SPERRY UNIVAC
Operating System/3 (OS/3)

Integrated Communications Access Method (ICAM) Transaction Control Interface (TCI)

User Guide

H

SPERRY UNIVAC

UP–8551 Rev. 2

SPERRY UNIVAC
Operating System/3 (OS/3)

Integrated Communications Access Method (ICAM) Transaction Control Interface (TCI)

User Guide

This document contains the latest information available at the time of preparation. Therefore, it may contain descriptions of functions not implemented at manual distribution time. To ensure that you have the latest information regarding levels of implementation and functional availability, please contact your local Sperry Univac representative.

Sperry Univac reserves the right to modify or revise the content of this document. No contractual obligation by Sperry Univac regarding level, scope, or timing of functional implementation is either expressed or implied in this document. It is further understood that in consideration of the receipt or purchase of this document, the recipient or purchaser agrees not to reproduce or copy it by any means whatsoever, nor to permit such action by others, for any purpose without prior written permission from Sperry Univac.

Sperry Univac is a division of the Sperry Rand Corporation.

This document was prepared by Systems Publications using the SPERRY UNIVAC UTS 400 Text Editor. It was printed and distributed by the Customer Information Distribution Center (CIDC), 555 Henderson Rd., King of Prussia, Pa., 19406.

PRINTED IN U.S.A.

PAGE STATUS SUMMARY

ISSUE: UP-8551 Rev. 2

Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level
Cover								
Title Page								
PSS	1							
Preface	1, 2							
Contents	1 thru 6							
1	1 thru 5							
2	1 thru 58							
3	1 thru 62							
4	1 thru 57							
5	1 thru 3							
Appendix A	1 thru 35							
Appendix B	1 thru 23							
Index	1 thru 6							
User Comment Sheet								

All the technical changes are denoted by an arrow (→) in the margin. A downward pointing arrow (↓) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow (↑) is found. A horizontal arrow (→) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.

Preface

This manual is one of a series designed to instruct and guide the programmer in the use of the operating system. This manual specifically describes the transaction control interface (TCI) of the integrated communications access method (ICAM). The intended audience is the novice programmer with a basic knowledge of communications data processing and the programmer whose experience is limited to systems other than SPERRY UNIVAC systems.

The information contained in this manual is presented as follows:

- SECTION 1. INTRODUCTION

  Describes the overall configuration, components, and primary purpose of the TCI.

- SECTION 2. NETWORK DEFINITION

  Describes how a TCI network is defined for system generation via a set of unique macroinstructions.

- SECTION 3. COMMUNICATIONS USER PROGRAM (CUP)

  Describes how the user can write a communications program using a set of unique macroinstructions within his applications program.

- SECTION 4. MESSAGE PROCESSING PROCEDURE SPECIFICATION (MPPS) MACROINSTRUCTIONS

  Describes an extended set of network definition macros that permit automatic processing and routing of messages in and out of the network.

- SECTION 5. ICAM LOADING AND OPERATOR COMMUNICATIONS

  Describes operator communications with ICAM and load procedures.

- APPENDIX A. PACKETS AND TABLES

  Describes the control packets, error conditions, parameters, and work area tables associated with the TCI.

- APPENDIX B. JOURNALING

  Describes the network definition, message processing procedure specifications, and data utility to provide a record of message events and/or permit cold restart of a system after failure.

Contents

APPENDIXES

FIGURES

TABLES

1. Introduction

1.1. GENERAL

The transaction control interface (TCI) provides you with simplified and efficient communications interactive message control procedures for the integrated communications access method (ICAM). It is designed to support the SPERRY UNIVAC Series 90 Information Management System 90 (IMS 90), but it also provides some additional enhancements not required by IMS 90.

The TCI allows your program to examine the first text fields of each incoming message and to selectively retrieve the message for immediate processing or defer it for later processing. Thus, you may specify concurrent processing of transactions without being involved with complex message send and receive procedures.

A high degree of compactness is achieved by providing an optional disk buffering feature (nonqueued) for both input and output messages. Disk-buffered input may also be designated with main storage or disk queueing for output. Messages may be staged in main storage for immediate processing or on disk for deferred processing. The logic of determination is directly under your control. Your program is scheduled automatically when each message arrives in the system and the first text fields have been transferred to your transaction terminal table. After examination, you may request the message immediately or may defer it for later processing.

A communications control area (CCA) is required to define the network for system generation (Section 2). A communications user program (CUP) is employed (Section 3). The CUP allows you to embed ICAM TCI macroinstructions within your application code to execute the communications services desired.

Your CUP is provided with a number of imperative macroinstructions to link TCI functions. A set of transaction terminal tables (TTT) that reside in your program area are the principal means of this control. You may construct your own table or use the declarative macro MTABLE to construct, format, and fill in most parameters. Once the tables are defined, you use the MOPEN and LNEREQ macros to acquire the specific network and lines previously generated by your network definition. The MOPEN macro performs the same functions that the NETREQ macro does in other ICAM interfaces.

To attach or receive messages from the files/queues, two imperative macros (MREAD/MWRITE) are provided. Additional imperative macros MDEFER and MALERT are provided to control the enhancement features of TCI. Expansion of these macros produces an SVC interrupt. The tables are internally converted to message control table (MCT) formats and transferred to remote device handlers (RDH). The RDH then creates an appropriate communications physical input/output control packet (CPIOCP) and passes control to the communications control routine (CCR) described in the fundamentals of ICAM user guide, UP-8194 (current version). The CCR uses the CPIOCP to construct buffer control words and issues and services commands to and from the communications adapter hardware subsystem, UP-8273 (current version).

Additional imperative macroinstructions are available to display and alter network status. Optional features are immediate return line (IRL), output delivery notification, contingency notification, and batch mode processing.

Figure 1-1 shows the relationship among your program, the supervisor, and the ICAM elements. ICAM in this interface consists of the communications control area (CCA), the MREAD/MWRITE message staging, queueing and/or buffering (MUST), the communications network controller (CNC), and remote device handlers (RDH). Implied in this structure is the link to your CUP provided by the macroinstruction set.

Following is a summary of the principal features available in the TCI environment:

- Complete asynchronous and concurrent operational capability for your communication task.

- Automatic program scheduling for each incoming message as it arrives in the system. This permits you to examine the first fields of text without reading the message.

- First text fields of all available input messages are continuously under the surveillance of your program. This promotes efficient concurrent processing of transactions.

- Optional disk buffering (nonqueued) for both input and output messages. Output can be distributed to seven levels with three priority designations. Input and output disk buffering are automatic with transient TCI; output disk buffering is not available with resident TCI.

- Automatic date and time stamping (optional) of incoming messages without using the MPPS routines.

- Source and transaction number stamping (optional) without using the MPPS routines.

- Ability to retain unsolicited output to a terminal while a multiple message transaction is in progress between your program and the terminal operator.

- Automatic scheduling of your program with a delivery notice for each outgoing message after it has been transmitted. This feature may be dynamically controlled by your program.

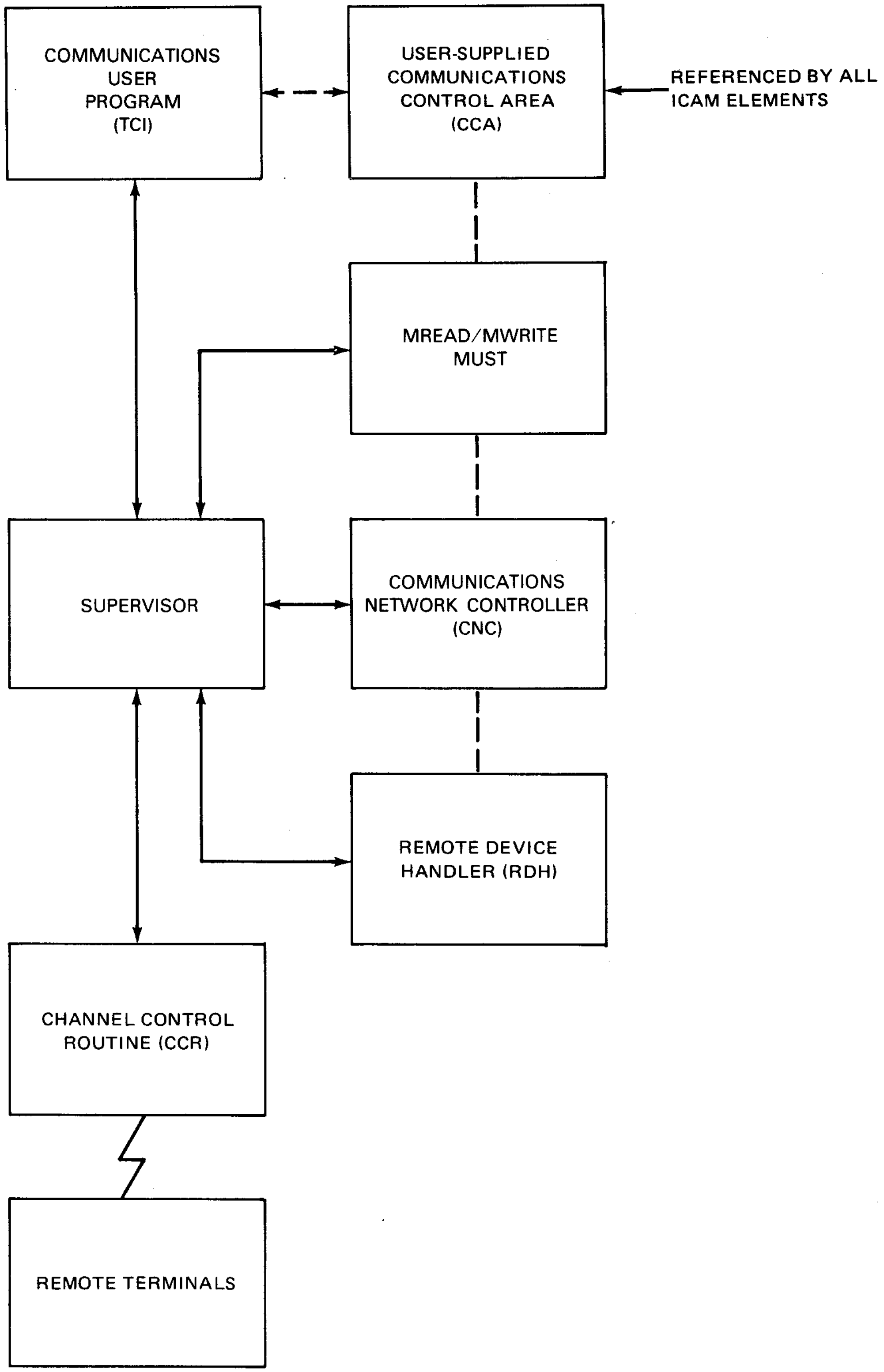

COMMUNICATIONS USER PROGRAM (TCI)
USER-SUPPLIED COMMUNICATIONS CONTROL AREA (CCA)
REFERENCED BY ALL ICAM ELEMENTS
MREAD/MWRITE MUST
SUPERVISOR
COMMUNICATIONS NETWORK CONTROLLER (CNC)
REMOTE DEVICE HANDLER (RDH)
CHANNEL CONTROL ROUTINE (CCR)
REMOTE TERMINALS

Figure 1—1. Transaction Control Interface (TCI)

- You may dynamically build and submit a multiple destination routing list with an outgoing message. This promotes efficient multiple routing of messages in a message switching or complex administrative environment.

- Direct user control of terminals and queued or buffered message priority levels via "hold" indicators in your program area.

- You may cancel an input message without a read operation.

- You have the ability to retransmit the last output message to a terminal without resubmitting the message.

With some limitations, TCI operates in an 8192-byte partition, which supports 10 UNISCOPE Display Terminals or SPERRY UNIVAC DCT 500 Data Communications Terminals.

1.2. TRANSIENT TCI

The transient TCI is composed of a series of program overlays that are under the control of a single root element that is resident in main storage. The functions associated with each overlay correspond to those functions that are normally identified with the processing of a message in a serial fashion. This procedure provides a small-scale ICAM that retains all of the desirable attributes of a resident ICAM, including reasonable device independence.

The transient mode of operation for the communications user program (CUP) provides communications support in an estimated 8K byte partition running under the supervisor as a symbiont. The design approach is oriented toward the 10-terminal, interactive-type user. Special techniques are employed to reduce message processing delays. A special loader is provided by ICAM so that an overlay may be loaded from disk with a single I/O request to the system access technique (SAT).

Transient TCI may be converted to resident TCI at CUP system generation time by selecting the appropriate configuration parameters. Total functional compatibility is retained between the two types of TCI so that a user may migrate to a resident mode without any reprogramming of his applications program. Full concurrent operational capability is maintained between ICAM and the user program.

No limit is placed upon the number of terminals per line; however, a transient user is limited to either one or two lines. The response time for a 2-line, 10-terminal (5 per line) configuration is identical to that of a 1-line, 10-terminal configuration.

The following restrictions apply to a transient TCI configuration and its associated operating environment:

- Only the UNISCOPE 100, UNISCOPE 200, DCT 475, and DCT 500 remote terminals are supported.
- Only a single terminal type in a specific configuration is supported.
- Only completed messages are transferred into and out of user-designated buffer areas. Message segmentation is not supported.
- Only the TCI interface in the transient mode is supported. No other communications activity (e.g., remote job entry) can operate concurrently with transient TCI.

  The size of the TCI user's input buffer (specified in input buffer prefix before issuance of MREAD) must always be the size of the maximum message he expects to receive, regardless of the actual size of the incoming message. (The size specified should be at least as large as MSGSZE specification on DISCFILE macro from CCA generation.)
- Transient MCP does not support automatic dialing or unattended answering. The availability of operator communication in transient mode is limited to marking specific terminals up and down.
- CCAs generated for transient TCI must specify the network buffer size in increments of 64 words (256 bytes) because of the sectorized disks. In addition, the number of network buffers is assumed to be one (any other value is ignored), and the threshold value must be zero.
- Transient ICAM users who have a 9600-baud direct connection line should specify the RETRY parameter with a decimal value of 6 for both input and output (RETRY=(6,6)).
- Transient TCI supports a maximum of 2 lines and a maximum of 10 terminals. ←
- Output messages sent to nonoperational terminals (i.e., nonexistent or not responding) cause transient TCI to be nonresponsive.

Also, the transient UNISCOPE remote device handler, after output, performs a specific poll for acknowledgment (ACK) of output and supplies a hexadecimal 10-byte buffer to receive the ACK. If the terminal operator presses the TRANSMIT key too quickly after the cursor comes back on output, the specific ACK may return more than just the ACK. In this case, the RDH will lose most of the input without notification to the user program. This loss can be avoided by telling the terminal operator to allow at least 1 second before pressing the TRANSMIT key.

When the CUP issues an MOPEN macro and the TCI disk file is not allocated or assigned in the job control stream, ICAM is cancelled with the error code X'461'. Control is not returned to the user at the MOPEN error return in this case.

2. Network Definition

2.1. GENERAL

The network definition macroinstructions are used in conjunction with the communications configuration parameters (COMMCT) to perform system generation (SYSGEN) of an ICAM network. (See system installation user guide/programmer reference, UP-8074 (current version) for ICAM SYSGEN procedures.) Defining a network means providing ICAM with:

- a description of the communications hardware (lines, terminals, and queues) in the network; and

- the ICAM services required to transfer and/or process the message data for a given communications application.

These network definition macroinstructions, in addition to linking the required software routines, construct a communications control area (CCA). A CCA contains all of the tables required to define and control a specific communications network configuration. Your programs can assemble CCAs tailored to your specific needs. The CCA also contains the necessary buffer pools to service the ICAM elements.

The network definition you specify for a basic ICAM system will, most often, include all the communications hardware present in the system; however, you can define any number of CCAs and, therefore, each CCA will probably include only a portion of the communications hardware in the system.

Multiple network (CCA) ICAM systems are intended primarily for installations that require multiprogramming of communications message processing programs. When multiple networks are defined for an ICAM system, they can be linked to ICAM by a request for the services of a particular communications network from a communications user program (CUP).

The loaded CCA is part of the ICAM module which is not in the user region and is uniquely assigned to your program. Alternatively, you can identify a CCA element on an SFT control card in your run stream. Job control will load this CCA at run time in a separate area of main storage and assign it to your program. When you terminate your program, the CCA area will be released to the system.

The macroinstructions required to generate a network CCA also contain parameters that are used exclusively by the SYSGEN facility. You may submit one or more CCA network definitions to a cataloged system run file and the system will automatically generate the required network tables and a loadable ICAM to support those network definitions.

A full CCA network definition is required by the transaction control interface (TCI) since it supports disk buffering. The following macroinstructions are required to generate this type of CCA:

- CCA

  Generates a control section

- BUFFERS

  Generates an activity request packet (buffer) pool and, if resident TCI, a network buffer pool

- LINE

  Generates a line control table

- TERM

  Generates a terminal and device control table

- DISCFILE

  Defines a disk file to be used for TCI and transient operation or disk queueing

- ENDCCA

  Marks the end of a CCA network definition

Collectively, these macros generate the CCA control section, an activity request packet (ARP) packet pool, one or more line control tables, and their associated terminal and device control tables. These macros must be assembled in a specific order. Further information regarding the use of these instructions and their parameters is provided in 2.2. The following order of macro presentation is required.

CCA

BUFFERS

$$\left\{\begin{array}{l}\text{LINE}_1\\ \text{TERM}_1\\ \text{TERM}_n\end{array}\right\}\quad \underline{\text{Line Group 1}}$$

$$\left\{\begin{array}{l}\text{LINE}_n\\ \text{TERM}_1\\ \text{TERM}_n\end{array}\right\}\quad \underline{\text{Line Group n}}$$

DISCFILE

ENDCCA

The following macros are not required in TCI, but are optional:

- PRCS

  Generates a process file table with queue control tables

- DLIST

  Generates a list of multiple destinations

- JRNFILE

  Generates a log or history file DTP and associated control fields

The CCA network generation for TCI is primarily the same as for STDMCP except for the following:

- TCI does not require process files.

- TCI requires queue tables only when in resident mode. That is, if disk-buffered output is desired, as it is in transient mode, no queue tables should be generated. Disk-buffered output is not supported in resident mode.

- TCI requires you to assign a disk file to be used exclusively by ICAM. This file is defined by the DISCFILE macro. The network summary will indicate the file allocation required for the network. The job stream for your CUP should contain the following control cards to assign the file for ICAM's use:

  ```
  // DVC logical-unit-number

  // VOL volume-serial-number

  // LBL file-identifier, volume-serial-number

  // LFD file-name,O,INIT
  ```

  The control stream you would need to allocate the file is:

  ```
  // DVC logical-unit-number

  // DVC volume-serial-number

  // EXT ST,C,O,BLK,(256,n)

  // LBL file-identifier,volume-serial-number

  // LFD file-name,O,INIT
  ```

The file need only be allocated once unless the file requirements change. The disk file size for the EXT job control card is in blocks and is based upon the following formula:

$$n = \left[\left[\frac{\left(\frac{a}{b - f} \right) + 255}{256} \right] \mathrm{x}\; c \;\mathrm{x}\; d \right] + e$$

where:

a

Is maximum message size (input and output) in bytes.

b

Is CCA data buffer size in bytes.

c

Is number of disk slots per terminal. If disk buffering output is specified, the number of slots is one for output and eight for input.

d

Is number of terminals in the network.

e

Is 160 for transient TCI and 0 for resident TCI.

f

Is one of the following:

- 68 if disk queueing is not used;
- 80 if disk queueing is used; or
- 92 if disk queueing and journaling are used.

2.2. NETWORK DEFINITION MACROINSTRUCTIONS

The macroinstructions necessary to define your communications network are described in alphabetic order (2.2.1 through 2.2.8). Examples of typical networks are shown in 2.3.

BUFFERS

2.2.1. Data Buffer Specifications (BUFFERS)

Function:

Defines and generates up to two buffer pools that are contained in the CCA. The two pools are an activity request packet (ARP) pool for ICAM use and a network data buffer pool for staging and queueing message text. The first three positional parameters define the network data buffer pool. The number of ARPs in the activity request packet pool is defined by the ARP keyword parameter. Each pool generated has its own control table.

The network data buffer pool consists of an explicit number of buffers and a data offset for network buffers used for editing purposes by ICAM elements. This instruction appears only once in a network definition.

Format:

LABEL	ΔOPERATIONΔ	OPERAND
not used	BUFFERS	num,size,thresh, ARP=integer $\left[,\text{DICE}=\left\{\begin{matrix}\text{n}\\ \text{1}\end{matrix}\right\}\right]$ [,STAT=YES]

Parameters:

num

Specifies, in decimal, the number of buffers to be generated in the network data buffer pool. Transient TCI assumes a value of 1 and ignores all others.

Number estimation is much more difficult than size, since it depends on application, traffic per line, and number of lines. ICAM gives output priority over input to minimize the number of buffers required. Therefore, you can best estimate by the number and size of output messages that can be in the system at any one time. Buffers are never assigned to an inactive line; they are only allocated after input is flowing.

size

Specifies the size, in words, of a network data buffer. Used only if positional parameter 1 is specified.

Main storage queueing requires a minimum of 25 words. This is based on a 20-word header and 21 possible MPPS blanks. Using the 20-word header as a start, estimate size so that 90 percent of the input messages fit in only one buffer. If your normal message is 10 words long and you're using MPPS, your buffer size would be 35 words long. It is not an ICAM requirement that a message fit within a single buffer, but it significantly improves throughput.

Disk queueing requires a minimum of 26 words. The disk queueing message header is 20 words in length.

A network with the log/journal feature requires a minimum of 29 words while the message header is 23 words in length.

For transient TCI, the size specified should be an increment of 64 and must agree with the value specified on the MSGSIZE keyword parameter associated with the DISCFILE macro.

thresh

A decimal number that specifies a threshold value for the buffer pool. When detected, it causes network activity to be degraded. This parameter is not supported in transient TCI.

To protect your network, a 10 to 15 percent threshold value is suggested. ICAM will shut down all input and send output until the number of available buffers is greater than this value.

ARP=integer

A comparative (relative) threshold value is generated automatically if positional parameter 3 is specified for network data buffers.

This parameter, if specified, defines the number of ARPs to be generated in a pool.

ARP packets are 10-word activity request packets used by ICAM for many purposes including CPIOCPs, MCTs, and schedule packets. The number required in a network varies, depending on the number of lines and type of network. The suggested minimum number to be configured is:

1. Resident TCI

   Seven per line plus four for the network plus the number of special functions listed.

2. Transient TCI

   Five per line plus four per network

3. Special Functions

   If using terminal queueing of UNISCOPE 100 multidrop lines, add one ARP per terminal. If issuing a long series of contiguous IRL calls to ICAM, add one ARP for every SVC over three in the series.

DICE=n

Specifies the maximum number of output DICE functions that require time fills. The default value is 1. This keyword generates n times 12 bytes of main storage for the network buffer. It is only specified when TRANSNT=YES is specified in the CCA macro.

STAT=YES

Generates separate buffer pool and ARP pool statistical areas that keep track of buffer pool and ARP pool use. The format of each area is seven words of control information followed by one word containing the number of deferred requests and one word containing the number of rejected requests. Requests are deferred and rejected because few or no buffers or ARPs are available. Following these words is the statistics area which has one word for every buffer or ARP in the pool. These give the number of buffers or ARPs still available after a buffer or ARP is used. For example, word 12 gives the number of times 3 buffers/ARPs were available (12 minus the first 9 words); word 28 gives the number of times 19 buffers/ARPs were available (28 minus the first 9 words). The format looks like this:

Word		
0	control information	
1	no. of buffers/ARPs generated	no. of buffers/ARPs currently available
2		
3		
4	control information	
5		
6	no. of deferred requests	
7	no. of rejected requests	
8	no. of times 0 buffers/ARPs available	
9	no. of times 1 buffer/ARP available	
10	no. of times 2 buffers/ARPs available	
11		
n	no. of times n–9 buffers/ARPs available	

The buffer and ARP statistics are reset to zero each time you reload ICAM. Whenever a buffer or ARP is allocated, ICAM accesses the statistics storage counter corresponding to the number of remaining unused buffers or ARPs available to ICAM and adds one to that counter. Each counter can hold a maximum value of 2,147,483,648 (2^{31}) before wraparound.

Figure 2–1 shows a buffer pool statistical area. Message traffic in this test system was particularly heavy, as can be seen by the tenth word in the poll that shows there were no buffers available 616 times (268_{16}). The ninth word shows that 562 times (232_{16}) buffer requests were rejected. If this is a normal message load, the user of this communications system should regenerate ICAM and declare more buffers because he's overloading the system.

Both the buffer pool and ARP pool use the same DSECT for the control table. For dump analysis, the location of the particular pool can be obtained by adding the value found in register 2 (the address of the CCA) to the value of TN#CARP for the ARP pool and to the value of TN#CDTAT for the data buffer pool. Address pointers to the buffer pool control tables are generated in the CCA control section.

NOTE:

The journal utility (JUST) can produce a network buffer pool activity report. See Appendix B for more information.

Examples:

```
      1        10     16
    ----------------------------
1.  |          BUFFERS ARP=2Ø
2.  |          BUFFERS 2,82,ARP=12
3.  |          BUFFERS 2Ø,256,3,ARP=1ØØ
4.  |          BUFFERS 5,64,Ø,ARP=2Ø
5.  |          BUFFERS 5,64,Ø,ARP=15,STAT=YES
```

1. Generates an activity request pool with 20 buffers. No network buffer pool is generated; a single control table is generated.

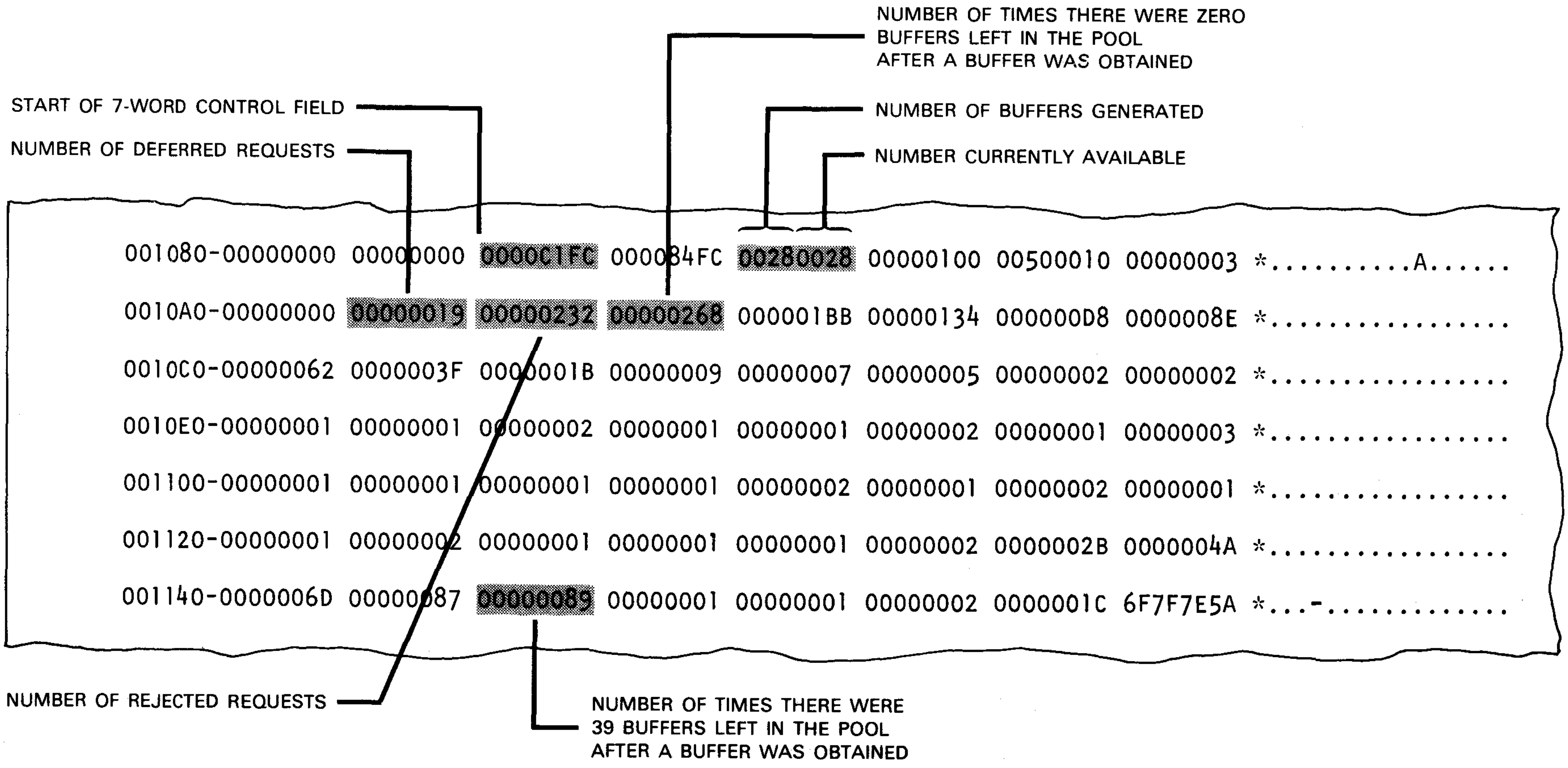
2. Generates two buffer pools: one for activity request packets and one for network buffers. A separate control table is generated for each pool. Threshold values for each pool will be zero. There are two network data buffers, 82 words in length, and there are 12 activity request packets.

3. Generates two buffer pools, one for activity request packets and one for network buffers. A separate control table is generated for each. The threshold value for the network buffers will be 3. There are 20 network buffers of 256 words each. There are 100 activity requet packets.

4. Generates two buffer pools with separate control tables. The network buffer pool has five 64-word buffers with a threshold value of zero. The ARP pool has 20 packets.

   Address pointers to the buffer pool control tables are generated in the CCA control section. Address pointers with a value of zero are generated for missing buffer pool specifications. All buffer pools are generated on a full-word boundary.

5. Provides a statistics area of the following format for each buffer pool.



```
001080-00000000 00000000 0000C1FC 000084FC 00280028 00000100 00500010 00000003 *...........A......
0010A0-00000000 00000019 00000232 00000268 000001BB 00000134 000000D8 0000008E *.................
0010C0-00000062 0000003F 0000001B 00000009 00000007 00000005 00000002 00000002 *.................
0010E0-00000001 00000001 00000002 00000001 00000001 00000002 00000001 00000003 *.................
001100-00000001 00000001 00000001 00000001 00000002 00000001 00000002 00000001 *.................
001120-00000001 00000002 00000001 00000001 00000001 00000002 0000002B 0000004A *.................
001140-0000006D 00000087 00000089 00000001 00000001 00000002 0000001C 6F7F7E5A *...-.............
```

START OF 7-WORD CONTROL FIELD

NUMBER OF DEFERRED REQUESTS

NUMBER OF TIMES THERE WERE ZERO
BUFFERS LEFT IN THE POOL
AFTER A BUFFER WAS OBTAINED

NUMBER OF BUFFERS GENERATED

NUMBER CURRENTLY AVAILABLE

NUMBER OF REJECTED REQUESTS

NUMBER OF TIMES THERE WERE
39 BUFFERS LEFT IN THE POOL
AFTER A BUFFER WAS OBTAINED

Figure 2—1. Statistics Area for Data Buffers or ARPs

CCA

2.2.2. Begin Network Specifications (CCA)

Function:

Identifies the beginning of a network specification and generally defines the type of network being configured. This macro also initiates ICAM generation.

Format:

LABEL	△OPERATION△	OPERAND
network-name	CCA	TYPE=(TCI) [,PASSWORD=password] [,TRANSNT=YES] [,CCAID=identifier] [,FEATURES=([{TRACEMIN}][,AUTOBUF][,OPCOM] {TRACEMAX} {TRACEDEV}] [,TOPRI][,BASIC][,OUTDELV] [,MULTIACT][,RESTART][,SEGMENTS])] [,JRNINIT=(x,[,RESTART][,PERF] [,STAT][,LOG][,JOURN][,ODNR])] [,GAWAKE={YES} {NO}]

Label:

network-name
A label up to four characters in length that uniquely identifies the network being configured. It is this name that is submitted with the NETREQ and NETREL macroinstructions.

Parameters:

TYPE=(TCI)
Defines the type of network being configured and the communications services that may be required of ICAM.

PASSWORD=password
A label up to eight characters in length that can be used to further identify the network being configured. If a password is assigned to a network, it must be identified in all requests for that network (MOPEN macroinstruction). Otherwise, the network will not be initialized and will not be accessible to the user program.

If omitted, a password will not be associated with the network.

TRANSNT=YES

Identifies the network is being generated for a transient configuration.

CCAID=identifier

A string of up to eight characters that provides a visible identifier for the CCA assembly listing on each page and in the generation summary report at the end of the listing.

FEATURES=

Used only by SYSGEN. Does not generate any code.

where:

TRACEMIN

Causes minimum trace module to be included in communications physical IOCS routine (CPIOCS). Do not use for lines greater than 19 kilobytes.

TRACEMAX

Causes maximum trace module to be included in CPIOCS. Do not use for lines greater than 19 kilobytes.

TRACEDEV

Extends trace module for CPIOCS (see communications physical interface, fundamentals of ICAM user guide, UP-8194 (current version)). Do not use for lines greater than 19 kilobytes.

AUTOBUF

Causes automatic line or data buffering to be included. Not required for remote device handlers supplied by Sperry Univac.

OPCOM

Causes operator communications island code for unsolicited type-ins to be included.

TOPRI

Causes the top priority feature to be included in message queueing routines. This parameter is not supported in transient TCI.

BASIC

Selects a basic TTY or UNISCOPE RDH. Selection of this parameter precludes unattended answering in the LINE macro. This parameter is not supported in transient TCI.

OUTDELV

Causes the output delivery notification feature for auxiliary device errors to be included.

MULTIACT

Selects a multiple priority version of activity control. This parameter is not supported in transient TCI.

RESTART

Causes selection of the disk queueing routine that makes use of the necessary checkpoints to enable a warm restart capability. This parameter is not supported in transient TCI.

SEGMENTS

Selects an enhanced MUST module that provides for segmented message traffic. This parameter is not supported in transient TCI.

JRNINIT=

where:

x

Determines the type of records collected:

1	=	ODNR,LOG,JOURN,RESTART
2	=	PERF,ODNR,LOG,JOURN,RESTART
3	=	All options

RESTART

Causes restart data to be recorded.

PERF

Causes performance records to be recorded.

STAT

Causes operational statistics to be recorded.

LOG

Causes message headers to be recorded.

JOURN

Causes complete message segments to be recorded.

ODNR

Causes output delivery notification records to be recorded.

This parameter is not supported in transient TCI.

$\text{GAWAKE}=\left\{\begin{matrix}\text{YES}\\ \text{NO}\end{matrix}\right\}$

Specifies that the GAWAKE feature of ICAM is or is not to be configured. This feature requires that module TZ$GAW01 be included. See fundamentals of ICAM user guide, UP-8194 (current version). This parameter is not supported in transient TCI.

Examples:

```
1        10   16
-----------------------------------------
NET1     CCA  TYPE=(TCI),PASSWORD=TCIUSER
              CCAID=NET1
NET2     CCA  TYPE=(TCI),FEATURES=(TRACEMAX)
NET3     CCA  TYPE=(TCI),TRANSNT=YES
```

DISCFILE

2.2.3. Disk File Definition (DISCFILE)

Function:

Defines a disk file for TCI and transient operation or a disk file for disk queueing. For nonqueued disk files (disk buffering), the parameter MSGSIZE=n is used in conjunction with other macros to generate one SAT DTF for the file and one SAT partition control appendage (PCA) for message buffering and one PCA if in transient mode. Also generates a register save area and work areas for SAT I/O. For queued disk files, the parameter FILEDIV=n is used in conjunction with other macros to generate one SAT DTF, two SAT PCAs, and the necessary file control information for disk queueing. This macro must immediately precede the ENDCCA macro.

Format:

LABEL	△OPERATION△	OPERAND
file-name	DISCFILE	{MSGSIZE=n FILEDIV=n[,VERIFY=YES]}

Label:

file-name

A label up to seven characters in length uniquely identifying a disk file. Must be identical to the file name on the // LFD control card in the CUP job control stream.

Parameters:

MSGSIZE=n

Specifies the maximum size of input or output message processed by ICAM. n is in bytes and is used to establish SAT disk addresses based on the number of terminals, number of 256-byte sectors per network data buffer, and the number of slots per terminal for buffering messages. The size in blocks is printed in the network summary.

For transient TCI, the number of bytes specified must agree with the buffer size specified in positional parameter 2 of the BUFFER macro.

FILEDIV=n

Specifies the percentage of disk space that disk queueing should allocate for the control area on disk. Disk queueing requires one 256-byte block (sector) for every 2000 disk buffers in the file, one sector for each queue using the file, and one sector for descriptor records.

NOTE:

The percentage specified must be at least one track of the file.

VERIFY=YES

Specifies to disk queueing that the SAT disk file should be generated with the verification option in the partition control appendage (PCA).

NOTES:

1. _The FILEDIV parameter must be specified for all disk-queued files. The file name specified must correspond to the file name specified on the LFD statement in the job control stream for assigning and allocating a SAT disk file._

Assignment		_Allocation_	
// DVC	_logical-unit-number_	_// DVC_	_logical-unit-number_
// VOL	_volume-serial-no._	_// VOL_	_volume-serial-no._
// LBL	_file-identifier_	_// EXT_	_ST,C,,CYL,n*_
// LFD	_filename_	_// LBL_	_file-identifier_
		// LFD	_filename_

*where n is the number of cylinders assigned to the file. Refer to the job control user guide, UP-8065 (current version) for more detailed information concerning parameters.

2. _Assuming that the buffer size is specified at 64 words, this gives 1 sector per buffer, and the file allocated is 5000 sectors with 10 queues. The required number of sectors for the control area would be ((5000//2000)=>3) + (10) + 1 = 14 sectors. Thus, the percentage for control space would be specified as 1. If there were only 140 sectors in the file, the calculation would be ((140//2000)=>1) + (10) + 1 = 13 and the percentage in this case is specified as 10 percent._

DLIST

2.2.4. Distribution List (DLIST)

Function:

Collectively identifies a list of destinations as a single destination. This macroinstruction enables a single user-specified instruction to denote a list of terminals, or LOCAPs, and the destinations for a message. An MWRITE instruction that references DLIST is equivalent to a group of MWRITE instructions that reference every destination in the DLIST, that is, where a separate transmission is performed for each terminal, DLIST, or LOCAP. Note that DLISTs cannot be nested more than once (i.e., a DLIST referenced by a DLIST cannot reference a third DLIST). A DLIST should not reference itself.

Format:

LABEL	△OPERATION△	OPERAND
DLIST-name	DLIST	dest-name$_1$,dest-name$_2$,...dest-name$_n$

Label:

DLIST-name

Is a label up to four characters long that uniquely identifies the subject distribution list.

Parameters:

dest-name (1 through n)

Identifies the terminals, process files, DLISTs, or LOCAPs as the destinations for a message. At least two destinations must be specified in each DLIST.

NOTES:

1. *A DLIST macroinstruction must be included even when executing only a user-defined DLIST.*

2. *Destination names must be unique in all DLIST macroinstructions in which either a PUTCP or ROUTE macroinstruction is directed. Thus, a destination name can appear only once in a DLIST or a group of DLISTs that are in the scope of either a PUTCP or ROUTE macroinstruction.*

3. *Unpredictable results will occur to an output auxiliary device not configured.*

Examples:

```
1        10    16

DLT1     DLIST TERI,TER2,DLT2,PFL1,PFL2,DLT3
DLT2     DLIST TER3,PFL3,LOC1,LOC2,TER3
DLT3     DLIST TER4,LOC3
```

2.2.5. End Network Specifications (ENDCCA)

Function:

Designates the end of the network specifications.

Format:

LABEL	△OPERATION△	OPERAND
not used	ENDCCA	not used

2.2.6. Journal File (JRNFILE)

Function:

Specifies that journalling is to be a feature of the network and indicates the type, size, and number of buffers to be allotted. Creates a DTF, staging areas, and associated statistical and control fields.

Format:

LABEL	△OPERATION△	OPERAND
filename	JRNFILE	$\text{TYPE=}\left(\text{SAT,}\begin{Bmatrix}\text{TAPE}\\ \text{DISC}\end{Bmatrix}\right)$ [,BUFF=(number,size,thresh,)]

Label:

filename
The logical file name. This label is required and serves to uniquely identify the output file. It is referenced as an operand by the MPPS macros JOURN and LOG. It must be identical to the file name on the // LFD control card.

Parameters:

TYPE=
Identifies the type of I/O interface required.

where:

SAT,TAPE
The OS/3 tape system access technique (TSAT) is used for this file. Standard labels are assumed.

SAT,DISC
The OS/3 disk system access technique (SAT) is used for this file. This is the default value. Standard labels are assumed.

BUFF=
Identifies the number, size, and threshold values for the journal staging areas.

where:

number

Is a decimal number specifying the number of staging areas associated with the file. The permitted values are from 3 to 10. The default value is 3.

size

Is a decimal number specifying the number of bytes in each staging area. For history files, the staging area should be large enough to contain two or more network buffers. For log files, some multiple of the message header segment should be the basis for the calculation. The journal routines add 9 bytes to each record and 24 bytes to each block output. If size is not specified, the value will be algorithmically determined to optimize throughput and minimize the change of stage area saturation.

thresh

Is a decimal number specifying the theshold value. When the number of staging areas available is less than the threshold value, the condition will be made known to the caller of the journal routine. The permitted values are from 0 to the number of staging areas specified. The default is 1.

The journal file is defined in the user's CUP job control language (JCL) as follows:

Label	Definition	Example
DVC	Logical unit no.	// DVC 60
VOL	Volume serial no.	// VOL VSN001
EXT	Defines a new file	// EXT ST,,1,CYL,50
LBL	Physical file identifier	// LBL JOURNAL001,VSN001
LFD	Logical file name referenced by the JRNFILE macro of ICAM	// LFD filename

2.2.7. Line Characteristics (LINE)

Function:

Identifies the characteristics of each line in a communications network. One instruction must appear for each line and be followed immediately by the terminal (TERM) and queue (QUEUE) macroinstructions associated with a line, if applicable. Default values for specific devices are described in the system installation user guide/programmer reference, UP-8074 (current version).

