byte 14	
2^7	Chain data
2^6	Chain command
2^5	Suppress length indication
2^4	Skip
Byte 15	
2^6	Chain buffer
2^0	Buffer terminate

Table 8-5. Device Status Descriptions (IORB)

Value	Meaning
2^7	Attention
2^6	Status modifier
2^5	Control unit end
2^4	Busy
2^3	Channel end
2^2	Device end
2^1	Unit check
2^0	N/A

ICW Field



For further information rearding the ICW, refer to Volume 2 of the *System 80 Models 8-20 Processor Complex Controllers Programming Reference Manual* (UP-9732).

Section 9
Device Trace Symbiont (DT)

The device trace symbiont (DT) is a diagnostic tool for monitoring communications and workstation activity. The DT symbiont lets you monitor input and output messages. It can be used in conjunction with the ITF facility, which is described in Section 8.

The purpose of the DT symbiont is to log in a file for later analysis some or all of the interrupts received from a channel or port. These interrupts are traced by swapping the system I/O and, after logging the necessary data, returning the interrupts to the system interrupt handlers. Logging to disk is accomplished by having the DT symbiont allocate a 5-cylinder SAT file (called SYSREC) when the symbiont is entered intially.

You can activate the device trace symbiont by entering DT at the console, workstation, or terminal. You can also enter optional trace commands. Several commands can be entered, separated by commas. If a command has an associated parameter, enclose it in parentheses.

Example

```
DT RECORD=(0311),STOP
```

You can also enter device trace commands as unsolicited messages, one at a time, in one of the following formats:

```
00 DT xxxx

00 DT xxxx=nnnn

00 DT xxxx=(nnnn)
```

If the symbiont is initialized with no comands, the following sequence occurs:

1. `ENTER DEVICE ADDRESS (HEXA) TYPE IN XXXX'`

   Enter the physical device address represented by XXXX (see 9.2 for further details on device addressing).

2. `DEVICE TRACE SYMBIONT READY`
   `TO START RECORDING ISSUE: 00 DT RECORD`

   After these messages are displayed, DT can accept an unsolicited command.

3. Operator entry of an unsolicited command

```
00 DT RECORD
```

initiates recording of I/O activity for the appropriate device(s).

4. To terminate the recording process, enter:

```
00 DT STOP
```

5. DT displays the number of records and pages stored in the trace file.

6. To display the contents of the trace file, enter one of the following:

```
00 DT REPLAY        to direct output to the printer

00 DT DISPLAY       to direct output to the console, workstation, or terminal
```

7. To terminate the DT symbiont, enter

```
00 DT END
```

9.1. Device Trace Commands

The following are the device trace commands:

BACKWARD
: Skips backward the number of blocks specified by the command (default = 8) or by a previous SKIP command. For example:

```
BACK=5          backs up 5 blocks

BACK            backs up according to a previous SKIP or 8 (default)
```

CONTINUE
: Resumes the halted command (DISPLAY, PRINT, or REPLAY).

DISPLAY
: Displays the contents of the trace file in one or two lines on the console, workstation, or terminal.

DUMP
: Cancels the DT symbiont with a dump. The trace file is not scratched.

EJECT
: Stops recording when the end of the trace file is reached.

END

Terminates the DT symbiont and scratches the trace file.

ERASE

Initializes DT file counters. This function is applicable for the following commands:

```
CONTINUE
DISPLAY
PRINT
RECORD
REPLAY
```

FORWARD

Skips forward the number of blocks specified by the command (default = 8) or by a previous SKIP command.

FULL

Logs the complete data buffer in addition to those items traced by the MAXIMUM command.

HALT

Stops the following DISPLAY, PRINT, and REPLAY commands temporarily.

MAXIMUM

Logs the IORB in addition to those items traced by the MINIMUM and MEDIUM commands.

MEDIUM

Traces the following items in addition to those provided by the MINIMUM command:

- 1st and 9th words of the DVCB
- First 4 characters of the program name
- First 3 IOST words (Models 8-20)

MINIMUM

Traces only the following data:

- Command code
- Device status
- First 2 sense bytes, if any
- 24 bytes of data

This is the default.

PRINT
Prints the contents of the last *n* blocks specified by the SKIP command. (To print the entire trace file, use REPLAY.)

RECORD
Activates recording of the interrupts by swapping the system I/O interrupt address with the DT symbiont trace facility. See 9.2 for further details.

Example:

```
RECORD=(0280)
```

REPLAY
Prints the contents of the entire trace file.

SHORT
Logs only the first 24 bytes of the data buffer. This is the default.

SKIP
Sets the number of blocks to skip forward/backward in the trace file (default = 8).