Format:

```
LABEL        △OPERATION△     OPERAND
line-name       LINE         DEVICE= ⎧ ⎛TTY⎡,⎛28⎞⎤⎞                                       ⎫
                                     ⎪ ⎜   ⎢ ⎨32⎬⎥⎟                                       ⎪
                                     ⎪ ⎜   ⎢ ⎨33⎬⎥⎟                                       ⎪
                                     ⎪ ⎜   ⎢ ⎨35⎬⎥⎟                                       ⎪
                                     ⎪ ⎝   ⎣ ⎝37⎠⎦⎠                                       ⎪
                                     ⎪ (DCT475)                                           ⎪
                                     ⎪ ⎛DCT500⎡,⎧TTY ⎫⎤⎞                                  ⎪
                                     ⎪ ⎝      ⎣ ⎩AUTO⎭⎦⎠                                  ⎪
                                     ⎪ (DCT1000,BATCH)                                    ⎪
                                     ⎨ (DCT1000,INTER[,UNISCOPE])                         ⎬
                                     ⎪ (RDHA[,RDHB][,RDHC][,RDHD])                        ⎪
                                     ⎪ (UNISCOPE[,KA])                                    ⎪
                                     ⎪ (1004)                                             ⎪
                                     ⎪ (9200)                                             ⎪
                                     ⎪ (9300)                                             ⎪
                                     ⎪ (DCT2000,line-buffer-length)                       ⎪
                                     ⎪ ⎛BSC,⎧line-buffer-length⎫ ⎞                        ⎪
                                     ⎪ ⎜    ⎩TOGGLE            ⎭ ⎟                        ⎪
                                     ⎪ ⎜   ,⎧ASCII   ⎫           ⎟                        ⎪
                                     ⎪ ⎜    ⎨EBCDIC  ⎬           ⎟                        ⎪
                                     ⎩ ⎝    ⎩TRANSCOD⎭           ⎠                        ⎭
                             ,TYPE=⎛line-speed[,FULL][,AUTO][,UNAT][,SWCH]⎞
                                   ⎝[,SYNC][,FLDQ][,CRTS][,NIDLE]         ⎠              ←
                             [,CALL=remote-device-phone-number]
                             [,ID=ca-port-number]
                             ⎡,INPUT=⎛⎧YES ⎫[,no-of-blanks][,filename]⎞⎤
                             ⎣       ⎝⎩name⎭                          ⎠⎦                 ←
                             [,DIALER=(port-number[,EON])]
                             [,LBL=size-in-words]
                             [,MPPS=(label[number-of-blanks])]
                             ⎡,RECONECT=⎧YES   ⎫⎤
                             ⎣          ⎩QUEUES⎭⎦
```

```
LABEL          △OPERATION△      OPERAND

                                [,RETRY=(n,m)]
                                [,RDLHQ=YES]
                                ⎡,STATS=⎰YES⎱⎤
                                ⎣       ⎱NO ⎰⎦
                                [,TIMEOUT=(n,m)]
                                ⎡,XLATE=⎛⎧YES           ⎫ ⎧YES           ⎫⎞⎤
                                ⎢       ⎜⎨NO            ⎬,⎨NO            ⎬⎟⎥
                                ⎣       ⎝⎩labeli,idi    ⎭ ⎩labelo,ido    ⎭⎠⎦
                                ⎡,LOW=⎰filename⎱⎤ ⎡,MEDIUM=⎰filename⎱⎤
                                ⎣     ⎱MAIN    ⎰⎦ ⎣        ⎱MAIN    ⎰⎦
                                ⎡,HIGH=⎰filename⎱⎤
                                ⎣      ⎱MAIN    ⎰⎦
```

Label:

line-name
A label up to four characters in length that identifies the line being defined.

Parameters:

DEVICE=
Identifies the types of devices connected on the line being defined and their mode of operation, if applicable. The specification in each keyword parameter must be coded in the order shown:

where:

TTY
The default condition assumes an ASCII teletypewriter device (TTY,35).

TTY, 28, or 32
Identifies a BAUDOT teletypewriter device.

TTY, 33, 35, or 37
Identifies an ASCII teletypewriter device.

DCT475
Identifies a DCT 475 terminal operating in teletypewriter mode.

DCT5ØØ,TTY
Identifies a DCT 500 series (DCT 500 and DCT 524) terminal operating in the teletypewriter mode.

DCT5ØØ,AUTO
Identifies one or more DCT 500 series terminals operating in the automatic mode. In AUTO mode, DCT 500 and DCT 524 terminals can be mixed on a line.

DCT1000,BATCH

Identifies one DCT 1000 terminal operating in the batch processing mode.

DCT1000,INTER

Identifies one or more DCT 1000 terminals operating in the interactive mode.

DCT1000,INTER,UNISCOPE

Identifies one or more DCT 1000 terminals operating in interactive mode with any of the following terminals: UNISCOPE 100/200, UTS 400 operating in UNISCOPE mode, UTS 400 operating in UTS 400 native mode, and UTS 400 Text Editor (TE).

NOTES:

1. *The forms DEVICE=(DCT1000,INTER,U100) and DEVICE=(DCT1000, INTER,U200) previously used are also acceptable.*

2. *Only the DCT500, AUTO, UNISCOPE 100, and UNISCOPE 200 device types are supported in transient TCI.*

RDHA[,RDHB][,RDHC][,RDHD]

Specifies the inclusion of one to four user-written remote device handlers in the ICAM load module. You should assign one of these labels to your handler, and ICAM automatially includes this RDH object module from SG$OBJ on the software release disk pack in the load module.

UNISCOPE

Identifies one or more of the following terminals:

UNISCOPE 100
UNISCOPE 200
UTS 400 (operating in UNISCOPE mode)
UTS 400 (operating in UTS 400 native mode)
UTS 400 Text Editor (TE)

NOTE:

The forms DEVICE=(U100) and DEVICE=(U200) previously used are also acceptable.

UNISCOPE,KA

Identifies one or more UTS 400 terminals that utilize Katakana/English keyboards. Note that this operand is not applicable to UTS 400 Text Editor (TE) or UTS 400 running in the UNISCOPE native mode.

1004

Identifies a 1004 card processor operating under control of an RMS1 plugboard.

```
92ØØ
```

Identifies a 9200 data processing system (REM-1) that is simulating a 1004 card processor operating under the control of an RMS1 plugboard.

```
93ØØ
```

Identifies a 9300 data processing system (REM-1) that is simulating a 1004 card processor operating under the control of an RMS1 plugboard.

```
DCT2ØØØ,line-buffer-length
```

Identifies a DCT 2000 terminal with a buffer length as specified in subparameter 2. Line buffer length is usually 84 or 132 bytes, with 84 being the default value. (The count for the TERM macro DEVICE parameter will be 80 or 128, with 80 being the default.)

```
BSC,{line-buffer-length},ASCII
    {TOGGLE            }
```

Identifies a binary synchronous controlled device. All code and text are translated to ASCII before transmission and from ASCII to EBCDIC after receipt. TOGGLE identifies a line with toggled buffers where length is based on line speed. Default is (BSC, 216, EBCDIC). Line buffer length is calculated as follows:

2m + mn for nontransparent mode

4m + mn for transparent mode

where:

m = number of records per block

n = number of data characters per record

```
BSC,{line-buffer-length},EBCDIC
    {TOGGLE            }
```

Identifies a binary synchronous controlled device. All code and text are in EBCDIC code.

```
BSC,{line-buffer-length},TRANSCOD
    {TOGGLE            }
```

Identifies a binary synchronous controlled device. All code and text are in TRANSCODE.

TYPE=

Identifies the characteristics of the line, most of which are directly dependent upon the hardware connected on the line. Line speed is the only required parameter; the remaining subparameters are optional and may be specified in any order.

where:

line-speed

Is a decimal number that identifies the baud rate at which data is exchanged between the terminal devices connected to this line and ICAM. This rate is a function of the communications line.

FULL

Indicates that the line is capable of full-duplex operation. Note that the remote device handlers of ICAM do not operate the terminals in the full-duplex mode, but can take advantage of the full-duplex operation to decrease turnaround time.

If omitted, a half-duplex line is assumed.

AUTO

Indicates that the line is equipped for automatic dialing. Specify only if subparameter SWCH is used. Autodial configurations must be defined first in the CCA macro (AUTOBUFF parameter). If multiple autodial lines are being used, they must be grouped contiguously as the first lines in the CCA. If the autodial lines are not defined first in the CCA, an error will occur in the assembly of the ICAM module. This parameter is not supported in transient TCI.

UNAT

Indicates that the computer automatically answers incoming calls (unattended answering) from a remote terminal via the swiched telephone network under program control without the need for operator intervention. The ID keyword and subparameter SWCH must be specified. This parameter is not supported in transient TCI.

This parameter may not be configured when the FEATURE=BASIC operand of the CCA macro is specified.

SWCH

Indicates that the line is a switched line (needs to be dialed, either automatically or manually).

If omitted, a private (dedicated) line is assumed and subparameters AUTO and UNAT are not required. If SWCH is specified, select subparameter AUTO or UNAT unless the host processor operator will manually dial the remote device. In this case, do not specify subparameter AUTO or UNAT.

SYNC

Indicates that the line hardware is set for the synchronous transmission of data.

If omitted, asynchronous transmission is assumed.

FLDQ

Indicates that full-duplex line queueing is desired. For standard ICAM RDHs, this feature is applicable to NTR and ILA disciplines; for other RDHs, omit this subparameter.

CRTS

Indicates the request-to-send signal is cleared in output/output sequences of messages conditions. This inhibits the sync character from being sent.

NIDLE

Indicates no line turn-around function is performed after each input message completion. If omitted, an automatic-line-around is performed in each input completion.

CALL=remote-device-phone-number

Is a numeric or alphanumeric telephone number that may be used to dial a terminal. This parameter must be specified if the AUTO subparameter is specified in the TYPE keyword.

ID=ca-port-number

A decimal number that is required for dedicated or switched lines that operate with unattended answering.

Values of 4 through 15 can be specified for the first communications adapter (CA) and 20 through 31 if a second CA is utilized.

INPUT=

where:

YES

Specifies that an input queue is desired for any terminal on this line. A GETCP addresses the terminal file name specified in subparameter 3 if YES is specified.

name

Is the 1- to 4-character symbolic name in the label field of the PRCS, MPPS, or TERM macro. This is the name the user should address on his GETCP calls to retrieve input messages from a terminal.

no-of-blanks

Is a decimal vlaue from 1 to 255 that indicates the number of bytes to be reserved in front of the input message for MPPS or user insertion of data.

filename

Is the 1- to 7-character DISCIFLE name of a disk file associated with an internal input queue (QCT) generated by the TERM macro. A file name should only be specified when YES is specified.

NOTE:

This keyword allows input from any terminal on this line to be directed to an input queue, a PRCS, or another terminal as an output message.

DIALER=(port-number[,EON])

A decimal number is required if automatic dialing is utilized. See values specified in the keyword parameter ID.

EON is an optional end of number subparameter that must be used if required by the automatic calling unit (ACU). This parameter is not supported in transient TCI.

LBL=size-in-words

The LBL parameter specifies the line buffer length to be used. It can be specified for all devices except the 1004 and 9200/9300 systems, DCT 2000, and binary synchronous controlled devices where TOGGLE (see DEVICE=BSC,TOGGLE) is not specified.

The length specified is a decimal value in words (one word equals four bytes). This parameter is not supported in transient TCI.

If this parameter is not specified, the following default values are generated:

Line Speed (maximum Baud rate)	Default Value (words)
300	4
2400	5
4800	10
9600	20

MPPS=

Identifies the MPPS to be used to process the message header data transmitted and received on the line, and the number of blank character positions ICAM should leave in each message header for MPPS data insertions (such as date and time stamping). More than one line may use the same MPPS. This parameter is not supported in transient TCI.

where:

label

Identifies the 1-to 4-character name of the MPPS (if used) to be used to process the message header data on a line, as defined in an MPPS MPSTART macroinstruction.

number-of-blanks

Specifies the number of blank character positions ICAM is to leave in each message header for MPPS data insertions.

The number of blanks applies only to incoming messages; i.e., the input data is offset by the specified number of characters. If MPPS is to do time, date, or sequence out count insertion on an output message that was created via a PUTCP, you must supply the correct number of blanks in front of the regular text or DICE commands.

If omitted, no blank character positions are reserved for MPPS data insertions. If data insertions are attempted by the MPPS, they are blocked and appropriate error bits are set in the message header prefix for subsequent error processing by the MPPS.

RECONECT=$\left\{\begin{array}{l}\text{YES}\\ \text{QUEUES}\end{array}\right\}$

Indicates user wants immediate main storage resident reconnection of unattended (call in) lines when original caller disconnects. RECONECT=YES means reconnection with output queues kept intact, which means next caller will receive queued output. RECONECT=QUEUES means that all output queues, including INTERCEPT, will be purged before reconnection.

There is no CUP or console notification of the reconnection (console will be notified of reconnection failure with LINE DOWN message); therefore, CUPs such as IMS will not be able to send normal READY messages except to the original caller.

This feature requires object module TZ$MCTI1. This parameter is not supported in transient TCI.

RETRY=

Specifies the number of error retries that the RDH is to execute before error notification; e.g., parity errors and time-out.

where:

n

Is a decimal value 1 to 255 for input.

m

Is a decimal value 1 to 255 for output.

NOTES:

1. *For binary synchronous communication (BSC) devices, input retries pertain only the host bidding for the line. All other retries are covered by the value specified for output.*
2. *For DCT 2000 terminals, input and output retries are determined from the value specified in the output parameter.*

If this parameter is not specified, the following default values are automatically generated:

Device	Input	Output
BSC	6	4
UNISCOPE/DCT 1000	6	4
DCT 500/TTY	0	0
DCT 2000	-	4
1004/9200/9300	6	4

RDHLQ=YES

Provides the necessary code allowing a message to be sent to a terminal while a previous message sent to another terminal on this multisation line is being transferred to its auxiliary device.

Only used with field-written remote device handlers supporting multistation lines and terminals on those lines incorporating auxiliary devices. When specified, terminal queueing must be specified.

STATS={YES NO}

Indicates whether terminal statistics are to be accumulated for the line and used by the journal utility. If STATS=NO, or the parameter is omitted, no statistics will be accumulated. This parameter is not supported in transient TCI. The statistics are kept for each terminal on the line. The TN#TSTAT field of each terminal control table (covered by the TN#TCT dummy control section) gives the relative address of the statistics area for that terminal. The format of the 3-word area is:

	Byte			
Word	0	1	2	3
1	no. of messages received		no. of input retransmission requests	
2	no. of messages transmitted		no. of output retransmission requests	
3	no. of polls sent		no. of no-traffic responses	

TIMEOUT=

Specifies the line time-out value in seconds. The values specified are the total times allowed for an input or output line buffer to be filled, emptied, or terminated. If time-out occurs, error retry logic is entered.

where:

n

Is a decimal value 1 to 32,767 for input.

NOTE:

For binary synchronous communications (BSC) controlled devices, this value is used when input text is expected or during receipt of input text. For DCT 500/TTY devices, this value is used after the first input text character is received.

m

Is a decimal value 1 to 32,767 for output.

If this parameter is not specified, the following default values are automatically provided:

Device	Input	Output
BSC	3	20
UNISCOPE/DCT 1000	3	20
DCT 500/TTY	512	20
DCT 2000	9	20
1004/9200/9300	4	20

```
XLATE= ( { YES          } , { YES          } )
       ( { NO           }   { NO           } )
       ( { labeli,idi   }   { labelo, ido  } )
```

where:

YES

Indicates that for input, use a standard translation table for this device. This is the default value.

NO

Indicates not to translate this input.

labeli

Indicates name of a user-supplied input translation table. The name must correspond to the name of one of the translation tables generated by the user in this CCA.

idi

Specifies a unique decimal value (1≤n≤240) identifying a user-generated input translation table within this network. This number makes it easy for ICAM to tell if the same input translation table already exists in the network. If the same number is used with a different labeli name, the table name used is the last one specified. The same input translation table may be identified in any number of LINE or TERM macros.

labelo

Specifies the name of a user-supplied output translation table for this line. The name specified must correspond to the name of the translation tables generated by the user in this CCA.

ido

Specifies a unique decimal value (1≤n≤240) identifying a user-generated output translation table within this network. This number makes it easy for ICAM to tell if the same table already exists in the network. If the same number is used with another labelo name, the table name used is the last one specified. The same output translation table may be identified in any number of LINE or TERM macros.

NOTES:

1. *The XLATE= parameters specified via the LINE macroinstruction are used for all terminals on a line for which no TERM macro XLATE parameter is specified. If an XLATE parameter is specified for a terminal via the TERM macro, it overrides the line information for that terminal only.*

2. *The idi and ido parameters refer to separate input and output tables; that is, there may be up to 240 input translation tables and 240 output tables.*

3. *Refer to Appendix A for a detailed description of user translation tables.*

```
LOW={filename}
    {MAIN    }
```

Specifies that the low priority queue is to be a 1- to 7-character file name of a disk file or located in main storage.

```
MEDIUM={filename}
       {MAIN    }
```

Specifies that the medium-priority queue is to be a 1- to 7-character file name of a disk file or located in main storage.

```
HIGH={filename}
     {MAIN    }
```

Specifies that the high-priority queue is to be a 1- to 7-character file name of a disk file or located in main storage.

NOTE:

The low, medium and/or high parameters specified in the LINE macroinstruction will be used for all terminals on that line for which a low, medium, or high parameter is not specified. If any of the preceding parameters are called in the TERM macroinstruction, it overrides the line parameters for that terminal only.

Examples:

```
1        10    16                                                      72
--------------------------------------------------------------------------
LNE1     LINE  CALL=6469Ø3Ø,                                           X
               DEVICE=(U1ØØ),                                          X
               TYPE=(2ØØØ,UNAT,SWCH,SYNC),                             X
               ID=4
LNE2     LINE  CALL=1759Ø3Ø,                                           X
               DEVICE=(TTY,33),                                        X
               TYPE=(11Ø,SWCH)
LNE3     LINE  DEVICE=(1ØØ4),                                          X
               TYPE=(24ØØ),                                            X
               ID=6
```

2.2.8. Input Process File (PRCS)

Function:

Defines a process file for use by a network as consisting of one, two, or three input queues. User program input messages are obtained from these queues by the GETCP macroinstruction, which references a user-defined (DTFCP) process file. This process file is referenced by the name (label) defined by this PRCS macroinstruction. If a process file is referenced by the user program without a priority level, the queues are accessed in descending priority order. Table 2–1 shows how queues are established and accessed.

Format:

LABEL	△OPERATION△	OPERAND
prcs-name	PRCS	$\left[\text{LOW=}\begin{Bmatrix}\text{filename}\\ \text{MAIN}\end{Bmatrix}\right]$ $\left[\text{,MEDIUM=}\begin{Bmatrix}\text{filename}\\ \text{MAIN}\end{Bmatrix}\right]$ $\left[\text{,HIGH=}\begin{Bmatrix}\text{filename}\\ \text{MAIN}\end{Bmatrix}\right]$

Label:

prcs-name
A 1-to 4-character label uniquely identifying up to three distinct input queues.

Parameters:

LOW=

where:

filename
Is 1- to 7-character file name of the disk file to be used for the low-priority queue of the subject PRCS file.

MAIN
Indicates that the low-priority queue of the subject PRCS file is to be located in main storage.

If omitted, refer to Table 2–1.

MEDIUM=

where:

filename
Is a 1- to 7-character file name of the disk file to be used for the medium-priority queue of the subject PRCS file.

MAIN
Indicates that the medium-priority queue of the subject PRCS file is to be located in main storage.

If omitted a medium-priority queue is not established. Refer to Table 2–1.

HIGH=

where:

filename
Is a 1- to 7-character file name of the disk file to be used for the high-priority queue of the subject PRCS file.

MAIN
Indicates the high-priority queue of the subject PRCS file is to be located in main storage.

If omitted, a high-priority queue is not established. Refer to Table 2–1.

Example:

```
1        10    16
------------------------------
PRF1     PRCS  LOW=MAIN
PRF2     PRCS  LOW=FILQU1,HIGH=MAIN
PRF3     PRCS  LOW=MAIN
```

Table 2—1. Process File (PRCS) Queue Relationships

PRCS Queue Specified			Queue Accessed by Queue Established in PRCS File	DTFCP/GETCP Level		
Low (L)	Medium (M)	High (H)		Low	Medium	High
No	No	No	L	L	L	L
No	No	Yes	H	H	H	H
No	Yes	No	M	M	M	M
No	Yes	Yes	H,M	M	M	H
Yes	No	No	L	L	L	L
Yes	No	Yes	H,L	L	L	H
Yes	Yes	No	M,L	L	M	M
Yes	Yes	Yes	H,M,L	L	M	H

2.2.9. Terminal Characteristics (TERM)

Function:

Identifies the characteristics of each terminal in a communications network. One of these instuctions must appear for each terminal. The terminals associated with a line are defined following the LINE macroinstruction. Default values for specific features are described in the system installation user guide/programmer reference, UP-8074 (current version).

Format:

```
LABEL          △OPERATION△      OPERAND
term-name        TERM           FEATURES= ⎧ (TTY)
                                          ⎪ ⎛⎧DCT475⎫⎞
                                          ⎪ ⎜⎨DCT5ØØ⎬⎟
                                          ⎪ ⎝⎩DCT524⎭⎠
                                          ⎪ (DCT1ØØØ)
                                          ⎪ ⎛U1ØØ ⎡,⎧ 96Ø ⎫⎤⎞
                                          ⎪ ⎝     ⎣ ⎩1Ø24 ⎭⎦⎠
                                          ⎪ ⎛U2ØØ ⎡,⎧1536 ⎫⎤⎞
                                          ⎪ ⎝     ⎣ ⎩192Ø ⎭⎦⎠
                                          ⎪ ⎛U4ØØ,⎧ 96Ø⎫⎞
                                          ⎪ ⎜     ⎪1Ø24⎪⎟
                                          ⎪ ⎜     ⎪1536⎪⎟
                                          ⎪ ⎝     ⎩192Ø⎭⎠
                                          ⎪ (U4ØØTE)
                                          ⎪ ⎛1ØØ4 ⎡,,⎧DLT3⎫⎤⎞
                                          ⎪ ⎝     ⎣  ⎩DLT1⎭⎦⎠
                                          ⎨ (DCT2ØØØ[,128][,SBLK])
                                          ⎪ ⎛⎧92ØØ ⎡,⎧ 96⎫⎤ ⎡,,⎧DLT3⎫⎤⎫⎞
                                          ⎪ ⎜⎪     ⎢ ⎨12Ø⎬⎥ ⎣  ⎩DLT1⎭⎦⎪⎟
                                          ⎪ ⎜⎨     ⎣ ⎩132⎭⎦          ⎬⎟
                                          ⎪ ⎜⎪93ØØ ⎡,,⎧DLT3⎫⎤          ⎪⎟
                                          ⎪ ⎝⎩     ⎣  ⎩DLT1⎭⎦          ⎭⎠
                                          ⎪  ⎛⎧2780⎫
                                          ⎪  ⎜⎩BSC ⎭
                                          ⎪  ⎜[,max-output-record-length]
                                          ⎪  ⎜⎡,⎧max-no-records/block⎫⎤
                                          ⎪  ⎜⎣ ⎩MULTI               ⎭⎦
                                          ⎪  ⎜⎡,⎧SECOND ⎫⎤
                                          ⎪  ⎜⎣ ⎩PRIMARY⎭⎦
                                          ⎪  ⎜[,TRANSPARENT]
                                          ⎪  ⎜[,transparent-input-record-length]
                                          ⎪  ⎜[,max-block-length]
                                          ⎪  ⎜⎡⎧MODEL1⎫⎤
                                          ⎪  ⎜⎢⎪MODEL2⎪⎥
                                          ⎪  ⎜⎢⎨MODEL3⎬⎥
                                          ⎩  ⎝⎣⎩MODEL4⎭⎦
```

(continued)

```
LABEL        △OPERATION△       OPERAND

                               [,ADDR=(rid,sid)]
                               [,ANSWER=(integer,C,text)]
                               ⎡,AUXn=⎛⎧PTP               ⎫⎞⎤
                               ⎢      ⎜⎪PTR               ⎪⎟⎥
                               ⎢      ⎜⎪PCH               ⎪⎟⎥
                               ⎢      ⎜⎨RDR               ⎬⎟⎥
                               ⎢      ⎜⎪PRNTR             ⎪⎟⎥
                               ⎢      ⎜⎪COP,did           ⎪⎟⎥
                               ⎢      ⎜⎪TP,did            ⎪⎟⎥
                               ⎣      ⎝⎩TCS,did1,did2     ⎭⎠⎦
                               [INPUT=(mpps-name[,number-of-blanks])]
                               ⎡LOW=⎧filename⎫⎤
                               ⎣    ⎩MAIN    ⎭⎦
                               ⎡MEDIUM=⎧filename⎫⎤
                               ⎣       ⎩MAIN    ⎭⎦
                               ⎡HIGH=⎧filename⎫⎤
                               ⎣     ⎩MAIN    ⎭⎦
                               [,PINTV=tenths-of-seconds]
                               [,PROFIL=n]
                               ⎡,DICE=⎛⎧ON ⎫,⎧NEWLINE⎫⎞⎤
                               ⎣      ⎝⎩OFF⎭ ⎩FORMS  ⎭⎠⎦
                               [,MSGWAIT=(message)]
                               ⎡,XLATE=⎛⎧YES         ⎫,⎧YES         ⎫⎞⎤
                               ⎢       ⎜⎨NO          ⎬ ⎨NO          ⎬⎟⎥
                               ⎣       ⎝⎩labeli,idi  ⎭ ⎩labelo,ido  ⎭⎠⎦
```

Label:

term-name
A 1- to 4-character label that identifies this terminal.

Parameters:

FEATURES=
Identifies the terminal device and some of the incorporated features. The specifications in each keyword parameter must be coded in the order shown.

where:

TTY
Identifies the terminal device as a teletypewriter (TTY).

```
⎧DCT475⎫
⎨DCT5ØØ⎬
⎩DCT524⎭
```
Identifies the terminal device as a DCT 500 series device.

DCT1000

Identifies the terminal device as a DCT 1000 terminal.

$$\mathtt{U100}\left[,\begin{Bmatrix}\mathtt{960}\\ \mathtt{1024}\end{Bmatrix}\right]$$

Identifies this terminal as a UNISCOPE 100 Display Terminal or UTS 400 terminal operating in UNISCOPE mode that is set to display 960 or 1024 characters (12 x 80 or 16 x 64 screen).

$$\mathtt{U200}\left[,\begin{Bmatrix}\mathtt{1536}\\ \mathtt{1920}\end{Bmatrix}\right]$$

Identifies this terminal as a UNISCOPE 200 Display Terminal or UTS 400 terminal operating in UNISCOPE mode that is set to display 1536 or 1920 characters (24 x 64 or 24 x 80 screen).

$$\mathtt{U400},\begin{Bmatrix}\mathtt{960}\\ \mathtt{1024}\\ \mathtt{1536}\\ \mathtt{1920}\end{Bmatrix}$$

Indicates that a UTS 400 is being used; 960, 1024, 1536, and 1920 refer to the screen size of the unit. Note that this operand does not apply to the UTS 400 operating in the UNISCOPE mode.

U400TE

Indicates the UTS 400 Text Editor (TE) terminal is used. The screen size is 24 x 80 (1920 characters).

$$\mathtt{1004}\left[,,\begin{Bmatrix}\mathtt{DLT3}\\ \mathtt{DLT1}\end{Bmatrix}\right]$$

Identifies the terminal device as a 1004 card processor system and indicates by specification the data line terminal (DLT1 or DLT3) being used. The extra comma denotes an unused parameter.

DCT2000 [,128][,SBLK]

Identifies this terminal as a DCT 2000 terminal and indicates the features it contains (128 indicates that the DCT 2000 terminal is set to receive and print 128 characters rather than 80 characters). SBLK indicates that it uses the short block feature.

$$\mathtt{9200}\left[,\begin{Bmatrix}\mathtt{96}\\ \mathtt{120}\\ \mathtt{132}\end{Bmatrix}\right]\left[,\begin{Bmatrix}\mathtt{DLT3}\\ \mathtt{DLT1}\end{Bmatrix}\right]$$

Identifies the terminal device as a 9200 system that is being interfaced with ICAM as a communication terminal. Also specifies the print line length as being 96, 120, or 132. Indicates by specification the data line terminal that is being used as either DLT1 or DLT3.

$$\mathtt{9300}\left[,,\begin{Bmatrix}\mathtt{DLT3}\\ \mathtt{DLT1}\end{Bmatrix}\right]$$

Identifies the terminal device as a 9300 system that is being interfaced with ICAM as a communication terminal. Indicates by specification the data line terminal that is being used as either DLT1 or DLT3. Note the extra comma, which denotes an unused parameter.

```
{2780}[,max-output-record-length][,{max-no-records/block}]
{BSC }                            [ {MULTI               }]

[,{SECOND }][,TRANSPARENT][,transparent-input-record-length]
[ {PRIMARY}]

[,max-block-length][,{MODEL1}]
                   [ {MODEL2}]
                   [ {MODEL3}]
                   [ {MODEL4}]
```

Specifies the parameters for a binary synchronous communications device.

where:

```
{2780}
{BSC }
```

Identifies the terminal device.

where:

`2780`

Specifies an IBM 2780, 3741, or 2780 compatible terminal.

`BSC`

Specifies general BSC procedures with no specific terminal-related functions assigned.

`max-output-record-length`

A decimal number that specifies the output print line length. If omitted, default is 80.

```
{max-no-of-records/block}
{MULTI                  }
```

A decimal number that specifies the maximum number of records that may be transmitted in a block. If MULTI is specified, this number is set to 7. If omitted, the default is 2.

```
{SECOND }
{PRIMARY}
```

Specifies the priorities of the RDH.

`TRANSPARENT`

Indicates that the terminal has the transparency feature. This may only be specified if the LINE macro specified DEVICE=(BSC,,EBCDIC).

`transparent-input-record-length`

A decimal number that specifies the fixed length of transparent input records. If zero, single-record transparent input of arbitrary length is assumed. In this case, ITB, DLE, STX will be passed to the user if present in the message. If omitted, the default is 80. This subparameter is applicable only if subparameter TRANSPARENT is used.

max-block-length

A decimal number that specifies the maximum number of characters (excluding STX and DLE, STX) that can be sent in a block. If omitted, 2780 compatible default is 200. If the maximum number of records per block is 2 or MULTI has been specified for the 2780 compatible mode, default is 400.

$$\left\{\begin{matrix}\text{MODEL1}\\ \text{MODEL2}\\ \text{MODEL3}\\ \text{MODEL4}\end{matrix}\right\}$$

Specifies the output devices available to the 2780. MODEL1 is the printer; MODEL2 is printer and punch; MODEL3 is printer (no card reader); MODEL4 is the punch.

Summarizing the default values for 2780/BSC mode we have:

FEATURES=(2780,80,2,SECOND)

FEATURES=(BSC,80,1,SECOND,,80,84)

NOTE:

Only the UNISCOPE 100, UNISCOPE 200, and DCT 500 terminals are supported in transient TCI.

ADDR=

Identifies the hardware address of the terminal as it was wired in the device by the field installation technicians. It must be specified for UNISCOPE 100 and UNISCOPE 200 terminals, UTS 400 terminals (all versions), DCT 1000 terminals, and the DCT 500 terminals in automatic mode.

where:

- DCT 500 terminals:

  The rid,sid definitions for DCT 500 terminals must be unique for each terminal on a line.

- UNISCOPE 100, UNISCOPE 200, UTS 400, and DCT 1000 terminals:

  The rid,sid definitions for these terminals are as follows:

rid

Specifies two hexadecimal digits that identify the remote identification device address wired in the terminal. All terminals in a polling group must have the same rid and all TERM macros for terminals in a polling group must be adjacent in the network definition. The allowable range of rid addresses is 21_{16} to $4F_{16}$.

sid

Specifies two hexadecimal digits that identify the station identification address wired in the terminal. Each terminal in a polling group must have a unique sid. The allowable range of sid addresses is 51_{16} to $6F_{16}$.

NOTE:

For UTS 400, each terminal in the group (master/slaves or terminal controller/slaves) must use the same rid. The sid for each terminal must be assigned in contiguous ascending order with the master (or the primary terminal) designated as the lowest value sid. If the screen bypass function is utilized, the sid designated for this function must be the next available sid (one greater than the sid assigned to the last actual terminal in the group).

ANSWER=

This parameter may be used to specify a site-identification constant against which site ID codes may be verified. This parameter is not supported in transient TCI.

where:

Device	Format	Usage
1004 card processor	integer, C, text (alphanumeric) characters	Checked against remote site ID
9200/9300 Series system	integer, C, text (alphanumeric) characters	Checked against remote site ID

integer

Is a decimal number indicating the number of characters comprising the text of subparameter 3.

C

Identifies the site ID constant as an alphanumeric character string.

text

Is the site ID. Text must be *n* nonblank alphanumeric characters.

AUXn=

All remote terminals associated with ICAM are assumed to include at least one input and/or output device, which is considered to be the primary device for I/O operations. In addition to the primary device, a remote terminal may also support a number of secondary device that are known as auxiliary devices.

The keyword parameters AUX1 through AUX12 apply to any terminal with one or more auxiliary devices. They must be used in the sequence from AUX1 to AUX12. A tape cassette subsystem (TCS) defined for a UNISCOPE terminal is an exception. When a TCS drive is defined by AUXn, then AUXn+1 is skipped.

When a UTS 400 terminal is configured, the AUXn=(TCS,did$_1$,did$_2$) operand refers to the tape cassette system or the diskette; the AUXn=(TP,did) operand refers to the 800 terminal printer or the 0786 printer. This parameter is not supported in transient TCI.

The auxiliary devices are specified as:

- DCT 1000

  The primary input and output devices for the DCT 1000 terminal are the keyboard and printer, respectively. In addition, up to three auxiliary devices can be defined for a DCT 1000 terminal. The AUXn parameter is:

```
AUXn=⎧PTP  ⎫
     ⎪PTR  ⎪
     ⎨PCH  ⎬
     ⎪RDR  ⎪
     ⎩PRNTR⎭
```

where:

PTP
Identifies the paper tape punch.

PTR
Identifies the paper tape reader.

PCH
Identifies the card punch.

RDR
Identifies the card reader.

PRNTR
Identifies the auxiliary printer.

The following values are substituted for the designated parameters.

Device Parameter	Device Address (Hex.) Wired in DCT 1000
PTP	7D
PTR	7C
PCH	7B
RDR	7A
PRNTR	7F

- UNISCOPE/UTS 400

The primary input and output devices for a UNISCOPE or UTS 400 terminal are the keyboard and the screen, respectively. For the UNISCOPE, up to 8 devices or 12 device addresses (did) can be specified. For the UTS 400, up to 8 devices on the 7-bit interface and/or 4 devices on the 8-bit interface (a maximum of 12 device addresses) can be specified. The allowable range for device addresses is 73_{16} to $7E_{16}$. For the UNISCOPE or UTS 400 terminals, the AUXn parameter is:

```
AUXn=(COP,did)
```

where:

COP
: Identifies the communications output printer.

did
: Specifies two hexadecimal digits giving the device address in the COP.

```
AUXn=(TP,did)
```

where:

TP
: Identifies the 800 terminal printer, or the 0786 printer

did
: Specifies the two hexadecimal digits identifying the device address in the TP.

AUXn=(TCS,did_1,did_2)

Identifies one of the two cassette drives (n) on the cassette subsystem regardless of the type of operation (n + 1 is also valid except for write operations), or the diskette.

where:

TCS
: Identifies the tape cassette system, or diskette.

did_1
: Specifies two hexadecimal digits giving the output device (write) address in the TCS for one of the two drives.

did_2
: Specifies two hexadecimal digits giving the input device (read) address in the TCS for the same drive.

Example:

```
1         10      16                                                        72
--------------------------------------------------------------------------------
                  AUX1=(TCS,73,74),                                        X
                  AUX3=(TCS,75,76)
```

- DCT500, DCT 524, DCT 475, and TTY

In addition to the primary input and output devices of the keyboard and the printer, the DCT 500, DCT 475, and TTY supply auxiliary device support for the paper tape reader and punch. The DCT 524 supplies auxiliary device support for the write and read cassette systems.

```
AUXn={PTP}
     {PTR}
```

where:

PTP
Identifies the paper tape punch or the cassette write head for the DCT 524.

PTR
Identifies the paper tape reader or the cassette read head for the DCT 524.

The following values are substituted for the designated parameters:

Device Parameter	Device Address (Hex.) Wired in DCT 500
PTP	75
PTR	76

- 1004 card processor and 9200 and 9300 systems

The primary input and output devices for the 1004 card processor are the card reader and page printer. In addition, the 1004 card processor, the 9200 system, and the 9300 system provide auxiliary device support for a single card punch. For these systems, the AUXn parameter is:

```
AUX1=(PCH)
```

where:

PCH
Identifies the card punch.

- DCT 2000

  The primary input and output devices for the DCT 2000 are the card reader and the page printer. In addition, the DCT 2000 supplies auxiliary device support for a single card punch. For the DCT 2000, the AUXn parameter is:

  ```
  AUX1=(PCH)
  ```

  where:

  PCH
  Identifies the card punch.

INPUT=
Identifies the MPPS macro.

where:

mpps-name
Is the 1- to 4-character name in the label field of the MPPS macro.

number-of-blanks
Is a decimal number from 1 to 255 indicating the number of bytes to be reserved in front of the input message for MPPS or user insertion of data.

```
LOW={filename}
    {MAIN    }
```

Identifies the low-priority output queue associated with the terminal. The output queue can be a 1- to 7-character name of the DISCFILE macroinstruction associated with the queue; or specifies that the queue is a main-storage queue. This parameter is not supported in transient TCI.

```
MEDIUM={filename}
       {MAIN    }
```

Identifies the medium-priority output queue associated with the terminal. The output queue can be a 1- to 7-character name of the DISCFILE macroinstruction associated with the queue; or specifies that the queue is a main-storage queue. This parameter is not supported in transient TCI.

```
HIGH={filename}
     {MAIN    }
```

Identifies the high-priority output queue associated with the terminal. The output queue can be a 1- to 7-character name of the DISCFILE macroinstruction associated with the queue; or specifies that the queue is a main-storage queue. This parameter is not supported in transient TCI.

NOTES:

1. *The QUEUES parameter used previously is also acceptable.*

2. *These parameters override the LINE priority parameter for the indicated terminal.*

PINTV=tenths-of-seconds

Indicates the time to allow between polls to the terminal in tenths of seconds. This parameter applies only to the UNISCOPE 100, UNISCOPE 200, and DCT 1000 terminals. All UNISCOPE or DCT 1000 terminals in the same poll group must specify the same poll interval. The default value is 10 (10 tenths=1 second). The TERM macro truncates the value to give an integral value in tenths of seconds. The allowable range is 10 to 2550 (1 to 255 seconds).

PROFIL=n

Assigns a profile number to a terminal for format edit processing. (See fundamentals of ICAM user guide, UP-8194 (current version) for a discussion on format edit.) The value n corresponds to the value assigned to an end user profile (EUP) macro via its PRFN keyword. This parameter is not supported in transient TCI.

$\text{DICE=}\left\{\begin{matrix}\text{ON}\\\text{OFF}\end{matrix}\right\},\left\{\begin{matrix}\text{NEWLINE}\\\text{FORMS}\end{matrix}\right\}$

Describes the use of DICE on a terminal basis. The default is (ON,NEWLINE).

where:

ON

Indicates that DICE is to be placed in all input messages by the ICAM remote device handler.

OFF

Indicates that DICE is not to be placed in input messages.

NEWLINE

Indicates that, on output, a set coordinates DICE or a *forms control with clear DICE* is converted to a new line position control DICE with $M=00_{16}$ and $N=00_{16}$. See 3.19.

FORMS

Indicates that, on output, a *set coordinates* DICE is converted to a *forms control* DICE. See 3.19.

NOTE:

NEWLINE and FORMS apply to all devices except CRT devices.

MSGWAIT=(message)

Indicates a computer message waiting. A 4-character message is sent to the terminal. After the terminal responds with a BEL (control G) and an ETX (control C), the waiting message is transmitted to the terminal. This parameter is only used with DCT 475, DCT 500, DCT 524, or TTY terminals. The default value is /CMW but any 4-character message may be substituted in the parameter operand.

NOTE:

No carriage return or line feed will accompany the default message. However, carriage return and line feed commands could be inserted as the first or last two characters of the 4-character message in the parameter operand, leaving two printable characters in the message. This may be accomplished by multipunching carriage return (0—9—5) and line feed (12—9—8—5) or by multipunching a DICE expression with a DLE (12—11—9—8—1) and the function code for new line (4). In either situation, the remote device handler will insert appropriate time fill for carriage return and line feed.

$$\mathrm{XLATE}=\left(\left\{\begin{array}{l}\mathrm{YES}\\ \mathrm{NO}\\ \mathrm{labeli,\ idi}\end{array}\right\},\left\{\begin{array}{l}\mathrm{YES}\\ \mathrm{NO}\\ \mathrm{labelo,\ ido}\end{array}\right\}\right)$$

where:

YES

Indicates that, for input, use a standard translation table for this device. This is the default value.

NO

Indicates not to translate this input.

labeli

Indicates the name of a user-supplied input translation table. The name must correspond to the name of one of the translation tables generated by the user in this CCA.

idi

Specifies a unique decimal value (1⩽n⩽240) identifying a user-generated input translation table within this network. This number makes it easy for ICAM to tell if the same input translation table already exists in the network. If the same number is used with a different labeli name, the table name used is the last one specified. The same input translation table may be identified in any number of LINE or TERM macros. The default is the input translation defined for the LINE macro associated with this terminal.

labelo

Specifies the name of a user-supplied output translation table for this device. The name specified must correspond to the name of the translation tables generated by the user in this CCA.

ido

Specifies a unique decimal value (1≤n≤240) identifying a user-generated output translation table within this network. This number makes it easy for ICAM to tell if the same table already exists in the network. If the same number is used with another labelo name, the table name used is the last one specified. The same output translation table may be identified in any number of LINE or TERM macros. The default is the input translation defined for the LINE macro associated with this terminal.

NOTES:

1. *The XLATE parameters specified via the LINE macro are used for all terminals on the line for which no XLATE parameter is specified. An XLATE parameter on a TERM call line overrides the line information for that terminal only.*

2. *The idi and ido parameters refer to separate input and output tables. There may be up to 240 input and 240 output translation tables.*

3. *The user-specified translation tables should appear at the end of a network (CCA) generation.*

4. *The continuation of DC statements is not acceptable; therefore, each statement must have a closing apostrophe in column 71 or before.*

5. *The standard translation tables are either 128 or 256 bytes long; user tables should normally be one of these two lengths.*

Examples:

```
1       10    16                                                         72
---------------------------------------------------------------------------
TRM1    TERM  ADDR=(29,53),                                              X
              FEATURES=(U1ØØ,1Ø24),                                      X
              PINTV=5Ø,                                                  X
              HIGH=MAIN
TRM2    TERM  ADDR=(29,54),                                              X
              FEATURES=(U1ØØ,1Ø24),                                      X
              PINTV=5Ø,                                                  X
              HIGH=MAIN
TRM3    TERM  ADDR=(28,51),                                              X
              FEATURES=(U1ØØ,1Ø24),                                      X
              PINTV=5Ø,                                                  X
              HIGH=MAIN
TRM4    TERM  ADDR=(28,52),                                              X
              FEATURES=(U1ØØ,1Ø24),                                      X
              PINTV=5Ø,                                                  X
              HIGH=MAIN
TRM5    TERM  ANSWER=(4,C,PSDC),                                         X
              FEATURES=(1ØØ4),AUX=(PCH)
TMØ1    TERM  FEATURES=(U4ØØ,192Ø),HIGH=MAIN,MEDIUM=MAIN,LOW=MAIN,       X
              ADDR=(21,54)AUX1=(COP,73),AUX2=(COP,74),                   X
              AUX3=(TCS,75,76),AUX5=(TCS,77,78),                         X
              AUX7=(TCS,D9,7A),AUX9=(TCS,7B,7C)
```

2.3. EXAMPLES OF TCI NETWORK DEFINITION

To define a TCI network, let's first construct a diagram such as Figure 2-2. A typical network might consist of two communications lines. We'll make line 1 a switched line with two UNISCOPE 100 display terminals. We'll make line 2 a dedicated line and connect a 1004 card controller to part 4 of the communications adapter. For the lines, let's specify synchronous, half-duplex transmission at 2000 baud.

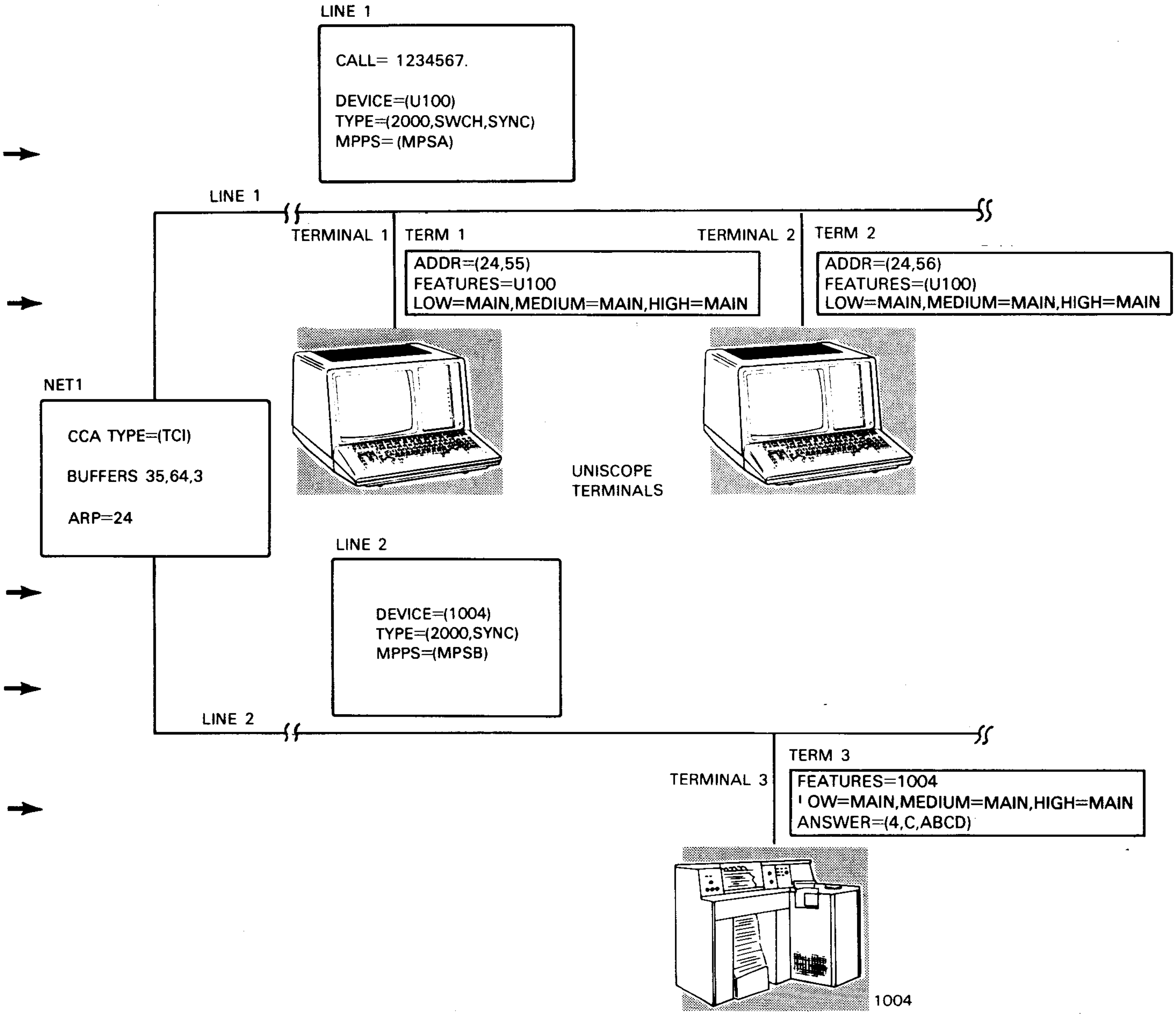


Figure 2—2. TCI Network with No Auxiliary Devices

The 1004 card controller provides a remote site-id of PDC (Philadelphia Development Center) and the remote device phone numbers are 123-4567 and 123-4568. (Do not use dashes in the coding.) Note that the network described in the ICAM direct data interface (DDI) user guide, UP-8549 (current version), and the TCI network are identical in appearance and parameters. Now, however, we must consider the queues that provide us with the TCI output queueing capability.

For the destination queues (output), let's assign each terminal three priority queues (high, medium, and low) in main storage.

We'll also assume that we have different message processing program specifications (MPPS) for the two lines. MPPS is described in Section 4, and it is sufficient here to note that we want to assign them names of MPSA and MPSB.

Now that we have identified the queues, the only additional specifications to complete the change from a DDI network to a TCI network is to allocate network buffers via the BUFFERS macro.

You can now take parameters directly from Figure 2-2 and write them on a coding form in the order described in 3.1. Example 1, illustrating Figure 2-2, is formatted to show a comparison between the coding required for a DDI network definition and that required for a TCI network definition. If you are familiar with DDI, the comparison clarifies the additions necessary to convert from one to the other. If you are not familiar with DDI, it highlights the instructions and parameters necessary to provide the output queueing features of TCI.

Example 1 (Conversion from DDI to TCI Network):

```
DDI Network                              Conversion to TCI Network

1       10    16                         1       10    16
NET1    CCA   TYPE=(DDI,3)←              NET1    CCA   TYPE=(TCI)
        BUFFERS ARP=2Ø  ←                        BUFFERS 35,64,3,ARP=24
LNE1    LINE  CALL=(1234567),
              DEVICE(U1ØØ),
              TYPE=(2ØØØ,SWCH,SYNC)
TRM1    TERM  FEATURES=(U1ØØ),
              ADDR=(24,55) ←                       LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
TRM2    TERM  FEATURES=(U1ØØ,1Ø24),
              ADDR=(24,56) ←                       MEDIUM=MAIN,HIGH=MAIN,HIGH=MAIN
LNE2    LINE  DEVICE=(1ØØ4),
              TYPE=(2ØØØ,SYNC),
              ID=4
TRM3    TERM  FEATURES=(1ØØ4),
              ANSWER=(4,C,ABCD)←                    LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
TCIFILE DISCFILE MSGSIZE=256
        ENDCCA
```

From the label of the CCA macro, an entry point label is automatically generated. This label has the form, CCA#name. In example 1, the label is CCA#NET1. To the simplified network in Figure 2-2, add a communications output printer (COP) and a tape cassette subsystem (TCS) to each of the UNISCOPE terminals. These are classified as auxiliary devices. For clarity in showing these additions, let's assume that all the parameters shown in Figure 2-2 remain the same. In the new network diagram (Figure 2-3), we'll show the additions necessary to define the auxiliary devices. Again, you can go directly from the diagram to the coding form to produce example 2.

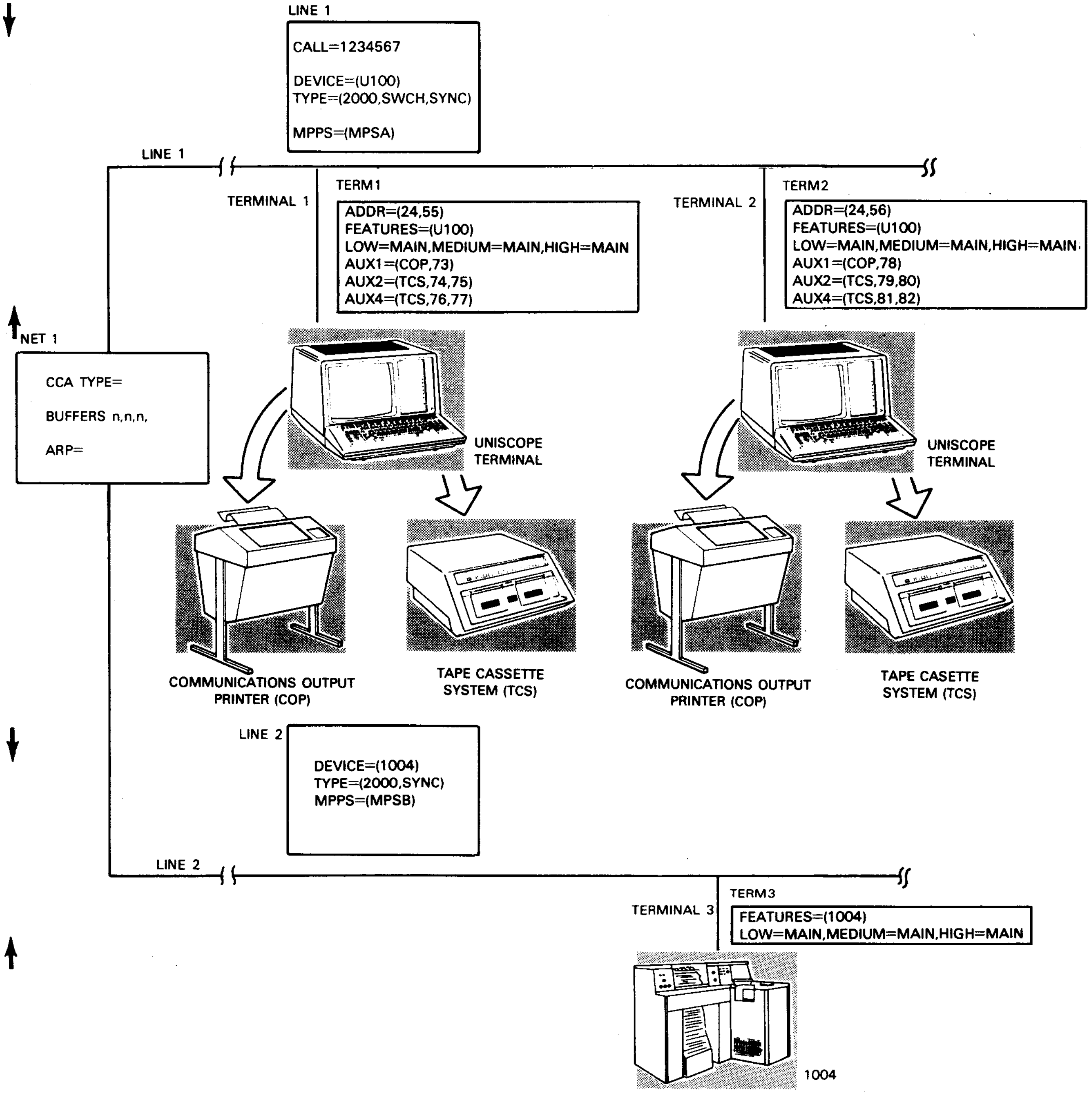


Figure 2—3. TCI Network with Auxiliary Devices

Example 2 (See Figure 2-3.):

```
1        10    16                                                     72
NET2     CCA   TYPE=(TCI)
         BUFFERS 35,64,3,ARP=24
LNE1     LINE  CALL=(1234567),                                         X
               DEVICE=(U1ØØ),                                          X
               TYPE=(2ØØØ,SWCH,SYNC)
TRM1     TERM  FEATURES=(U1ØØ),                                        X
               ADDR=(24,55),                                           X
               AUX1=(COP,73),                                          X
               AUX2=(TCS,74,75),                                       X
               AUX4=(TCS,76,77),                                       X
               LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
TRM2     TERM  FEATURES=(U1ØØ,1Ø24),                                   X
               ADDR=(24,56),                                           X
               AUX1=(COP,78),                                          X
               AUX2=(TCS,79,8Ø),                                       X
               AUX4=(TCS,81,82),                                       X
               LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
LNE2     LINE  CALL=(1234568),                                         X
               DEVICE=(1004),                                          X
               TYPE=(2000,SYNC),                                       X
               ID=4
TRM3     TERM  FEATURES=(1004),                                        X
               LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN,                         X
               ANSWER=(4,C,PSDC)
TCIDTF   DISCFILE
         ENDCCA
```

When you write the communications user program (CUP), this is the network and line(s) that are activated. A sample CUP using this network is illustrated in 3.6.

Advancing to example 3, you see a more complex network of seven lines with various combinations of terminals and auxiliary devices. Drawing the same basic diagram of lines and terminals as Figures 2-2 and 2-3, you could easily produce the network definition shown in example 3. A detailed analysis of this network definition follows the example.

Also shown are the SYSGEN parameters required for system generation that are described in detail in the systems installation user guide/programmer reference, UP-8074 (current version).

Example 3:

```
     1        10    16                                                      72
     ---------------------------------------------------------------------------
     COMMCT                                   (SYSGEN)
     IDEV     CCA   TYPE=(TCI)
              BUFFERS 3Ø,64,5,ARP=46
(1)  LØØ1     LINE  DEVICE=(UNISCOPE),TYPE=(96ØØ,FULL,SYNC),ID=4
(2)  SCP1     TERM  ADDR=(23,54),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,               X
                    AUX1=(COP,73),AUX2=(TCS,74,75),AUX4=(TCS,76,77),         X
                    LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
(3)  SCP2     TERM  ADDR=(23,55),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,               X
                    AUX1=(TP,73),LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
(4)  SCP3     TERM  ADDR=(23,56),FEATURES=(U2ØØ,192Ø),PINTV=1Ø,              X
                    LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN

(5)  LØØ2     LINE  DEVICE=(DCT1ØØØ,INTER,U1ØØ),                             X
                    TYPE=(48ØØ,AUTO,SWCH,SYNC),DIALER=(11),CALL=2345678,     X
                    LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
(6)  SCP4     TERM  ADDR(24,55),FEATURES=(U1ØØ,1Ø24),PINTV=1Ø
(7)  SCP5     TERM  ADDR=(24,56),FEATURES=(U1ØØ,1Ø24),PINTV=1Ø
(8)  DCT1     TERM  ADDR=(25,55),FEATURES=(DCT1ØØØ,PINTV=5Ø
(9)  DCT2     TERM  ADDR=(25,56),FEATURES=(DCT1ØØØ),PINTV=5Ø,                X
                    AUX1=(PTR),AUX2=(PTP),AUX3=(RDR),AUX4=(PCH)

(10) LØØ3     LINE  DEVICE=(U1ØØ),TYPE=(2ØØØ,SWCH,SYNC),CALL=2957841
(11) SCP6     TERM  ADDR=(21,51),FEATURES=(U1ØØ,1Ø24),PINTV=1Ø,              X
                    LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN

(12) LØØ4     LINE  DEVICE=(DCT5ØØ,AUTO),TYPE=(3ØØ,SWCH),                    X
                    LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
(13) DCF1     TERM  ADDR=(23,54),FEATURES=(DCT5ØØ),                          X
                    AUX1=(PTP),AUX2=(PTR)
(14) DCF2     TERM  ADDR=(23,55),FEATURES=(DCT524),                          X
                    AUX1=(PTP),AUX2=(PTR)
(15) DCF3     TERM  ADDR=(23,56),FEATURES=(DCT5ØØ)

(16) LØØ5     LINE  DEVICE=(DCT5ØØ,TTY),TYPE=(3ØØ,SWCH) LOW=QTY1

(17) TTY1     TERM  FEATURES=(DCT5ØØ)

(18) LØØ6     LINE  DEVICE=(TTY,35),TYPE=(11Ø,UNAT,SWCH),ID=9
(19) TTY2     TERM  FEATURES=(TTY),AUX1=(PTR)

(20) LØØ7     LINE  DEVICE=(DCT5ØØ,TTY),TYPE=(11Ø,SWCH)
(21) TTY3     TERM  FEATURES=(DCT524),AUX1=(PTP),AUX2=(PTR),                 X
                    HIGH=MAIN,MEDIUM=MAIN,LOW=MAIN

(22) TCIDTF   DISCFILE MSGSIZE=256
              ENDCCA
              MCP   MCPNAME=C1 MCPVOL=ICAM#2    (SYSGEN)
                    CACH=(Ø4,IDEV,1)
                    CACH=(Ø5,IDEV,2)
                    CACH=(Ø6,IDEV,3)
                    CACH=(Ø7,IDEV,4)
                    CACH=(Ø8,IDEV,5)
                    CACH=(Ø9,IDEV,6)
                    CACH=(1Ø,IDEV,7)

     END
```

Line/Terminal — Description

① L001 — A synchronous, dedicated, full-duplex line connected to ports 4 and 5 of the communications adapter (CA). Direct Connection Modules (DCMs) are used to achieve the full-duplex 9600 baud. The line terminates two UNISCOPE 100 terminals and one UNISCOPE 200 terminal. All three display terminals are in one poll group with a common rid 23_{16}. The poll interval for this group is 1 second with the display terminals connected to a terminal multiplexer.

② SCP1 — A UNISCOPE 100 terminal with a 960-character (12 x 80) CRT supporting one tape cassette system (TCS) and one communications output printer (COP).

③ SCP2 — A UNISCOPE 100 terminal with a 12 x 80 screen supporting one Model 800 terminal printer.

④ SCP3 — A UNISCOPE 200 terminal with a 1920-character (24 x 80) screen. No auxiliary device is supported.

⑤ L002 — A synchronous switched line for which automatic dialing has been specified. The telephone number is specified in the CALL parameter. An automatic calling unit (ACU) to be used in autodialing is connected to port 11 of the CA as specified by the DIALER parameter. The line terminates two UNISCOPE 100 terminals and two DCT 1000 terminals.

⑥
⑦ SCP4, SCP5 — A UNISCOPE 100 terminal with a poll rate of 1 second and a 1024-character (16 x 64) screen.

⑧ DCT1 — A DCT 1000 with primary input/output devices of keyboard and printer.

⑨ DCT2 — A DCT 1000 with auxiliary devices supported consisting of paper tape reader/punch, card reader, and card punch in addition to its primary input/output devices.

⑩ L003 — A synchronous switched line connected to port 6 of the CA. The line terminates one UNISCOPE 100 terminal. Output is queued at three priority levels.

Line/Terminal	Description
⑪ SCP6	A UNISCOPE 100 terminal with 960-character (12 x 80) CRT and a poll rate of 1 second.
⑫ L004	An asynchronous switched line connected to port 7 of the CA. It terminates two DCT 500 terminals and one DCT 524 terminal that are in automatic mode (addressed). The baud rate switch on each terminal is set at 300. Even though the rid is the same for each DCT 500, they are polled separately. The poll rate is approximately 2 seconds times the number of terminals (and auxiliary devices when requested for input) on the line.
⑬ DCF1	A DCT 500 with paper tape reader/punch as the auxiliary device.
⑭ DCF2	A DCT 524 with a TCS as an auxiliary device.
⑮ DCF3	A DCT 500 with no auxiliary devices.
⑯ L005	A 300-baud rate, asynchronous, switched line connected to port 8 of the CA. It terminates a DCT 500 in teletypewriter mode.
⑰ TTY1	A DCT 500 terminal with no auxiliary devices.
⑱ L006	A 110-baud rate, asynchronous, switched line connected to port 9 of the CA. It terminates a Model 35 teletypewriter. The line is used for unattended answering, which means that the remote terminal will dial in to make a connection.
⑲ TTY2	A Model 35 teletypewriter with a paper tape reader (PTR) as an auxiliary device.
⑳ L007	A 110-baud rate, asynchronous, switched line connected to port 9 of the CA. It terminates a DCT 524 in teletypewriter mode.
㉑ TTY3	A DCT 524 with a TCS as an auxiliary device (PTP and PTR).
㉒ TCIDTF	The definition for the TCI disk file.

The previous examples all used main storage queues. Example 4 llustrates a resident TCI network with disk queueing. In the TERM macro, the LOW operands give the name of a disk file defined at the end of the network description. This example is shown in Figure 2–4 with no auxiliary devices and in Figure 2–5 with auxiliary devices. In Figure 2–5, only the parameters defining the auxiliary device specifications are shown. All other parameters are identical to Figure 2–4.

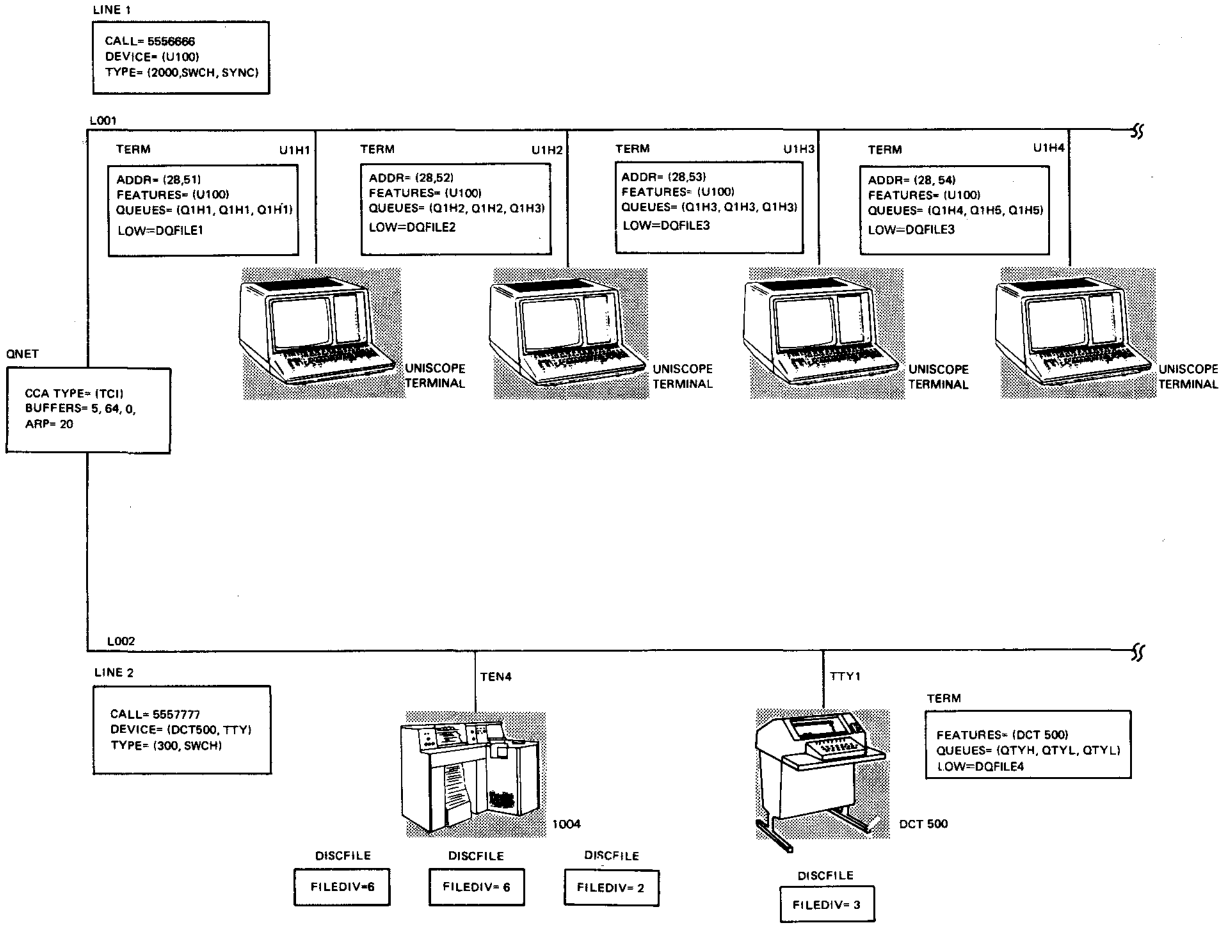


Figure 2—4. Disk-Queued TCI Network with No Auxiliary Devices

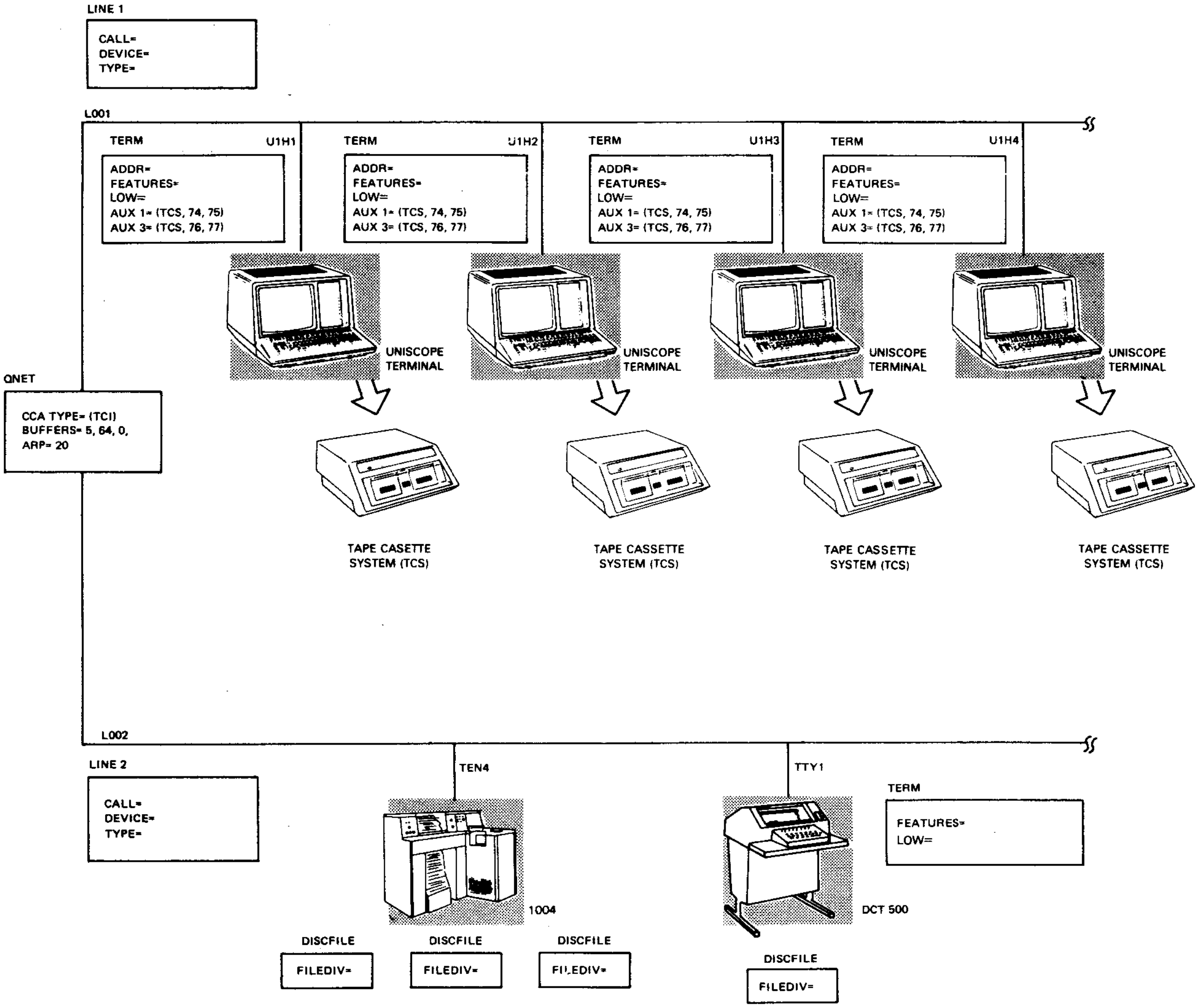


Figure 2—5. Disk-Queued TCI Network with Auxiliary Devices

Example 4:

```
    1        10    16                                                     72
    ------------------------------------------------------------------------
①   ONET     CCA   TYPE=(TCI),PASSWORD=DISCNET
             BUFFERS 5,64,Ø,ARP=2Ø,FEATURES=OUTDELV
②   LØØ1     LINE  DEVICE=(U1ØØ),TYPE=(2ØØØ,SWCH,SYNC),CALL=5556666
    U1H1     TERM  ADDR=(28,51),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,              X
                   LOW=DQFILE1,                                            X
③                  AUX1=(TCS,74,75),AUX3=(TCS,76,77)
    U1H2     TERM  ADDR=(28,52),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,              X
                   LOW=DQFILE2,AUX1=(TCS,74,75),                           X
                   AUX3=(TCS,76,77)
    U1H3     TERM  ADDR=(29,53),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,              X
                   LOW=DQFILE3,                                            X
                   AUX1=(TCS,74,75),AUX3=(TCS,76,77)
    U1H4     TERM  ADDR=(29,54),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,              X
                   LOW=DQFILE3,AUX1=(TCS,74,75),                           X
                   AUX3=(TCS,76,77)
④   LØØ2     LINE  DEVICE=(DCT5ØØ,TTY),TYPE=(3ØØ,SWCH),CALL=5557777,ID=6
    TTY1     TERM  FEATURES=(DCT5ØØ),LOW=DQFILE4
    TEN4     TERM  FEATURES=(1ØØ4),ANSWER=(5,C,BLUEBELL)
⑤   DQFILE1  DISCFILE FILEDIV=6
    DQFILE2  DISCFILE FILEDIV=6
    DQFILE3  DISCFILE FILEDIV=2
    DQFILE4  DISCFILE FILEDIV=3
    TCIDTF   DISCFILE MSGSIZE=256
             ENDCCA
```

Line/Terminal	Description
①	The output delivery notification feature is specified.
②	A synchronous, switched line. The line terminates four UNISCOPE 100 terminals, two with rid 28 and two with rid 29. Each UNISCOPE 100 terminal has a 960-character screen and a poll interval of 1 second. Line speed is 9600 baud.
③	An auxiliary device tape cassette subsystem (TCS) is specified.
④	An asynchronous, switched line with line speed of 300 baud on port 6 of the CA. The line terminates a DCT 500 in teletypewriter mode and a 1004.
⑤	Disk files are defined for disk queueing. Three files are used in L001 and one file is used in L002 for disk-queued output.

NOTE:

Examples 1, 2, and 3 have the queues residing on disk while example 4 has the queues residing in main storage.

Example 5 shows the macros and their associated parameters that are required to generate a CCA for a transient TCI.

Example 5:

```
      1                 10          16                                                                  72
      ----------------------------------------------------------------------------------------------------
      TCI1              CCA         TYPE=(TCI),PASSWORD=CONFID1,                                        X
                                    CCAID=TCIFONE,TRANSNT=YES,                                          X
                        BUFFERS     1,256,,ARP=9
  ①   LNE1              LINE        DEVICE=(U1ØØ),                                                      X
                                    TYPE=(2ØØØ,SWCH,SYNC)
  ②   TRM4              TERM        ADDR=(24,51),PINTV=1Ø,                                              X
                                    FEATURES=(U1ØØ,96Ø)
      TCIDTF            DISCFILE         MSGSIZE=96Ø
                        ENDCCA
```

Line/Terminal	Description
① LNE1	A synchronous, switched line. It terminates a single UNISCOPE 100 terminal. Baud rate is 2000.
② TRM4	A UNISCOPE 100 terminal 960-character (12 x 80) CRT and a poll rate of 1 second.

3. Communications User Program (CUP)

3.1. GENERAL

In an ICAM communications network, the communications user program (CUP) determines the processing performed on messages transmitted and received in the network. The CUP consists of three functional parts: acquiring/releasing, sending/receiving, and altering/displaying. Deferred user service transient (DUST) macros are used to acquire, release, alter, or display networks or lines. Declarative and imperative macros are provided for the actual sending and receiving. The interface with files is defined by the declarative macroinstruction for the most part, while services required of ICAM are executed through the imperative macros.

A declarative macro generates and formats a packet or table with fixed and variable sets of constants that will be examined, altered, and dynamically manipulated by a particular ICAM software element. Since the declarative macro generates constants, it may not be coded within the executable portion of your program. Each packet associated with a discrete function has a unique descriptive label and acronym used to reference that packet thereafter. The declarative macro is intended to be the raw intelligence that describes particular functions requested by your program. The description of the function in the packet is intended to work with an imperative macro that passes control to the required software element for execution of the function jointly described by both the declarative and imperative macros.

The imperative macros, by convention, use your program registers 1 and 0 as parameter passing vehicles. Normally, R1 is preloaded with the address of the description packet before issuing the imperative macros. Generally, R1 is loaded by a load-address (LA) instruction. R0 is loaded by a load (L) instruction. R0 is reserved for supplemental parameter passing and, normally, each byte is reserved for particular information.

Whenever ICAM gives you control (i.e., SVC completion, NETREQ macro error return, etc.), it does so by placing the address in your TCB PSW. For this reason, therefore, all points of control passed from ICAM to you must be to your resident code (i.e., use of VCONs for return addresses is not supported by ICAM). In addition, all tables or packets supplied to ICAM via an SVC must be in the user's resident main storage.

ICAM is an event-driven method of controlling communications networks. Figure 3-1 shows the interrupt and entry point scheme for this event-driven control. Once the required information packet is constructed by a declarative macro, you can initiate an event via an imperative macro provided for the distinct application-oriented interface chosen.

Three distinct entry points in your program may be passed control by the ICAM actions, subsequent to an imperative macro (entry point A). An immediate return line (IRL) passes control to your program entry point B where cover registers are intact. When program B finishes its work and is in a waiting state pending scheduling of its completion address, it must execute a CYIELD imperative macro to release control. There are no parameters associated with CYIELD. A discussion on CYIELD and tasking procedures is contained in the fundamentals of ICAM user guide, UP-8194 (current version).

An imperative macro has an associated completion address (packet address or implied address) in your program (entry point C) to which control is passed whenever status presentation is necessary. IRLs gain control prior to a completion address in a contention situation (simultaneous occurrence of two events). The completion address is not restricted to a successful completion status and the functional interface determines what, if any, error status is presented at this entry point. You must reestablish the cover registers (registers 2 through 15) at entry point C.

Entry point D is an optional contingency or error processing entry point in your program. Normally, it is used to signal an inability to perform a required function. You must reestablish the cover registers (2 through 15) at entry point D.

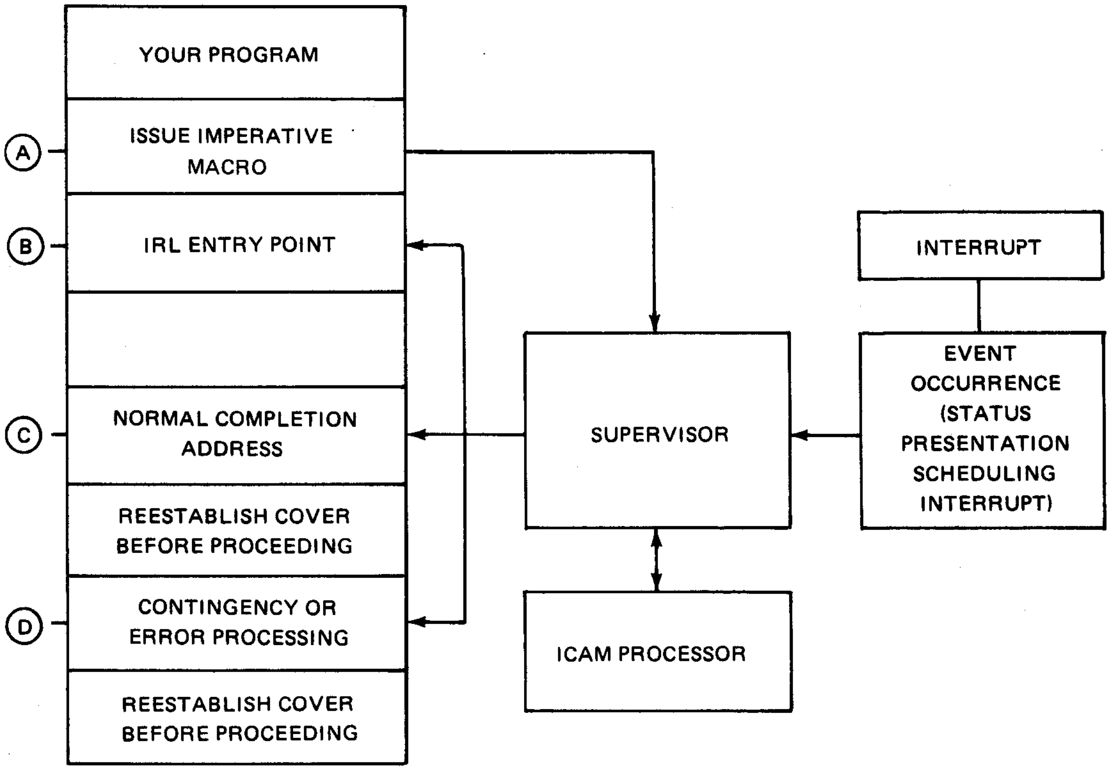
3.2. ACQUIRING AND RELEASING COMMUNICATIONS FACILITIES

The macroinstructions in this functional group are used to:

- acquire and initialize a network;

- begin processing for an entire network (a line in a network or one or more terminals on a line); and

- terminate processing for a network, a line, or one or more terminals.


YOUR PROGRAM
A
ISSUE IMPERATIVE MACRO
B
IRL ENTRY POINT
C
NORMAL COMPLETION ADDRESS
REESTABLISH COVER BEFORE PROCEEDING
D
CONTINGENCY OR ERROR PROCESSING
REESTABLISH COVER BEFORE PROCEEDING
SUPERVISOR
ICAM PROCESSOR
INTERRUPT
EVENT OCCURRENCE (STATUS PRESENTATION SCHEDULING INTERRUPT)

LEGEND:

Ⓐ At entry point of your user program, an imperative macro is issued.

The resultant interrupt passes control through the supervisor to an appropriate ICAM software processor.

Ⓑ If an immediate return line (IRL) capability is provided, control will be returned to your program entry point B with cover registers intact. No event occurrence is implied by the IRL. The execution of the CYIELD imperative macro suspends your program until control is passed to either your completion address or your contingency address.

Ⓒ When an event occurs that delivers normal completion status, the appropriate entry point C of your program will be passed control. Entry point C is not restricted to successful completions.

Ⓓ When a unique contingency occurs, control will be passed to entry point D if a unique contingency address parameter is provided.

Figure 3—1. Event-Driven Control of ICAM

LNEREL

3.2.1. Terminate Linkage (LNEREL)

Function:

Terminates the linkage between a channel and a line described in a communications network. Linkage termination occurs after validation of the LNEREL macro call. Any output messages that have not been processed remain on the output queues. If the line is later reopened via an LNEREQ macro call (3.2.2), any remaining output messages on the output queues will then be processed.

The error conditions that may be encountered during the execution of this instruction are described in Table A-6. This instruction is not supported in transient TCI.

Format:

```
LABEL          △OPERATION△      OPERAND
[symbol]         LNEREL         line-name
                                ⎡,MF=⎧⎛⎧C⎫ [,prefix-code]⎞           ⎫⎤
                                ⎢    ⎪⎜⎨D⎬               ⎟           ⎪⎥
                                ⎢    ⎨⎝⎩N⎭               ⎠           ⎬⎥
                                ⎢    ⎪⎛E,⎧parameter-list-label⎫⎞     ⎪⎥
                                ⎢    ⎪⎝  ⎩(1)                 ⎭⎠     ⎪⎥
                                ⎣    ⎩(L)                            ⎭⎦
```

Parameters:

line-name
: Identifies the line for which the LNEREL functions are to be initiated.

MF=
: Identifies a parameter list whose address may be explicitly specified or implicitly passed to the called routine through register 1. See S-type macroinstructions described in the fundamentals of ICAM user guide, UP-8194 (current version).

Example:

```
1        10     16
-----------------------
         LNEREL LNE3
```

3.2.2. Initiate Linkage (LNEREQ)

Function:

Initiates the required linkage between a channel and a line described in a communications network. This instruction affects all the terminals on a line, as defined in the network definition. On polled lines, this instruction initiates the necessary polling procedures.

The error conditions that may be detected during the execution of this instruction are described in Table A-7. This instruction is not supported in transient TCI.

NOTE:

The functions performed by this instruction can be initiated as a parameter to the MOPEN macroinstruction.

Format:

```
LABEL       △OPERATION△     OPERAND
----------------------------------------------------------------------
[symbol]      LNEREQ        line-name
                            ⎡,MF=⎧⎛⎧C⎫[,prefix-code]⎞              ⎫⎤
                            ⎢    ⎪⎜⎨D⎬              ⎟              ⎪⎥
                            ⎢    ⎨⎝⎩N⎭              ⎠              ⎬⎥
                            ⎢    ⎪⎛E,⎧parameter-list-label⎫⎞       ⎪⎥
                            ⎢    ⎪⎝  ⎩(1)                 ⎭⎠       ⎪⎥
                            ⎣    ⎩(L)                              ⎭⎦
```

Parameters:

line-name
: Identifies the line for which the LNEREQ functions are to be initiated.

MF=
: Identifies a parameter list whose address may be explicitly specified or implicitly passed to the called routine through register 1. See S-type macroinstructions described in the fundamentals of ICAM user guide, UP-8194 (current version).

Example:

```
1       10     16
-----------------------
        LNEREQ LNE1
```

NOTE:

All fields in the MTABLES macroinstruction, except those set at network generation, are cleared via a NETREL macroinstruction. Therefore, when a CUP issues a MOPEN macro after a NETREL macro before leaving main storage, you should be careful how to reestablish the fields in the TTT.

3.2.3. Network Initialization (MOPEN)

Function:

Activates previously defined CCA network and initializes the TTT entries. All functions of the MOPEN macroinstruction are automatically performed. ICAM requires that the tables generated by the MTABLE macro be in their initial state when MOPEN is executed. If a disk file has not been allocated or assigned, an error code is displayed. In resident mode, control returns to the user. In transient mode, ICAM is cancelled.

The error conditions that can be detected during execution of this instruction are described in Table A-10.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	MOPEN	network-name ,ERRET=error-address ,TCSADDR=table-name [,PASSWORD=password] [,LNEREQ={(YES) (NO)}] [,RESTART=YES] [,MF={((C D N)[,prefix-code]) (E,{parameter-list-label (1)}) (L)}]

Label:

[symbol]
Is an alphanumeric character string from one to four characters long that identifies and addresses the instruction.

Parameters:

network-name
Identiifes the 4-character name of the CCA. This must correspond to the name given at network generation time in the CCA macroinstruction.

ERRET=error-address

Identifies the symbolic address of the error return address if the network cannot be intialized. The error address must be in the root phase of the user program.

NOTE:

When an error occurs for which you receive control at ERRET, register 0 will contain the error.

TSCADDR=table-name

Identifies the symbolic name used in the label field of the MTABLE macroinstruction.

PASSWORD=password

Is a character string from one to eight characters long that identifies the password, if any, embedded in the network to be activated. The password in a network is specified by the optional PASSWORD parameter of the CCA macroinstruction. This parameter must be specified if, in fact, a password is in the subject network.

If omitted, the network must have been generated without a password specified for the MOPEN macroinstruction to be accepted.

LNEREQ=

where:

YES

Indicates that the line request (LNEREQ) functions are to be automatically executed for all the lines that are defined in the network. This specification must be used when using transient mode TCI.

NO

Indicates that none of the line request functions are to be initiated. Instead, an LNEREQ macroinstruction must be issued for each line to be activated.

If omitted, LNEREQ=(NO) is assumed.

RESTART=YES

Specifies that a warm restart is to be performed on all disk queues associated with the named network. This causes all queues to be refreshed to the last point at which a queue checkpoint was taken and the queues placed on hold. This parameter is not supported in transient TCI.

If omitted, a normal MOPEN is performed initializing all queues as empty.

NOTES:

1. *This parameter should not be used the first time a network is requested since there is no information on disk with which to refresh the queues.*

2. *When specified for a network with all main storage queues, the parameter is meaningless and is ignored.*

MF=

Identifies a parameter list whose address may be explicitly specified or implicitly passed to the called routine through register 1. See S-type macroinstructions described in the fundamentals of ICAM user guide, UP-8194 (current version).