STATUS
Displays the current device trace symbiont status.

STOP
Stops recording interrupts by resetting the I/O interrupt address to the appropriate handler. This also stops PRINT and DISPLAY.

TIME
Sets the interval (in seconds) bewteen timer interrupts (default = 30).

TRASTBL
Specifies the translation table to be used when displaying/printing the contents of the trace file.

TRAS=E
Uses EBCDIC. This is the default.

TRAS=A
Uses ASCII.

UNIT
Changes the current device address to the new address of a device or group of devices to be traced. See 9.2 for further details.

WRAP
Continues tracing at the beginning of the file when trace recording has reached the end of the file.

9.2. Physical Addressing

A physical address is specified as a 4-character entry in the RECORD command. The format is:

```
ccud
```

where:

cc
: Is the hexidecimal channel number with a leading zero (e.g., 08, 0D).

u
: Is the unit position of the physical address.

d
: Is the decimal position of the physical address.

If an X is specified in the unit position of the physical address, DT traces I/O activity of all units with the same common channel and subchannel numbers.

Each line (SLCA) can be subdivided into three addresses depending on whether the function desired is control, input, or output.

Example 1 - Half-Duplex Line

`0D5X` Trace all activity on channel 13 port 5.

`0D50` Trace only control commands with their associated buffers for channel 13 port 5.

`0D51` Trace input/output commands and their associated buffers for channel 13 port 5.

`0DXX` Trace all activity on all communications lines connected to channel 13.

Example 2 - Full-Duplex Line

`028X` Trace all activity on channel 2 port 8.

`0280` Trace only control commands with their associated buffers for channel 2 port 8.

`0281` Trace only output commands and their associated buffers for channel 2 port 8.

`0292` Trace only input buffers and their associated buffers for channel 2 port 9.

Section 10
ICAM Edit Dump (IED)

10.1. General Description

The ICAM edit dump utility is a symbiont that dumps selected groups of ICAM tables for diagnostic purposes. It supplements the system dump (SYSDUMP). You can use the edit dump to snapshot ICAM while it is running. Use it any time ICAM has a program exception. When you take both an edit dump and a system dump, run the edit dump before the system dump.

The ICAM tables that you can dump are:

- General information tables
  - ICAM general information table
  - GUST general information table
  - Activity control queues
  - CCA address table
- Line link table
- ITF trace
- ICAM task control blocks
  - MCP task control block
  - Subtask task control block
- CCA control section
- Buffer pools
  - ARP
  - Network
  - UDUCT (DCA data unit control table)
  - Link buffers (Not applicable to model 7E)
  - MCPBUF pool (model 7E only)

- Destination table
- Hexadecimal dump of CCA
- RIM (remote interface manager) queues
- Session analysis (session control entries)
- End user tables
  - Communication user programs
  - Line vector table
  - Terminal control table
  - Process file
  - Distribution list

10.2. Executing the Edit Dump

To load the edit dump utility, type in at the system console or workstation:

```
IED
```

The console or workstation screen displays three messages in series. You must respond to each message before the next message is displayed. Edit dump executes after you respond to the third message.

Message 1:

```
TERMINATE AFTER DUMP   (Y)ES OR (N)O?
```

Response:

(Y)ES
: Terminate edit dump after processing command in last message.

(N)O
: Do not terminate; more dumps will be taken. You can respond with N, NO, or any other character.

Message 2:

```
OUTPUT REASON FOR DUMP OR (N)O?
```

Response:

State the reason for the dump. Limit your explanation to 40 characters. An EOT entry results in a blank page.

Message 3:

```
IED800 - ENTER COMMAND OR (H)ELP
```

Response:

Enter the command that gives you the type of dump you want (see 10.3 and Table 10-1). If you are not sure which command is best, enter H or HELP. A table of the available commands is displayed. You can also enter T, causing the edit dump to terminate without a message.

After processing a command, the edit dump terminates if you respond YES to the first message. If you respond NO, the edit dump displays the second message again.

10.3. Selecting the Correct Command Option

Although several options are available when you select the tables to be jumped, the two we recommend are A and Y. If you suspect that you have a problem with physical I/O or action control (ACTCON), enter a Y to get the tables needed for diagnostics related to those components. For problems in other segments of ICAM, enter an A to get the appropriate tables.

The other command options provide smaller selected groups of tables suitable for limited purposes. The command options are described as follows (Table 10-1 lists the ICAM tables and the command options that produce the edit dumps of the tables):

Command	Description
A	Dumps all tables except line link tables and end user tables
B	Dumps buffer pools only
C,*name-1*	Dumps all tables except line link tables, ITF trace, and CCA hexadecimal dump. *name-1* is the label of the CCA macro.