NETREL

3.2.4. Release Network (NETREL)

Function:

Releases the facilities comprising a communications network. The releasing of facilities occurs after validation of the NETREL macro call. Anv outgoing messages that have not been processed remain on the output queues. However, these messages can be lost if the last user of ICAM does a NETREL since ICAM will terminate and go out of main storage. Use of the QDEPTH macroinstruction or programming delays prior to executing the NETREL macro will avoid loss of messages.

The error conditions that may be detected during execution of this instruction are described in Table A-8. This instruction is not supported in transient TCI.

Format:

```
LABEL       ΔOPERATIONΔ    OPERAND
[symbol]    NETREL         network-name
                           ⎡,MF=⎧⎛⎧C⎫[,prefix-code]⎞          ⎫⎤
                           ⎢    ⎪⎜⎨D⎬              ⎟          ⎪⎥
                           ⎢    ⎨⎝⎩N⎭              ⎠          ⎬⎥
                           ⎢    ⎪⎛E,⎧parameter-list-label⎫⎞   ⎪⎥
                           ⎢    ⎪⎝  ⎩(1)                 ⎭⎠   ⎪⎥
                           ⎣    ⎩(L)                          ⎭⎦
```

Parameters:

network-name
Specifies the network to be released as defined by the CCA macroinstruction.

MF=
Identifies a parameter list whose address may be explicitly specified or implicitly passed to the called routine through register 1. See S-type macroinstructions described in the fundamentals of ICAM user guide, UP-8194 (current version).

Example:

```
1         10     16
--------------------------
          NETREL NET1
```

3.3. SENDING AND RECEIVING MESSAGES

The macroinstructions comprising this functional group transfer messages between the ICAM and your CUP. If user-specified message size specifications are omitted, ICAM rejects the macro submitted and transfers control to the error address specified in the declarative control packet or returns to the user program in line.

3.3.1. TCI Declarative Macroinstructions

A single declarative macroinstruction (MTABLE) is required to generate the necessary transaction terminal tables (TTT), together with their associated control section and table extent areas for use of your program. These tables are initialized when the MOPEN imperative macroinstruction is executed by your program. The MOPEN automatically processes all NETREQ functions associated with ICAM.

A second declarative macro (DLIST) is available to be used in conjunction with the MWRITE imperative macroinstruction. It can be used to reference a list of destinations to which a message can be sent with a single macroinstruction. DLIST is not available in transient TCI.

MTABLE

3.3.1.1. Define the Transaction Tables (MTABLE)

Function:

Generates a set of tables based upon the number of terminals specified and, optionally, an extent area for your program. A control section is generated and contains scheduling addresses in your program.

Format:

LABEL	△OPERATION△	OPERAND
symbol	MTABLE	TERMS=number-of-terminals ,CCA=cca-name ,COMPL=label ,INOTICE=label ,ERROR=label [,SINPUT=YES] [,DELIVERY=label] [,PREVIEW=number-of-words] [,EXTAREA=number-of-words] [,DATIME=YES] [,SOURCE=YES] [,TRANSNO=YES] [,FREEINPT=YES] $\left[,\mathrm{OVERUN}=\begin{Bmatrix}\mathrm{YES}\\\mathrm{NO}\end{Bmatrix}\right]$ $\left[,\mathrm{DSECTS}=\left(\begin{Bmatrix}\mathrm{YES}\\\mathrm{NO}\end{Bmatrix},\begin{Bmatrix}\mathrm{LIST}\\\mathrm{NOLIST}\end{Bmatrix}\right)\right]$

Label:

symbol
: Is an alphanumeric character string from one to four characters long that specifies the address of the beginning of the table area. It is forced to a full-word boundary. This label must be specified since it is also required as a keyword parameter in the MOPEN macro.

Parameters:

TERMS=number-of-terminals
: Is a required parameter that specifies the number of terminals declared in the CCA network definition.

CCA=cca-name
: Is a required parameter that is the name (up to four characters) of your CCA network. It must correspond to the name given in the label field of the CCA macroinstruction.

COMPL=label

Is a required parameter that specifies the address (in your program) to be given control when a user's service request, such as MREAD or MWRITE, has been completed. You may regain control prior to completion by exercising the IRL option.

INOTICE=label

Is a required parameter that specifies your address to be scheduled each time that a message arrives in the system. When control is received at this address, register 1 is the address of the TTT with the message. The beginning of the text is in the previous area.

ERROR=label

Is a required parameter that specifies your address to be scheduled when a network type contingency occurs.

SINPUT=YES

Is an optional parameter specifying that you will not be scheduled at your input notification address for every input message. This requires you to survey your transaction control table control section input summary before every CYIELD.

DELIVERY=label

Is an optional parameter that specifies your address to be scheduled for each message after it has been transmitted from the system. Scheduling assumes delivery to a specified destination.

PREVIEW=number-of-words

Is an optional parameter that specifies the number of words, in the TTT, which contain the text of incoming messages for you to examine or preview. If not specified, a 3-word (12-byte) field will be generated.

EXTAREA=number-of-words

Is an optional parameter that specifies the number of words to be generated as a table extent area for each TTT entry. The address of this area is generated in the TTT when this parameter is specified; otherwise, the field has a value of zero. (This area is under user control.)

DATIME=YES

Is an optional parameter that, when specified, causes date and time stamping of incoming messages without using MPPS routines.

SOURCE=YES

Is an optional parameter that caues incoming messages to be stamped with the source terminal's name.

TRANSNO=YES

Is an optional parameter that causes incoming messages to be stamped with a unique transaction number code. This code is sequential and is incremented each time that a message is stamped.

FREEINPT=YES

Is an optional parameter that permits concurrent processing of multiple input messages from the same terminal by your program. The default for this parameter is controlled input, which means that subsequent input after the first will be inhibited by TCI until the response is completed for the previous input message. The controlled input requires an enable input indicator setting in the output prefix flag byte on an MWRITE macroinstruction or the issuance of an MALERT macro to enable input in order for your program to receive more input from the terminal.

OVERUN=YES

Is an optional parameter that permits you, by means of the preview area, to look at any input messages received while a previous input is still being held by ICAM.

Without this logic, TCI input logic rejects any and all input messages, without operator or user notification, from a terminal under the following conditions:

- You had previously inhibited input from that terminal by issuing an MALERT TSTATUS function with the inhibit flag set in TM#TTIUS of the TTT.

- An input message had been previously submitted by that terminal and ICAM still has it under its control since you have not retrieved it by means of an MREAD macro or canceled it by an MALERT CANCELIN function with the cancel flag set in TM#TTIUS of the TTT.

When input overrun is specified, it will only alter the second condition. The input logic in TCI will provide you with only a preview of the incoming message. If the incoming message text is larger than the preview area, you will see only the preview text portion. This message is known as an overrun message and is not subject to an MREAD nor can you cancel it. It is possible that a number of overrun messages may be received before a normal message is acknowledged by your program. In this case, the preview area in the TTT will contain the last overrun message received. You should be aware of the following information concerning the use of the overrun feature:

- You receive control at your notification address for a normal message and only the first overrun message.

- The input summary in the TTT control section is not incremented for overrun messages.

- You can detect overrun messages by continually surveying the input TCI flag byte TM#TTTCI for a TM#TTIOS flag bit setting.

- For overrun messages, TCI alters only the preview area and the input auxiliary function and device fields, and sets the overrun flag bit TM#TTIOS. It increments the input transaction count TM#TTIT and sets TM#TTNCE. All other fields of the TTT apply to the normal message.

- The normal message is not affected. It can still be retrieved or canceled.

- When an overrun occurs, the input auxiliary field refers to the overrun message. If you choose to continue with a normal message, its auxiliary fields will be passed.

- The overrun flag is never cleared by TCI. You must always clear it.

DSECTS= $\left(\begin{Bmatrix}\text{YES}\\\text{NO}\end{Bmatrix},\begin{Bmatrix}\text{LIST}\\\text{NOLIST}\end{Bmatrix}\right)$

Is an optional parameter that allows you to control the placement of the TCI DSECTS in your assembly, the listing or nonlisting of DSECTS in your assembly, and the presence of DSECTS within the range of your MTABLE macroinstruction.

DLIST

3.3.1.2. Distribution List (DLIST)

Function:

Same as 2.2.4 but available to the communications user program rather than at network definition time. Collectively identifies a list of destinations as a single destination. This macroinstruction enables a single user-specified instruction to denote a list of terminals as the destinations for a message. An MWRITE instruction that references a DLIST is equivalent to a group of MWRITE instructions that reference every destination in the DLIST, that is, where a separate transmission is performed for each terminal, or DLIST. Note that DLISTs cannot be nested more than once (i.e., a DLIST referenced by a DLIST cannot reference a third DLIST). A DLIST should not reference itself. This instruction is not supported in transient TCI.

Format:

LABEL	△OPERATION△	OPERAND
DLIST-name	DLIST	dest-name$_1$,dest-name$_2$,...dest-name$_n$

Label:

DLIST-name
Is a label up to four characters long that uniquely identifies the subject distribution list.

Parameters:

dest-name (1 through n)
Identifies the terminals, or DLISTs, that may be specified by another DLIST as the destinations for a message. At least two destinations must be specified in each DLIST.

Examples:

```
1        10    16
DLT1     DLIST TER1,TER2,DLT2,PFL1,DLT3
DLT2     DLIST TER3,TER4
```

NOTES:

1. *An implied DLIST, which is a PUTCP to either a CCA or a line name, is restricted.*

→ 2. *A DLIST with line queueing is also restricted.*

3.3.2. TCI Imperative Macroinstructions

The following imperative macroinstructions control message transfers between your program and the transaction control interface:

- MREAD

  This instruction causes a complete message to be transferred into your designated buffer area. A required parameter is the address of the associated TTT; the network buffers that originally contained the message are immediately released to the network buffer pool after the transfer to the user's area.

- MWRITE

  This instruction causes a complete message to be transferred from your designated buffer area into the designated message queue, either main or disk storage. Multiple queueing priorities may be designated for outgoing messages. The address of the associated TTT is a required parameter.

- MDEFER

  Your program is automatically given control to examine each message as it arrives in the system. You must respond to this scheduling by executing either the MREAD or the MDEFER macroinstruction. The MDEFER instruction causes the message to be written into a dedicated terminal slot on disk, or if disk is not specified, to be retained in the network buffer pool and linked into the associated TTT. A subsequent MREAD instruction is required to retrieve the message. No further input is solicited from the associated terminal until you respond to the input message. The address of the TTT is a required parameter.

- MALERT

  Your program may alert the TCI that there has been a change made in the TTT control indicators or that special functions are to be performed by TCI that are not related to any particular transaction.

- MOPEN

  This instruction activates both your previously defined CCA network and your TTTs. In addition, polling is optionally initiated on all lines in this network. ICAM requires that the tables generated by the MTABLE macro be in their initial state when the MOPEN macro is executed.

MREAD

3.3.2.1. Deliver Input Message (MREAD)

Function:

Transfers message text into the designated work area. Only complete messages are transferred with this instruction. The character count of the requested message is contained in the TTT. At MREAD execution time, the TM#TTAVL bit is reset by TCI in the submitted TTT.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	MREAD	$\left\{\begin{array}{l}\texttt{ttt-address}\\\texttt{(1)}\end{array}\right\}$

Label:

[symbol]
Is an alphanumeric character string one to four characters in length that identifies and addresses this instruction.

Parameters:

ttt-address
Is the symbolic label of a TTT entry. These reserved labels are generated automatically by the MTABLE macroinstruction and have the form TTTnnn, where nnn is the number of the terminal entry.

(1)
Indicates that the address of the TTT has been preloaded into register R1.

NOTE:

Prior to execution of the MREAD macroinstruction, the address of your input message prefix (Table A—4) with its attached text buffer must be placed in the TM#TTUWA field of the TTT. The input message prefix address must be aligned on a half-word boundary.

3.3.2.2. Transmit Output Message (MWRITE)

Function:

Transfers a message from the designated buffer area into one or more destination queues when output queueing is specified. If disk buffering is specified for output, the message is written into a designated message slot prior to transmission. In addition, the source terminal's input message may be canceled by this instruction if the previous text supplied information that meant an MREAD macro was not required.

Format:

LABEL	△OPERATION△	OPERAND

[symbol]	MWRITE	{ttt-address}
		{(1)}

Label:

[symbol]
Is an alphanumeric character string from one to four characters long that identifies and addresses this instruction.

Parameters:

ttt-address
Is the symbolic label of the ttt entry. These reserved labels are generated automatically by the MTABLE macroinstruction and have the form TTTnnn, where nnn is the number of the terminal entry.

(1)
Indicates that register R1 has been preloaded with the address of the TTT.

NOTE:

Prior to execution of the MWRITE macroinstruction, the address of your output message prefix (Table A—3) with its attached text buffer must be placed in the TM#TTUWA field of the TTT. The output message prefix address must be aligned on a full-word boundary. The output message prefix also contains a listing of the auxiliary device special function codes available to TCI.

The MALERT macroinstruction must be used in conjunction with some of the flag settings in the output prefix. For example, if TM#TDRSD is set in the output prefix, some future event (i.e., input) should set an MALERT SVC for retransmission or no retransmission. Only one output message per terminal may be held for retransmission.

MDEFER

3.3.2.3. Defer Input (MDEFER)

Function:

Causes an input message to be recorded in an associated disk message slot or, alternatively, to be linked from the associated TTT. This macro is executed in response to your automatic scheduling when an input message arrives in the system. This macro allows the main storage buffers in ICAM to be cleared until the message is cancelled or retrieved (via the MREAD macro).

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	MDEFER	{ttt-address (1)}

Label:

[symbol]
Is an alphanumeric character string from one to four characters long that identifies and addresses this instruction.

Parameters:

ttt-address
Is the symbolic label of a TTT entry. These reserved labels are generated automatically by the MTABLE macroinstruction and have the form TTTnnn, where nnn is the number of the terminal entry.

(1)
Indicates that the address of the TTT has been preloaded into register R1.

3.3.2.4. Control Indicator (MALERT)

Function:

Permits your program to alert TCI to a change in the control indicators in the TTT. These indicators may cause an output message to be retransmitted or an input message to be canceled, or may signal a change in the hold status of an output priority level (Table 3–1).

Format:

LABEL	△OPERATION△	OPERAND

[symbol]	MALERT	{ttt-address} , {function-symbol}
		{(1)} {(Ø)}

Label:

[symbol]
Is an alphanumeric character string from one to four characters long that identifies and addresses this instruction.

Parameters:

ttt-address
Is the symbolic label of a TTT entry. These reserved labels are generated automatically by the MTABLE macroinstruction and have the form TTTnnn, where n is the number of the terminal entry.

(1)
Indicates that the address of the TTT has been preloaded into register R1.

function-symbol
Is a symbol that indicates the type change in a transaction process or a change in the status of a TTT unrelated to any particular transaction.

(0)
Indicates that the function code for the appropriate symbol shown in Table 3-1 has been preloaded into the low-order byte of register R0.

NOTE:

Control is never received back at the SVC completion address. When an error is encountered, only in the RETRANS function will you receive control back at the contingency address with a 'No Message Available' status, and only if there was no message to retransmit. There is no return control under any other conditions although you can use an IRL.

Table 3—1. MALERT Function Relationships

Function Symbol	Function Code	TTT Flag Setting*	Description
RETRANS**	TM#TCART	N/A	Retransmit the last message to a terminal
NORETRAN*	TM#TCNRT	N/A	Do not retransmit the last message
CANCELIN	TM#TCANI	N/A	Cancel current input message from a terminal
ENABLIN	TM#TCAEI	N/A	Enable input from a terminal
TSTATUS	TM#TCTS	TM#TTHIN	Inhibit input from a terminal
TSTATUS	TM#TCTS	TM#TTHHY	Set high priority queue on hold for output
TSTATUS	TM#TCTS	TM#TTHMD	Set medium priority queue on hold for output
TSTATUS	TM#TCTS	TM#TTHLW	Set low priority queue on hold for ouput
TSTATUS	TM#TCTS	TM#TTRHY	Release high priority queue hold
TSTATUS	TM#TCTS	TM#TTRMD	Release medium priority queue hold
TSTATUS	TM#TCTS	TM#TTRLW	Release low priority queue hold
TSTATUS	TM#TCTS	TM#TTCAN	Cancel input
CLEARQUE	TM#TCQUE	TM#TTCHY	Cancel output on high priority queue
CLEARQUE	TM#TCQUE	TM#TTCMD	Cancel output on medium priority queue
CLEARQUE	TM#TCQUE	TM#TTCLW	Cancel output on low priority queue

*These flags must be set prior to issuing the MALERT macroinstruction.

**These functions have meaning only with output messages that were submitted with TM#TORSD set in the output prefix.

3.4. DISPLAYING AND ALTERING NETWORK STATUS

The macroinstructions composing this group are used to:

- suspend and resume the transmission of messages from queues;
- interrogate network status;
- alter polling sequences; and
- interrogate CCA tables.

Note that none of these instructions are available with transient TCI.

CCACPY

3.4.1. Copy Selected Network Information (CCACPY)

Function:

Retrieves line name, logical line number, terminal index, line terminal count, and device control tables from the communications control area (CCA) and stores the information in the specified user program location.

The error conditions that may be encountered during the execution of this instruction are described in Table A-12.

Format:

LABEL	△OPERATION△	OPERAND

[symbol]	CCACPY	{workarea$_1$-address,length$_1$,workarea$_2$-address,length$_2$}
		{(1)}

Parameters:

workarea$_1$-address
: Identifies the address in the user program at which the names of active terminals are stored. The end of the terminal list is marked by a word of hexadecimal F(FFFFFFFF$_{16}$).

(1)
: Indicates that register 1 contains the address of a user-generated parameter list.

length$_1$
: Indicates the number of bytes in workarea$_1$.

workarea$_2$-address
: Identifies the address in the user program where the CCA information is to be stored. This information is shown in Figures A-5, A-6, and A-7.

length$_2$
: Indicates the number of bytes in workarea$_2$.

Example:

```
1        10    16

READCCA  CCACPY TERMLIST,LTRMLST,CCAINF,LCCAINF
         CCACPY TLIST,LTLIST,CCADATA,LCCAD
```

QCLEAR

3.4.2. Clear Designated Queues (QCLEAR)

Function:

Clears queues of all messages without transmitting. All buffers are released. The function can apply to a single queue, the queues associated with a single terminal, or all queues associated with a line. This macro provides the same capability as the MALERT macroinstruction with function symbol of CLEARQUE. This instruction is not supported in transient TCI.

Error conditions that may occur during the execution of this instruction are described in Table A–12.

Format:

LABEL	ΔOPERATIONΔ	OPERAND
[symbol]	QCLEAR	⎧L,line-name ⎨Q,line-name,queue-name ⎨T,terminal-name [,{LOW MED HIGH}] ⎩(1) ⎫⎬⎭

Parameters:

L

Indicates that the clear function pertains to all the queues associated with all the terminals defined as part of the specified line (line name).

Q

Indicates that the clear function pertains to a specific queue (queue name) on an specific line (line name).

T

Indicates that the clear function pertains to all the queues associated with a specific terminal (terminal name) or to a specific priority queue of the specified terminal.

(1)

Indicates that the address of a QCLEAR parameter list is contained in register 1. The format of the parameter list is described in Figure A–8.

line-name

Identifies the line associated with the queue or queues.

terminal-name

Identifies the terminal associated with the queue or queues.

$\left\{\begin{array}{l}\text{LOW}\\\text{MED}\\\text{HIGH}\end{array}\right\}$

Indicates the low, medium, or high priority queue of the specific terminal.

queue-name

Identifies the name of the queue (a label up to four characters long) to be cleared.

Example:

```
1        10     16
------------------------------------
         QCLEAR Q,LNE3,Q0Ø3
         QCLEAR L,LNE2
         QCLEAR (1)
         QCLEAR T,TRM3,HIGH
```

QDEPTH

3.4.3. Queue Message Count (QDEPTH)

Function:

Indicates the number of messages in a single queue or in a group of queues. The number of messages in each queue is stored in a user-specified work area. The fields comprising this work area are illustrated in Figure A–8. This instruction is not supported in transient TCI.

The error conditions that may be detected during the execution of this instruction are described in Table A–12.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	QDEPTH	{Q,line-name,queue-name,workarea-name,workarea-length T,terminal-name,workarea-name,workarea-length (1)}

Parameters:

Q
: Indicates that a message count is desired for only a specific destination queue.

T
: Indicates that message counts are to be stored for all the destination queues associated with specified terminal.

(1)
: Indicates that the address of a QDEPTH parameter list identifying the queue to be examined and the work area address is contained in register 1. The format of the parameter list is described in Figure A–8.

line-name
: Identifies the line associated with the queue for which the message counts are to be obtained.

terminal-name
: Identifies the terminal for which queue message counts are to be obtained.

workarea-name

Identifies the label of the area in the communications user program where the message counts are to be stored (Figure A–8).

queue-name

Identifies the queue for which message counts are to be stored.

workarea-name

Same as parameter 3.

workarea-length

Indicates the workarea length in bytes.

workarea-length

Same as parameter 4.

Example:

```
1           10     16
--------------------------------------------------
            QDEPTH Q,LNE3,Q003,QSZ,WORKLEN
            QDEPTH (1)
```

QHOLD

3.4.4. Hold Output Transmission (QHOLD)

Function:

Suspends transmission of output messages from a queue or group of queues. The suspension can apply to a single queue, the queues associated with a single terminal, or all the queues associated with a line. The suspension remains in effect until a QRELSE macroinstruction is issued by the communications user program by the system operator via the system console. The user program (or the MPPS) may continue to transfer messages to a suspended queue. This macro provides the same capability as the MALERT macroinstruction with function symbol of TSTATUS and appropriate flag setting in the TTT. It is not supported in transient TCI.

The error conditions that may be detected during execution of this instruction are described in Table A-12.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	QHOLD	{ Q,line-name-queue-name T,terminal-name [,{HIGH LOW MED}] L,line-name (1) }

Parameters:

Q

Indicates that the hold function pertains to a specific queue (queue name) on a specific line (line name).

T

Indicates that the hold function pertains to all the queues associated with a specific terminal (terminal name) or to a specific priority queue of the specified terminal.

L

Indicates that the hold function pertains to all the queues associated with all the terminals defined as part of the specified line (line name).

(1)

Indicates that the address of a QHOLD parameter list is contained in register 1. The format of the QHOLD parameter list is illustrated and described in Figure A–9.

line-name

Identifies the line associated with the queue or queues.

terminal-name

Identifies the terminal with the queue or queues.

queue-name

Identifies the name of the queue (a label up to four characters long) to be held.

```
⎧HIGH⎫
⎨LOW ⎬
⎩MED ⎭
```

Indicates that the specified priority queue is to be suspended.

Example:

```
1           10     16
---------------------------------------------
            QHOLD  Q,LNE3,Q003
            QHOLD  L,LNE2
            QHOLD  (1)
            QHOLD  T,TRM3,HIGH
```

QRELSE

3.4.5. Resume Output Transmission (QRELSE)

Function:

Allows transmission of output messages from a previously held queue or group of queues. This macro provides basically the same capability as the MALERT macro with function symbol of TSTATUS and the appropriate flag setting in the TTT. It is not supported in transient TCI.

The error conditions that may be detected during execution of this instruction are described in Table A-12.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	QRELSE	{ Q,line-name-queue-name T,terminal-name [,{HIGH LOW MED}] L,line-name (1) }

Parameters:

Q

Indicates that a specific queue is to be released.

T

Indicates that all the queues associated with a specific terminal or with a specific priority of the named terminal are to be released.

L

Indicates that all queues associated with a specific line are to be released.

(1)

Indicates that register 1 contains a parameter list identifying the queue or queues to be released. The format of the QRELSE parameter list is described in Figure A-9.

line-name

Identifies the line associated with the queue or queues.

terminal-name

Identifies the terminal associated with the queue or queues.

queue-name

Identifies the name of the queue (a label up to four characters long) to be released.

$\left\{\begin{matrix}\text{HIGH}\\ \text{LOW}\\ \text{MED}\end{matrix}\right\}$

Indicates that the specified priority queue is to be released.

Example:

```
1            10      16
________________________________________
             QRELSE  L,LNE3
             QRELSE  T,TRM6
             QRELSE  (1)
             QRELSE  T,TRM3,HIGH
```

3.4.6. Change Telephone Number (TRMREP)

Function:

Changes the phone number in the line control table. The error conditions that may be detected during execution of this instruction are described in Table A–12.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	TRMREP	line-name,terminal-name,workarea-address ,FIELDS=(CALL)

Parameters:

line-name
: Identifies the line to which the subject terminal is connected.

terminal-name
: Identifies the subject terminal.

workarea-address
: Address of the work area in the CUP that contains a new phone number that replaces the number in the line control table. The phone number in the line control table was taken from the CALL parameter of the LINE macroinstruction.

FIELDS=(CALL)
: Replaces the phone number in the line control table phone directory with a new number residing in the CUP work area.

The phone number information in the CUP work area (Figure 3–2) starts at CUP work area +29. The first byte contains the total count, in binary, of the number of dialing digits in the phone number. The phone number then follows in decimal characters. The phone number cannot exceed 48 characters in length. (The number of characters in the new phone number must equal the number of characters in the number it replaced.

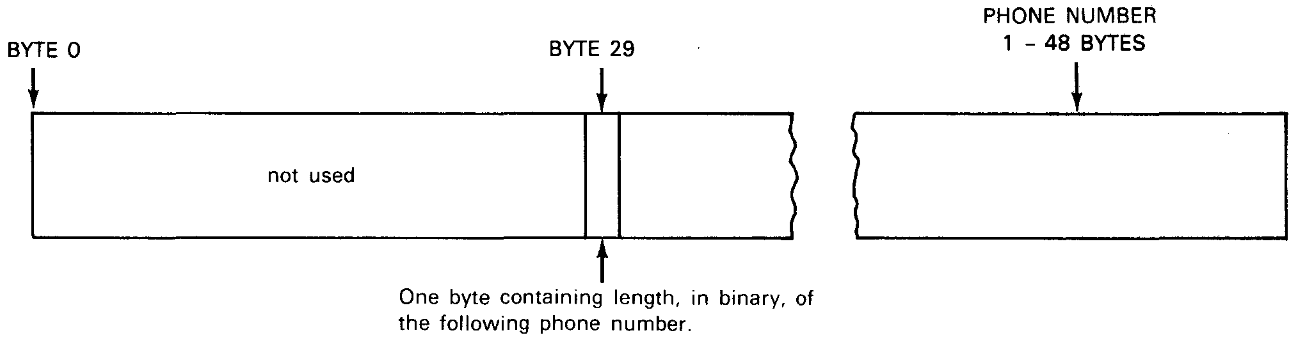
When auto dialing is used, you can put a hyphen in the normal place in the phone number to cause a 1.1-second pause. This delay allows a connection to be made before more dial characters are sent.

Example:

```
1        10    16
_____________________________________________________
         TRMREP L014,T003,TRWM,FIELDS=(666—5708)
```

NOTE:

The TRMREP macroinstruction must be issued after network activation to an inactive line.



One byte containing length, in binary, of the following phone number.

Figure 3—2. TRMREP Work Area Functional Field Description

3.5. RELINQUISHING AND ACQUIRING COMMUNICATIONS CONTROL

The CYIELD macroinstruction is issued by a CUP program to relinquish control following a series of MREAD/MWRITE/MDEFER/MALERT macroinstructions that had the immediate return line set (IRL). CYIELD is not necessary if one of these macros is issued without IRL set.

The CYIELD macroinstruction (3.5.1) passes control to ICAM activity control via the supervisor. If an activity is outstanding for that task, your program is immediately awakened (scheduled) and given control via the supervisor. If no activity is outstanding, the task is suspended. You may activate (awaken) the communications task at any time by executing the CAWAKE macroinstruction (3.5.2) from any task. The occurrence of an activity within ICAM for the communications task activates the task, provided that it is in an idle (CYIELD) condition.

It is recommended that all noncommunications I/O be performed under a separate task (a task other than a communications task) so that communications activity directed to the communications task by ICAM will continue to flow while printer/reader/disk I/O is being performed by a subtask. Synchronization is accomplished between your communications task and your subtask via the CAWAKE macro.

For your TCI CUP, the only time you will be given control following the CYIELD macroinstruction is if a CAWAKE macro is issued from another task. This is done because the TCI always gives you control at one of the unique identification addresses supplied in your MTABLE macro. For example, COMPL=/INOTICE=/ERROR=/DELIVERY=.

3.5.1. Release of Control (CYIELD)

Function:

When you wish to release control of your program to wait for I/O completion, you must enter a CYIELD condition either by issuing the imperative macro MREAD/MWRITE/MDEFER/MALERT with its associated information packet not set for immediate return line (IRL), or by issuing a CYIELD macro. If control is not released properly, your program will not be reactivated (scheduled) after completion of I/O sequences.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	CYIELD	

Label:

[symbol]
Is an optional alphanumeric character string (symbolic name) from one to eight characters long that identifies the specific instruction.

CAWAKE

3.5.2. Awake Communications Task (CAWAKE)

Function:

If your program is operating in a multitask environment, any noncommunications task or island code may need to awaken (activate) a communications task. This can only be done by issuing a CAWAKE macro with the communications task in a CYIELD condition. This will give the communications task control at the instruction immediately following the macroinstruction that put the communications task into the CYIELD condition.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	CAWAKE	

Label:

[symbol]
Is an optional alphanumeric character string (symbolic name) from one to eight characters long that identifies the specific instruction.

3.6. EXAMPLES OF TCI COMMUNICATIONS USER PROGRAMS

In Section 2, we constructed typical networks using network definition macros. We can now use the network shown in Example 4 (Figure 2–4) to execute a simple communications user program (CUP) for TCI. First, define the input file information by using the declarative macro MTABLE. We have previously defined five terminals, and input completion address is INCMPL, and input notification is INSTRT. We also have named the error routine address INBUG. Thus, the declarative macro is:

```
1       10    16                                                      72
-------------------------------------------------------------------------
TCIDTFI  MTABLE TERMS=5,COMPL=INCMPL,INOTICE=INSTRT,ERROR=INBUG
```

The labels specified are defined in the executable code as the labels of the specific routines desired.

The declarative macro itself is specified in the nonexecutable portion of the coding. In the executable portion of the code, we need to request the desired network and line for input:

```
         MOPEN QNET,TCSADDR=TCIDTF1,ERRET=INERRET
```

We must next set the address of the input message prefix into the terminal table and set the byte count in the message prefix:

```
         MVI   TM#TTUWA,TM#TIMSG-2
         MVI   TM#TICNT,1Ø96
```

These statements, along with the imperative macro MREAD or MDEFER, are placed in the appropriate sequence in the applications coding to perform reception of data. You determine which function to use through examination of your preview area. This is explained further in Example 2. When these instructions are executed, you will receive input completion at the label you specified in your applications program.

For output, let's expand the MTABLE to include output delivery notification. Our macro might then look like:

```
TCIDIFI  MTABLE TERMS=5,COMPL=INCMPL,DELIVERY=DNOTICE                X
               INOTICE=INSTRT,ERROR=INBUG
```

Again, let's set the address of the output message prefix into the transaction terminal table (TTT) and set the byte count in the message prefix:

```
         MVI   TM#TTUWA,TM#TOMSG-2
         MVI   TM#TOCNT,1Ø96
```

An additional function for output is to specify the terminal to which we wish to send the message, such as

```
1        10    16
_________________
MVC      TM#TODID,TERM1
```

where we have defined TERM1 to be the label of a terminal previously defined in the network definition.

In this example, we have also stated that we wanted delivery notice; therefore, we would have to specify some identifier in the constants and place this identifier in the message prefix. For example:

```
MVC     TM#TOPAR,DELNOTE
```

Now, all we need for output is the imperative macro MWRITE. These statements also are placed in the appropriate sequence in the applications coding to perform transmission of data.

To illustrate this, all of the statements described previously for input and output are shown in example 1:

Example 1:

```
        START
        BALR   R12,Ø
        USING  *,R12
        MOPEN  QNET,TCSADDR=TCIDTF1,ERRET=INERRET,LNEREQ=(LØØ1)
        .                             (Indicates user
        .                             {application
        .                             (code.
        BALR   12,UREAD1
        .                             (Indicates user
        .                             {application
        .                             (code.
        BALR   12,UWRITE1
        .
        .
        .
UREAD1  DS     0H
        MVC    TM#TTUNA1,TM#TIMSG-2   Set input message prefix
        MVI    TM#TICNT,1Ø96          Set byte count.
        MREAD  (1)
        BR     12
UWRITE1 DS     0H
        MVC    TM#TTUWA,TM#TOMSG-2    Set output message prefix.
        MVI    TM#TOCNT,1Ø96          Set byte count
        MVC    TM#TODID,TERM1         Set terminal
        MVC    TM#TOPAR,DELNOTE       Set delivery notice
        MVC    TM#FLG,TM#TOENI        Enable input
        MWRITE (1)
        EOJ
```

```
1        10    16                                                     72
TCIDTFI  MTABLE CCA=QNET,TERMS=5,COMPL=INCMPL,INOTICE=INSTRT,          X
                ERROR=INBUG
DELNOTE  DC    C'MSG1'
         DC    Y(TM#TIMSG)
TM#TIMSG DS    1F
WORKIN   DS    256F
         DC    Y(TM#TIMSG)
TM#TOMSG DS    1F
WORKOUT  DS    256F
TERM1    DC    C'U1H1'
TERM2    DC    C'U1H2'
TERM3    DC    C'U1H3'
TERM4    DC    C'U1H4'
         END
```

Note that the input message prefix and output message prefix, along with the respective text buffer addresses, are defined within the user area. In addition, the names of the terminals defined by the network definition are specified in the CUP in order that they can be appropriately used in the output message prefix for transmission of data.

If we had constructed the network using auxiliary devices (as shown in Figure 2-4) and wanted to transmit to one of these devices, we would have to perform two simple additional functions prior to the MWRITE:

1. Enter the auxiliary device number into the TM#TODEV field of the output message prefix, as it was previously specified via the AUXn parameter of the TERM macro. For instance, if we had chosen to send (write) this data to the first TCS drive on the terminal labeled U1H1 (AUX1), we would have coded a BAL instruction such as:

   ```
   MVI   TM#TODEV,1
   ```

2. Enter the appropriate auxiliary device command into the TM#TOFVN field of the output message prefix, as listed in Table A-3. In this case, since we have chosen to write a block, the command is TM#TORDB and we would have to write a BAL instruction like this:

   ```
   MVI   TM#TOFUN,TM#TORDB
   ```

For delivery notice we could define a constant such as:

```
DELNOTE  DC    C'MSG△'
```

Each time we prepare to send a message, we could increment the last characters before setting it into TM#TOPAR of the output message prefix. This field is then returned in register 0, while the delivery status and TTT address are contained in register 1 (3.13). Input notification and SVC completion processing are described in 3.14 and 3.15. If we wanted to make use of the preview feature, we could use an identification of BRISTOLAB in the constants. So we set aside 5 words (20 bytes) for the preview area in the input MTABLE. The first 20 bytes of an incoming message are then automatically placed in the TM#TTEXT field for examination. This could easily be accomplished through a BAL instruction such as:

```
10    16
-----------------------
CLC   TM#TTEXT,RECORDID
```

where RECORDID is the label of the preview identifier. This comparison could then be used to determine whether we wished to defer this record for later processing by placing it on disk via the MDEFER macro or process the entire message by means of the MREAD macro.

Example 2 shows the use of some of the alter/display macros and how they can be employed within a CUP. The input is determined to be processed or deferred. If it is to be processed, the text is read. On successful completion, the CCACPY macro is used to copy information concerning the line and terminal into the user area for a history file. This gives a parameter list, terminal name list, and CCA information list. QCLEAR is then used to clear any queues before releasing the network.

Example 2:

```
1        10    16                                                      72
--------------------------------------------------------------------------
HERE     START Ø
         BALR  2,Ø
         USING PREVUE,2
PREVUE   CLC   TM#TTEXT,RECORDID
         BNE   LATER
UREAD1   MOPEN QNET,TCSADDR=TCIDTF1,ERRET=INERRET
         MVC   TM#TTUWA,TM#IMSG-2
         MVI   TM#TICNT,1096
         MREAD (1)
INCMPL   .                           ⎧Application Text
         .                           ⎨Processing
         .                           ⎩Routine
         CCACPY TRMNAME,EIGHT,INDATA,SIXTEEN
         QCLEAR T,U1H1
         BE    ENDJOB
INERRET  .                           ⎧Application Error
         .                           ⎨Processing
         .                           ⎩Routine
INBUG    .                           ⎧Application SVC
         .                           ⎨Completion
         .                           ⎩Processing routine
LATER    MDEFER (1)
ENDJOB   NETREL
         EOJ
         DC    Y(TM#TIMSG)
```

```
1        10    16
TM#TIMSG DS    1F
WORKIN   DS    256F
         DC    Y(TM#TOMSG)
TM#TOMSG DS    4F
WORKOUT  DS    256F
TCIDTFI  MTABLE TERMS=1,CMPL=INCMPL,INOTICE=PREVUE,ERROR=INBUG
EIGHT
SIXTEEN  DC    F'16'
TRMNAME  DS    XL'8'
INDATA   DS    XL'16'
         END
```

3.7. CONCURRENT TRANSACTION PROCESSING

Concurrent transaction processing is the ability of your program to simultaneously activate and control more than one transaction at a time. Since this capability is applications oriented, it may be assumed that there is no relationship between terminals and application functions. Further, it is assumed that applications are identified by command functions or identifiers, which are embedded in message text and are, therefore, subject to your control.

The ICAM/TCI communications interface attempts to keep all message partitions filled with incoming messages by continuously soliciting input. You may stop input at any time by setting an indicator in the transaction terminal table. As each message arrives in the system, it is submitted to your program via the transaction terminal table for examination. The program may accept the message for immediate processing or may cause it to be recorded in the appropriate message slot for deferred processing. When disk staging is specified, deferred processing results in the message being written into the logical disk record that is associated with the originating terminal.

3.8. IMMEDIATE RETURN LINE PROCESSING

The immediate return line (IRL) feature of TCI permits your programs to operate in a completely asynchronous environment with a single communications task.

To exercise the IRL feature, merely set an indicator when executing an MREAD/MWRITE/MDEFER/MALERT macro. When set, you will receive control back at the point where the SVC was executed and with the calling environment restored. The completion of the requested function is scheduled at an address that is given in the control section of your transaction table area. If the IRL indicator is not set, your program is suspended pending completion of the requested function; however, scheduling still occurs at the address specified in the control section of the table area when completion occurs. The SVC completion address given in the control section of your transaction table may be modified prior to executing any of the service calls (SVC) just mentioned. The IRL indicator is never reset by TCI. You must reset it at all times.

Caution must be exercised when using the SVC call with an IRL. Flag indicators in your transaction tables and message information in your message prefixes established prior to executing the SVC should not be redefined until SVC completion. This is because of delays in ICAM caused by disk I/O in processing the SVC.

3.9. ACTIVITY SCHEDULING

All activities of your program that are scheduled by TCI are placed on a user queue on a first-in, first-out basis. When your communications task is active (busy), no scheduling action is performed by TCI. When releasing control, your communication task must execute a CYIELD macroinstruction. This passes control to TCI, which then activates the next outstanding activity or an SVC without IRL set, if any, for that task. When a requested function reaches completion and your task is idle, your program is activated immediately. User programs are permitted to have only a single task to interface with any ICAM communications.

NOTE:

Completed activities may not be scheduled back to your program in the order that they were initiated. You must examine both registers 1 and 0 to determine the activity completed.

3.10. DISK BUFFERING AND MESSAGE PARTITIONS

A terminal-related message partition is defined and allocated for each terminal that is associated with a TCI network configuration. Associated with each disk-based message partition is a transaction terminal table (TTT) entry that is related to a specific terminal. These table entries are generated in a contiguous manner and are logically numbered from 1 to n. The value is directly related to the order in which terminals are defined in the standard network generation macroinstructions. The value is not related to the communication line or lines.

When you select disk buffering at ICAM generation time, a disk-partition file is created by using the system access technique (SAT). The number of message slots created in the SAT partition depends upon whether disk buffering is specified for input only or for both input and output. This file must be created and allocated by your job stream.

For resident TCI, a single input message slot is created for each terminal in your network definition. For transient TCI, nine message slots are created for each terminal: one for input and eight for output.

The eight output message slots are arranged in a priority configuration of high, medium, and low. The first two slots of the eight are top and high, respectively, the third is medium, and the remaining five are low priority. You may place any of the three priorities on hold by setting appropriate indicators in the transaction terminal table.

3.11. USER CONTROL TABLES

Message transfers are controlled via the TTT. A single entry is required for each terminal that is defined in your network definition. A control section is defined for the set of terminal entries that contains scheduling information for the TCI program.

3.12. CONTINGENCY NOTIFICATION PROCESSING

TCI schedules your contingency notification address whenever it is determined that it cannot recover from an error situation. Under some conditions, when ICAM determines that neither you or it can recover from an error condition, ICAM will cancel your job. The cancel codes indicating the reason for the error are contained in Table A-13, as well as in the systems messages programmer/operator reference, UP-8076 (current version).

The error codes which will cause a return to your contingency address are contained in Table 3-2.

Table 3—2. Contingency Error Codes

Label*	Comment	R1 Value
ELDW	Line disconnected**	Line name (from CCA)
ELER	Line down-not disconnected**	Line name (from CCA)
EODE	Disk error during output**	TTT-address
ENMA	User issued MALERT for RETRANS but no message available to retransmit	TTT-address

*All labels have the prefix TM#T. Flags are passed in byte 0 of R0.
**TCI notifies console operator of error condition also.

3.13. DELIVERY NOTIFICATION PROCESSING

This optional feature in TCI gives you the ability to provide your own journaling or history of output messages. It also gives you the ability to receive notification of output errors received while attempting to send output to auxiliary devices. (See word 3 of A.1.3.)

You always receive the TTT address of the output destination in R1 and a status flag in the upper byte. If the flag TM#TDNEN is set, the message was transmitted successfully. The output message is always released from ICAM control at this point unless you had specified the message to be held for retransmission. In this case, ICAM will not look at the message again until you issue an MALERT. If you elect to issue an MALERT for retranmission of the message, you will receive output delivery notification for the second time. The message will then be removed from ICAM control and will not be subject to retransmission. If you had issued an MALERT not to retransmit, the message at that time is removed from ICAM control.

If the flag byte is not equal to the value of TM#TDNEN, it indicates that the output was being delivered to an auxiliary device on that terminal but the auxiliary device would not accept the output. In this case, the message is always held until the error is corrected and the message sent successfully. (See Table 3-3 for a description of the delivery notice flag settings contained in the upper byte of R1.) If byte 0 indicates a TM#TDNAX error, the auxiliary device is presented by modifying the bits of the applicable TM#TDDS1, TM#TDDS2, TM#TDDS3, and TM#DDS4 status via the OR assembler instruction.

Table 3—3. Delivery Notification Error Flags

Label*	Comment
DDNA	Terminal marked down — message held.
DNAX	Auxiliary device down — message held.
DDS1	UNISCOPE auxiliary status 1 — message held. Unexplainable status; TCS function inoperative.
DDS2	UNISCOPE auxiliary status 2 — message held. Printer out of paper, inoperative or in test mode; tape cassette end-of-tape. For end-of-tape, you perform a rewind with a mode 0 search to address t0000, where t is track 1 or 2. The rewind must be followed by a report address if further tape operations are to be performed. The rewind and report address must be at a different priority queue than the queue used for the TCS function, which was not performed because end-of-tape was reached. The report address results in input that causes the TCS function (which failed because of end-of-tape) to be sent to the terminal.
DDS3	UNISCOPE auxiliary status 3 — message held. Data error on TCS. Several attempts at backward-one-block and repeat of the TCS function have been made by RDH to no avail. The number of attempts at retry is specified by the LINE macro of the CCA. Default is 4.
DDS4	UNISCOPE auxiliary status 4 — message held. No response. May be disconnected or trying to read a blank tape.
DOAB	Abort output/break received.
DNEM	Successful output completion.
DLNO	Line down/disconnected.