Command	Description
D,*name-1*	Dumps all tables except line link tables and end user tables. *name-1* is the label of the CCA macro.
E,*name-2*	Dumps end user tables. *name-2* is the label of the LINE, TERM, LOCAP, PRCS, or DLIST macro.
I	Dumps ICAM trace facility (ITF). Includes the general information tables
S	Dumps RIM queues and session control entries
Y	Physical I/O or ACTCON dump. Dumps all tables except the hexadecimal dump of the CCA

Notes:

1. *If you do not include a name (CCA macro label) or specify an invalid name for option C or D, nothing is displayed on the console to indicate this.*

2. *If you do not include a name or specify an invalid name for option E, the message END USER NOT IN CCA appears on the printout from the edit dump.*

Table 10-1. ICAM Tables and Edit Dump Options

ICAM Tables	Command Options									Comments
	A	B	C	D	E	I	S	Y		
General information tables	X		X	X		X		X		■ ICAM general information table ■ GUST general information table ■ Activity control queues ■ Primary address table
Line link table								X		
ITF trace	X					X		X		The ITF trace is dumped only if the ICAM trace facility is loaded.

continued

Table 10-1. ICAM Tables and Edit Dump Options (cont.)

ICAM Tables	Command Options									Comments
	A	B	C	D	E	I	S	Y		
ICAM task control blocks	X		X	X				X		Both the MCP task control block (TCB) and the subtask control block dumps include the following tables: ■ Last activity control function ■ TCB extensions ■ TCB blocks ■ ILR trace areas ■ ILR save areas
CCA control section	X	X	X	X				X		
Buffer pools	X	X	X	X				X		For the buffer pools (ARP, network, UDUCT, link), trailing words of all zeros are deleted from the buffer displays.
Destination table	X		X	X				X		
Hexadecimal dump of CCA	X			X						
RIM queues	X		X	X			X	X		
Session analysis	X		X	X			X	X		A session analysis for each session control entry.
End user tables			X		X			X		The end user tables in the dump are formatted as follows: ■ Communications user program (CUP) – LOCAP – User task control block (TCB) – Last activity control function – TCB extension – TCB stacks – ILR trace area – ILR save area – QUEUE analysis ■ Line vector table – Type – Line buffers – Port analysis ■ Terminal control table – Type – Line type – Device control table – Queue analysis ■ Process file – File specification – Queue analysis ■ Distribution list

Section 11
UNIX System Access Module (UNXSAM)

This Section does not apply to model 7E.

11.1. General Description

The UNIX System Access Module (UNXSAM) allows you to access a UNIX operating system to run UNIX applications and shell commands from a local workstation, or from an OS/3 UTS 4000 or SVT class terminal or workstation.

UNXSAM acts as a bridge between OS/3 and a UNIX operating system. To the UNIX operating system, UNXSAM emulates an SVT 1220 or 1210 terminal by transforming UTS data into UNIX data. For the UTS terminal, UNXSAM transforms UNIX data into UTS data. UNIX security procedures are in force as soon as communication with a UNIX operating system is established.

Using UNXSAM requires no changes to the UNIX operating system hardware or software. UNIX lines are physically connected to the System 80 processor as full-duplex lines through an asynchronous port. You can use up to 10 lines to connect the two operating systems.

Figure 11-1 shows the UNXSAM operating environment. UNXSAM accesses the UNIX operating system through ICAM and the UTS terminal/workstation through OS/3 Interactive Services. The ICAM network definition must include a UNIX operating system definition (see 11.3).

Because of differences in hardware and data entry, SVT 1220 emulation is not complete:

- The UNIX display on the UTS terminal or workstation differs from an SVT 1220 display.
- Additional entry operations (from the function keys) are required when working from the local workstation or UTS terminal or workstation.
- The SVT 1220 edit mode of operation is not supported.

Because UTS input is transmitted one line at a time instead of one character at a time, some UNIX applications may not execute properly using UNXSAM.

UNXSAM informational and error messages are documented in the *OS/3 System Messages Reference Manual* (UP-8076).

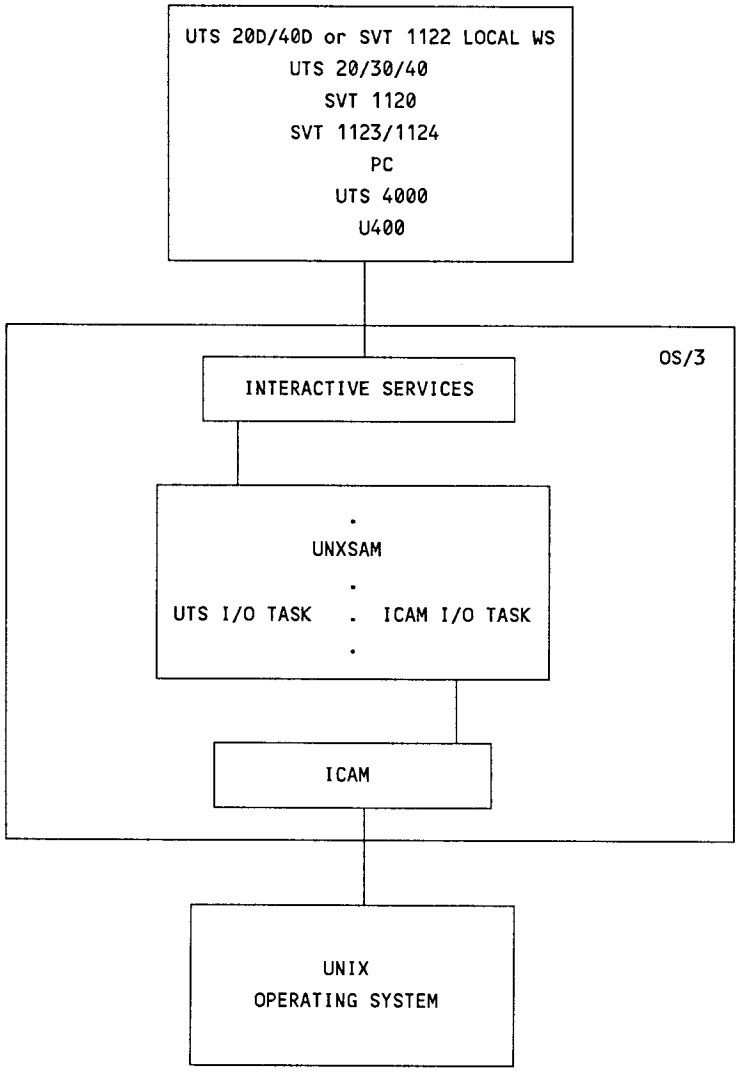


Figure 11-1. UNXSAM Operating Environment

11.2. Accessing UNIX from a UTS Terminal

11.2.1. Initiating UNXSAM

Use standard OS/3 LOGON and LOGOFF commands to initiate and terminate sessions with the OS/3 system.

Initiate UNXSAM by entering the interactive services RV command in one of the following formats:

```
RV UNXSAM
```

or

```
RV UNXSAM,[,L=xxxx][,P=zzzzzzzz]
```

where:

xxxx
: Is the 1- to 4-character name of the ICAM network generated for the UNIX line connection. The default is the CCA name ASYN.

zzzzzzzz
: Is a 1- to 8-character ICAM password for the UNIX network.

You can execute an RV UNXSAM command for each physical UNIX line connected to the System 80 processor.

The following job stream illustrates the initiation procedure. Refer to the *Job Control Language Programming Guide* (UP-9986) for job control statement definitions.

```
// JOB UNXSAM&&,N,,,2
// OPTION JOBDUMP
// GBL L=ASYN,P=
// DVC 20   // LFD PRNTR
// DVC 200  // UID $Y$MAS   // USE SFS  // LFD XDIB
// LBL UNXFKDEF   // LFD FKCDIB
// EXEC RC$UNX
// PARAM &L,&P
/&
```

11.2.2. UNIX Input/Output through UNXSAM

Once connected to a UNIX system through UNXSAM and ICAM, your terminal/workstation is both physically and logically connected to the UNIX operating system.

The input to UNIX you enter at the UTS terminal is one of the following:

- Data required by the UNIX operating system
- Function key entry to the UNXSAM module

Output from UNIX is displayed on the UTS as closely as possible to the data format that would be displayed on a UNIX operating system SVT terminal.

UNXSAM sends an SOE prompt to the UTS as the last character of every transmission from the UNIX operating system.

You can send data to the UNIX operating system in one of the following ways:

- Following the SOE prompt
- After signaling UNXSAM with a function key entry (see Table 11-1)

11.2.3. Using the Function Keys

Table 11-1 shows the UTS function key input from the local workstation or UTS terminal or workstation.

Table 11-1. UTS Function Key Input

Function Key	Description
FK1	Display/exit HELP screen (UNXHELP), which provides a narrative description of the function keys. Function key 1 is a toggle key. If the HELP screen is not displayed, press FK1 to display it. If the HELP screen is displayed, press FK1 to resume normal processing and restore the screen image.
FK2	Enter/exit block mode. Function key 2 is a toggle key that takes you in and out of block mode from line mode.

continued

Table 11-1. UTS Function Key Input (cont.)