* All labels have the prefix TM#T.

We recommend that you send your auxiliary device output at a different priority than your primary device output. This is because you may wish to perform auxiliary device error recovery by sending control messages to the primary. These messages can only be retrieved from the output queues if on a different priority level than the auxiliary device output messages.

NOTE:

Delivery notification of output errors is not performed by TCI unless you had specified the OUTDELV parameter on the FEATURES keyword of the CCA macro. You must also have specified the DELIVERY=address parameter of the MTABLES macro and not set the TM#TOSDN flag of the output message prefix when you issue MWRITE. If all of these conditions are not met, delivery notification will not occur.

3.14. INPUT NOTIFICATION PROCESSING

When you receive control for input processing, byte 3 of register 0 will contain error statuses. Input error codes are described in Table 3-4. A status of 0 in byte 3 indicates successful input. These flags are also located in TM#TTTCI of the TTT and these flags remain there until the input message is read or cancelled.

Table 3—4. Input Notification Error Codes

Error Label *	Comment
IPE	Input parity/block check error
RUN	Truncated input message
EOF	Batch end-of-file

*All labels have the prefix TM#TT.

3.15. SVC COMPLETION PROCESSING

All M-function SVC macroinstructions except the MOPEN and MALERT macroinstructions return to the user-supplied SVC completion address as defined in the transaction control section of your MTABLE macro. You may dynamically alter this address field prior to issuance of an SVC macroinstruction. The MREAD, MWRITE, MALERT, and MDEFER macroinstructions require a TTT address as a parameter in R1. At SVC completion, the TTT address passed in R1 upon entrance is returned in R1. R0 at return will be 0 if successful with no errors. Error codes returned in R0 are specified in Table 3-5.

Table 3—5. SVC Completion Error Codes

Label*	Comment
Bytes 0 and 1 (Fatal Errors – SVC Not Completed)	
EBND	Boundary error; table alignment
ELMT	Limits error; address not in job region
EDCD	Decode error; invalid function
ENMA	No message available
EDST	Missing or invalid destination
ENBA	To be supplied
Bytes 2 and 3 (Nonfatal Errors – SVC Completed)	
EIPE**	Input parity/block check
ETRN**	Truncated input message
BEOF	Batch end-of-file
EIPR†	Parity error

* All labels have the prefix TM#T.

** Input flag values may range from 10_{16} to $3F_{16}$.

† Output flag values may range from 40_{16} to $6F_{16}$.

Since the SVC completion address may be common to all the TCI macroinstructions, your program can decode the upper byte of R1 to determine which macroinstruction is being returned. This may be determined by comparing the upper byte with the DSECT label associated with a particular macroinstruction (Table 3-6).

Table 3—6. DSECT/Macro Function Relationships

DSECT Label*	Associated Macro
CRD	MREAD
CWRT	MWRITE
CDFR	MDEFER
CALT	MALERT
CSWT	MSWITCH

* All labels have the prefix TM#T.

3.16. BATCH MODE PROCESSING

TCI provides you with the ability to read cards from a batch terminal and send print or punch images to a batch terminal. You are made aware of being in batch mode by the setting of the TM#TTBAT flag after execution of the MOPEN macro.

You receive an input batch end-of-file indication when the last card image is received from a batch terminal. Conversely, you must set the output batch end-of-file in your output prefix when you send the last output image to a batch device.

3.17. DLIST/MULTIPLE DESTINATION ROUTING

ICAM provides for the routing of messages to more than one destination within the communications network. A message may be directed to as many destinations as the configured resources can handle. This feature is not supported in transient TCI. Three methods have been provided to enable the ICAM user to direct messages to multiple destinations:

1. CCA-defined DLIST

2. User-defined DLIST

3. Implied DLIST

Through the TCI interface the ICAM user program may direct a message via a DLIST to multiple destinations using the MWRITE macro. The user program must set up in the destination identifier field (TM#TODID) of the output message prefix the name of a CCA-defined DLIST, the name of a CCA or line, or the address of a user-defined DLIST. The TTT address specified on the MWRITE will be used to access the user message address.

Using the MPPS DIRECT and ROUTE macros, messages may be sent to a DLIST or multiple destinations. Only CCA-defined DLISTs are available for use by the MPPS macros.

An invalid destination specified in DLIST will cause the message transfer to be aborted and the message will not be delivered to any destinations specified in the list.

3.17.1. CCA-Defined DLIST

A CCA-defined distribution list is a list of destinations to which a message may be directed. This is a static list built at CCA generation time and is unchangeable for the life of the CCA.

3.17.2. User-Defined DLIST

The DLIST declarative macro may also be used within the user program or the user may construct his own DLIST dynamically. The format of the DLIST defined within the user region is shown in Figure 3–3.

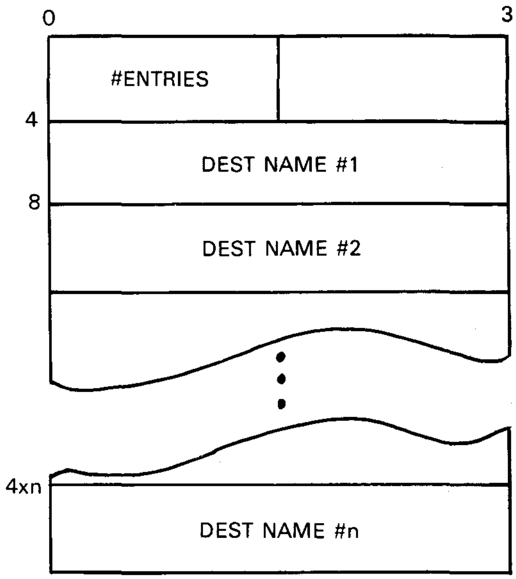


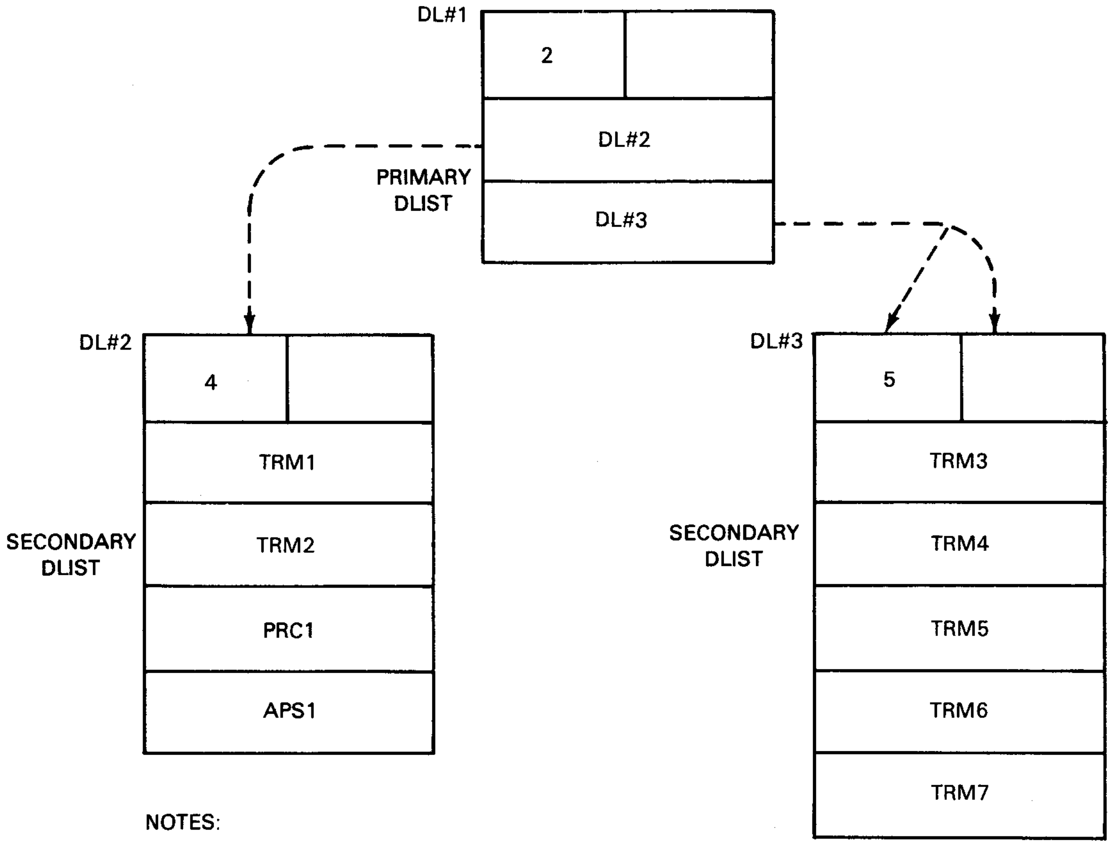
Figure 3—3. DLIST Format

3.17.3. Implied DLIST

ICAM allows the user to direct messages to a CCA or to a line defined within the CCA. When a message is directed to a CCA/line, it will be delivered to all the terminals in the CCA/line with which the user is currently in session. This feature is not supported in transient TCI.

3.17.4. Nested DLIST

ICAM allows for one level of nesting within a DLIST (Figure 3-4). The primary or first DLIST may have as possible destinations other DLISTs defined within the CCA. Secondary level DLISTs may not point to a tertiary DLIST, thus limiting nesting to one level.



NOTES:

DL#2, 3 are names of DLISTs.
TRM1, 2, ..., 7 are names of terminals.

Figure 3—4. Example of Nested DLIST

3.18. SPECIAL CONSIDERATIONS

3.18.1. Teletypewriter Restrictions

If a teletypewriter operator terminates input from the keyboard with an EOT character (disconnect function), a single character text message will be sent to the user at his input notification address and the preview area will contain the EOT character. This is considered a real input message and is subject to the same processing as other input messages. In addition, the user will receive control at his contingency address with a line down-disconnected status. This will enable him to issue a LNEREL and a LNEREQ macro to reestablish the necessary controls in case the line is configured for unattended answering.

3.18.2. Terminal Restrictions

Each input message (inquiry) from a terminal requires an output (reply) message to that terminal before any additional input will be accepted by the ICAM/TCI interface (from that terminal).

3.18.3. CUP Initialization and Termination

When a TCI CUP attaches to ICAM via the MOPEN macro, it must be guaranteed that ICAM has been loaded into main storage. The following sequence of events describes a CUP initialization and termination:

1. The console operator loads ICAM.

2. CUP1 issues a MOPEN macro to its dedicated CCA.

3. CUP2 issues a MOPEN macro to its dedicated CCA.

4. CUP1 terminates, but CUP2 stays active.

5. If CUP2 terminates, ICAM exits, or CUP1 could issue another MOPEN macro to its dedicated CCA and start processing again.

A termination macro, such as EOJ, will force a network release if a NETREL has not been previously issued.

3.19. USER PROGRAM DATA FORMAT CONSIDERATIONS

The remote device handlers (RDHs) create all required control characters and text delimiters on output messages. These control and text delimiters are removed from incoming messages. In addition, all needed code translations are performed by the RDHs.

3.19.1. Forms and Screen Control of Remote Terminals

The RDHs offer you two methods of causing functions as line feeds, form feeds, carriage returns, or cursor position control at a particular row and column on a display (CRT screen). Your program can be device (terminal) dependent by embedding a character string within your text that will translate to unique control characters for a particular terminal as has been traditional user responsibility in the past. The second alternative is to use an ICAM device independent control expression (DICE) word.

3.19.2. Device Independent Control Expressions

A device independent control expression (DICE) comprises a 4-character sequence found in the text portion of a message. On output, you place device-independent control expressions into the message text. On input, the remote device handlers are responsible for recognizing and converting device-oriented control information into an equivalent device independent control expression.

Format:

select character	function code	m field	n field

where:

select character
Is a unique character designating the start of a DICE. This character, a data link escape (DLE) control character in EBCDIC (10_{16}), must be used only to designate the start of a DICE.

function code
Is a 1-byte field defining the operation to be performed on output, or defining the device control sequence recognized on input. (Valid DICE function codes are listed in Table 3–7).

m field and n field
Qualify the DICE function code. These fields are treated as parameters to the DICE function code; their actual definition varies and is determined by the individual DICE function.

3.19.2.1. DICE Functions

The appropriate ICAM RDH replaces each output DICE with the position control sequence required by the terminal to produce the DICE-defined positioning. The appropriate ICAM RDH replaces device-oriented control information from the terminal with an input DICE. Table 3–7 lists the DICE function codes for output and input, along with the appropriate user commands for each function code. These commands can be used in a DICE generation macro.

Table 3—7. DICE Function Codes

DICE-Function Command	Value	Function	Output	Input
ZO#COORD	01_{16}	Set coordinates	X	X
ZO#FORM	02_{16}	Forms control	X	X
ZO#FORMA	$0A_{16}$	Forms control with clear (protected/unprotected)	X	
ZO#ERSLN	$0B_{16}$	Erase to end of line	X	
ZO#FORMC	03_{16}	Forms control with clear (unprotected)	X	
ZO#POS	04_{16}	New line control	X	X
ZO#POSC	05_{16}	New line control with clear	X	
ZO#CUR	06_{16}	Current position control	X	X
ZO#CURC	07_{16}	Current position control with clear	X	
ZO#BEG	08_{16}	Beginning of current line control	X	X
ZO#TABS	09_{16}	Set tab stop at an absolute position	X	

Example:

```
    1        10    16
①  NEWLINE  ZO#POS 0,0
②  COORD1   ZO#COORD 5,10
```

① This DICE sequence causes movement to a new line.

② New text will be started at line 5, column 10, due to this DICE.

3.19.2.2. DICE Code Generation

Procs are provided for IMS and user programs to generate the device independent control expression codes.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	DICE-function	m,n

Label:

[symbol]
An optional alphanumeric character string from one to eight characters long that identifies the specific instruction line.

Operation:

DICE-function
The user specifies the appropriate name from the command column of Table 3–7 for the desired DICE function.

Parameters:

m
A decimal number (0–255) indicating the number of lines or rows (horizontal) the terminal should advance before starting output of the message (Table 3–9).

n
A decimal number (0–255) indicating the number of spaces or columns (vertical) to the right the terminal should space before starting output of the message (Table 3–9).

3.19.2.3. Interpretation of DICE

In using DICE, the user program does not need to be aware of the terminal type. A particular DICE denotes the same positioning on any terminal. There are some exceptions that result from limitations of the terminal.

The interpretation of a DICE by the RDH is controlled by the following factors:

1. DICE function code

2. DICE m and n fields

3. The terminal involved

4. The particular device on the terminal being used

The ICAM RDHs currently provide device-independent support for three classes of remote terminal devices:

1. Hard copy character-oriented devices, such as the SPERRY UNIVAC Data Communications Terminal 475 (DCT 475), Data Communications Terminal 500 (DCT 500), Data Communications Terminal 524 (DCT 524), and Data Communications Terminal 1000 (DCT 1000), and TELETYPE* teletypewriter models 28, 32, 33, 35, and 37.

2. Hard copy page printer devices, such as the SPERRY UNIVAC 1004 Card Processor System, Data Communications Terminal 2000 (DCT 2000), and 9200/9300 Systems, and the IBM 2780.

3. CRT-type terminals, such as the UNISCOPE 100 and 200 Display Terminals, and the UTS 400.

Table 3–8 defines the primary output device and the primary input device for each terminal type. Tables 3–9 and 3–10 contain the interpretation of DICE from the viewpoint of the primary device of the remote terminal.

Table 3—8. Primary Devices

Terminal	Primary Output Device	Primary Input Device
Character-oriented	Printer	Keyboard
Page printing	Printer	Card reader
CRT	Screen	Keyboard

In addition to the specified primary devices, each terminal has the ability to support one or more auxiliary devices. The auxiliary devices suggested by each terminal are listed in Table 3–11. The DICE output function codes for the COP are interpreted in Table 3–12.

Trademark of Teletype Corporation

Table 3—9. Interpretation of DICE Output Function Codes for Primary Devices (Part 1 of 3)

Function	Command	① m	n	Character-Oriented Devices ②	CRT Devices	Page Printing Devices (n Is Not Interpreted)
Set coordinates	01_{16}	m_a	n_a	Action is optional. ③	Move cursor to row m and column n.	Action is optional. ③
Forms control	02_{16}	m_a	n_a	Form feed, carriage return, and advance to line m and column n (m-1 line feeds and n-1 spaces to the right)	Move cursor to row m and column n.	Top of form and advance to line m (m-1 line feeds)
Forms control with clear (unprotected)	03_{16}	m_a	n_a	Action is optional. ③	Move cursor to row m and column n, and clear unprotected data to end of screen.	Action is optional. ③
New line control	04_{16}	m_r	n_r	Carriage return, line feed, followed by m line feeds and n spaces to the right	Move cursor to beginning of next line. Then move cursor m lines down and n columns to the right.	Advance (m+1) lines.
New line control with clear	05_{16}	m_r	n_r		Same as 04_{16} except area between start and end positions is cleared.	
Current position control	06_{16}	m_r	n_r	m line feeds and n spaces to the right	Move cursor m lines down and n columns to the right.	Advance m lines.
Current position control with clear	07_{16}	m_r	n_r		Insert n spaces if non-significant space suppression is allowed. If not, insert n DC3 characters. M is not interpreted. ④	
Beginning of current line control	08_{16}	m_r	n_r	Carriage return followed by m line feeds and n spaces to the right	Move cursor to beginning of current line. Then move cursor m lines down and n columns to the right.	
Set tab stop at an absolute position	09_{16}	m_a	n_a	No line feed; one space to right	Set tab stop at row m and column n.	
Forms control and clear, protected and unprotected	$0A_{16}$	m_a	n_a	Action is optional. ③	Move cursor to row m and column n and clear protected and unprotected data to end of screen.	Action is optional.
Clear to end of line	08_{16}	n_a	n_a	No action	Cursor does not move. Clear unprotected data to end of file or end of first unprotected field, whichever occurs first.	Advance 0 lines

NOTES:

① m_a and n_a are absolute values of m and n. m_r and n_r are relative displacements of m and n. For CRT terminals, the home position is $(m_a,n_a)=(1,1)$.

For character- or page-oriented devices that allow position to top of form, the top-of-form position is $(m_a,n_a)=(1,1)$.

Absolute positions of m_a and n_a may range as follows:

m_a ranges 1 to r

Table 3—9. Interpretation of DICE Output Function Codes for Primary Devices (Part 2 of 3)

NOTES: (cont)

where:

r = maximum number of rows (CRT), or maximum number of lines per page.

n_a ranges 1 to c

where:

c = maximum number of columns (CRT), or maximum number of character positions per line.

Relative displacements of m_r and n_r may begin at zero and range to the bottom and right margin of the screen or page.

If a value of m or n falls outside of the legal range, that value of m or n will cause the following action:

m_a or $n_a = 0$ is interpreted as m_a or $n_a = 1$

m or n > maximum allowable m or n—action will vary depending on the remote terminal:

- UNISCOPE display terminals—wraparound will occur on screen.
- Character-oriented terminals—will give different results depending on the characteristics of the device.

② Most character-oriented terminals can be strapped to handle the carriage return (CR) character and the line feed (LF) character as follows:

- CR

  1. print mechanism moves to beginning of the same line; or
  2. print mechanism moves to the beginning of the same line followed by a line feed.

- LF

  1. line feed (no column change); or
  2. line feed followed by return of the print mechanism to the beginning of the new line.

To achieve device independence between terminal types, the character-oriented terminals must use the first option for CR and the first option for LF if the DICE function is ZO#CUR or ZO#BEG.

The first option should be used if the character-oriented terminals are a part of a message switch environment.

Certain terminals do not have a form feed capability (i.e., some TTY terminals). For these terminals, the DICE expressions that specify form feed will result in line feed instead.

③ The set coordinates function (ZO#COORD) or the forms control with clear function (ZO#FORMC), when acted upon by character-oriented or page-printing terminals, will vary in its action, depending on the usage of the DICE keyword parameter of the TERM macroinstruction at CCA generation time:

$$\text{TERM} \qquad \text{...,DICE=}\left(\left\{\begin{matrix}\text{FORMS}\\ \text{NEWLINE}\end{matrix}\right\}\right)\text{,....}$$

If FORMS is specified, the set coordinates function will be interpreted as the form control function.

If NEWLINE is specified, the set coordinates function and the forms control with clear function will result in a carriage return, line feed for character-oriented terminals, or advance one line for page-oriented terminals; m and n are not interpreted.

If the DICE parameter is not specified, the default option will be to NEWLINE.

Table 3—9. Interpretation of DICE Output Function Codes for Primary Devices (Part 3 of 3)

NOTES: (cont)

4 The UNISCOPE display terminal suppresses nonsignificant spaces on each line (except for the line containing the cursor) when text is transmitted to the processor or printed locally on the COP or TP.

The user program may send data to the UNISCOPE screen which has significant blank segments that include the last column of the screen. If this data is transmitted from the terminal to the processor or is printed locally on the COP or TP, the blank segments must consist of nonspace characters that are nondisplayable. The DC3 character meets these qualifications. The ICAM interface provides the user program the capability to prevent nonsignificant space suppression on the UNISCOPE display terminal. The "current position control with clear" is the only DICE function which can be used to perform a clear function if the user program is preventing nonsignificant space suppression.

NOTE:

The ASCII to EBCDIC translation table is modified so that the DC3 character is translated to a space 40_{16} for input from the UNISCOPE display terminal.

Table 3—10. Interpretation of DICE Output Function Codes for Primary Devices

Function	Command	m	n	Character-Oriented Devices	CRT Devices	Page Printing Devices (n Not Interpreted)
Set coordinates	01_{16}	m	n	Not used	m and n represent the start of entry (SOE) cursor coordinates.	Not used
Forms control	02_{16}	01	01	Form feed	Not used	Not used
New line control	04_{16}	00	00	Carriage return, line feed	Carriage return	Not used
Current position control	06_{16}	01	00	Line feed	Not used	End of input card
Beginning of current line control	08_{16}	00	00	Carriage return	Not used	Not used

NOTE:

The user can specify that the RDH is not to create input DICE. This is done at CCA generation time with the DICE keyword parameter to the TERM proc:

```
TERM . . . ,DICE=(OFF), . . . .
```

The default is DICE=(ON).

Table 3—11. DICE Usage for Auxiliary Devices

Remote Terminals	Auxiliary Device	DICE Usage
UNISCOPE	Tape cassette system (TCS) Communications output printer (COP) 800 terminal printer (TP)	DICE is applied to the COP. ①
UTS 400	Tape cassette system (TCS) Communications output printer (COP) 0786 printer 800 terminal printer (TP) Diskette	DICE is applied to the COP. ①
DCT 1000	Card reader/card punch Paper tape reader/punch	DICE is applied as if the output/input is to/from the primary device, even though it is for the auxiliary device. ②
DCT 500/TTY	Paper tape reader/punch	
DCT 524	Tape cassette (TCS) in paper tape read and write only	
Batch terminals	Punch	DICE is used for end of network buffer sentinel. No form control action is taken.

NOTES:

① If the print transparent option is not used, DICE is applied to the UNISCOPE screen even though the output is sent to an auxiliary device of the UNISCOPE terminal. In this case, the format of the data printed on the COP or TP is identical to the screen format. Nonsignificant space suppression by the UNISCOPE terminal may have to be prevented to keep the formats identical (see note 4 to Table 3–9).

The full capability of DICE cannot be applied to the COP because of hardware characteristics. All data to a UNISCOPE auxiliary device passes through the UNISCOPE terminal. When DICE is applied to the COP, the use of print transparent mode means that no carriage returns are transferred to the COP. Line feeds and form feeds take a memory position in the UNISCOPE memory and are nondisplayable. These characters are passed to the COP where:

- An LF causes a line feed followed by return of the print mechanism to the beginning of the new line.

- An FF causes a page eject and positioning of the print mechanism at the beginning of the first line of the form.

The COP has no tabbing capability.

The above characteristics are reflected in the interpretation of DICE output function codes for the COP as shown in Table 3–11.

For messages sent to a UNISCOPE auxiliary device with transparent transfer, the cursor to home (ESC e) sequence is inserted at the beginning of the text by the rDH.

② The control characters that are generated from the DICE function are always created for the primary device of a character-oriented device, even though the use program is sending to an auxiliary device. The message and these control characters (carriage returns, line feeds, form feeds, and spaces) will be punched/written by the output auxiliary device that was specified by the user program or was switch-selected by the terminal operator. If the punched/written data is later read by the terminal's input auxiliary device, the carriage returns, line feeds, and form feeds are converted to input DICE as specified in Table 3–9.

Table 3—12. Interpretation of DICE Output Function Codes for the COP

Function	Command	m	① n	Communications Output Printer (COP) ②
Set coordinates	01_{16}	m_a	n_a	Action is optional. ③ Default is: line feed
Forms control	02_{16}	m_a	n_a	Form feed, line feed, and advance to line m and column n (m-1 line feeds and n-1 spaces to the right)
Forms control w/clear	03_{16}	m_a	n_a	Action is optional. ③ Default is: line feed
New line control	04_{16}	m_r	n_r	Line feed, followed by m line feeds and n spaces to the right
New line control w/clear	05_{16}	m_r	n_r	
Current position control	06_{16}	m_r	n_r	Insert n spaces if nonsignificant space suppression is allowed. ④ If not, insert n DC3 characters. m is not interpreted.
Current position control with clear	07_{16}	m_r	n_r	
Beginning of current line control	08_{16}	m_r	n_r	
Set tab stop at an absolute position	09_{16}	m_r	n_r	

NOTES:

① , ② , ③ , and ④ same as notes for Table 3–9.

3.20. OBTAINING AN ICAM DSECT

Occasionally, you need to look at the DSECTs that ICAM uses. These DSECTs are stored in the procs named XTMDSCTS and XDECTS located in the $Y$MAC library of the SYSRES disk pack. The DSECTs may be obtained by calling and assembling the proc calls TM#DSECT and TN#DSECT, respectively. When no operands are specified in the referenced proc call, all DSECTs and EQUATEs in the proc are listed. Individual DSECTs or groups of DSECTs may be listed by specifying the coded names of the DSECTs desired as operands in the proc call. Table 3-13 provides the operands that may be specified, a description of the DSECT or group of DSECTs accessed, and the user interface involved for the TM#DSECT proc call; Table 3-14 does the same for the TN#DSECT proc call. A jobstream that was used to obtain a set of DSECTs is shown in the following example.

The format for calling the XTMDSCTS proc is:

LABEL	△OPERATION△	OPERAND
	TM#DSECT	See Table 3-13

The format for calling the XDSECTS proc is:

LABEL	△OPERATION△	OPERAND
	TN#DSECT	See Table 3-14

Example:

```
1         10     16
--------------------------------------------------------------------------
// JOB JHB
// DVC 2Ø
// LFD PRINT
// DVC 3Ø
// LFD CARD
// ASM
/$
 START
 TN#DSECT MCT                               GET MCT DTF (TN#MCT)
 END
/*
/&
// FIN
```

Table 3—13. TM#DSECT Proc Call Details

DSECT Name	Description	User Interface	Operand Specification	
			Individual Selection	Group Selection
TM#PRCS	Process File DTF	STDMCP	PRCS	GETPUT
TM#DEST	Destination File DTF	STDMCP	DEST	GETPUT
TM#TCS	Transaction Control Table	TCI	TCS	USERTCI
TM#TTT	Transaction Terminal Table	TCI	TTT	USERTCI
TM#TOMSG	Output Message Prefix	TCI	OMSG	USERTCI
TM#TIMSG	Input Message Prefix	TCI	IMSG	USERTCI
	Delivery Notification Statuses (EQUATEs)	TCI	EQUS	USERTCI

Table 3—14. TN#DSECT Proc Call Details

DSECT Name	Description	User Interface	Operand Specification	
			Individual Selection	Group Selection
TN#ARP	Activity Control SVC Decode ARP	N/A	ACTARP	BACTGRP
TM#ARP	MUST ARP	N/A	ARP97	N/A
TN#ACTB	Basic Activity Control Table	N/A	BASTAB	BACTGRP
TN#BPOOL	ARP/Buffer Pool Control Table	N/A	BPOOL	N/A
TN#CNTRL	CCA Control Section	N/A	CCACON	CCAGRP
TN#PARP	CPIOCP Packet	CPI	CPIOCP	PIOGRP
TN#DCT	Auxiliary Device Control Table	DUST	DCT	CCAGRP
TN#EDTBL	Destination Table	N/A	DESTBL	CCAGRP
TN#DLIST	Distribution/Fast Select List	N/A	DLIST	CCAGRP
NETREQ/NETREL	LNEREQ/LNEREL Error Code EQUATEs	DUST	N/A	DUST
TQ#DSCTS	NETREQ/NETREL DUST Macro Table	DUST	N/A	DUST
TQ#LINE	Line Request/Release DUST Macro Table	DUST	N/A	DUST
TN#GEN	ICAM General Information Table	N/A	GENTAB	BACTGRP
TN#GTCBS	User TCB Directory and Activity Queue Table	N/A	GENTAB	BACTGRP
TN#LCT	Line Macro Table (LCT)	N/A	LCT	CCAGRP
TN#PLINE	CACH Macro Table (Line Link Table (LLT))	N/A	LLT	PIOGRP
TN#MCT	Message Control Table (MCT)	DDI	MCT	DDIGRP
TN#MSG	Network Buffer Prefix	N/A	MSGPRE	N/A
TN#VARP	Overlay Control/Operator Communications ARP	N/A	OVARP	N/A
TN#FPRCS	PRCS Macro Table	N/A	PRCS	CCAGRP
TN#TCT	TERM Macro Table (TCT)	N/A	TCT	CCAGRP
TN#ARTME	Timer Activity Request Packet	N/A	TIMARP	DDIGRP
TN#QCT	Queue Control Table	N/A	QCT	CCAGRP

3.21. USER ISLAND CODE CONSIDERATIONS

All user SVC calls issued to ICAM from island code must be done with IRL requested. If the user fails to specify an IRL, or if the SVC is a CYIELD macro call, the user is permanently suspended.

4. Message Processing Procedure Specification (MPPS) Macroinstructions

4.1. GENERAL

TCI users are provided with a set of optional processing routines and macroinstructions that enable automatic analysis of incoming and outgoing messages. The result of this analysis enables ICAM to process and direct every message transmitted or received in a network.

MPPS functions are available to users of the resident transaction control interface (TCI) only. They are not available with transient TCI. MPPS instructions are submitted with the CCA network definition macroinstructions at SYSGEN time.

4.2. SPECIFYING AN MPPS

A message processing procedure specification (MPPS) defines the procedure that ICAM is to execute automatically on each segment of each message transmitted in a communications network, without regard to any CUP. An MPPS becomes, in effect, a subroutine of automatic functional macroinstructions that ICAM executes on behalf of a CUP. An MPPS can provide:

- message switching;
- time and date stamping;
- source validation;
- error indications;
- input message sequence checking;
- output message sequence number insertion;
- testing for message size and number limits;
- separation of messages by type or terminal;
- routing of messages destined for active terminals;

- rerouting of messages destined for downed terminals;
- retransmittal of messages not received error free; and
- logging of messages and error indications.

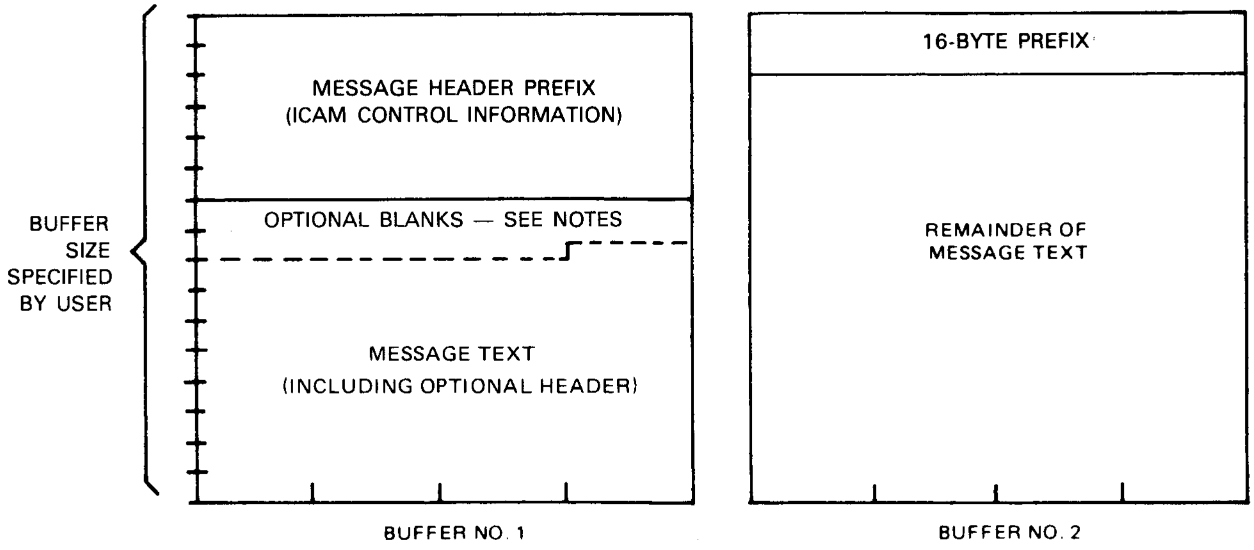
To take advantage of the MPPS facilities, you must establish a standard message format for the messages transferred on each line in a network. This is because the control mechanism (message header scan pointer) that directs processing of an MPPS routine moves only in a forward direction. Thus, you must specify your MPPS instructions in the same sequence as the control information contained in the message. If you want to build control information into your message, that information must be placed in the first part of the message (message header). You must also take network buffer sizing into consideration.

4.2.1. Buffer Size Considerations

As you can see in Figure 4–1, messages are placed in network buffers before they are delivered to a communications user program (CUP) on input or before they are turned over to a remote device handler (RDH) on output. These network buffers are provided automatically by the ICAM software; however, the size of the network buffers is determined by the value you specify in the BUFFERS macro at CCA generation. Each network buffer has message header information that occupies space at the beginning of the buffer for ICAM control purposes.

The control header (message header prefix) in the first buffer for a message is always much larger than that in any succeeding buffers that may be required to hold a message. It is always 80 bytes long if main storage queueing is used, or 104 bytes if disk queueing is specified. Subsequent buffers contain only 8-byte headers used for linkage purposes. Each buffer is filled with your message text starting at the first byte following the header.

If you write an MPPS routine as part of your CCA, it is important to remember that the MPPS routines examine and act upon only that portion of the first buffer immediately following the message header prefix up to the end of the first buffer. Therefore, when you specify buffer size, you must ensure that your message text control information will fit into *your* portion of the first network buffer, i.e., minimum buffer size is always at least message header prefix size plus your message text used for control purposes (message header).


BUFFER SIZE SPECIFIED BY USER
MESSAGE HEADER PREFIX (ICAM CONTROL INFORMATION)
OPTIONAL BLANKS — SEE NOTES
MESSAGE TEXT (INCLUDING OPTIONAL HEADER)
BUFFER NO. 1
16-BYTE PREFIX
REMAINDER OF MESSAGE TEXT
BUFFER NO. 2

NOTES:

1. Input messages are automatically prefixed with the number of blanks (X'40') specified by the user in the MPPS operand of the LINE macro or the INPUT parameter of the TERM macro. If date and time stamping is to be performed on output messages, the user must insert the proper number of blanks as part of his message data or DICE commands.

2. Blanks (X'40') must be provided by the user before the text if insertion is to take place on an output message created by a PUTCP.

Figure 4—1. Example of Network Buffer Use

4.2.2. Date and Time Stamping

If you want your messages date and time stamped, the MPPS routines will perform this service. However, for output messages, your CUP must provide blank spaces at the beginning of the message text for this information. For input messages, ICAM automatically prefixes the incoming message text with the proper number of spaces. (See the DATSTP macroinstructions for more details.)

4.2.3. Message Header Scan Pointer

The control element that provides MPPS processing is called the message header scan pointer. The scan pointer always points to the message header field that is currently being processed. When creating an MPPS, you must be aware of the scan pointer and how it operates.

The sequence of the macroinstructions contained in an MPPS routine must be arranged so that the scan pointer always moves forward, never backward. Further, the message header scan pointer is initialized to point to either the last blank character position at the beginning of the text or to the last byte in the message header prefix. The message header scan pointer may continue to be advanced, through the execution of certain MPPS instructions, until it reaches the end of the buffer in which it is operating. When the end of the buffer is reached, the end of the message header is also reached.

4.2.4. MPPS Error Analysis

When you write your MPPS routine, you can request tests for error conditions that may have occurred during processing. Certain MPPS macroinstructions (CANCELM, ERRMSG, REROUTI, REROUTO, and RETRANS) contain mask operands that allow you to specify the test you want performed. Table 4-1 provides a list of error masks that can be specified in the various MPPS macroinstructions. By using the error masks, you can send diagnostic messages to terminals, cancel messages, etc.

Table 4—1. MPPS Error Conditions

Error Mask	Explanation
TN#MNDST	No destination was found during MPPS processing.
TN#MBDST	An invalid destination was found during MPPS processing.
TN#MEOH	End of message header text was reached during MPPS scan of text.
TN#MBINS	Not enough blanks were reserved for data insertion (DATSTP, TIMSTP, SEQOUT).
TN#MBSOR	Source name specified in the message header text is incorrect, but is a valid terminal name.
TN#MNSOR	Source name specified in the message header text is incorrect.
TN#MBSQI	Input sequence count in the message header text is incorrect.
TN#MNSQI	Sequence in count was not placed in the message header prefix during receive header processing.
TN#MNSQO	Sequence out count was not placed in the message header prefix during send header processing.
TN#MCERR	Channel error
TN#MLERR	Line error
TN#MTERR	Terminal and/or auxiliary device error
TN#MBID	Journal stage area threshold reached
TN#MTOUT	Journal record staging error

Note that when you specify an error mask in an MPPS macro, you can reference one or more error conditions shown in the table. A 2-byte mask is generated and compared against the TN#MERRB field in the message header prefix. If any of the specified errors occur, the MPPS function is executed; therefore, you are free to specify a 2-byte logical expression when supplying an error mask.

Example:

```
    1        10     16
1.           CANCELM TN#MBDST++TN#MEOH
2.           REROUTO X'FFFF'— —(TN#MEOH++TN#MBINS),ALTD,M
```

1. Cancel the message if a bad destination was found or if the end of header was reached during the scan.

2. Reroute the output message to the medium queue of the alternate destination (of the original output terminal) if any error other than "end of header" or "no insertion" took place.

4.2.5. MPPS Message Processing

MPPS processing is divided into three subgroups for input message processing and three similar subgroups for output message processing. These subgroups are identified by the MPPS delimiter macroinstructions. Functional MPPS macroinstructions may be specified within each subgroup to perform the specific operations that you may desire. The functional macros that you may use within a subgroup are identified in Table 4–2.

The first type of MPPS subgroup is identified by the RECSEG macro for input message processing and the SENSEG macro for output message processing. Functional macros specified within these subgroups define the processing of all message buffers used to handle a message. For instance, if you specify a functional macro within one of these subgroups, all segments* of a message are processed according to that request. For example, if it takes six message buffers to contain a complete message and message logging is requested, all six message segments are logged.

The second type of MPPS processing subgroup is identified by the RECHDR (input) macro and the SENHDR (output) macro. It is the job of these subgroups to execute and process the functional macros that you specified to analyze the header information that forms the first part of your message text.

Generally, a message segment and the available text space in a message buffer mean the same thing.

Table 4—2. MPPS Delimiter and Functional Macroinstructions

Receive Group Macroinstructions			Send Group Macroinstructions		
Subgroup	**Delimiter**	**Functional**	**Subgroup**	**Delimiter**	**Functional**
	MPSTART[1]		Send segment	SENSEG	JOURN LOG
Receive segment	RECSEG	JOURN LOG			
Receive header	RECHDR[1]	ADVANCE BRANCH DATSTP DIRECT[2] IFSOURCE MSGLMT MSGTYP ROUTE[2] SEQIN SOURCE TIMSTP	Send header	SENHDR[1]	ADVANCE BRANCH DATSTP MSGTYP SEQOUT TIMSTP
Receive end	RECEND[1]	CANCELM ERRMSG MSGLMT REROUTI	Send end	SENEND[1]	ERRMSG REROUTO RETRANS
Receive post	RECPST[1]		Send post	SENPST[1]	

NOTES:

[1] Denotes required MPPS macroinstructions.

[2] One of these instructions must be executed for each message; otherwise, a no-destination error is reported in the message header prefix error field. If a no-destination error is detected by RECPST, the message will still be processed by TCI.

The third type of MPPS processing subgroup deals with error and end of message processing and is identified by the RECEND (input) and SENEND (output) macroinstructions. Functional processing in this subgroup begins after a complete message has been received or transmitted. At this time, any previously detected errors are acted upon. Functional macros in this subgroup enable you to cancel a message, reroute a message, send yourself a diagnostic message, etc.

The end of MPPS input message processing is indicated by the RECPST macroinstructions The end of MPPS output message processing is denoted by the SENPST macroinstructions, which also indicates the end of all MPPS processing for a message.

4.2.6. Delimiter Macroinstructions

There are nine delimiter macroinstructions that are used to separate and identify a particular sequence of functional macroinstructions within an MPPS. One of these macroinstructions, the MPSTART instruction, is used to identify the beginning of an MPPS (for one or more lines) to ICAM and is always required to be the first macroinstruction in every MPPS.

The remaining eight macroinstructions are divided into two groups: the receive group, which operates on incoming messages on a line, and the send group, which operates on outgoing messages on a line. The receive group macroinstructions must always precede the send group macroinstructions. Within each of these groups, the macroinstructions must be coded in the following sequence if they are used:

1. Receive group

   a. Receive segment (RECSEG) – Start receive segment

   b. Receive header (RECHDR) – Start receive header analysis

   c. Receive end (RECEND) – Begin receive-end processing

   d. Receive post (RECPST) – End of receive MPPS

2. Send group

   a. Send segment (SENSEG) – Start send segment

   b. Send header (SENHDR) – Start output header analysis

   c. Send end (SENEND) – Begin output message end processing

   d. Send post (SENPST) – End of output MPPS

MPSTART

4.2.6.1. Begin MPPS (MPSTART)

Function:

Identifies the beginning of an MPPS to ICAM. This instruction must be the first in every MPPS.

Format:

LABEL	△OPERATION△	OPERAND
mpps-name	MPSTART	unused

Label:

mpps-name

Is 1- to 4-character alphanumeric that must be the same as that specified in the MPPS parameter of a LINE macroinstruction or in the INPUT parameter of a TERM macroinstruction.

Example:

```
1       10    16
MPPS    MPSTART
```

4.2.6.2. Begin Receive Segment (RECSEG)

Function:

Identifies the beginning of the receive-segment subgroup, which is concerned with processing header and text segments of incoming messages. This instruction is not needed if processing of text segments by ICAM is not required.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	RECSEG	unused

4.2.6.3. Begin Receive Header (RECHDR)

Function:

Identifies the beginning of the receive-header subgroup containing the functional macroinstructions related to processing a message header for incoming messages. If it is determined that the segment being processed is a header segment, the functional macroinstructions included in the receive-header subgroup are executed. Also, the message header scan pointer (4.2.3) is set at this time. If the segment contains text only, the receive-header coding is bypassed. This instruction must immediately follow the RECSEG subgroup (if present) or MPSTART macroinstruction.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	RECHDR	unused

4.2.6.4. Begin Receive End (RECEND)

Function:

Identifies the beginning of the receive-end subgroup containing the functional macroinstructions required for end-of-message processing. When processing control is transferred to this instruction, the text comprising an input message is processed (unaffected by any error conditions that may have been detected while processing) in the receive-header subgroup. Only after the end-of-text sentinel is detected are the end-of-message and error processing macroinstructions specified in the receive-end subgroup executed.

This macroinstruction is always required and must follow the last functional macroinstructions under the RECHDR subgroup.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	RECEND	unused

RECPST

4.2.6.5. End Receive Group (RECPST)

Function:

Required to signify the end of all instructions in the receive-segment, receive-header, and receive-end subgroups. The instruction must be the last instruction in the receive group. If a valid destination has not been received for a given message, the message will be processed by TCI.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	RECPST	unused

4.2.6.6. Begin Send Segment (SENSEG)

Function:

Identifies the beginning of the send-segment subgroup, which is concerned with the processing of both header and text segments of outgoing messages. This macroinstruction is not needed if processing of text segments by ICAM is not required. If present, this instruction should be the first in the send group and should immediately follow the RECPST macroinstruction.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	SENSEG	unused

SENHDR

4.2.6.7. Begin Send Header (SENHDR)

Function:

Identifies the beginning of the send-header subgroup, which contains the functional macroinstructions required to process the message header segment of outgoing messages. If it is determined that the segment being processed is a header segment, the functional macroinstructions in the send-header subgroup are executed. If the segment contains text only, the send-header subgroup is bypassed. This instruction is required and must immediately follow either the SENSEG subgroup or RECPST instruction.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	SENHDR	unused

4.2.6.8. Begin Send End (SENEND)

Function:

Identifies the beginning of the send-end subgroup containing the functional macroinstructions required for end-of-message processing. The functional macroinstructions in this subgroup are executed only after an entire message has been sent to a remote terminal. After output completion is detected, control passes to the error processing macroinstructions following the SENEND macroinstruction. The SENEND macroinstruction is required and should follow the SENHDR subgroup.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	SENEND	unused

4.2.6.9. End Send Group (SENPST)

Function:

Required to signify the end of all instructions in the send group.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	SENPST	unused

4.2.7. Functional Macroinstructions

Functional macroinstructions are available to perform the actual processing and verification checks on the information contained in a message header. When using these instructions, remember that the message header scan pointer, which serves as the key to identify the field being processed, can move only in a forward direction. Thus, sequencing of these macroinstructions in an MPPS must be relative to the sequence of the information fields contained in the message header being processed. The instructions that move the message header scan pointer leave it pointing to the last character of the processed field.

The following macroinstructions are described in alphabetical order for easy reference:

- ADVANCE
- BRANCH
- CANCELM
- DATSTP
- DIRECT
- ERRMSG
- IFSOURCE
- JOURN
- LOG
- MSGLMT
- MSGTYP
- REROUTI
- REROUTO
- RETRANS
- ROUTE
- SEQIN
- SEQOUT
- SOURCE
- TIMSTP

ADVANCE

4.2.7.1. Advance Message Header Scan Pointer (ADVANCE)

Function:

Advances the message header scan pointer over message header text fields that are not to be processed by the MPPS. If the end of header is reached during execution of an ADVANCE instruction, the end-of-header flag is set in the message header prefix error field (Table 4–2) and processing control is transferred to the RECEND or SENEND macroinstruction, as applicable.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	ADVANCE	integer,[character-string]

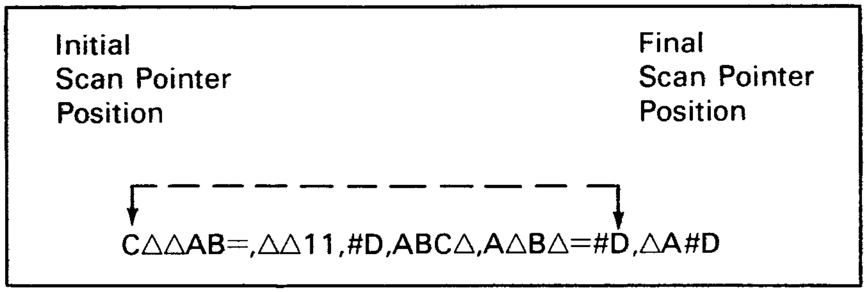
Parameters:

integer

Is a decimal number specifying the number of nonblank character positions to advance the message header scan pointer if positional parameter 2 is omitted, or a decimal number from 1 to 8 specifying the number of characters contained in positional parameter 2. In the first coding example given for this instruction, the message header scan pointer advances to the fourth nonblank character position it detects, then stops.

character-string

From one to eight alphanumeric characters identifying the character string the message header scan pointer is to look for when advancing. The scan pointer stops, pointing to the last character referenced by this parameter, after it has advanced beyond all previously identified characters in the character string, uninterrupted by any nonreferenced characters, excluding blanks. For example, given the following message header fragment



C△△AB=,△△11,#D,ABC△,A△B△=#D,△A#D

and the following coding example, the scanning mechanism checks the following 5-character blocks, looking for a match with the specified character-string:

Step	Test
1.	AB=,1
2.	B=,11
3.	=,11,
4.	,11,#
5.	11,#D
6.	1,#D,
7.	,#D,A
8.	#D,AB
9.	D,ABC
10.	,ABC,
11.	ABC,A
12.	BC,AB
13.	C,AB=
14.	,AB=#
15.	AB=#D

As indicated, the desired character string will be found on the 15th compare operation.

NOTE:

This parameter must be expressed in alphanumeric format and may not reference any blank characters.

If omitted, positional parameter 1 advances the scanning mechanism.

Example:

```
1        10    16
____________________________
         ADVANCE 4
           .
           .
           .
         ADVANCE 5,AB=#D
```

BRANCH

4.2.7.2. Bypass Processing (BRANCH)

Function:

Transfers processing control unconditionally to another portion of the MPPS to allow the program to bypass processing not required for a particular message type.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	BRANCH	symbolic-branch-address

Parameter:

symbolic-branch-address
Specifies the symbolic address of the macroinstruction to which control is transferred.

Example:

```
1        10    16
_______________________
         BRANCH DEST
```

4.2.7.3. Cancel Input Error (CANCELM)

Function:

As an input error procedure, eliminates all reference to an input message where a specified error occurred, by preventing its inclusion in an input queue.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	CANCELM	mask

Parameter:

mask

Is a logical expression specifying the bits in the error field of the message header prefix to be tested. If any of the specified error bits are set, the message is, essentially, thrown away.

Example:

```
1       10     16
----------------------------------------------
        CANCELM X'FFFF'— —(TN#MEOH++TN#MBINS)
        CANCELM TN#MBSQI
```

DATSTP

4.2.7.4. Insert Current Date (DATSTP)

Function:

Inserts the current date in the message header segment of an incoming or outgoing message. Either a calendar or ordinal date may be selected. This instruction may appear in the receive-header or send-header subgroups. If sufficient space is available in the message header to insert the date, all of the previously scanned message header data is shifted left the appropriate number of characters and the date is inserted. At the conclusion of this operation, the scan pointer, which never moves during the execution of this instruction, points to the last character in the date stamp.

If enough space is not available, the MPPS bad insertion flag is set. Control then passes to the next instruction in line.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	DATSTP	[{N J}]

Parameters:

N

Indicates that the date is to be inserted in calendar form: △xx/xx/xx.

J

Indicates that the date is to be inserted in ordinal form: △yy/ddd.

Example 1:

```
1            10     16
DATE         DATSTP J
```

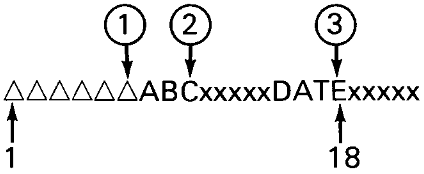
A second example (following) shows the use of the DATSTP macro in a typical MPPS routine.

Example 2:

```
1        10      16
-----------------------------
MPS1     MPSTART
         RECHDR
         MSGTYP 3,ABC,BR1
         ADVANCE 4,DATE
         DATSTP J
         BRANCH DIR
BR1      MSGTYP,3,XYZ,BR2
```

This example also illustrates the operation of the message header scan pointer.

When the message is first received, six blanks (△) are inserted by ICAM because MPPS= (MPS1,6) was specified in the LINE macro or INPUT=(MPS1, 6) was specified in the TERM macro. The input message would appear as follows:



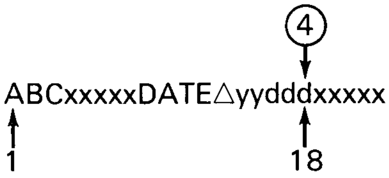
The message header scan pointer's position is indicated by the circled number following execution of the following macroinstructions:

① Initial scan position – Set by RECHDR.

② Position after MSGTYP – Three characters, ABC, were found so the branch was not taken.

③ Position after ADVANCE – The 4-character DATE field was found.

④ Position after DATSTP – In the following illustration of the same message, note that the message header scan pointer did not move; the text preceding the pointer was shifted to the left and the date was inserted. Following this macro, the pointer is still at position 18.



If insufficient space was specified in the LINE macro, the bad insertion bit would have been set (Table 4–1).

NOTE:

On output messages, the CUP must supply sufficient blanks for insertion.

DIRECT

4.2.7.5. Direct Incoming Message (DIRECT)

Function:

Directs an incoming message to the destination specified in the operand field. (Compare with ROUTE macroinstruction, 4.2.7.15.) If an invalid destination is specified in this instruction, the bad destination flag is set in the error field of the message header prefix and processing continues with the macroinstruction next in line.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	DIRECT	$\left\{\begin{array}{l}\text{T,terminal-name}\\ \text{D,distribution-list-name}\\ \text{ALTD,}\\ \text{SOURCE,}\end{array}\right\}\left[,\left\{\begin{array}{c}\text{H}\\ \text{M}\\ \text{L}\end{array}\right\}\right]\text{[,FSEL]}$

Parameters:

T
: Indicates that positional parameter 2 is the name of a terminal.

D
: Indicates that positional parameter 2 is the name of a distribution list.

ALTD
: Indicates that the alternate destination field of the source terminal contains the destination code for this message.

SOURCE
: Indicates that the message is routed back to its orginator.

terminal-name
: Specifies the name of one of the terminal entries defined in the communications control area (CCA).

distribution-list-name
: Specifies the name of a distribution list defined in the CCA.

$\left\{\begin{matrix}\mathtt{H}\\\mathtt{M}\\\mathtt{L}\end{matrix}\right\}$

Indicates the high, medium, or low priority queue, respectively, of the identified destination.

FSEL

Indicates that the terminal specified by positional parameter 2 is the name of a DCT 500 terminal with fast-select list to which the message should be directed.

If omitted, the specified destination is assumed to be not a fast-select device.

Example:

```
1        10     16
----------------------------------
TYPE1    DIRECT D,NYC1
TYPE2    DIRECT T,PHIL,H,FSEL
TYPE3    DIRECT ALTD,,H
TYPE4    DIRECT SOURCE,,L
```

NOTE:

Every time the DIRECT macro is executed, the destination name and corresponding queue address are updated; therefore, the message will be sent only to the destination specified by the last input destination macro. To have the message sent to more than one destination, a DLIST could be specified on the DIRECT or REROUTI macro call, or more than one destination name (including a DLIST) could be specified in the message text that is referenced during the execution of a ROUTE macro.

ERRMSG

4.2.7.6. Send Error Message (ERRMSG)

Function:

If a specified error occurs, causes an error message to be sent to a designated output terminal.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	ERRMSG	mask, {terminal-name SOURCE process-file}, 'error-text'

Parameters:

mask

Is a logical expression specifying the bits in the error field of the message header prefix to be tested. If any of the specified bits are set, the error message comprising positional parameter 3 is output to the device identified by positional parameter 2.

terminal-name

Identifies the output terminal to which the error message in positional parameter 3 is output.

SOURCE

Specifies that error messages are to be returned to the source terminal that caused the error condition.

process-file

Identifies the process file to which the error message in positional parameter 3 is output.

'error-text'

Specifies the text of the error message to be output to the terminal specified in positional parameter 2. The text must be enclosed in apostrophes.

Example:

```
1           10      16
------------------------------------------------------------------------
            ERRMSG  TN#MBINS,LP1,'NOT ENOUGH ROOM FOR DATA INSERTION'
               .
               .
               .
            ERRMSG  TN#MNDST,SOURCE,'NO DESTINATION SPECIFIED'
```

IFSOURCE

4.2.7.7. Message Handler (IFSOURCE)

Function:

As an input procedure, allows processing of an incoming message if the source terminal name in the message header prefix matches a specific terminal name.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	IFSOURCE	terminal-name,symbolic-branch-address

Parameters:

terminal-name
: Identifies a specific terminal name to be compared with the source terminal name in the message header prefix. If a match is detected, control passes to the symbolic address specified in positional parameter 2; otherwise, processing continues with the next inline functional instruction.

symbolic-branch-address
: The symbolic address of another functional macroinstruction to which control is transferred if a match is detected between positional parameter 1 and the source terminal name.

Example:

```
1        10     16
-----------------------------------
         IFSOURCE TØØ1,SENDTRM1
         IFSOURCE TØØ4,SENDALTD
         DIRECT T,TRM3,M
           .
           .
           .
         BRANCH REND
SENDTRM1 DIRECT T,TRM1,L
           .
           .
           .
         BRANCH REND
SENDALTD DIRECT ALTD,,H
           .
           .
           .
REND     RECEND
```

Explanation:

If the message came from T001, it is directed to the low queue associated with the TRM1 terminal.

If the message came from T004, it is directed to the high queue associated with the alternate destination of T004.

NOTE:

If no ALTD exists for T004, the bad destination flag is set and the message may be lost if proper error recovery is not specified in the RECEND subgroup.

If the message came from neither T001 or T004, it is directed to the medium queue associated with terminal TRM3.

JOURN

4.2.7.8. Create Journal Record (JOURN)

Function:

Creates a journal record for the current message. This record consists of the message header prefix and all associated text and is transferred to the specified output file.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	JOURN	$filename_1$

Parameter:

filename

Identifies the file in which the journal entries are to be written. This identifier must be defined in the CCA as the label of the JRNFILE macro.

Example:

```
1        10    16
JRNL     JOURN JFILE
```

4.2.7.9. Log Current Message (LOG)

Function:

Transfers the current message header prefix to the specified output file. The log record also contains an index for location of the associated journal entry, if any.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	LOG	filename

Parameter:

filename
Identifies the data management log file established for the logging of messages. This identifier must be defined in the CCA as the label field of the JRNFILE macro.

Example:

```
1        10    16
LOG      LOG   LOGA
```

MSGLMT

4.2.7.10. Limit Messages (MSGLMT)

Function:

Limits the number of messages that can be transmitted by a terminal during a single polling pass. This instruction is optional for user written handlers with polled lines and may appear in the receive-header or receive-end subgroup; however, it may be executed only once per message. It applies only if the PLIMIT keyword parameter of the TERM macro (2.2.10) was not supplied for the terminal on the line.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	MSGLMT	[message-limit]

Parameter:

message-limit
Is a decimal integer specifying the maximum number of messages to be accepted from a terminal during one polling pass. When the maximum number of messages is transmitted by a terminal, the polling mechanism advances to the next terminal in the polling sequence. This message limit applies to all the terminals connected to lines controlled by the MPPS in which it appears.

If omitted, it is assumed that poll limits were specified when the terminal table entries were created at network definition time.

Example:

```
1        10    16
LMT1     MSGLMT 1Ø
```

4.2.7.11. Identify Message Type (MSGTYP)

Function:

Separates incoming or outgoing messages into two or more types to allow for different processing of each type. Use of this instruction is optional, and it may appear in both the receive-header and send-header subgroups.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	MSGTYP	[integer,character-string,symbolic-branch-address]

Parameters:

integer

Specifies a decimal number from 1 to 8 that indicates the number of characters in positional parameter 2.

If omitted, processing continues with the next inline functional instruction.

character-string

Specifies a string of up to eight alphanumeric characters, including special symbols, that identifies a particular message type. This character string is compared with the message type field in the message header currently being processed. If a match is detected, the scan pointer is updated and processing continues with the macroinstructions immediately following this instruction. If no match is detected, the scan pointer is not moved, and processing control is transferred to the symbolic address identified by positional parameter 3. If end of header is reached while scanning for message type, the end-of-header flag is set in the message header prefix, and the processing control is transferred to the RECEND or SENEND macroinstruction, as applicable.

If omitted, processing continues with the next inline instruction.

symbolic-branch-address

Specifies the symbolic address of another functional or delimiter macroinstruction to which control is transferred if no match is detected between the characters comprising positional parameter 2 and the message type field in the message header.

If omitted, processing continues with the next inline instruction.

Example:

```
1        10    16
_________________________________
         MSGTYP 3,ABC,TYPE2
            .
            .
            .
TYPE2    MSGTYP 3,XYZ,TYPE3
```

4.2.7.12. Reroute Input Error (REROUTI)

Function:

As an input error procedure, reroutes an input message, where a specified error occurred, to a designated terminal. The message may be rerouted to its originator, to an alternate destination as specified in the terminal table of the source terminal, or to any other specific terminal.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	REROUTI	mask, $\left\{\begin{array}{l}\text{SOURCE}\\ \text{ALTD}\\ \text{terminal-name}\\ \text{dlist}\end{array}\right\}\left[,\left\{\begin{array}{c}\text{H}\\ \text{M}\\ \text{L}\end{array}\right\}\right]$

Parameters:

mask
Is a logical expression that selects the error indicators set in the error field of the message header prefix to be tested.

SOURCE
Indicates that the message must be rerouted to its originator.

ALTD
Indicates that the alternate destination field of the source terminal contains the destination code for this message.

terminal-name
Identifies a specific terminal to which the message is to be directed.

dlist
Identifies a specific distribution list.

$\left\{\begin{array}{c}\text{H}\\ \text{M}\\ \text{L}\end{array}\right\}$
Identifies the high, medium, or low priority queue, respectively, of the destination identified by positional parameter 2.

Example:

```
1               10      16
---------------------------------------------------------------
                REROUTI TN#MBSQI,SOURCE
                   .
                   .
                   .
                REROUTI TN#MNDST++TN#MBDST,REM1,M
```

NOTE:

Every time the REROUTI macro is executed, the destination name and corresponding queue address are updated; therefore, the message will be sent only to the destination specified by the last input destination macro. To have the message sent to more than one destination, a DLIST could be specified on the DIRECT or REROUTI macro call, or more than one destination name (including a DLIST) could be specified in the message text that is referenced during the execution of a ROUTE macro.

4.2.7.13. Reroute Output Error (REROUTO)

Function:

As an output error procedure, directs an output message, where a specified error occurred, to another destination. The message may be directed to the alternate destination specified in the terminal table of the intended terminal or to a specific terminal.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	REROUTO	$\text{mask,}\left\{\begin{array}{l}\text{ALTD}\\ \text{terminal-name}\end{array}\right\}\left[,\left\{\begin{array}{c}\text{H}\\ \text{M}\\ \text{L}\end{array}\right\}\right]$

Parameters:

mask

Is a logical expression that selects the error indicators set in the error field of the message header prefix to be tested.

ALTD

Indicates that the alternate destination field of the intended terminal contains the destination code for this message.

terminal-name

Identifies a specific terminal to which the output message is to be directed.

$\left\{\begin{array}{c}\text{H}\\ \text{M}\\ \text{L}\end{array}\right\}$

Identifies the high, medium, or low priority queue, respectively, of the specified destination.

If omitted, the low priority queue is assumed.

Example:

```
1       10    16
________________________________
        REROUTO TN#MEOH,ALTD,H
          .
          .
          .
        REROUTO X'FFFF',TØØ1,L
```

NOTE:

Following output completion, the MPPS SENEND subgroup is given the opportunity to redirect the current output message. Only one function (rerouting, retransmitting, or intercepting) may be performed, and this function is the first one in the subgroup that is executed for that particular message.

4.2.7.14. Retransmit Output Error (RETRANS)

Function:

As an output error procedure, retransmits the output message if a specified error occurred. The number of transmissions attempted is a characteristic of the device handler.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	RETRANS	mask

Parameter:

mask
Is a logical expression that selects error indicators set in the error field of the message header prefix to be tested.

Example:

```
1        10    16
_____________________________
         RETRANS TN#MTERR++TN#MLERR
```

NOTE:

Following output completion, the MPPS SENEND subgroup is given the opportunity to redirect the current output message. Only one function (rerouting, retransmitting, or intercepting) may be performed, and this function is the first one in the subgroup that is executed for that particular message.

ROUTE

4.2.7.15. Scan for Routing (ROUTE)

Function:

Scans the destination field of incoming messages in the message header to determine the routing of each message. Unlike the DIRECT macroinstruction, this instruction enables the originator of each message to specify its destinations.

The ROUTE macro differs from the DIRECT macro in that it allows the *terminal* operator to specify the actual destinations of the message as he is keying in the message. (The destination names must correspond to the terminal names specified in a network.)

NOTES:

1. *If an invalid destination is detected during the scan, the MPPS bad destination flag is set; however, scanning will continue until the EOA character is found. Output will be to all valid destinations provided error processing does not alter the destination.*

2. *If a terminal operator wishes to route a message to more than one location, he may specify the label of one or more terminal names, process file names, or the DLIST destinations in the input text. Multiple ROUTE macros should not be issued to route messages because each time a ROUTE macro is issued, the destination names and queue address are updated. This causes the message to be sent to the destinations detected during the last execution.*

3. *The ROUTE function scans the message text for terminal operator specified destinations. To allow for multiple destinations, the routine builds a dummy DLIST and places into it each destination found in the input text. Because the DLIST processor allows only one nesting of DLISTs, a nested DLIST cannot be handled as a destination for a message. Only the dummy DLIST is handled as a valid destination for the message.*

4. *If end-of-header (in the network buffer) is reached during the scan, the MPPS end-of-header and bad destination flags are set.*

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	ROUTE	eoa-character[,integer] $\left[,\left\{\begin{matrix}H\\M\\\underline{L}\end{matrix}\right\}\right]$

Parameters:

eoa-character
Identifies the 1-byte EBCDIC character that is to terminate scanning of the message header destination field. A space is required between the last destination name and the eoa character.

integer
Is a decimal number, from 1 to 4, specifying the number of *nonblank* characters in each destination code present in the message header. This parameter can be used only when all the destination codes present in the destination field are equal in length.

If omitted, the destination codes are assumed to be of varying lengths from one to four characters and are separated by one or more blank characters.

```
⎧H⎫
⎨M⎬
⎩L⎭
```

Indicates the high, medium, or low priority queue, respectively, of the specified destination.

Example 1:

ROUTE statement:

```
1       10    16
_______________________
        ROUTE #,,H
```

Message text:

```
         INITIAL POSITION         FINAL SCAN
       OF MESSAGE HEADER            POINTER
          SCAN POINTER             POSITION
                ↓                      ↓
xxxROUTE△TEST△PHIL△NYC△WASH△#
```

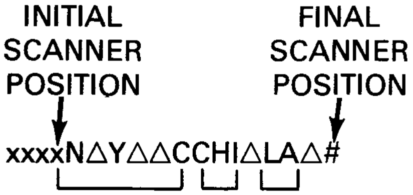
In this example, the message is placed on the high queues, for PHIL, NYC, and WASH, provided they are all valid destinations.

Example 2:

ROUTE statement:

```
1            10      16
________________________________
             ROUTE #,3,L
```

Message text:



This message is placed on the low queues for NYC and CHI. The MPPS bad destination flag is set because LA is a 2-character destination name.

Example 3:

MPPS and network statements in the CCA:

```
1            10      16
_____________________________________________________
TRM1         TERM
TRM2         TERM
TRM3         TERM                            CCA
TRM4         TERM
TRM5         TERM                            entries
DL1          DLIST   TRM1,TRM2
DL2          DLIST   TRM3,TRM4,TRM5
DL3          DLIST   TRM2,DL2
MPP1         MPSTART
             RECHDR
             ADVANCE  1,*
             ROUTE    $                      MPPS
             RECEND
             RECPST                          entries
             SENHDR
             SENEND
             SENPST
```

Message text	Action
*TRM1 $	The message is sent to TRM1.
*TRM1 TRM3 $	The message is sent to TRM1 and TRM3.
*TRM2 DL2 $	The message is sent to TRM2, TRM3, TRM4, and TRM5.
DL1 DL2 $	The message is sent to TRM1, TRM2, TRM3, TRM4, and TRM5.
*TRM1 DL3 $	The bad destination bit is set because two levels of nesting are detected: 1. First level - TRM1, DL3 2. Second level - TRM2, DL2. The user should use an MPPS error macro (CANCELM, ERRMSG, REROUTI) to handle this situation.
TRM3 DL2 $	Unpredictable results - the queuer cannot handle a message being queued to the same file twice (TRM3 DL2 = TRM3, TRM3, TRM4, TRM5).

SEQIN

4.2.7.16. Verify Input Sequence Number (SEQIN)

Function:

Verifies the source terminal message sequence number (supplied by the operator) in the header segment of an incoming message. If the message sequence number is out of sequence, the MPPS bad sequence count flag is set and processing continues with the next-in-line instruction. If end of header is reached during this scan, the MPPS end-of-header flag is set and control is transferred to the RECEND macroinstruction.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	SEQIN	[integer]

Parameter:

integer

Is a decimal number from 1 to 5 indicating the number of nonblank characters comprising the input sequence number in the input sequence count field of a message header. This parameter can be used only when all the input sequence numbers are equal in length (i.e., if SEQIN 3 was specified, then 001 through 999).

If omitted, the input sequence numbers are assumed to be of varying lengths and are separated from subsequent fields by one or more blank characters.

Example:

```
1       10    16
        SEQIN 2
```

4.2.7.17. Insert Output Sequence Number (SEQOUT)

Function:

Inserts an output sequence number in the message header of an outgoing message in the form △nnnn, where the number of digits (n) is specified by parameter 1. If not enough room is in the message header to insert the sequence number, the MPPS bad insertion flag is set, and processing continues with the next-in-line macroinstruction. If the output sequence number can be inserted in the message header, the data already in the message header is shifted left the appropriate number of character positions and the output sequence number is inserted. In either event, the scan pointer does not move from its original position. If an output sequence number is inserted, however, the scan pointer points to the last (units) digit of the sequence number as a result of the shift-left operation. Output sequence numbers are recycled automatically when they reach their maximum limit.

Format:

LABEL	△OPERATION△	OPERAND
[symbols]	SEQOUT	integer

Parameter:

integer
Is a decimal number from 2 to 6 specifying the number of places the output sequence number is to occupy.

Example:

```
1          10     16
--------------------------------
           MPSTART
              .
              .
              .
           SENHDR
           MSGTYP 3,ABC,XYZ
           SEQOUT 3
           BRANCH SENEND
XYZ        SEQOUT 6
SENEND     SENEND
```

Using the coding shown, the following paragraphs illustrate how three output messages would be handled.

- Output message 1

  Before MPPS:

  ```
  SCAN POINTER
       ↓
  △△△△△△ABCxxxxxx
  ```

  After MPPS:

  ```
     SCAN POINTER
          ↓
  △△△ABC△nnxxxxxx
  ```

  where nn is the output sequence count of the destination terminal.

  Since the first three characters of the message are ABC, the scan pointer points to the C following the execution of the MSGTYP instruction. For the sequence-out count, a check is made to make sure at least three blanks (△) precede the text. The blanks are there, so the scanned text (ABC) is shifted left three positions (the number specified on the SEQOUT-call).

  The sequence-out count, originally obtained from the destination terminal's terminal table, is converted to EBCDIC. A leading blank followed by the 2-byte sequence-out count is inserted in the text. The scan pointer is not moved although it now points to the units field of the sequence-out count.

- Output message 2

  Before MPPS:

  ```
  SCAN POINTER
       ↓
  △△△△△△XYZxxxxxx
  ```

  After MPPS:

  ```
  SCAN POINTER
       ↓
  △nnnnnXYZxxxxxx
  ```

  where nnnnn is the output sequence count of the destination terminal.

Since the first three characters of the message are not ABC, the scan pointer is not advanced and control passes to the SEQOUT 6 instruction. Since six blanks precede the text, there is space for insertion. All scanned data (in this case, no data has been scanned) is shifted left six positions and the EBCDIC sequence-out count is inserted (a leading blank followed by the 5-byte count). The scan pointer is not moved, although it now points to the units field of the sequence-out count.

- Output message 3

Before MPPS:

```
SCAN POINTER
      ↓
      ABCxxxxxx
```

After MPPS:

```
   SCAN POINTER
        ↓
      ABCxxxxxx
```

Since the first three characters of the message are ABC, the scan pointer advances to the C following execution of the MSGTYP instruction. A check is made for three blanks preceding the text. The blanks are not present, so no insertion takes place (the corresponding error bit is set). The scan pointer is not moved nor is any data shifted.

NOTE:

On output, the CUP must supply sufficient blanks for insertion.

SOURCE

4.2.7.18. Verify Source Terminal Identification (SOURCE)

Function:

Verifies the source terminal identification as it is coded in the source terminal field of the message header segment of incoming messages. This instruction compares the source terminal field of the message header with the source terminal field in the message header prefix.

If the source name specified in the message header does not agree with the source code in the message header prefix, and it is not a valid source terminal name as known by ICAM, the invalid source flag is set in the error field of the message header prefix. If the source name specified in the message header does not agree with the source code in the message header prefix, but is a valid source terminal name, the bad source flag is set. In both cases, processing continues with the next-inline macroinstruction.

If the end-of-header is reached while scanning the source terminal identification field, the end-of-header flag is set and processing control is transferred to the RECEND macroinstruction.

Format:

LABEL	△OPERATION△	OPERAND
[symbol]	SOURCE	[integer]

Parameter:

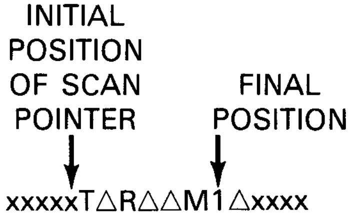
integer
: Is a decimal number from 1 to 4 indicating the number of nonblank characters in the source terminal identification field. This parameter should be specified only when all source terminal identification codes, which are to be processed by this particular SOURCE macroinstruction, are of equal length.

If omitted, the source terminal identification codes are assumed to be of varying length from 1 to 4 characters and are separated from subsequent fields by one or more blank characters.

Examples:

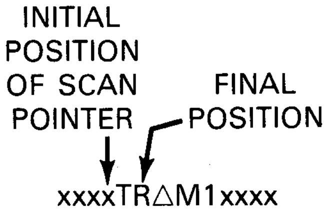
```
    1        10     16
   ┌──────────────────────────
1. │         SOURCE  4
2. │         SOURCE
```

1. In this example, if the following message is received, processing would be as follows:



If TRM1 is a valid source terminal, processing continues and no flags are set. If it is a valid terminal name (i.e., it was specified in the CCA) but is not the source name of the terminal sending the message, the bad source flag is set. If TRM1 is an invalid name, no source flag is set.

2. In this example, the following message would be processed as follows:



Because the length of the source name is not specified, the scanner stops at a blank character and the two characters TR are compared against the true name of the source terminal as stated in Example 1.

4.2.7.19. Insert Time of Day (TIMSTP)

Function:

Inserts the current time of day in the message header segment of incoming or outgoing messages. The format of the time stamp is △hh:mm(△00:00-△23:59). At the conclusion of this instruction, the message header scan pointer points to the last (units) minute digit. If not enough room is in the message header to insert the current time of day, the MPPS bad insertion flag is set and processing is transferred to the next-in-line macroinstruction. If the time can be inserted, the data already scanned is shifted left six character positions and the time is inserted.

Format:

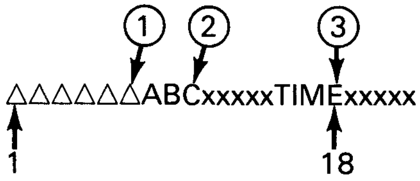
LABEL	△OPERATION△	OPERAND
[symbol]	TIMSTP	unused

Example:

```
1           10      16
_________________________________
MPS2        MPSTART
            RECHDR
            MSGTYP 3,ABC,BR1
            ADVANCE 4,TIME
            TIMSTP
            BRANCH DIR
BR1         MSGTYP 3,XYZ,BR2
```

This example shows the use of the TIMSTP macro in a typical MPPS routine. It also illustrates the operation of the message header scan pointer.

When the message is first received, six blanks (△) are inserted by ICAM because MPPS=(MPS2,6) was specified in the LINE macro or INPUT=(MPS2, 6) was specified in the TERM macro. The input message would appear as follows:



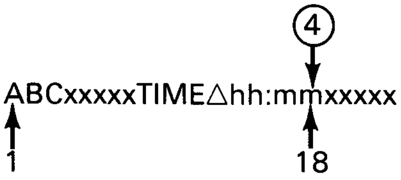
The message header scan pointer's position is indicated by the circled number following execution of the following macroinstructions:

① Initial scan position – Set by RECHDR.

② Position after MSGTYP – Three characters, ABC, found; branch not taken.

③ Position after ADVANCE – The 4-character TIME field was found.

④ Position after TIMSTP – In the following illustration of the same message, note that the message header scan pointer did not move; the text preceding the pointer was shifted to the left and the time was inserted. Following this macro, the pointer is still at position 18.



If insufficient space was specified in the LINE or TERM macro, the bad insertion bit would have been set (Table 4–1). Note that, on *output* messages, the CUP must supply sufficient spaces (blanks) to allow the insertion of the time stamp.

4.3. EXAMPLES OF MPPS USAGE

Example 1:

```
1        10    16                                                       72
--------------------------------------------------------------------------
MPS1     MPSTART
         RECHDR
TYPRØ1   MSGTYP    4,ALTD,TYPRØ2  IF FIRST 4 CHARS NOT ALTD, GO TYPRØ2
         IFSOURCE  TRM7,DIRTRM5   IF FROM TRM7 GO TO DIRTRM5
         DIRECT    ALTD,,M        IF NOT, GO TO MED PRIORITY ALTD
         TIMSTP                   INSERT TIME
         DATSTP    J              INSERT ORDINAL DATE
         BRANCH    REND
DIRTRM5  DIRECT    T,TRM5,M       GO TO TRM5 MED PRIORITY QUEUE
         SEQIN     4              CHECK 4-CHARACTER SEQUENCE COUNT
         BRANCH    REND
TYPRØ2   MSGTYP    4,SRCE,ROUTEIN IF FIRST 4 CHARS NOT SRCE, GO ROUTEIN
         DIRECT    SOURCE,,H      IF FROM SOURCE GO TO HIGH QUEUE
         DATSTP    N              INSERT CALENDAR DATE
         SOURCE                   VERIFY SOURCE ID
         BRANCH    SEQI
ROUTEIN  ROUTE     *,4,L          ROUTE TO DESTINATIONS SPECIFIED
SEQI     SEQIN                    CHECK SEQUENCE NO:
REND     RECEND
         CANCELM   TN#MEOH        END OF HDR DURING SCAN,CANCEL INPUT
         ERRMSG    TN#MBSOR++TN#MNSOR,TRM6,'INCORRECT OR INVALID SOURCE    X
               TERMINAL NAME'     SEND MSG TO TRM6 IF SOURCE ERROR
         REROUTI   TN#MNDST++TN#MBDST,SOURCE,M   INVALID OR NO DEST
         RECPST
         SENHDR
TYPSØ1   MSGTYP    4,ALTD,TYPSØ2  IF FIRST 4 CHARS NOT ALTD, GO TYPSØ2
         IFSOURCE  TRM7,TIMDAT    IF FROM TRM7 GO TO TIMDAT
         SEQOUT    4              INSERT 4-CHARACTER SEQUENCE NO. (ØØØØ)
         BRANCH    SEND
TIMDAT   TIMSTP                   INSERT TIME
         DATSTP    N              INSERT CALENDAR DATE
         BRANCH    SEQO
TYPS02   MSGTYP    4,SRCE,SEQO    IF NOT SRCE, GO TO SEQO
         TIMSTP                   INSERT TIME
SEQO     SEQOUT    6              INSERT SIX-CHAR:SEQUENCE NO:
SEND     SENEND
         REROUTO   TN#MTERR,ALTD,H  REROUTE TO ALTD IF TERMINAL ERROR
         ERRMSG    TN#MBINS,TRM6,'NOT ENOUGH SPACE FOR INSERTIONS'
         SENPST
```

Programming Notes:

MPPS program MPS1 will process messages in the following manner:

1. Message type ALTD (other than source terminal TRM7)

   - Input

     – Message is directed to the alternate destination of the source terminal with a medium priority queue.

  - Time is inserted.

  - Ordinal date is inserted.

- Output

  - Output sequence number is inserted (△nnn).

- Terminal Messages

  - Input message from terminal:

    ALTD△REMAINING MESSAGE TEXT

  - After completion of input side of MPPS, the message looks like:

    ALTD△10:00△74168△REMAINING MESSAGE TEXT

  - After completion of output side of MPPS, the message looks like:

    ALTD△001△10:00△74168△REMAINING MESSAGE TEXT

2. Message type ALTD (the source terminal is TRM7).

- Input

  - Message is directed to TRM5 with a medium priority queue.

  - Sequence in number is required on the input message and must be four characters long.

- Output

  - Time is inserted.

  - Calendar date is inserted.

  - Sequence-out number is inserted (△nnnnn).

- Terminal Messages

  - Input message from terminal (TRM7):

    ALTD△0001△REMAINING MESSAGE TEXT

  - After completion of input side of MPPS, the message looks like:

    ALTD△0001△REMAINING MESSAGE TEXT

- After completion of output side of MPPS, the message looks like:

  ALTD△10:00△74/06/17△00001△0001△REMAINING MESSAGE TEXT

  where the first sequence number is the output sequence number and the second is the input sequence number.

3. Message type SRCE

   ■ Input

   - Message is directed back to the source terminal with a high priority queue.
   - Calendar data is inserted.
   - Source terminal name required on input message in variable length (one to four characters).
   - Input sequence number required on input message in variable length (one to four characters).

   ■ Output

   - Time is inserted.
   - Output sequence number is inserted (△nnnnn).

   ■ Terminal Messages

   - Input message from terminal:

     SRCE△TRM3△02△REMAINING MESSAGE TEXT

   - After completion of input side of MPPS, the message looks like:

     SRCE△74/06/17△TRM3△02△REMAINING MESSAGE TEXT

   - After completion of output side of MPPS, the message looks like:

     SRCE△10:00△00002△74/06/17△TRM3△02△REMAINING MESSAGE TEXT

4. Messages that do not have a message type of ALTD or SRCE

   ■ Input

   - Message will be routed to the destination(s) stated at the beginning of the message and placed on the appropriate low priority queue(s).
   - Input sequence number, variable length, is required on input message.

- Output

  - Output sequence number is inserted (△nnnnn).

- Terminal Messages

  - Input message from terminal:

    TRM3△TRM4*△003△REMAINING MESSAGE TEXT

  - After completion of input side of MPPS, the message looks like:

    TRM3△TRM4*△003△REMAINING MESSAGE TEXT

  - After completion of output side of MPPS, the message looks like:

    △003△TRM3△TRM4*△003△REMAINING MESSAGE TEXT

5. Error processing for all messages

- Receive Side

  - Message is cancelled if end-of-message header text was reached during MPPS scan.

  - Error message is output to terminal TRM6 if an incorrect source terminal name was entered or an invalid terminal name was entered on the message requiring source validation.

  - Reroute of message back to the source terminal if no destination or bad destination is found.

- Send Side

  - Reroute of messages to the alternate destination if a terminal error occurs.

  - Error message is output to terminal TRM6 if there are not enough blank positions reserved in the message header text (TIMSTP, DATSTP, SEQOUT).

In the network definition as shown in Example 2, any message input to the CUP from the UNISCOPE 100 Display Terminal (UIH1) is placed in the UIH2 queue through MPS1. The time is inserted at the front of the message in the form △hh:mm. If for some reason the time is not inserted, the message is canceled.

No special processing is performed by MPS1 for a message output to the UNISCOPE 100 terminal unless an error occurs during initial transmission, in which case the message is retransmitted.

Example 2:

```
1        10    16                                                     72
ONET     CCA   TYPE=(TCI),PASSWORD=DISCNET,FEATURES=OUTDEV
         BUFFERS 5,64,Ø,ARP=2Ø
LØØ1     LINE  DEVICE=(U1ØØ),TYPE=(2ØØØ,SWCH,SYNC),CALL=5556666,      X
               MPPS=(MPS1,6)
U1H1     TERM  ADDR=(28,51),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,             X
               HIGH=Q1H1,MEDIUM=Q1H1,LOW=Q1H1
U1H2     TERM  ADDR=(28,52),FEATURES=(U100,96Ø),PINTV=10,             X
               HIGH=Q1H2,MEDIUM=Q1H2,LOW=Q1H2
U1H3     TERM  ADDR=(29,53),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,             X
               HIGH=Q1H3,MEDIUM=Q1H3,LOW=Q1H3
U1H4     TERM  ADDR=(29,54),FEATURES=(U1ØØ,96Ø),PINTV=1Ø,             X
               HIGH=Q1H4,MEDIUM=Q1H5,LOW=Q1H5
Q1H1     QUEUE TYPE=TERM,DISC=DQFILE1
Q1H2     QUEUE TYPE=TERM,DISC=DQFILE1
Q1H3     QUEUE TYPE=TERM,DISC=DQFILE2
Q1H4     QUEUE TYPE=TERM
Q1H5     QUEUE TYPE=TERM,DISC=DQFILE2
LØØ2     LINE  DEVICE=(DT5ØØ,TTY),TYPE=(3ØØ,SWCH),CALL=5557777,       X
               ID=6,MPPS=(MPS2,9)
TTY1     TERM  FEATURES=(DCT5ØØ),HIGH=QTYH,MEDIUM=QTYL,LOW=QTYL
TEN4     TERM  FEATURES=(1004),ANSWER=(8,C,BLUEBELL)
QTYH     QUEUE TYPE=TERM
QTYL     QUEUE TYPE=TERM,DISC=DQFILE3
QTN4     QUEUE TYPE=TERM,DISC=DQFILE4
MPPSA    MPSTART
         RECHDR
         DIRECT T,U1H2        SEND TO U1H2 TERMINAL QUEUE
         TIMSTP               INSERT TIME
         RECEND
         CANCELM TN#MBINS     CANCEL IF INSERTION ERROR
         RECPST
         SENHDR
         SENEND
         RETRANS X'FFFF'      RETRANSMIT IF ANY ERRORS ON OUTPUT
         SENPST
MPS2     MPSTART
         RECHDR
         MSGTYP 3,TTY,TYPE2   CHECK 3 CHARS',IF NOT TTY GO TO TYPE2
         DIRECT T,U1H3        IF TTY,PUT IN U1H3 TERMINAL QUEUE
         BRANCH REND
TYPE2    MSGTYP 4,1ØØ4,TYPE3  CHECK 4CHARS',IF NOT 1ØØ4 GO TO TYPE3
         ADVANCE 4,DATE       SCAN FOR 4 CHAR' WORD 'DATE'
         DATSTP J             INSERT ORDINAL DATE
         DIRECT T,TEN4        IF 1ØØ4,PUT IN TERMINAL QUEUE
         BRANCH REND
TYPE3    DIRECT SOURCE,,L     PUT IN LOW PRIORITY OF SOURCE AND SEND BACK
REND     RECEND
         RECPST
         SENHDR
         MSGTYP 4,1ØØ4,NOSEQOUT    IF NOT 1ØØ4, GO TO NOSEQOUT
         SEQOUT 3                  IF 1ØØ4, INSERT SEQ.NO
NOSEQOUT SENEND
         REROUTO TN#MLERR,U1H1     IF LINE ERROR, NOTIFY U1H1
         SENPST
DQFILE1  DISCFILE FILEDIV=6
DQFILE2  DISCFILE FILEDIV=6
DQFILE3  DISCFILE FILEDIV=2
DQFILE4  DISCFILE FILEDIV=3
         ENDCCA
```

Any message input via the teletypewriter or the 1004 card processor is processed by MPS2. If the first three characters of the message type are TTY, the message is placed in the U1H3 queue. If the first four characters of the message type are 1004, the message header scan pointer is moved forward to the message header field DATE and the ordinal date is inserted in the form △yyddd. The message is placed in the U1H4 queue.