Function Key	Description
FK3	Enter control sequence. Function key 3 allows you to enter an SVT/UNIX special key sequence and transmit it to UNIX. See "Entering Control Sequence/Text" for details.
FK4	Set operation mode (8-bit/application). Function key 4 is a toggle key that sets 8-bit and/or application mode on and/or off.
FK5	Space bar (line mode)/repeat key (block mode) Function key 5 operates differently depending on the mode you are in. - Line mode - UNIX MORE function: reserves 20 additional lines for input. (Incoming UNIX data to be displayed at the UTS is deleted until you press the ENTER key.) - Block mode - Repeats last command sequence/text character.
FK6 - FK20	Display/transmit defined function key string. Use function keys 6 through 20 to display the data string defined for that particular key. UNXSAM transmits the data string to UNIX if in block mode, or if so specified in the string. See "Defining Function Keys 6-20" for details.
FK21	Set debug SNAP on/off. Function key 21 is a toggle key that activates/deactivates the OS/3 SNAP calls to print UNIX and UTS data traffic.
FK22	Terminate UNXSAM UNIX session. Function key 22 causes the current UNXSAM session to terminate. A control D is transmitted to UNIX in order to terminate any UNIX process that may be executing.

Entering Control Sequence/Text

In response to function key 3, UNXSAM moves the cursor to line 24, clears the line, and displays an SOE prompt. You can enter either text or control sequences. No carriage return is appended to the end of the transmission. UNIX data received during the entry of FK3 input is not displayed and is discarded.

Table 11-2 shows the special key sequences and their UNXSAM default function key assignments (where applicable), which are also listed on the UNXSAM HELP screens. You can enter one or more special key sequences after entry via FK3 or in block mode. The UNXSAM < delimiter must be entered before each key identifier (for example, enter <AU for arrow up). The key identifiers can be entered in uppercase or lowercase.

Table 11-2. UNXSAM Control Sequences

Key	Default	UNIX Key Description
AU	7	Arrow up (scan up)
AD	8	Arrow down (scan down)
AR	6	Arrow right (scan right)
AL	9	Arrow left (scan left)
FD	-	Find
IH	-	Insert here
RM	-	Remove
SL	18	Select
PS	20	Previous screen
NS	19	Next screen
ES	10	Escape
EN	-	Enter
Pn	-	PF1 - PF4, PF6 - PF9
Pnn	-	PF10 - PF20
SP	17	Space
RT	13	Carriage return
LF	12	Line feed
BS	11	Backspace
DL	-	Delete
HT	16	Horizontal tab
Cx	-	UNIX control character. x specifies the specific control character: A - Z, a - z, 2- 8, [,], ?, /, , or space.

Defining Function Keys 6-20

You can assign specific character strings to function keys 6-20. Press function key 6 from the the HELP screen to display the default assignments for function keys 6-20, which are also listed in Table 11-2. To redefine any key, enter the desired data characters and/or UNXSAM control sequences in the Current field for the appropriate function key. To reinstate the default value for a function key that has been changed, erase the current field or enter a space in the first character position.

When you have redefined the desired function keys, press XMIT. UNXSAM validates the data and redisplays the screen, with error messages, if necessary. (You must correct any errors before you can exit from this screen unless you choose to terminate the session via function key 22.) You may further modify the definitions or press function key 6 to accept the screen.

When you accept the screen, the values for function keys 6-20 are saved in a UNXSAM OS/3 system file (UNXFKDEF), which is a MIRAM keyed file. The user-id is used as the access key. Each time you load and execute UNXSAM, it accesses the UNXFKDEF file to determine if any function key values have been redefined. If not, the default values are used. The default values are primarily used for screen mode.

Before saving any nondefault function key definitions, you must allocate the UNXFKDEF file by entering the following command. As a result of this definition execution, the file is cataloged.

```
RV UNXALLOC,[,V=vol]
```

where:

vol
: Is the volume on which the file is allocated. The default is the SYSRES device.

The following job stream illustrates the file allocation/cataloging procedure. A minimum size file is defined.

```
// JOB UNXALLOC
// GBL V=RES
// FKCDIB ALLOC UNXFKDEF,VOL=&V,EXT=(MI,C,0,TRK,10)
// CAT FKCDIB
/&
```

11.3. ICAM Configuration

To communicate with a UNIX system, you must include a definition of the UNIX line in your ICAM configuration.

11.3.1. Network Definition

To configure ICAM for UNXSAM, specify the following parameters in the LINE and TERM macros:

- LINE macro

  `DEVICE=(TTYEM)`
  Specifies a line emulating a full-duplex, uncontrolled, TTY-like device.