If neither TTY nor 1004 are the beginning characters of the message, it is directed back to the source by being placed on the low priority queue of the originating terminal.

A message being output to the TTY or the 1004 card processor passes through MPS2. If the first four characters of the message are 1004, the output sequence number is inserted immediately following the characters 1004 in the form △nn. If a line error occurs during output, the message is rerouted to U1H1.

We could use the name CUP shown in Section 3 to execute this network that used to execute the network with no MPPS. The only change to the network definition program is to append an MPPS parameter to the LINE macro. The network definition inclusion of the MPPS instructions then results in all the automatic processing being performed when the CUP is executed.

5. ICAM Loading and Operator Communications

5.1. GENERAL

The integrated communications access method (ICAM) is an extension of the supervisor. It handles data communications tasks. The commands necessary to initialize and/or execute ICAM consist of commands to load the module and operator communications commands to mark up or down the lines, terminals, or ports on which the module is operating.

5.2. LOADING ICAM

Function:

The Cn/Mn command brings in the load module specified to handle the communications task.

Format:

$$\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}$$

where:

n = 1–9

When ICAM load modules are generated, they are named C1-C9 or M1-M9. The response is:

ICAM READY

5.3. OPERATOR COMMUNICATIONS

The operator is sometimes required to type in instructions to ICAM to facilitate processing. These type-ins have the following format:

$$\text{00}\triangle\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}\triangle\text{cc f,}\left\{\begin{matrix}\text{xxxx}\\\text{xx}\end{matrix}\right\}\text{, jj}$$

where:

$\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}$

Is a 2-character name of an ICAM load module (C1–C9) or (M1–M9).

cc

Is a 2-character command.

f

Is a 1-character facility type (L=line, P=port, T=terminal).

xxxx

Is a 1- to 4-character line/terminal and is terminated by a comma. The xxxx field is associated with a user's network, whose job number is jj. If f is a T, it is a terminal. If f is an L, it is a line. P indicates a 2-character value of a port on the CA.

xx

Is a 2-character port indication on the CA.

jj

Is a 2-digit job number.

The following unsolicited type-ins are the only ones that are valid:

$\text{00}\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}\text{UP L, xxxx, jj} = \text{request (open) specified line*}$

$\text{00}\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}\text{DO L, xxxx, jj} = \text{release (close) specified line*}$

$\text{00}\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}\text{UP T, xxxx, jj} = \text{mark terminal specified available (up)}$

$\text{00}\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}\text{DO T, xxxx, jj} = \text{mark terminal specified unavailable (down)}$

$\text{00}\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}\text{UP P, xx, jj} = \text{mark port specified available (up)*}$

$\text{00}\left\{\begin{matrix}\text{Cn}\\\text{Mn}\end{matrix}\right\}\text{DO P, xx, jj} = \text{mark port specified unavailable (down)*}$

**These commands are not supported with transient TCI.*

A typical response to an unsolicited type-in is:

```
MC#90 LINE xxxx MARKED UP, USERjj
```

See system messages programmer/operator reference, UP-8076 (current version), for additional ICAM messages.

5.4. CONSOLE MODIFICATION OF JOURNAL RECORDING

To be supplied

Appendix A. Packets and Tables

A.1. TRANSACTION TERMINAL TABLE

The transaction terminal table (TTT) control section packet is one of a set of request packets for the transaction control interface. This packet can be constructed by defining the needed information as a series of constants or through use of the declarative macro MTABLE.

Figure A—1 depicts the packet format. Its usage is illustrated by an English language field description within the allocated fields. Figure A—2 depicts the entry information following the control section. Tables A—1 and A—2 give detailed byte descriptions of these fields, relating them to the applicable DSECT byte and bit labels. The transaction control section packet associated with this interface has the label TM#TCS, which relates it to the DSECT of that name. Each label in the table has two parts: a prefix and suffix. All labels in Table A—1 are prefixed by TM#TC. The transaction table entry packet has the label TM#TTT, while all labels in Table A—2 are prefixed by TM#TT.

You must generate two work area prefix packets called the output message prefix and the input message prefix. These packets are used by TCI for processing. The output packet has the label TM#TOMSG, and all labels in Table A—3 are prefixed by TM#TO. The input packet has the DSECT label TM#TIMSG, and all labels in Table A—4 are prefixed by TM#TI.

A.1.1. Transaction Terminal Table Control Section

The transaction control section of the MTABLE set contains scheduling addresses and various flags that apply to network level status reporting. In addition, it identifies your network CCA tables. Figure A—1 and Table A—1 show the control section format. Following is a description of each entry in this section:

- Word 1 - CCA Network Name (TM#TCCA)

  The 4-character name of your network-defined tables (set by the MTABLE macro). The name must correspond to the name that is specified at ICAM system generation time in the label field of the CCA macro. The MOPEN macroinstruction automatically executes the functions of the NETREQ macroinstruction using this network name.

<table>
<tr><th>Byte</th><th>0</th><th>1</th><th>2</th><th>3</th><th>Word</th></tr>
<tr><td>0</td><td colspan="4">CCA network name</td><td>1</td></tr>
<tr><td>4</td><td>preview size</td><td colspan="3">input notification address</td><td>2</td></tr>
<tr><td>8</td><td>sectors per slot</td><td colspan="3">delivery notice address</td><td>3</td></tr>
<tr><td>11</td><td>unused</td><td colspan="3">SVC request completion address</td><td>4</td></tr>
<tr><td>17</td><td>unused</td><td colspan="3">contingency address</td><td>5</td></tr>
<tr><td>20</td><td>▨ network flags</td><td colspan="3">▨ reserved</td><td>6</td></tr>
<tr><td>24</td><td>control flags</td><td>reserved</td><td colspan="2">input summary</td><td>7</td></tr>
<tr><td>28</td><td colspan="4">reserved</td><td>8</td></tr>
<tr><td>32</td><td colspan="4">reserved</td><td>9</td></tr>
<tr><td>36</td><td colspan="4">reserved</td><td>10</td></tr>
</table>

NOTE:

▨ Shaded areas are system-supplied parameters

Figure A—1. Transaction Terminal Table Control Section Functional Field Description

- Word 2 – Preview Size and Input Notification Address (TM#TCPRE and TM#TCIAD)

Field 1 – Preview Size

Contains the number of bytes in which message text is placed in the TTT from input messages prior to scheduling your program at the input notification address. Leading blanks, DICE sequences, or field control character (FCC) sequences are skipped over prior to transferring the text into the preview area. (FCC sequences are five bytes of field control characters for the UTS 400.)

Field 2 – Input Notification Address

Is the address in your program area that ICAM schedules as an activity for each message as it arrives in the system. A parameter passed to your program is the address of the associated TTT (in register 1). Error flags concerning the input message are passed in register 0 (see 3.14).

Table A—1. Transaction Terminal Table Control Section Detailed Field Descriptions

Word	Field					Content	Comment	Byte
	Word/Bit Label Suffix *	Type and Length	Set **					
			D	U	P			
1	CA	CL4	X			CCA network name	Keyword CCA	0
2	PRE	XL1	X			Size of preview area	In TTT input only	4
	IAD	XL3	X			Input notification address	Keyword INOTICE	5
3	SCT	XL1	X			No. of disc sectors/termslot	Sector = 256 bytes	8
	DAD	XL3	X			Delivery notice address	Output message address in your program (optional)	9
4	OP1	XL1				Unused byte		12
	SVC	XL3	X			User SVC completion address	All M-functions return to this address in your program. Keyword COMPL.	13
5	OP2	XL1				Unused byte		16
	ONT	XL3	X			User contingency address	Error return address. Keyword ERROR	17
6	NET	XL1			X	Network status flags		20
	LIN	XL3			X	Line status flags		21
7	FLG	XL1				Control flags		24
	SDN		X			Suppress delivery	Keyword DELIVERY	
	SIN		X			Suppress input	Keyword SINPUT	
	OVR		X			Input overrun — status notification desired	Keyword OVRUN	25
	OP4	XL1						
	NPI	XL2			X	Input summary		26
8	TIM	XL4				Reserved		28
9	PRC	XL4				Reserved		32
10	SZE	XL4				Reserved		36

* All labels have the prefix TM#TC. The DSECT label of this table is TM#TCS.

** D indicates set by declarative macro.
U indicates to be set by user.
P indicates set by software processor.

- Word 3 — Sectors per Slot and Delivery Notice Address (TM#TCSCT and TM#TCDAD)

Field 1 — Sectors per Slot

Sectors per slot is the number of 256-byte sectors that comprise a disk slot. The delivery notice address is the address in your program area that is scheduled for each output message after it has been transmitted from the system. If this field is zero, no scheduling will be performed.

Field 2 — Delivery Notice Address

The delivery notice parameter in the message prefix area of the transmitted message is returned as a parameter to your program (in register 0). The delivery notification status (see 3.13) is returned in the upper byte of register R1. The TTT address is returned in the three low-order bytes of register R1.

- Word 4 — SVC Request Completion Address (TM#TCOP and TM#TCSVC)

Field 1 — Unused (TM#TCOP1)

Field 2 — SVC Completion Address

See 3.15.

- Word 5 — Contingency Address (TM#TCOP2 and TM#TCONT)

Field 1 — Unused (TM#TCOP2)

Field 2 — Contingency Address

See 3.12.

The address in your program area that is scheduled as an activity when an error of a network nature occurs and may be recoverable or unrecoverable. This may be associated with disk I/O errors, CA subsystem failures, buffer pool depletions, or line errors. Depending upon the type, the status may be returned in register R0 or in the transaction control section.

- Word 6 — Network Flags and Line Flags (TM#TCNET and TM#TCLIN)

The network flag field contains network-oriented status flags. Indicators that affect the operation of the total system are reported in this field. Notification is at the contingency address. The 3-byte field TM#TCLIN is currently undefined.

- Word 7 - Control Flags and Input Summary (TM#TCFLG, TM#TCOP4, and TM#TCNPI)

Field 1 - TM#TCFLG

This 1-byte field contains indicators set by the MTABLE macroinstruction. They are used to suppress delivery notice or suppress input notification when the user is busy. Another indicator can be set when input overrun status notification is desired.

Field 2 - Unused (TM#TCOP4)

Field 3 - Input Summary (TM#TCNPI)

The input summary field is a 2-byte field indicating the number of input messages that have been received by ICAM but not yet processed by the user program.

- Word 8 – Reserved

  The entire 4-byte field is reserved for TCI use.

- Word 9 – Reserved

  The entire 4-byte field is reserved for TCI use.

- Word 10 – Reserved

  The entire 4-byte field is reserved for TCI use.

A.1.2. Transaction Terminal Table Entry Section

A transaction terminal table (TTT) entry is created for each terminal that is defined in your CCA network generation. These tables reside in your program area and are referenced by both you and TCI. Figure A–2 and Table A–2 show the format of this table. Following is a description of each entry in the table:

- Word 1 – Tflags, NCC, and Slot Bits (TM#TTFLG, TM#TTSL1, and TM#TTSL2)

  The 1-byte Tflags field indicates the terminal's status and current terminal activity. The most significant bit of this byte is the IRL indicator for all service requests. Additional bits indicate the last terminal table entry, whether the terminal is up or down, and whether an input message is available.

  The 1-byte NCC field contains a count of the number of characters skipped prior to transferring text into the message preview area. This generally means the number of blanks or DICE sequence characters that preceded the text.

  The bits contained in the two slot bytes are used to indicate the presence of a message in the disk terminal slots when disk buffering is specified. The first byte indicates an input message present; the second byte indicates an output message present on any of the four priority slots. The remaining four bits indicate presence of any additional low-priority output messages.

- Word 2 – Terminal name (TM#TTNAM)

  This 4-character name is the destination or source name of the terminal as defined by the TERM macroinstruction at CCA network generation time. The MOPEN macroinstruction causes this field to be initialized with the name from the CCA network tables.

- Word 3 – Tflags and Message Header Address (TM#TTFLG2 and TM#TTMSG)

  The 1-byte field (Tflags 2) is used to indicate additional status and terminal activity, such as batch mode processing active.

  The message header address field contains the address of an input message header when main storage queueing is specified for input. The message is not linked in a specific message queue but is staged in the network buffer pool in the CCA. This field is not a user field.

Byte	0	1	2	3	Word
0	Tflags 1	▨ NCC	▨ disk message		1
			▨ slot bits (high med)	▨ slot bits (low)	
4	▨ terminal name				2
8	Tflags 2	message header address			3
12	▨ input message count		▨ output message count		4
16	user flags — input	▨ TCI flags — input	user flags — output	▨ TCI flags — output	5
20	terminal number		reserved		6
24	message prefix address				7
28	▨ input disc sector address				8
32	▨ input sequence count		▨ line number		9
36	auxiliary device/ special function (input)	auxiliary device index	▨ character count		10
40	user table address				
44	↑ message text preview area (variable) ↓				

NOTE:

▨ Shaded areas are system-supplied parameters

Figure A—2. Transaction Terminal Table Entry Section Functional Field Description

- Word 4 — Input Message and Output Message Count (TM#TTIT and TM#TTOT)

  The input field contains the accumulated total of input messages from this terminal.

  The output field contains the accumulated total of output messages to this terminal.

- Word 5 — Input/Output Flags

  Field 1 — User Flags — Input (TM#TTIUS)

  This 1-byte field provides indicators for you to initiate functions that are to be performed by TCI. Some of these are:

  — Cancel input message

  If set, this flag causes the MWRITE or MALERT macro to perform a CANCELIN function to release the input message. The indicator is cleared by TCI upon completion of the SVC.

  — Inhibit input

  If set, this flag causes the MALERT macro to perform a TSTATUS function to inhibit input on a terminal. This indicator is cleared upon completion of the SVC.

  — Date and time stamping

  If set, this flag causes MREAD processing to transfer the stamping to your input message prefix.

  — Input overrun

  If set, this flag indicates that an input message was received while another was still in process. Preview shows overrun message.

  — Transaction number stamping

  If set, this flag causes MREAD processing to transfer the stamping to your input message prefix.

  — Source stamping

  If set, this flag causes MREAD processing to transfer this stamping to your input message prefix.

Field 2 — TCI Flags — Input (TM#TTTCI)

This 1-byte field provides indicators for TCI to notify you of certain conditions associated with a message. See 3.14.

Some of these are:

— Over-run condition

— Input parity error (unbuffered device)

— Truncated input message (user-supplied buffer too small for message)

These indicators will be supplied to you in byte 3 of register 0 on either MREAD SVC completion or input notification and remain until the next input message is received.

Field 3 — User Flags — Output (TM#TTOUS)

This byte field provides indicators for you to initiate functions or set conditions that are to be performed by TCI when processing the MALERT TSTATUS function (Table 3—1). Some of these are:

— Hold high priority output queue

— Hold medium priority output queue

— Hold low priority output queue

— Remove high priority queue outputs

— Remove medium priority queue outputs

— Remove low priority queue outputs

— Release high priority queue outputs

— Release medium priority queue outputs

— Release low priority queue outputs

You are responsible for setting these indicators prior to executing the MALERT macroinstruction for the TSTATUS function. These indicators are cleared upon completion of the SVC except for the hold indicators. These are not cleared until the MALERT macro is executed, indicating a release of the output queues.

Field 4 — TCI Flags — Output (TM#TTQUE)

This 1-byte field is used by TCI to save the priority of output messages that you may wish to retransmit.

- Word 6 – Terminal Number (TM#TTNUM and TM#TTOPN)

Field 1 – Terminal Number (TM#TTNUM)

The terminal number is the sequential number of the terminal with a range of 1 to n and is derived from the order in which the terminals are defined in the CCA network generation.

Field 2 – TM#TTOPN

This 2-byte field is reserved for TCI use.

- Word 7 – Message Prefix Address (TM#TTUWA)

This 4-byte field identifies the current message prefix address to be processed by TCI for either MREAD/MWRITE or MSWITCH macros. This address must be within your program area, on a half-word boundary, and contain a message prefix dependent on the function to be processed.

- Word 8 – Input Disk Section Address (TM#TTSLT)

This 4-byte field is used exclusively by TCI and contains the input disk sector address used to calculate disk addresses for input and/or output message flow for the terminal.

- Word 9 – Input Sequence Count (TM#TTSEQ and TM#TTRSV) and Line Number (TM#TTLNM)

Field 1 – Input Sequence Count

This 2-byte field contains a unique input transaction sequence count of the last input received by this terminal. You can use this field to determine the order in which all current input came into the system.

Field 2 – Line Number

The sequential number of the line from 1 to n to which the terminal is attached. The MOPEN macroinstruction causes this field to be initialized with the sequential line number.

- Word 10 – Auxiliary Device/Special Function, Auxiliary Device Index, and Input Character Count (TM#TTIAX and TM#TTICT)

Field 1 – Auxiliary Device/Special Function (TM#TTIFC) and Auxiliary Device Index (TM#TTIDV)

This 2-byte field identifies input from an auxiliary device attached to a station or to special function keys on a UNISCOPE terminal. It has the same format and definition as defined for the process file DTF of the standard MCP.

Field 2 — Character Count (TM#TTICT)

This 2-byte field contains the total character count of the input message text that is currently available. It permits you to allocate a precise buffer size prior to executing the MREAD macroinstruction.

Field 3 - Message Text Preview Area (TM#TTEXT)

This variable-length field contains the absolute text that is contained at the beginning of an input message. It is not altered by TCI until a subsequent message is received from the terminal. The size of this area is defined in the MTABLE macroinstruction, with a default value of 12 bytes.

■ Word 11 - User Table Address (TM#TTUDS)

This 4-byte field contains the address of the extension area for this terminal, if specified in the MTABLE macroinstruction. A user table is generated for each terminal through the EXTAREA parameter. These extension tables are for the exclusive use of your program. A value of zero in this field indicates that no table has been generated.

Table A—2. Transaction Terminal Table Entry Section Detailed Field Descriptions (Part 1 of 3)

Word	Field					Content	Comment	Byte
	Word/Bit Label Suffix*	Type and Length	Set**					
			D	U	P			
1	FLG	XL1				IRL & terminal flags		0
	IRL			X		Immediate return line flag	User flag	
	DWN				X	Terminal down flag	Set by MOPEN	
	OUT				X	Output in progress	Set by TCI	
	IN				X	Input in progress	Set by TCI	
	LST		X			Last TTT flag	Set by MOPEN	
	AVL				X	Input MSG available	Set by TCI	
	INH		X			TCI controlled inhibit input	Set by MTABLE	
	DER				X	DISC error input	Set by TCI	
	NCC	XL1			X	Number of control characters and/or DICE characters not shown in preview	Set by TCI	1
	SL1	XL1			X	Input slot flags		2
	INP				X	Input disk slot full	Set by TCI	
	RTS				X	Retransmit requested	Set by MALERT	
	SL2	XL1			X	Output slot flags	Set by TCI	3
	TOP				X	Top priority slot full	Transient only	
	HGH				X	High priority slot full	Transient only	
	MED				X	Medium priority slot full	Transient only	
	LW1				X	Low priority slot 1 full	Transient only	
	LW2				X	Low priority slot 2 full	Transient only	
	LW3				X	Low priority slot 3 full	Transient only	
	LW4				X	Low priority slot 4 full	Transient only	
	LW5				X	Low priority slot 5 full	Transient only	
2	NAM	CL4			X	Terminal name	(Set to TCT name) by MOPEN	4

*All labels have the prefix TM#TT. The DSECT label of this table is TM#TTT.
**D indicates set by declarative macro.
U indicates to be set by user.
P indicates set by software processor.

Table A—2. Transaction Terminal Table Entry Section Detailed Field Descriptions (Part 2 of 3)

Word	Field					Content	Comment	Byte
	Word/Bit Label Suffix*	Type and Length	Set**					
			D	U	P			
3	FLG2	XL1			X	Terminal flag byte 2	Set by MOPEN	8
	BAT				X	Terminal is batch device	Set by MOPEN	
	MSG	XL3			X	TCI working storage location	Input – netbuf address set by TCI	9
4	IT	XL2			X	Number of input transactions	Set by TCI	12
	OT	XL2			X	Number of output transactions	Set by TCI	14
5	IUS	XL1		X		User input flags	Set by user	16
	CAN			X		Cancel input flag	Cleared by TCI	
	HIN			X		Inhibit input until MALERT macro issued to enable input	Cleared by TCI MALERT ENABLEIN	
	DAT		X			Time/date stamping on input	Set by MTABLE	
	RNM		X			Transaction number stamping on input	Set by MTABLE	
	SRC		X			Source stamping on input	Set by MTABLE	
	IOS				X	Input overrun occurred	Set by TCI	
	TCI	XL1			X	TCI input error flags	Set by TCI	17
	IPE				X	Input parity/block check error		
	RUN				X	Truncated input message		
	EOF				X	Batch end-of-file		
	TCO	XL2				User output flags		18
	HHY					Hold high priority output	Cleared by TCI on MALERT when release flag is set	
	HMD					Hold medium priority output		
	HLW					Hold low priority output		
	RHY			X		Release hold on high priority output	Cleared by TCI or MALERT	
	RMD			X		Release medium priority hold		
	RLW			X		Release low priority hold		
	CHY			X		Clear high priority output		
	CMD			X		Clear medium priority output		
	CLW			X		Clear low priority output		
	QHD			X		All hold flags set		
	QRL			X		All release flags set		
	CQU			X		All clear flags set		
	QUE				X	TCI Output flags	Set by TCI	19
	HQH				X	High queue on hold for restransmit		
	MQH				X	Medium queue on hold for retransmit		
	LQH				X	Low queue hold for retransmit		
6	NUM	XL2	X			Terminal number	Set by MOPEN	20
	OPN	XL2	X			reserved		22

*All labels have the prefix TM#TT. The DSECT label of this table is TM#TTT.
**D indicates set by declarative macro.
U indicates to be set by user.
P indicates set by software processor.

Table A—2. Transaction Terminal Table Entry Section Detailed Field Descriptions (Part 3 of 3)

Word	Field					Content	Comment	Byte
	Word/BIT Label Suffix*	Type and Length	Set**					
			D	U	P			
7	UWA	XL4		X		Message prefix address	Set by user	24
8	SLT	XL4			X	Input slot	Set by MOPEN	28
9	SEQ	XL2			X	Input sequence count	Set by TCI	32
	LNM	XL2			X	Line number	Set by MOPEN	34
10	IAX	XL2			X	Auxiliary Device/Special Function and Auxiliary Device Index	Set by TCI	36
	IFC				X	Auxiliary Device/Special Function	1-byte field	
	CMW				X	Send computer message	Set by TCI	
	FK1				X	Function key 1 received	Set by TCI	
	FK2				X	Function key 2 received	Set by TCI	
	FK3				X	Function key 3 received	Set by TCI	
	FK4				X	Function key 4 received	Set by TCI	
	IDV				X	Auxiliary Device Index	1-byte field	37
	ICT	XL2			X	Input character count	Set by TCI	38
11	UDS		X			Address of user work area for this terminal	Set by MTABLE	40
	EXT					Equate for start of text in preview area.	Set by TCI	44

*All labels have the prefix TM#TT. The DSECT label of this table is TM#TTT.

**D indicates set by declarative macro.
U indicates to be set by user.
P indicates set by software processor.

A.1.3. Output Message Prefix Format

An output message prefix (OMSG) entry is required for each MWRITE and MSWITCH macroinstruction. These prefixes reside in your program area and are referenced by both you and TCI. Figure A—3 and Table A—3 show the format of this prefix.

Byte	0	1	2	3	Word
0	output flags	output priority	paging flags		1
4	destination ID				2
8	delivery notice parameter				3
12	auxiliary device/ special function	auxiliary device index	output character count		4

Figure A—3. Output Message Prefix Functional Field Description

Following is a description of each entry in the output message prefix:

- Word 1 — Output Flags, Output Priority, and Paging Flags (TM#TOFLG, TM#TOPRI, and TM#TOPGE)

  Field 1 — Output Flags

  This 1-byte field indicates various functions to be performed by TCI for the output message. Some of these functions are:

  — Cancel input and deliver output

  Indicates to TCI that the preview text was sufficient for processing the input message and that an MREAD is not required. TCI knows then to remove the current input message identified by the terminal specified as a parameter on the MWRITE macroinstruction.

  — Enable input of output EOM

  Indicates to TCI that you configured the TCI-controlled inhibit input function by your MTABLES macro generation and now desire more input from the terminal after output is delivered to that terminal.

  — Dequeue output message and hold next output delivery

  Indicates to TCI, after the message is delivered by the MWRITE to its destination, to set that queue on hold so that no more output is delivered from that queue until you do an MALERT TSTATUS function to release it. This will put the queue on hold whether or not there are messages on the queue. The message just delivered will be released unless you also had set the 'do not dequeue output message' function flag. In this case the hold will be set and the message will not be released.

  — Do not dequeue output message

  Indicates that a possible retransmission of the output message may be required and that TCI should not release this output message after it is delivered to its destination for the first time. A requirement for this function is that at some later point (either at delivery notice or next input notification) you will issue an MALERT macro to retransmit or not retransmit (same as dequeue/release) that output message.

  — Suppress delivery notice

  Indicates to TCI that you do not desire a delivery notification for this output message.

  Field 2 — Output Priority

  This 1-byte field contains the priority at which the output message is to be queued.

If disk buffering is specified for output, it identifies the disk terminal slot into which the message will be written prior to transmission.

Field 3 — Paging Flags

This 2-byte field provides paging control for a series of output messages. Specific functions and usage are currently undefined.

- Word 2 — Destination Identifier (TM#TODID)

This 4-byte field identifies one or more destinations for the message. The field may take one of the following forms:

1. The 4-character name of a terminal as defined by a TERM macroinstruction during CCA network generation. This name is also the same name placed into each TTT by the MOPEN macroinstruction.

2. The name of a destination list (DLIST) as defined in the CCA network generation.

3. If the high-order byte of this field is zero, it is interpreted as the address of a user's routing list that has been constructed dynamically by your program. This list may contain any number of destination names. No name in this list may reference a DLIST, which is a predefined group code.

- Word 3 — Delivery Notice Parameter (TM#TOPAR)

This 4-byte field is returned to your program in register 0 when delivery notification scheduling is performed after the message is delivered to its destination. It is intended to uniquely identify the message to your program. An additional parameter passed to delivery notification is the address of the TTT associated with the destination to which the message was delivered in register 1.

- Word 4 — Auxiliary Device/Special Function, Auxiliary Device Index, and Output Character Count (TM#TOAUX and TM#TOCNT)

Field 1 — Auxiliary Device/Special Function

A bit-oriented field used to control an auxiliary device or handle special features of a terminal.

Field 2 - Auxiliary Device Index

This field contains the logical auxiliary device index. Refer to the AUXn parameter of the TERM macro (2.2.10).

Field 3 — Output Character Count

This 2-byte field specifies the number of characters of message text that immediately follow this prefix.

Table A—3. Output Message Prefix Detailed Field Descriptions

Word	Field					Content	Comment	Byte
	Word/Bit Label Suffix*	Type and Length	Set**					
			D	U	P			
1	FLG	XL1		X		Output message flags byte		0
	CIN					Cancel input and deliver output		
	ENI			X		Enable input at output EOM		
	RSD			X		Do not dequeue message		
	SDN			X		Supress delivery notification		
	HLD			X		After delivery, release message, set hold on queue to deliver no more output.		
	EOF				X	Batch end-of-file		
	PRI	XL1		X		Output priority flags byte		1
	TPB			X		Top break priority		
	TOP			X		Top priority		
	HIG			X		High priority		
	MED			X		Medium priority		
	LOW			X		Low priority		
	PGE	XL2		X		Output paging flags	Not yet defined	2
2	DID	CL4		X		Destination identifier		4
3	PAR	XL4		X		Delivery notice parameter		8
4	AUX	XL2		X		Auxiliary Device/Special function and Auxiliary Device Index		12
	FUN					Auxiliary Device/Special Function		
	MWK or MWS					Notify terminal msg. waiting		
	MWK					Notify DCT 1000 message waiting and sound alarm.		
	PRT ++ TAT					Load program on diskette (UTS 400)		
	PRT					Print(write) to TCS, COP, NIP		
	RDB					Read a block on TCS		
	BOB					Backspace one block on TCS		
	SRH					Search TCS		
	RPA					Report address on TCS		
	PRF					Print form	UTS 400 only	
	XAL					Transfer all	UTS 400 only	
	XVR					Transfer variable	UTS 400 only	
	XCG					Transfer changed	UTS 400 only	
	PRE					Prevent nonsignificant space suppression		
	TAT					Transparent auxiliary device transfer.		
	DEV					Auxiliary device index	One byte	13
	CNT	XL2				Number of text characters in output message.		14

*All labels have the prefix TM#TO. The DSECT label of this table is TM#TOMSG.

**D indicates set by declarative macro.
U indicates to be set by user.
P indicates set by software processor.

A.1.4. Input Message Prefix Format

Figure A—4 and Table A—4 show the format of your input message prefix area that is associated with the MREAD macroinstruction. The first word of the prefix is required, while the remaining four are optional. If the field in words 2 through 5 were not specified in the MTABLE macroinstruction, they are deleted as entries in the prefix. When deleted, they are replaced by the area where text commences. Following is a description of the entries in this prefix area:

- Word 1 — Character Count and Reserved Bytes (TM#TICNT and TM#TIRSV)

  Field 1 — Character Count

  This 2-byte field contains a count of the number of text bytes in the input message. The count includes 4 for the first word plus the optional fields, if specified, in words 2 through 5.

  Field 2 — Auxiliary Device/Special Function and Auxiliary Device Index

  This 2-byte field identifies input from an auxiliary device attached to a station or to special function keys on a UNISCOPE display terminal. This field is also contained in the TTT during preview of the input message. It is passed to the input message prefix since, at the time of MREAD completion, the TTT field may relate to an input overrun message and not the message being transferred to you by the MREAD macro.

- Word 2 — Source Name (TM#TISRC) (Optional)

  The 4-character name of the terminal from which the message was received. This name corresponds to the name specified in the TERM macroinstruction during CCA network generation.

- Word 3 — Date Stamp (TM#TIDAT) (Optional)

  The current date, in ordinal form, when the message arrived into the system.

- Word 4 — Time Stamp (TM#TITIM) (Optional)

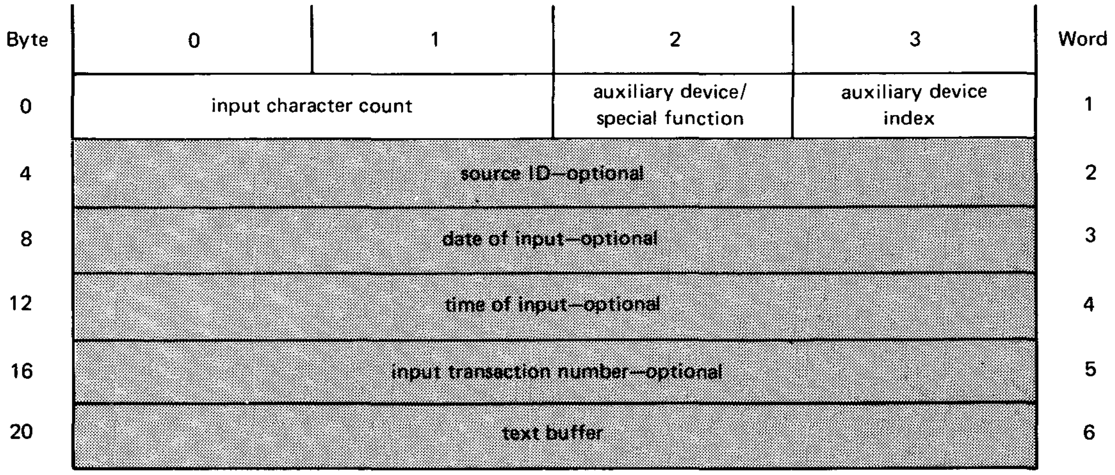
  The time of day, in milliseconds, when the message arrived into the system.

- Word 5 — Sequential Transaction Number Stamp (TM#TITRN) (Optional)

  A unique number assigned to the input message. This number is maintained in the CCA and is incremented after each assignment to any input message.

- Word 6 — Text Buffer

  The input message area.



Byte	0	1	2	3	Word
0	input character count		auxiliary device/ special function	auxiliary device index	1
4	source ID—optional				2
8	date of input—optional				3
12	time of input—optional				4
16	input transaction number—optional				5
20	text buffer				6

NOTE:

Shaded areas are system-supplied parameters

Figure A—4. Input Message Prefix Functional Field Description

Table A—4. Input Message Prefix Detailed Field Descriptions

Word	Field					Content	Comment	Byte
	Word/Bit Label Suffix *	Type and Length	Set **					
			D	U	P			
1	CNT FNC DEV	XL2 XL1 XL1		X	 X X	Size in bytes of input buffer Auxiliary device/special function Auxiliary device index		0 2 3
2	SRC	CL4			X	Source ID of input – optional		4
3	DAT	XL4			X	Date of input – optional	Ordinal date	8
4	TIM	XL4			X	Time of input – optional	Binary millisecond time of day	12
5	TRN	XL4			X	Input transaction number – optional		16
6	LNG	Variable			X	Input message		20

* All labels have the prefix TM#TI. The DSECT label of this table is TM#TIMSG.

** D indicates set by declarative macro.
U indicates to be set by user.
P indicates set by software processor.

A.2. ERROR RECOVERY PROCEDURE FOR A CUP

The user can construct his own error recovery procedure through the use of MPPS. (See Section 4.)

A.3. DUST ERROR PROCESSING PROCEDURES

Each of the deferred user service transient (DUST) macroinstructions generates, in conjunction with the supervisor, a parameter table into which control information concerning the processing status of each instruction is stored. When an error condition is detected during processing of one of the DUST instructions, an error field in the table is set to relate the error condition. Processing of the instruction is then halted, and control is transferred to an error return address, if possible. In addition, the error status is stored in the low order two bytes of register 0. When control is returned to an error address, register 1 points to the parameter table generated by the macro.

The error return address for DUST functions is specified via the ERRET operand of the MOPEN macroinstruction. When ICAM is unable to return control to your user program at the address specified in the ERRET parameter, control is returned (inline) to the next instruction in your program following the incomplete DUST request. In addition, certain illogical requests may cause ICAM to cancel your user program. (See Table A-13.)

By calling the proc TQ#X, the user program can access the DSECTs that map the various tables as shown in Table A—5.

Table A—5. TQ#X Labels for Mapping Common Part of DUST Function Tables

Label	Length	Content
TQ#XERR	OH	Error half word
TQ#XER1	XL1	Error byte 1
TQ#XER2	XL1	Error byte 2

A.4. ERROR CONDITION TABLES

Tables A-6 through A-12 contain the error conditions that may be detected during processing of any of the DUST imperative macroinstructions. The error conditions described are stored in parameter tables generated by the macroinstructions themselves. A list of cancel conditions is contained in Table A-13.

Table A—6. LNEREL Error Conditions

Byte	Error Code	Cause/Condition ①	Result
TQ#LER1	TU#DDTAE	LNEREL table outside user region	②
	TU#DTBE	LNEREL table not full-word aligned	②
	TU#DDTLE	LNEREL table length incorrect	②
	TU#DFE	LNEREL table flag incorrect	②
TQ#LER2	TU#DLNNE	Line name specified does not exist.	③
	TU#DLAR	Line name specified has already been released or was never requested.	③

NOTES:

① The following ICAM cancel codes may occur:

450 Network not active; either no MOPEN or no NETREL has been done.

441 No available ARP in user network to process DUST request.

② Control is returned inline following this request. The request is not honored.

③ Control is returned at the address specified in the ERRET operand of the MOPEN macro. The request is not honored.

Table A—7. LNEREQ Error Conditions

Byte	Error Code	Cause/Condition ①	Result
TQ#LER1	TU#DDTAE	LNEREQ table outside a user region	②
	TU#DDTBE	LNEREQ table not full-word aligned	②
	TU#DDTLE	LNEREQ table length incorrect	②
	TU#DFE	LNEREQ table flags incorrect	②
TQ#LER2	TU#DLNNE	Line name specified nonexistent	③
	TU#DLAR	Line name specified already active	③
	TU#DNLLT	Line cannot be mapped; no available port	③
	TU#DNCCT	No available communications adapter (CA) subsystems available for this line	③
	TU#DLCAE	CA initialization error	③

NOTES:

① The following ICAM cancel codes may occur:

450 Network not active; either no MOPEN or no NETREL has been done.

441 No available ARP in user network to process DUST request.

② Control is returned inline following the request. The request is not honored.

③ Control is returned at the address specified to the ERRET operand of the MOPEN macro. This request is not honored.

Table A—8. NETREL Error Conditions

Byte	Error Code	Cause/Condition ①	Result
TQ#NER1	TU#DDTAE	NETREL table outside user region	②
	TU#DDTBE	NETREL table not full-word aligned	②
	TU#DDTLE	NETREL table length incorrect	②
	TU#DFE	NETREL table flags incorrect	②
	TU#CCAE	CA initialization error	③

NOTES:

① The following ICAM cancel code may occur:

450 Network not active; either no MOPEN or no NETREL has been done previously. Except for this condition', the NETREL is performed in all cases.

② Control is returned inline following the request. The request is not honored.

③ Control is returned at the address specified to the ERRET operand of the NETREQ/MOPEN macro. This request is not honored.

Table A—9. NDETACH Error Conditions

To Be Supplied

Table A—10. MOPEN Error Conditions

Byte	Error Code	Cause/Condition ①	Result
TQ#NER1	TU#DDTAE	MOPEN table outside user region	②
	TU#DDTBE	MOPEN table not full-word aligned	②
	TU#DTLE	MOPEN table length incorrect	②
	TU#DFE	MOPEN flag incorrect	②
	TU#DRAN	Network requested already active	④
	TU#DERRA	Error return address not in user region	②
	TU#DPASS	Invalid password; password or macro does not match MOPEN password.	②
	TU#DDSKE	Disk error opening a file	③
	TU#DDSKA	Attach error occurred – disk queueing.	③
	TU#DDSKR	A disk error occurred while reading a file – disk queueing.	③
	TU#DDSKF	File error (file characters do not match those in CCA) – disk queueing.	③
	TU#DTCIA	MOPEN of TCI network invalid TCS address	③
	TU#DTCER	TCI network does not match CCA network.	③
	TU#DCCAE	Unable to load CCA due to loader error; main storage saturation to duplicate CCA	③
	TU#DDSOE	SAT error returned when attempting to open the TCI disk file	③
	TU#DCSAT	CCA saturation – ARP unavailable	③
	TU#DUSAT	User saturation – ICAM user slot unavailable to log in user	③
TQ#NER2	TU#DNLLT	One or more lines in the network could not be mapped to a physical port.	③
	TU#DNCCT	No available CA tables for one or more lines	③
	TU#DLNNE	Line name specified nonexistent	③
	TU#DLAR	Line name specified already active	③
	TU#JERR	Journal file initialization error has occurred.	③
	TU#DLCAE	CA initialization error	③

NOTES:

① The following ICAM cancel codes may occur:

440 User has done a MOPEN but has no CCA.

441 No ARP available to process the user request.

② Control is returned inline following this request. The request is not honored.

③ Control is returned at the address specified in the ERRET operand of the MOPEN macro. The request is not honored.

④ User program is cancelled.

Table A—11. NATTACH Error Conditions

To Be Supplied

Table A—12. QDEPTH/QHOLD/QRELSE/CCACPY/QCLEAR/TRMREP Error Conditions

Byte	Error Code	Cause/Condition	QDEPTH	QHOLD	QRELSE	CCACPY	QCLEAR	TRMREP
TQ#QERR	TQ#QLNF	Line not found.	Y	Y	Y	N	Y	Y
	TQ#QTNF	Terminal not found.	Y	Y	Y	Y	Y	Y
	TQ#QNF	Queue not found.	Y	Y	Y	N	Y	N
	TQ#QNPF	Process file not found.	Y	N	N	N	Y	N
	TQ#QTLG	Data beyond table or program area.	Y	N	N	Y	N	N
	TQ#QTNFW	Table not on full word boundary.	Y	N	N	Y	N	N
	TQ#QIQT	Invalid queue priority	N	N	N	N	N	N
	TQ#QWAE	Work area error.	Y	N	N	Y	N	N
	TQ#NDTH	Phone number too long	N	N	N	N	N	Y

NOTES:

Y indicates that this error may be set for this macro.

N indicates that this error is never set for this macro.

A.5. ICAM CANCEL CONDITIONS

When any of the errors listed in Table A–13 occurs, the task causing the error is cancelled and the operator is notified via the system console. The appropriate error code will be displayed in the job control console message for job termination. The error code will also appear on the first line of a cancel dump printout.

Table A—13. ICAM Cancel Conditions

Error Code	Cause/Condition
430	TCI user cannot be rescheduled because ARP pool is empty.
440	DUST user has made an MOPEN call but has no CCA.
441	DUST user has made a valid DUST call but ARP pool is empty.
442	DUST user has made a DUST call other than MOPEN without previously doing an MOPEN to open the network.
443	DUST user has made an invalid overlay call.
444	DUST user has made an MOPEN but no user control slot is available (i.e., the maximum number of users has been exceeded).
450	Invalid ICAM SVC call due to one of the following: 1. Illegal subfunction code following the SVC instruction caused by not using ICAM macros or by not configuring ICAM to support the macro used. 2. Network request in progress or being released.
460	ICAM has experienced a program exception and is canceling itself and all users from the system.
461	ICAM has experienced an unrecoverable disk error and is canceling itself and all its users from the system.
480	Disk queueing has experienced an unrecoverable disk error and is canceling the user of the file in error.
481	Disk queueing is unable to secure an ARP for processing.

A.6. CUP DISPLAY/ALTER PARAMETER AND WORK AREA TABLES

Figures A–5 through A–9 describe the fields for the display/alter macroinstructions of a CUP. The information to be saved is stored in parameter tables generated by the associated macroinstructions. Table A–14 presents TRMREP work area field descriptions.