  `TYPE=(4800,FLDQ,FULL)`
  Specifies a two-way simultaneous transmission line.

  `LBL=`
  Omit the LBL operand. A default value of 64 words is generated when DEVICE=(TTYEM) is specified.

- TERM macro

  `FEATURES=(TTYEM)`
  Specifies a full-duplex, uncontrolled, TTY-like device.

  `INPUT=(YES)`
  Creates an input message queue.

  `DICE=OFF`
  Indicates that device independent control expressions are not placed in input messages.

  `XLATE=(NO,NO)`
  Indicates that no input translation or output translation takes place for this terminal.

11.3.2. Example ICAM Configuration for UNXSAM

Figure 11-2 shows a sample ICAM configuration for UNXSAM. In this example, the System 80 processor appears to the UNIX operating system as an SVT 1210/1220 terminal.

For a complete description of an ICAM configuration, refer to the *Models 3-6 and 8-20 ICAM Operations Guide* (UP-9745).

```
     1                                                                  72
     |                                                                  |
1.   COMMCT
2.   ASYN       CCA      TYPE=(GBL,,N),                                X
                         FEATURES=(OPCOM,OUTDELV)
3.             BUFFERS  30,256,3,ARP=30,STAT=YES
4.   UNX0       LOCAP    TYPE=(STDMCP),LOW=MAIN,DUSTERR=INLINE
5.   ASY1       LINE     DEVICE=(TTYEM),                               X
                         ID=06,                                        X
                         TYPE=(4800,FLDQ,FULL)
6.   UX00       TERM     FEATURES=(TTYEM),                             X
                         INPUT=(YES),                                  X
                         LOW=MAIN,                                     X
                         DICE=OFF,                                     X
                         XLATE=(NO,NO)
7.             SESSION  PRIMARY,EU1=(UNX0),EU2=(UX00)
8.             ENDCCA
9.             MCP      MCPNAME=C1
               CACH=(06,4800,FULL)
10.  END
```

Notes:

1. Signals the start of an ICAM generation.

2. DCA global network.

3. Specifies the buffering requirements.

4. LOCAP name. A maximum of 10 LOCAPs can be specified. Each LOCAP must have an associated LINE specified, and each LINE must have an associated TERM specified. LOCAP names must be UNX0 through UNX9 (0 through 9).

5. Full-duplex asynchronous line using the TTYEM handler. Full duplex must be specified.

6. Terminal name specification indicating TTYEM. The TERM names must be UX00 through UX09 (00 through 09).

7. Establishes a static session between the LOCAP and the terminal.

8. Indicates the end of the CCA generation.

9. Indicates the name of the ICAM network and specifies the channel hardware assignments.

10. Indicates the end of the ICAM generation.

Figure 11-2. Sample ICAM Configuration for UNXSAM

Section 12
DCP/Telcon Load and Dump Facilities

12.1. Load Facility

A DCP connected to an OS/3 system on a UDLC connection can be downline loaded.

A DCP connected to an OS/3 system on a channel connection can be cross-channel loaded.

12.1.1. File Creation

The file to be loaded (downline or cross-channel) must be created on a Series 1100/2200 system and transported via magnetic tape to the OS/3 system. Refer to the appropriate Communications Delivery Level Installation Guide and Configuration Guide for information on generating Telcon software.

Then copy the Telcon generation output file to magnetic tape using these commands:

1. `@asg,tj t.,u9s,telcon,,ring`
2. `@asg,a telcon*load`
3. `@copout,o telcon*load.prc105,t.`
4. `@mark t.`
5. `@fre,l`

where:

1. Assigns the tape to transport the loadable DCP software.
2. Assigns the Telcon load file (telcon*load), which contains the loadable DCP software. The actual filename created depends on a response supplied during the Telcon generation procedure.
3. Copies the omnibus element from the Telcon load file to the tape. The Telcon filename (telcon*load.prc105) shown here is an example. The element name depends on responses supplied during Telcon generation.
4. Appends a file mark to the tape.

5. Frees the files previously assigned.

This tape is then transported to the OS/3 system where you create a loadable file by entering the following command at the OS/3 system console:

```
RV TELCRFM,,DEV= [ ccuu ] ,VOL= [ volume ],ALLOC= [ Y ]
                 [ ccss ]      [ RES    ]       [ N ]
                 [ ccmn ]
```

where:

cc
: Is the channel address. Valid values are:

  02 for UDLC connections (models 3-6)

  13, 15 for UDLC connections (models 8-20)

  01, 02, 03, 06, 07 for channel connections (model 8)

  01 - 06 for channel connections (models 10-20)

  08, 09, or 10 for UDLC connections (model 7E)

uu
: Is the control unit address. Required for channel connections on models 8-20. Valid values are decimal 08 to 15.

ss
: Is the SLCA device address. Required for UDLC connections on models 3-6 and 8-20. Valid values are decimal 01 to 14.

mn
: Is the COMMDLP address. Required for UDLC connections on model 7E.

  m
  : Is the control unit address: 1-7 for COMMDLPs on channel 8; 0-7 for COMMDLPs on channels 9 or 10.

  n
  : Is the COMMDLP line number (0-7).

volume
: Is the volume serial number of the disk pack on which the file containing the loadable code is to reside. If omitted, the SYSRES pack is assumed.

ALLOC
: Y specifies that the file will be allocated during this run. N specifies that the file already exists. N is the default.

Note: *If the DCP load file is meant to be available for loading through both a channel and a UDLC connection, then TELCRFM must be run for each address.*

Examples

1. The following example creates a new file on SYSRES containing the DCP loadable code. The new file is cataloged and named DCP.LOAD.1505.

```
RV TELCRFM,,DEV=1505,ALLOC=Y
```

2. The following example overwrites an existing file on SYSRES containing the DCP loadable code. The file is named DCP.LOAD.0712.

```
RV TELCRFM,,DEV=0712,ALLOC=N
```

3. The following example creates a new file on volume PACK01 containing the DCP loadable code. The new file is cataloged and named DCP.LOAD.1308.

```
RV TELCRFM,,DEV=1308,ALLOC=Y,VOL=PACK01
```

12.1.2. ICAM Generation

If a DCP is to be downline loaded or cross-channel loaded, the following parameter must be included on the CCA macro:

```
CCA    DCPLOAD=YES
```

12.1.3. System 80 Load Procedure

Perform the following sequence prior to downline or cross-channel loading:

1. Load the ICAM symbiont.
2. Run GUST.
3. Initialize the line over which the DCP load sequence is to occur. (GUST automatically does this unless the line has been previously marked down, in which case you must enter an unsolicited command to reinitialize the line.)

Note: *In order to avoid reinitializing the line, it is recommended that you start the downline/cross-channel load procedure at the DCP before GUST is executed.*

12.1.4. DCP Load Procedure

Refer to the *DCP/OS Operators Reference* (UP-11541) for information on initiating downline or cross-channel loading from a DCP.

12.2. Dump Facility

You can capture a DCP dump on your OS/3 system, write the dump to tape, and print the dump on a Series 1100/2200 system. Here's a brief overview of the procedure you follow; additional details are provided in 12.2.1 through 12.2.6.

1. On your OS/3 system, a MIRAM formatted DCP dump file is created by ICAM when the DCP initiates a dump transfer.
   - A DCP connected to an OS/3 system on a UDLC line is upline dumped to a file on OS/3.
   - A DCP connected to an OS/3 system on a channel is cross-channel dumped to a file on OS/3.
2. Execute the TELDUMP run stream to create a Series 1100/2200 formatted magnetic tape.
3. Transport the tape to a Series 1100/2200 system and use the TELCON COPYD utility to copy the DCP dump file from the tape to a Series 1100/2200 word addressable dump file.
4. Use the word addressable dump file as input to the TELFOR program. TELFOR converts the dump file so that it can be used as input to the DCPDUMP program. The DCPDUMP program prints the dump file.

12.2.1. ICAM Generation

To indicate the upline or cross-channel dump capability, the following parameter must be included on the CCA macro:

```
CCA    DCPLOAD=YES
```

12.2.2. System 80 Load Procedure

Perform the following sequence prior to upline or cross-channel dumping:

1. Load the ICAM symbiont.
2. Run GUST.
3. Initialize the line over which the DCP dump sequence is to occur. (GUST automatically does this unless the line has been previously marked down, in which case you must enter an unsolicited command to reinitialize the line.)

Note: *In order to avoid reinitializing the line, start the upline/cross-channel dump procedure at the DCP before GUST is executed.*

12.2.3. Creating the Dump File on System 80

Note: In order to perform this procedure, the original DCP load file created on OS/3 using the run stream TELCRFM (see 12.1.1.) must be available and cataloged.

To upline or cross channel dump DCP memory to your OS/3 system, set the DCP toggle switches to the OS/3 UDLC line or channel address. Then, press the PROG LOAD button on the DCP.

The DCP transmits the DCP dump file to OS/3, which creates a MIRAM file on the SYSRES device and writes the DCP dump file to the newly created file. Before you dump DCP memory, check that there is enough space on the SYSRES device for the dump file. The size of the dump file depends on the amount of DCP memory. For example, a dump file taken from a DCP 15 with a memory size of four megabytes requires 28 cylinders on an 8433 disk device.

The name of the MIRAM file on the SYSRES is:

```
DCP.DUMP.{ ccuu | ccss | ccmn }.yymmdd.hhmmss
```

where:

cc
: Is the channel address. Valid values are:

  02 for UDLC connections (models 3-6)

  13, 15 for UDLC connections (models 8-20)

  01, 02, 03, 06, 07 for channel connections (model 8)

  01 - 06 for channel connections (models 10-20)

  08, 09, or 10 for UDLC connections (model 7E)

uu
: Is the control unit address. Required for channel connections on models 8-20. Valid values are decimal 08 to 15.

ss
: Is the SLCA device address. Required for UDLC connections on models 3-6 and 8-20. Valid values are decimal 01 to 14.

mn
: Is the COMMDLP address. Required for UDLC connections on model 7E.

m

Is the control unit address: 1-7 for COMMDLPs on channel 8; 0-7 for COMMDLPs on channels 9 or 10.

n

Is the COMMDLP line number (0-7).

yymmdd

Is the date the MIRAM file was created. It comes from the file name.

hhmmss

Is the time the MIRAM file was created. It comes from the file name.

When the dump transfer finishes, ICAM reads the DCP load file and reloads the DCP.

12.2.4. Creating a Series 1100/2200 Formatted Magnetic Tape

On your OS/3 system, create an unlabeled DCP dump tape in Series 1100/2200 format by entering the following command at the system console:

```
RV TELDUMP,,DEV= { ccuu } ,DATE=yymmdd,TIME=hhmmss
                 { ccss }
                 { ccmn }
```

where:

cc

Is the channel address. Valid values are:

02 for UDLC connections (models 3-6)

13, 15 for UDLC connections (models 8-20)

01, 02, 03, 06, 07 for channel connections (model 8)

01 - 06 for channel connections (models 10-20)

08, 09, or 10 for UDLC connections (model 7E)

uu

Is the control unit address. Required for channel connections on models 8-20. Valid values are decimal 08 to 15.

ss

Is the SLCA device address. Required for UDLC connections on models 3-6 and 8-20. Valid values are decimal 01 to 14.

mn
: Is the COMMDLP address. Required for UDLC connections on model 7E.

  m
  : Is the control unit address: 1-7 for COMMDLPs on channel 8; 0-7 for COMMDLPs on channels 9 or 10.

  n
  : Is the COMMDLP line number (0-7).

yymmdd
: Is the date the DCP dump file was created. It comes from the file name.

hhmmss
: Is the time the DCP dump file was created. It comes from the file name.

The TELDUMP job executes load module TZ@DMP which displays two messages on the system console:

MC#614 ENTER TELCON VERSION, DEFAULT IS: TELCON8R1

MC#615 ENTER FILE ID VERSION, DEFAULT IS: OS3DCPDUMP

Respond to these messages by entering information for the header record on the tape. The names you enter are used for console display information on the Series 1100/2200 system to display the tape identification that is being used.

When TELDUMP finishes, transport the tape to the Series 1100/2200 system.

12.2.5. Copying the DCP Dump File to a Series 1100/2200 File

On a Series 1100/2200 system, use the TELCON category 3 COPYD utility to copy the DCP dump file from tape to a word addressable dump file. You can find the TELCON category 3 utilities and related documentation at the end of the TELCON release tape.

The OS/1100 commands to copy the tape to the word addressable dump file are:

1. `@asg,up qual*filename.,d///`
2. `@prt,t qual*filename.`
3. `@asg,tj t.,u9,dcpdump,,noring`
4. `@qual*library.copyd t.,qual*filename.`
5. `@free t.`
6. `@free qual*filename.`

where:

1. Creates and assigns a word addressable dump file on disk.
2. Checks the word addressable dump file.
3. Assigns the DCP dump tape created on OS/3 to a temporary Series 1100/2200 tape file.
4. Executes the COPYD utility which copies the tape file assigned in step 3 to the word addressable dump file assigned in step 1.
5. Frees the Series 1100/2200 tape file.
6. Frees the word addressable dump file.

12.2.6. DCP Dump Processing

You can print the word addressable dump file on a Series 1100/2200 system using the DCPDUMP program. For information on running DCPDUMP, refer to the *Telcon Dump Analysis Guide,* UP-10301.

Index

H

I

J

K

L

M

N

O

P

R

S

T

U

NOTES

NOTES

NOTES

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL
First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL
First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL
First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990