Byte	0	1	2	3	Word
0	error half word		unused	function code	1
4	CCA information table address				2
8	terminal-name list address				3
	CCA information table length		terminal-name list length		4

Byte	Field		Content	Comment	Word
	Label* Suffix	Type and Length			
0	ERR	H	Error half word	Bits set by ICAM	1
		XL1	Unused		
3	FUNC	XL1	Function code indicator	Identifies CCACPY function	
4	CAI	F	Address of CCA information table	Stored by ICAM in this user workarea.	2
8	TNT	F	Address of terminal name	Stored by ICAM in this user workarea.	3
12	LEN1	XL2	No. bytes in CCA information table		4
14	LEN2	XL2	No. bytes in terminal name list		

*All labels are prefixed TQ#C. The DSECT for this table is TQ#CCT.

Figure A—5. CCACPY Parameter List Field Descriptions

Byte	0	1	2	3	Word
0	terminal name 1				1
4	terminal name 2				2
8	terminal name 3				3
	. . . .				
n	terminal name n				n

Byte	Field		Content	Comment	Word
	Label* Suffix	Type and Length			
0	TRMI	CL4	Terminal name	Table consists of a series of these items.	1

*The label is prefixed TC#C. The DSECT for this table is TC#CCINP.

Figure A—6. CCACPY Terminal Name List Field Descriptions

Byte	0	1	2	3	Word
0	terminal name 1				1
4	line name for terminal 1				2
8	unused	flag byte (UNISCOPE)	number of terminals on line		3
12	terminal index		logical line number		4
16	device control table 1				5
	. . . . .				
n	device control table n				n

Byte	Field		Content	Comment	Word
	Label* Suffix	Type and Length			
0	TRM	CL4	Terminal name		1
4	LINE	CL4	Line name		2
8	RESV	XL1	Unused		3
	U100	XL1	Flag indicator	Indicates screen type (64/80 column)	
	NRT	H	No. of terminals on line		
12	TIDX	H	Terminal index		4
	LLN	H	Logical line number		
16	DCT	F	Device control table name	All DCTs linked to the TCT are included as part of the table entry.	5

*All labels are prefixed TC#C. The DSECT for this table is TC#CCOTP.

Figure A—7. CCA Information Table Field Descriptions

BYTE	0	1	2	3	WORD
0	work area length		number of queues		1
4	name				2
8	name of low priority queue				3
12	low priority number of messages		name of medium priority queue (1st byte)		4
16	name of medium priority queue (2nd byte)		medium priority number of messages		5
20	name of high priority queue				6
24	high priority number of messages		name of intercept queue (1st half)		7
28	name of intercept queue (2nd half)		intercept queue number of messages		8

Byte	Field		Usage			Word
	Label* Suffix	Type and Length	Process File	Queue	Terminal	
0	WAL	H	Number of bytes in WKAR	Same	Same	1
	NQ	H	Number of queues – 1, 2, or 3	1	3	
4	NAM	CL4	Name of PRCS	Name of LINE	Name of TERM	2
8	LOW	CL4	'LOW'	Name of QUEUE	Name of low priority QUEUE assigned to TERM	3
12	LCNT	H	Number of messages on low queue	Number of messages on QUEUE	Number of messages on low priority queue	4
14	MED	CL4	'MED' if NQ > 1; blank otherwise	(Not used or required)	Name of medium priority QUEUE assigned to TERM	
18	MCNT	H	Number of messages on medium queue if NQ > 1; 0 otherwise		Number of messages on medium priority queue	5
20	HIGH	CL4	'HIGH' if NQ = 3; blank otherwise		Name of high priority queue assigned to TERM	6
24	HCNT	H	Number of messages on high queue if NQ = 3; 0 otherwise		Number of messages on high priority queue	7
	INTR	CL4	(Not used or required)		Name of intercept queue, if any (QInn or blank)	
30	ICNT	H			Number of messages on intercept queue, or 0	8

*All labels are prefixed TQ#Q. The DSECT for this table is TQ#QDWA and must be aligned on a half-word boundary.

Figure A—8. QDEPTH Work Area Field Descriptions

BYTE	0	1	2	3	WORD
0	error code		flag byte	function code	1
4	process file, line, or terminal name				2
8	work area length	work area address			3
12	queue name *				4

*Used only when positional parameter 1 of QDEPTH, QHOLD, QRELSE, or QCLEAR is Q.

Byte	Field		Content	Comment	Word
	Label* Suffix	Type and Length			
0	ERR	H	Error flags	Set by ICAM	1
2	FLAG	YL1	Type indicator	Differentiates P, L, T and Q types	
3	FUNC	YL1	Function indicator	Indicates which macroinstruction was coded	
4	PLTN	CL4	Process file, line or terminal name		2
8	WALG	YL1	Work area length in bytes	Used only by QDEPTH	3
9	WORK	A3	Work area address	Used only by QDEPTH	
12	NAME	CL4	Queue name		4

*All labels are prefixed: TQ#Q. The DSECT name is TQ#QDSCT.

**For QTRANS functions, TQ#QFN1 and TQ#QFN2 are equates for TQ#QPLTN and TQ#QWALG corresponding to the two file names.

Figure A–9. QDEPTH/QHOLD/QRELSE/QCLEAR Parameter List Field Descriptions

Table A—14. TRMREP Work Area Detailed Field Descriptions

Field		Content			Comments
Label * Suffix	Type and Length				
RESV	H	(Unused)			Reserved for further system use
CNTL	XL2	Bit flags denoting fields to be copied or replaced			Set by ICAM according to the FIELDS subparametres; or if FIELDS omitted, set by user program
Byte 0		Bit	Field	TRMREP Subparam	
		KQUH	QUEH	QUEUEH/QUEUES	High queue name
		KQUM	QUEM	QUEUEM/QUEUES	Medium queue name
		KQUL	QUEL	QUEUEL/QUEUES	Low queue name
		KALT	ALTD	ALTD	Alternate destination name
		KADD	ADDR	ADDR	Addressing RID, SID
		KFRQ	FREQ	FREQ	Polling frequency
		KPLM	PLIM	PLIM	Polling limit
Byte 1		KPAD	PADR	POLL	Polling RID, SID, DID
		KRSP	RSLN, RESP	RESP	Response or answer-back
QUEH	CL4	Name of high priority queue			First QUEUES subparameter in TERM definition
QUEM	CL4	Name of medium priority queue			Second QUEUES subparameter in TERM definition
QUEL	CL4	Name of low priority queue			Third QUEUES subparameter in TERM definition
ADDR	XL2	RID, SID for device addressing			ADDR parameter in TERM definition
FREQ	XL2	Minimum time, in ms, between polling passes			PINTV parameter in TERM definition
PLIM	XL2	Limit on the number of input messages to be accepted from the terminal in a single polling pass			PLIMIT parameter in TERM definition
RSLN	XL1	Length of response or answer-back			First ANSWER subparameter in TERM definition
PADR	XL3				
	XL1	(Unused)			Reserved for future use
RESP	XL1	First byte of response or answer-back if provided			

*Labels are prefixed TQ#W. The DSECT name is TQ#TCWRK.

A.7. USER TRANSLATE TABLES

A.7.1. Specifying User Translate Tables

When specifying user translate tables, you specify the XLATE keyword parameter with its subparameters labeli, labelo, idi, and ido. You can use the XLATE keyword parameter associated with either the LINE or TERM macro call. When the XLATE keyword parameter on the TERM macro is used, the line information for that terminal is overridden. The LINE macro is discussed in 2.2.7, and the TERM macro is discussed in 2.2.10.

Each unique input and output translate table should be assigned a unique value. These values are stored separately, so that duplication between input and output values is permissible.

During CCA generation, a table of translate table addresses is built. This table contains the addresses of any translate table needed to support the CCA. The addresses of the standard translate tables appear first, followed by the user-specified tables. Each time a user-specified translate table is detected, its address (labeli/labelo) is placed in the appropriate slot. The index to this table is the idi/ido subparameter that has a decimal value ranging from 1 to 240.

A.7.2. Examples of User Translate Tables

Some typical examples of user translate tables are presented in this discussion.

In the following examples, assume that the input modification value is m and the output modification value is n:

```
     1        10    16
    ---------------------------------
1. |         XLATE=(INPUTA,1,OUTPUTA,1)
2. |         XLATE=(INPUTA,1,OUTPUTB,2)
3. |         XLATE=(INPUTC,3,OUTPUTC,3)
4. |         XLATE=(INPUTD,4,OUTPUTD,3)
```

1. The address of INPUTA, A(INPUTA) is stored into the table at the base plus (1 + m) x 4; A(OUTPUTA) at the base plus (1 + n) x 4.

2. The address of INPUTA, A(INPUTA) is the same as in example 1; however, the address of A(OUTPUTB) is at the base plus (2 + n) x 4.

3. The address of INPUTC, A(INPUTC) is stored into the table at the base plus (3 + m) x 4; A(OUTPUTC) is stored at the base plus (3 + m) x 4.

4. The address of INPUTD, A(INPUTD) is stored into the table at base plus (4 + m) x 4; the address of A(OUTPUTD) is stored at the base plus (3 + n) x 4. Notice that this overlays OUTPUTC from example 3.

Figure A-10 shows the address of the translate tables if they were processed.

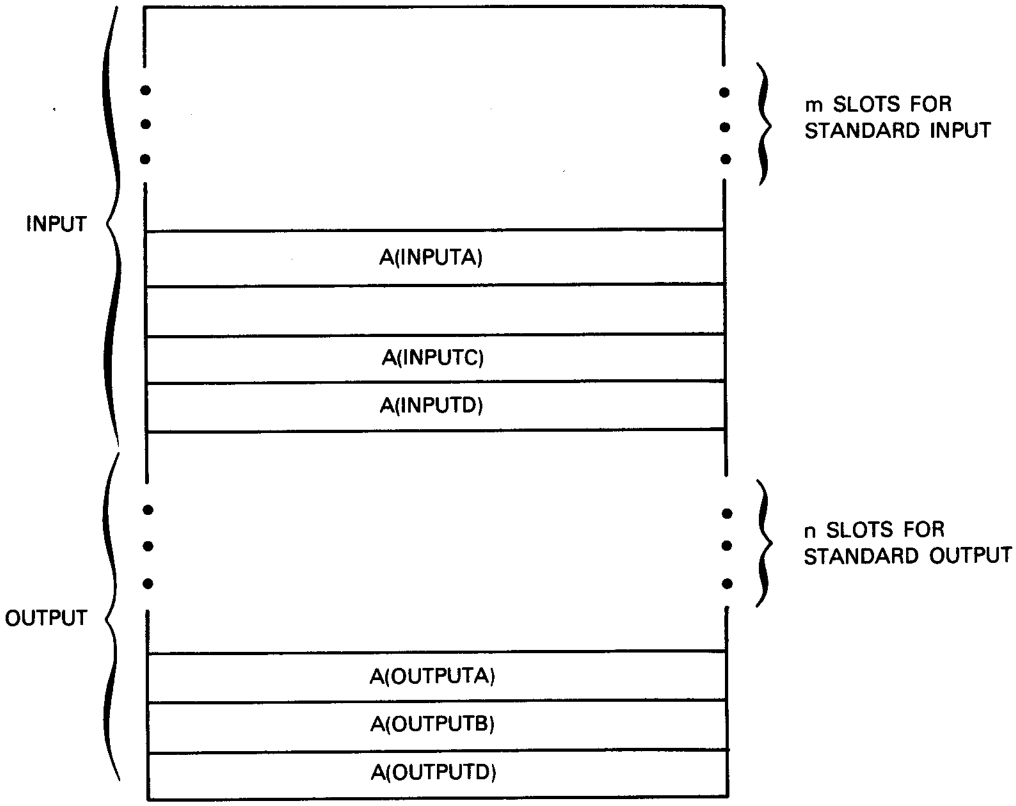


Figure A—10. Typical User Translate Table

Looking at Figure A-10, notice that there is an empty slot in the input portion. This resulted because an idi of 2 was not specified. To conserve main storage, you should specify consecutive numbers for both input and output. Also, notice that there is no A(OUTPUTC) because an ido of 3 was also specified for OUTPUTD.

As mentioned earlier, you have the option of using the XLATE parameter associated with either the LINE or TERM macro call. The following examples show the differences between the two macro calls.

```
     1        10    16
1.   L1       LINE  XLATE=(INPUTA,1,OUTPUTA,1)
     T1       TERM  XLATE=(,OUTPUTB,2)
     T2       TERM  XLATE=(INPUTB,2)
2.   L2       LINE  XLATE=(NO)
     T3       TERM  XLATE=(YES,YES)
     T4       TERM
3.   L3       LINE  XLATE=(,YES)
     T5       TERM  XLATE=(OUTPUTC,3)
     T6       TERM  XLATE=(INPUTC,3,NO)
```

1. L1 indicates that two user-supplied translate tables are specified (INPUTA and OUTPUTA). For T1, the input translation is INPUTA, which is the default from L1, while the output translation is OUTPUTB. For T2, the input translation is INPUTB, while the output translation is OUTPUTA, which is the default from L1.

2. There is no input translation for L2, and the standard translation is used on output. For T3, both the input and output translations are standard. For T4, there is no input translation for input, which was defaulted from L2; the output translation is standard, which was also defaulted from L2.

3. The standard translation tables are used for both input and output. For T5, the standard translation table is used for input, which was defaulted from L3, while the output translation table is OUTPUTC. For T6, the input translation table is INPUTC, and there is no output translation.

A.7.3. CCA Generation of User Translate Tables

All the user-supplied translate tables must be generated within the CCA. Therefore, when you are doing your ICAM SYSGEN, the user translate tables must be included before the ENDCCA card in the COMMCT section. Figure A—11 is an example of a CCA using translate table substitution; only the parameters pertaining to this function are supplied.

```
1        10     16                                                     72
---------------------------------------------------------------------------
COMMCT
NET1     CCA    params
         BUFFERS params
L1       LINE   params,XLATE=(INPUTA,1,OUTPUTA,1),
T1       TERM   params,XLATE=(,OUTPUTB,2),
T2       TERM   params,XLATE=(INPUTB,2),
L2       LINE   params,XLATE=(NO),
T3       TERM   params,XLATE=(YES,YES),
T4       TERM
L3       LINE   params,XLATE=(1,YES),
T5       TERM   params,XLATE=(OUTPUTC,3)
T6       TERM   params,XLATE=(INPUTC,3,NO),
INPUTA   DC     XL16'
          .     translate table entries
          .     normally 256 bytes
          .
         DC     XL16'
INPUTB   DC     XL16'
          .     translate table entries
          .
          .
         DC     XL16'
```

Figure A—11. Example of a Translate Table Substitution (Part 1 of 2)

```
1           10     16
------------------------------------------------------------
INPUTC      DC     XL16'                                    '
             :     translate table entries
             :
             :
            DC     XL16'                                    '
OUTPUTA     DC     XL16'                                    '
             :     translate table entries
             :
             :
            DC     XL16'                                    '
OUTPUTB     DC     XL16'                                    '
             :
             :
             :
            DC     XL16'                                    '
OUTPUTC     DC     XL16'                                    '
             :
             :
             :
            DC     XL16'                                   '
            ENDCCA
            MCP
            MCPNAME= params
            CACH=(params)
             :
             :
             :
            CACH=(params)
END
// FIN
```

Figure A—11. Example of a Translate Table Substitution (Part 2 of 2)

If the network in Figure A—11 were generated, the following index values would appear in the terminal tables:

- T1

  Input: 1 + m (m = number of standard input tables)

  Output: 1 + n (n = number of standard output tables)

- T2

  Input: 2 + m (m= number of standard input tables)

  Output: 1 + n (n = number of standard output tables)

- T3

  Input: standard input value for particular line type

  Output: standard output value for particular line type

- T4

  Input: standard input value for no input translation

  Output: standard output value for particular line type

- T5

  Input: standard input value for particular line type

  Output: 3 + n (n = number of standard output tables)

- T6

  Input: 3 + m (m = number of standard input tables)

  Output: standard value for no output translation

Appendix B. Journaling

B.1. GENERAL

The journal utility (JUST) is an optional utility that retrieves message text and statistical data. This data was previously written to the history file via MPPS. The journal utility is executed as an offline batch routine either to print this statistical data in an edited form or write it to disk.

The records processed by the journal utility are:

- JOURN records containing message text and related control data such as message waiting, message transmitted, or function key signal received.
- ODNR records indicating the successful delivery of the message.
- PERF records containing line and terminal performance data.
- STATS records containing buffer utilization data
- LOG records containing only ICAM journal header data.

To use journaling, you must generate MPPS at network generation via the LINE macroinstruction.

In addition to the journal utility retrieving, printing, and writing functions, the journal utility and ICAM, collectively, provide the user with a cold-restart function. During communications processing, ICAM can write restart records to the history file. If during processing, hardware failure occurs, you can execute the journal utility to reconstruct the ICAM message queues using the cold restart records. Following the reconstruction, you can request a warm restart of ICAM at network request time via the RESTART parameter in the NETREQ macroinstruction.

B.2. NETWORK DEFINITION FOR JOURNALING

The following macroinstructions are used to define journaling:

LABEL	△OPERATION△	OPERAND

cca-name	CCA	TYPE=(STDMCP),
		JRNINIT=(x[,RESTART]
		[,PERF][,STAT]
		[,LOG][,JOURN][,ODNR])

file-name	JRNFILE	TYPE= SAT, TAPE ,BUFF=(number,size,thresh) DISC
line-name	LINE	[,STATS=YES]
	BUFFERS	[,STAT=YES]
[symbol]	journ	filename

When specifying the JRNINIT parameter, x specifies the integer (1, 2, 3) determining the types of records collected. RESTART allows restart records to be written, PERF requests the writing of line and terminal performance data, and STAT requests the writing of buffer utilization data. JOURN requests the writing of message text and certain control information, ODNR allows for output delivery notification to be written, and LOG allows only journal buffer data to be written. All records are written to the history file.

The TYPE parameter must be STDMCP while the other option parameters may be included as shown in Section 2.

The JRNFILE macroinstruction defines the history file where ICAM writes the designated records. The following coding shows how you might define a network for system generation to include journaling:

```
1        10    16                                                      72

COMMCT
CT33     CCA   TYPE=(STDMCP),PASSWORD=CTCOM33,                         X
               CCAID=CTCOM33,                                          X
               JRNINIT=(3,PERF,STAT,JOURN,ODNR),                       X
               FEATURES=(TOPR1,TRACEMAX,OPCON,SEGMENTS)
         BUFFERS 1Ø,64,1,ARP=36,STAT=YES
LNØ1     LINE  DEVICE=(U1ØØ),                                          X
               TYPE=(2ØØØ,SWCH,SYNC,CONNECT),                          X
               LBL=32,                                                 X
               MPPS=(MP33,2Ø),                                         X
               STATS=YES
TRM1     TERM  ADDR=(29,54),                                           X
               FEATURES=(U1ØØ,96Ø),                                    X
               LOW=MAIN,                                               X
               ALTD=TRM3,                                              X
               INTERCPT=YES
TRM2     TERM  ADDR=(29,53),                                           X
               FEATURES=(U1ØØ,96Ø),                                    X
               LOW=MAIN,                                               X
               ALTD=TRM3,                                              X
               INTERCPT=YES
                .
                .
                .
```

```
1        10    16                                                    72
______________________________________________________________________
PRC1     PRCS  LOW=MAIN,MEDIUM=MAIN,HIGH=MAIN
JRN1     JRNFILE TYPE=(SAT,TAPE)
DLT1     DLIST TRM1,TRM2,TRM3,TRM4
DLT2     DLIST DLT1,TRM5,TRM6,TRM7,TRM8
MP33     MPSTART
  .   )
  .   }  MPSTART
  .   )     .
            .
            .

         RECSEG
         JOURN JRN1
         RECHDR
         ADVANCE,1,*
            .
            .
  .         .
  .
  .       * GO

            .
            .
            .
TIGO        MSGTYP 2,GO,TIDL
            DIRECT P,PRC1,H
            BRANCH TIRECEND
  .
  .
  .
```

In this example, you should note the following items:

- The JRNINIT parameter in the CCA macroinstruction indicates the record types to be collected; namely, performance, statistics, journal, and output delivery notification records.
- The STAT parameter in the CCA macroinstruction and the STAT=YES parameter in the BUFFERS macroinstruction requests the writing of buffer utilization data.
- The PERF parameter in the CCA macroinstruction and the STATS parameter in the LINE macroinstruction requests the writing of line activity data.
- The JRNFILE macroinstruction indicates the name of the history file (JRN1) and the file type (SAT, tape file) to be created.
- The JOURN macroinstruction in the MPPS is required for the history file to be created. The name must be identical to that of the JRNFILE macroinstruction.

B.3. REPORTING FACILITY

The journal utility reporting facility can produce an edited display, a summary report, a statistical report, or a combination of all the reports. Table B-1 lists the control statement and the function performed.

Table B—1. Journal Utility Control Statements

Control Statement	Function
BSTAT	Produces a report showing the network and ARP buffer utilization using STAT records.
SELECT	Lists all history file records (except restart); also produces a report similar to BSTAT and a report similar to SUM; or any combination of these.
SUM	Produces a report showing line and terminal usage within ICAM using PERF records.
USER	Permits a user-written data reduction routine working with the journal utility to produce a unique report from the history file.

B.3.1. Control Statements

The control statements for the journal utility take the same format as any other system utility control statements, namely;

```
△OPERATION△OPERAND
```

However, you should note the following items:

- Control statement labels are not permitted and the operation must be preceded by a space, and followed by at least one space.

- The maximum number of characters for a control statement is 255, including commas, parentheses, and dashes.

- If a control statement contains multiple operands, they are connected by implied AND operators; therefore, all conditions must be met before a record is selected. For example,

  ```
  △SELECT△JEC=JOURN,DATE=78001
  ```

  This control statement causes the printed report to contain journal header and continuation records with the date Jan 1 1978. All other records in the history file are ignored.

- When more than one report is required during an execution, an additional disk work file must be allocated via the // WORK1 jproc.

- Normally, 7000_{16} bytes should be specified in the JOB control statement.

B.3.1.1. Buffer Statement (BSTAT)

Function:

Generates a 2-page summary report listing, that lists network buffer utilization data, and ARP network buffer utilization data at network release time.

Format:

△OPERATION△	OPERAND
BSTAT	TIME=NET

Parameters:

TIME=NET
Specifies that the journal utility produce a report from the buffer statistics data that ICAM wrote to the history file at network release.

Example:

```
1         10    16
BSTAT    TIME=NET
```

NOTE:

A typical BSTAT report is shown in Figure B—2.

SELECT

B.3.1.2. SELECT Statement

Function:

Output all records of the history file subset defined by its operand to either a printer or SAT file. For printer output, the records are edited and the following three reports are produced:

- Journal message and output delivery notification
- Buffer utilization summary
- Line/terminal performance summary

The journal message and delivery notification report lists the journal header (JOURN) and continuation records, and the output delivery notification records (ODNR). The journal header records contain the first portion of the message (or the entire message if small enough) and associated control data. The journal continuation records contain the remaining part of the message. Although most journal and output delivery notification record fields are self-explanatory, you should note the following:

- SEQ-NO indicates the message sequence indicator (MSI). Both journal header and continuation records have the same number. Also, the sequence number of an output delivery notification record refers to a previously printed journal record.
- DATE and TIME indicate the date and time that the records were written to the history file.
- COMPLETE MESSAGE indicates the entire message is contained in the journal header record.
- INITIAL SEGMENT indicates the remaining portion of the message is contained in one or more continuation records (same MSI).
- TEXT BYTES THIS FINAL SEGMENT indicates this continuation record contains the final message segment.
- TEXT IS indicates the data following is the message text; all nonprintable characters appear as blanks in the listing.
- ODNR indicates the output delivery notification record.

The statistic report is produced from the STATS records listing the network and ARP buffer pool data (similar to BSTAT report). The performance report is produced from the PERF record and lists line activity data for each terminal in the network (similar to SUM).

Format:

```
△OPERATION△    OPERAND

SELECT              [FILE=file-name]
                    ⎡,JEC=⎧JEC-code                                   ⎫⎤
                    ⎣     ⎩(JEC-code1,JEC-code2,...JEC-coden)         ⎭⎦
                    ⎡,MSI=⎧MSI-number                  ⎫⎤
                    ⎣     ⎩MSI-number1-MSI-number2      ⎭⎦
                    ⎡,DATE=⎧yyddd          ⎫⎤
                    ⎣      ⎩yyddd1-yyddd2   ⎭⎦
                    ⎡,TIME=⎧hhmmss            ⎫⎤
                    ⎣      ⎩hhmmss1-hhmmss2    ⎭⎦
                    ⎡,TERMD=⎧t-d-name                                   ⎫⎤
                    ⎣       ⎩(t-d-name1,t-d-name2,...t-d-namen)         ⎭⎦
                    ⎡,TERMS=⎧t-s-name                                   ⎫⎤
                    ⎣       ⎩(t-s-name1,t-s-name2,...t-s-namen)         ⎭⎦
```

Parameters:

FILE

Designates the LFD name of a SAT file. This permsits the history file records (except RESTART) to be transcribed unedited to the designated file. This file can then be used as a primary input file by the journal utility at a later date.

JEC

Designates the types of records to be printed in either the report or transcribed into a disk file. Unspecified record types found in the history file are ignored. However, when the JEC parameter is omitted, all record types in the history file are selected except the RESTART records. The value may be one or a list of the following:

JOURN	Journal header and journal continuation records
ODNR	Output delivery notification report records
STATS	Buffer statistical records
PERF	Line and terminal performance records
LOG	Message header segments

MSI

Indicates only records containing the designated message sequence indicator are to be retrieved. The indicator is a 4-byte binary value and can be stated as either a decimal or hexadecimal number. Each record type has a separate set of sequence numbers starting with 1.

DATE

Indicates only records containing the designated date are to be retrieved. The date is yyddd where:

yy
: Is the year, with decimal values from 00–99.

ddd
: Is the day of the year, with decimal values from 001–366.

TIME
: Indicates only records containing the designated time are to be retrieved. The time is hhmmss,

where:

hh
: Hours in the decimal range 00–23.

mm
: Minutes in the decimal range 00–59.

ss
: Seconds in the decimal range 00–59.

TERMD/TERMS
: Indicates only records containing the designated destination and source terminal names are to be retrieved. The value is specified as a single 4-character name or series of names. This parameter should only be specified in conjunction with JEC=JOURN.

NOTE:

If you omit all the parameters, all history file records except the restart record are edited and printed.

```
1       10    16
________________
 SELECT JEC=PERF
 SELECT
```

NOTE:

A typical SELECT report is shown in Figure B—6.

B.3.1.3. SUM Statement

Function:

Generates a message traffic summary report from the performance (PERF) record.

Format:

△OPERATION△	OPERAND
SUM	[,TERMD={t-d-name (t-d-name$_1$,t-d-name$_2$,...t-d-name$_n$)}] [,TERMS={t-s-name (t-s-name$_1$,t-s-name$_2$,...t-s-name$_n$)}]

Parameters:

TERMD/TERMS
Indicates only the data from the designated terminal names are printed. The value is specified as a single 4-character name or series of names. (TERMS and TERMD are equivalent)

NOTE:

If the parameters are omitted, the entire report is produced.

Example:

```
1                10     16
__________________________
 SUM
```

NOTE:

A typical SUM report is shown in Figure B—4.

B.3.1.4. USER Statement

Function:

Generates a linkage to a user-defined data reduction routine. Through this linkage, the journal utility passes the specified history file records to the user's data reduction routine.

Format:

```
△OPERATION△      OPERAND
USER             LOAD=mod-name
                 [,DATE={yyddd              }]
                 [      {yyddd1-yyddd2      }]
                 [,TIME={hhmmss               }]
                 [      {hhmmss1-hhmmss2      }]
                 [,JEC={JEC-code                                }]
                 [     {(JEC-code1,JEC-code2,...JEC-coden)      }]
                 [,TERMD={t-d-name                                }]
                 [       {(t-d-name1,t-d-name2,...t-d-namen)      }]
                 [,TERMS={t-s-name                                }]
                 [       {(t-s-name1,t-s-name2,...t-s-namen)      }]
```

Parameters:

LOADM

Defines a user load module by its name. This keyword permits a single value only. The value is the name of a load module created by the user.

DATE

Indicates only records containing the designated date are to be retrieved. The date is yyddd,

where:

yy

Is the year, with decimal values from 00-99.

ddd

Is the day of the year, with decimal values from 001-366.

TIME

Indicates only records containing the designated time are to be retrieved. The time is hhmmss,

where:

hh

Hours in the decimal range 00–23.

mm

Minutes in the decimal range 00–59.

ss

Seconds in the decimal range 00–59.

JEC

Designates the types of records to be passed to the user-defined data-reduction routine. Unspecified record types found in the history file are ignored. However, when the JEC parameter is omitted, all record types in the history file are selected except the restart records. The value may be one or a list of the following:

JOURN	Journal header and journal continuation records
ODNR	Output delivery notification report records
STATS	Buffer statistical records
PERF	Line and terminal performance records
LOG	Message header segments

TERMD/TERMS

Indicates only records containing the designated destination and source terminal names are to be retrieved. The value is specified as a single 4-character name or series of names. This parameter should only be specified in conjunction with JEC=JOURN.

Example:

```
1        10    16                                                72
_________________
 USER  LOADM=ULOD
```

NOTE:

The required coding to be incorporated in the user's data reduction routine for generating the linkage to the journal utility is presented in Figure B—7.

B.4. EXECUTING THE JOURNAL UTILITY (JUST)

The following coding example shows a typical job control stream executing the journal utility. The input file (history file created by ICAM and MPPS) must be defined as INPUT1 in the LFD job control statement. A printer file must be defined with the filename of PRNTR. In addition, if multiple journal utility statements (BSTAT, SUM, SELECT, USER) are used, a work file must be defined via the // WORK1 jproc.

```
 1. | // JOB SELECT,,,7000
 2. | // DVC 20 // LFD PRNTR
 3. | // DVC 100  // VOL S10309 // LFD INPUT1
 4. | // OPTION JOBDUMP
 5. | // EXEC JUST
 6. | /$
 7. |  SELECT
 8. | /*
 9. | /&
10. | // FIN
```

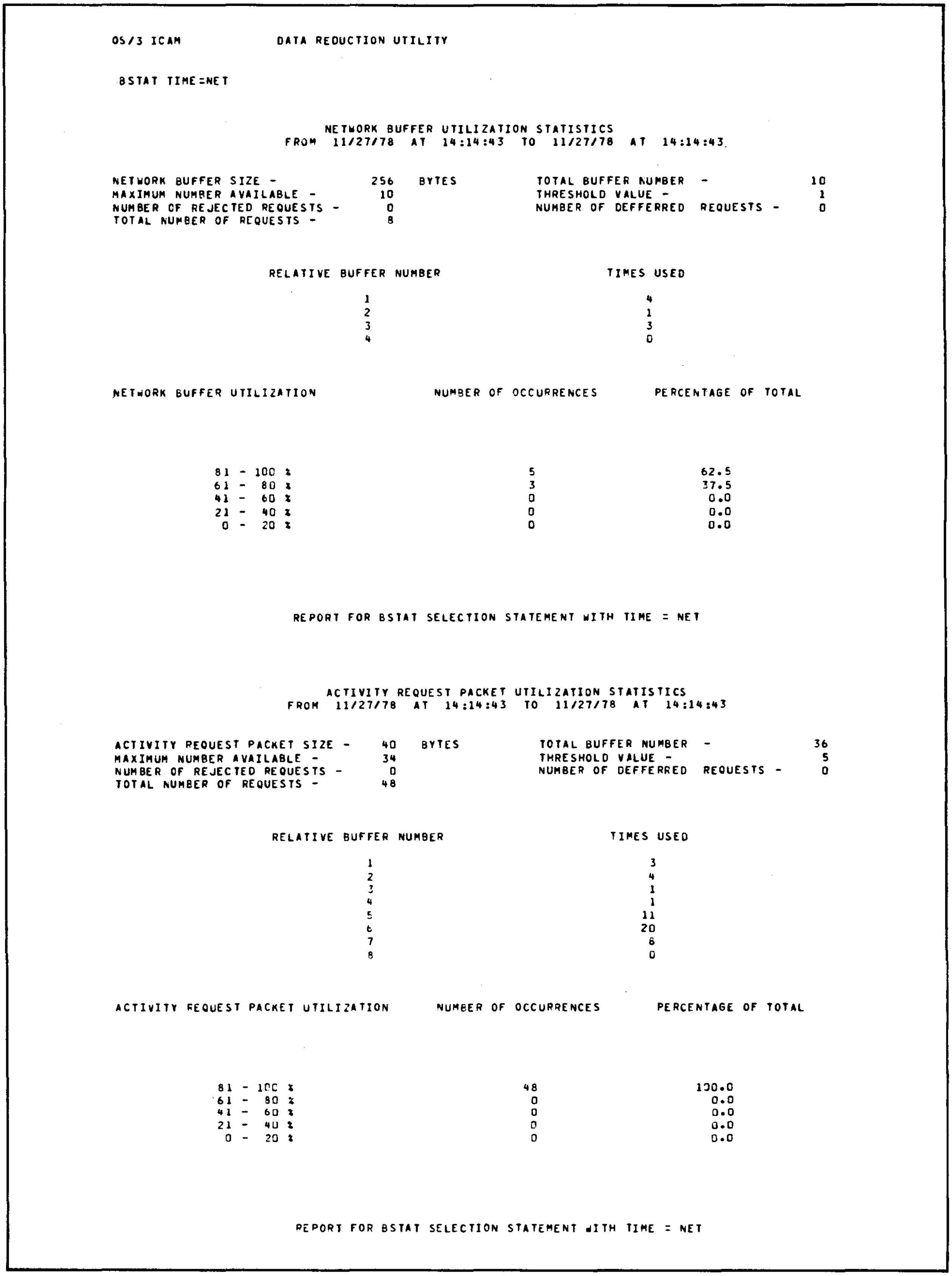
1. Main storage size specification on JOB card
2. System printer file specification
3. History file specification
4. Desire dump upon abnormal termination
5. Execution of the journal utility
6. Start of journal utility statements
7. SELECT journal utility statement
8. End of journal utility statements
9. End of job
10. End of card input

B.5. TYPICAL JOURNAL UTILITY REPORTS

Figure B-1 shows the job control stream to execute the journal utility's BSTAT statement; Figure B-2 is the printout. Figure B-3 is the job control stream to execute the journal utility's SUM statement; Figure B-4 is the printout. Figure B-5 is the job control stream to execute the journal utility's SELECT statement and Figure B-6 is the printout.

```
// JOB BSTAT,,,7000
// DVC 20 // LFD PRNTR
// DVC 100  // VOL S10309 // LFD INPUT1
// EXEC JUST
/$
 BSAT TIME=NET
/*
/&
// FIN
```

Figure B—1. Typical BSTAT Job Control Stream



```
OS/3 ICAM            DATA REDUCTION UTILITY

 BSTAT TIME=NET


                        NETWORK BUFFER UTILIZATION STATISTICS
                  FROM  11/27/78  AT  14:14:43  TO  11/27/78  AT  14:14:43


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 -         8


                 RELATIVE BUFFER NUMBER                    TIMES USED

                            1                                   4
                            2                                   1
                            3                                   3
                            4                                   0


NETWORK BUFFER UTILIZATION               NUMBER OF OCCURRENCES       PERCENTAGE OF TOTAL




                 81 - 100 %                         5                      62.5
                 61 -  80 %                         3                      37.5
                 41 -  60 %                         0                       0.0
                 21 -  40 %                         0                       0.0
                  0 -  20 %                         0                       0.0




                    REPORT FOR BSTAT SELECTION STATEMENT WITH TIME = NET




                       ACTIVITY REQUEST PACKET UTILIZATION STATISTICS
                  FROM  11/27/78  AT  14:14:43  TO  11/27/78  AT  14:14:43


ACTIVITY REQUEST PACKET SIZE -    40   BYTES         TOTAL BUFFER NUMBER  -          36
MAXIMUM NUMBER AVAILABLE -        34                 THRESHOLD VALUE -                5
NUMBER OF REJECTED REQUESTS -      0                 NUMBER OF DEFFERRED  REQUESTS -  0
TOTAL NUMBER OF REQUESTS -        48


                 RELATIVE BUFFER NUMBER                    TIMES USED

                            1                                   3
                            2                                   4
                            3                                   1
                            4                                   1
                            5                                  11
                            6                                  20
                            7                                   8
                            8                                   0


ACTIVITY REQUEST PACKET UTILIZATION      NUMBER OF OCCURRENCES       PERCENTAGE OF TOTAL




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




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

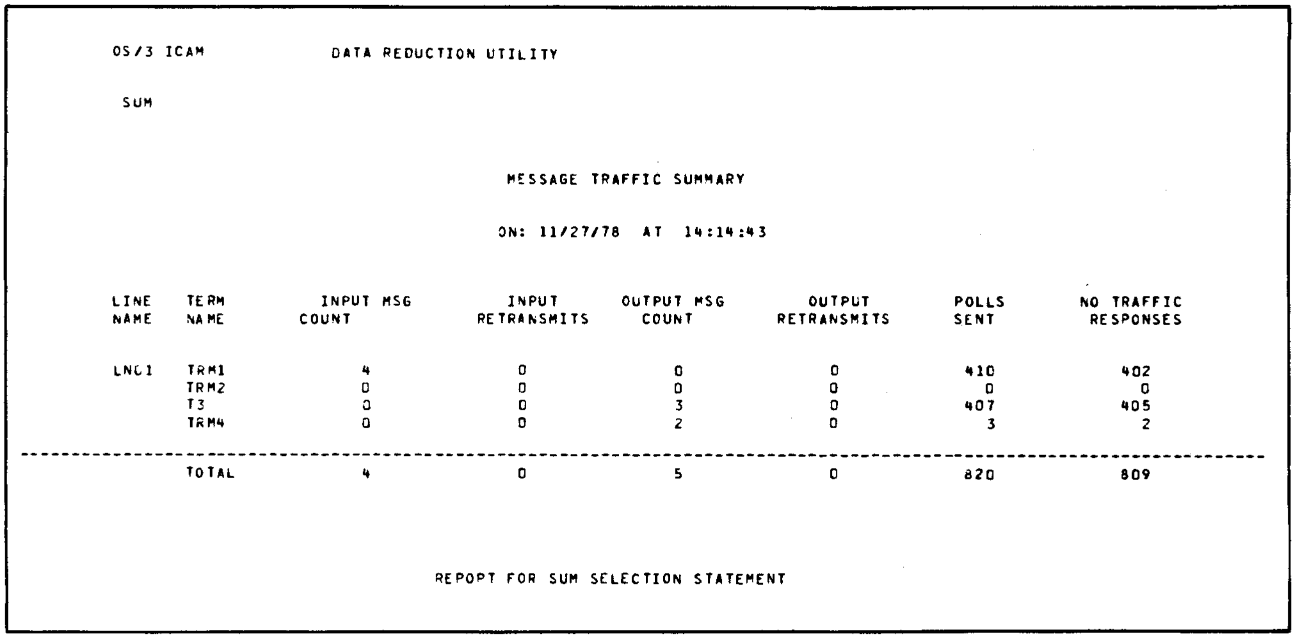
Figure B—2. BSTAT Report

NOTE:

The Relative Buffer Number and the Times Used items in Figure B—2 describe the utilization of the buffer pool from NETREQ to NETREL. The user can analyze this information to determine if fewer or more buffers and ARPs are required.

```
// JOB SUM,,,7ØØØ
// DVC 2Ø // LFD PRNTR
// DVC 1ØØ // VOL S1Ø3Ø9 // LFD INPUT1
// EXEC JUST
/$
 SUM
/*
/$
// FIN
```

Figure B—3. Typical SUM Job Control Stream



```
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


                               REPOPT FOR SUM SELECTION STATEMENT
```

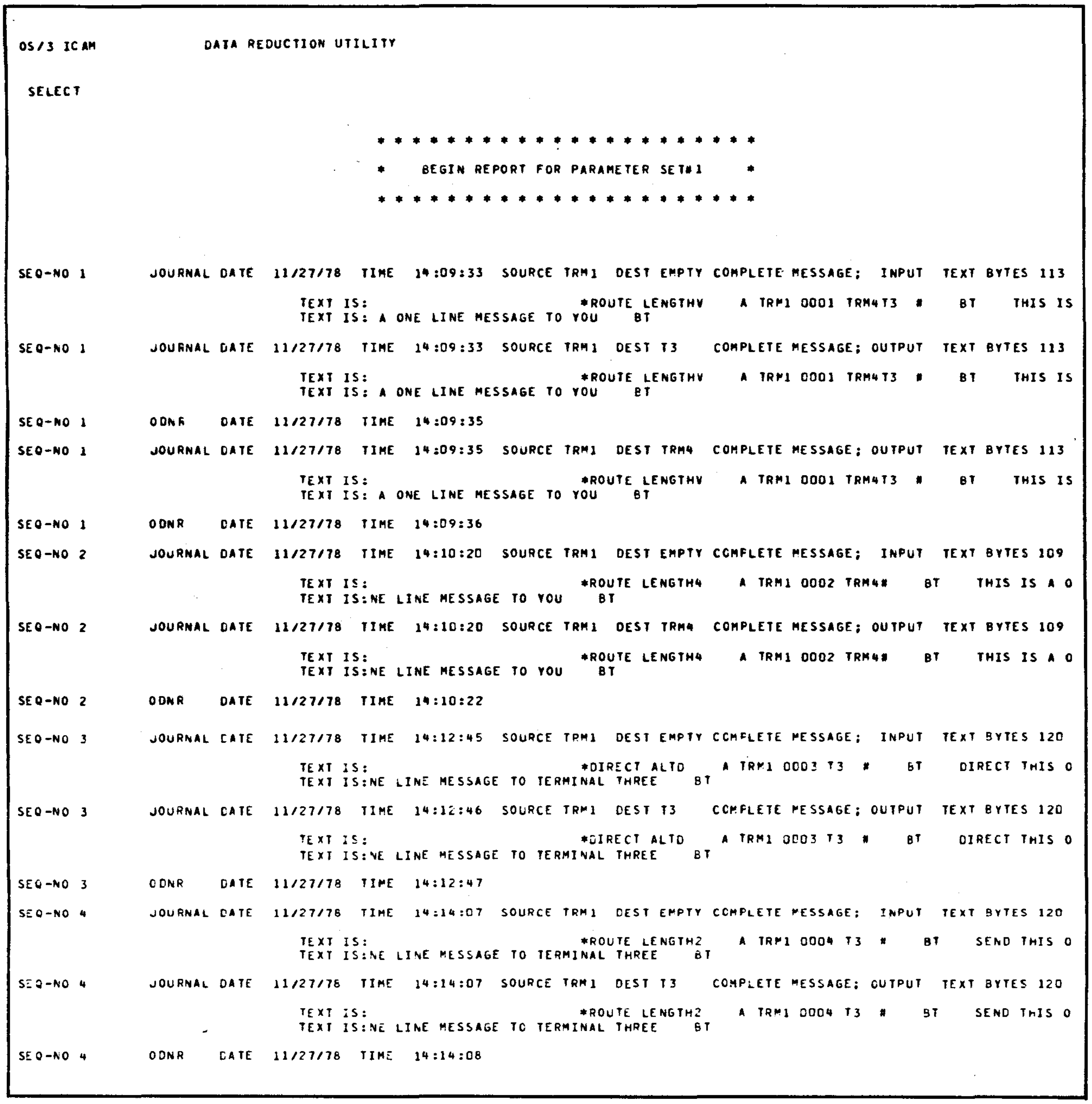
Figure B—4. SUM Report

NOTE:

Any error messages encountered during the communication processing are only counted in the POLLS SENT field (Figure B—4). Therefore, cross-footing is not possible.

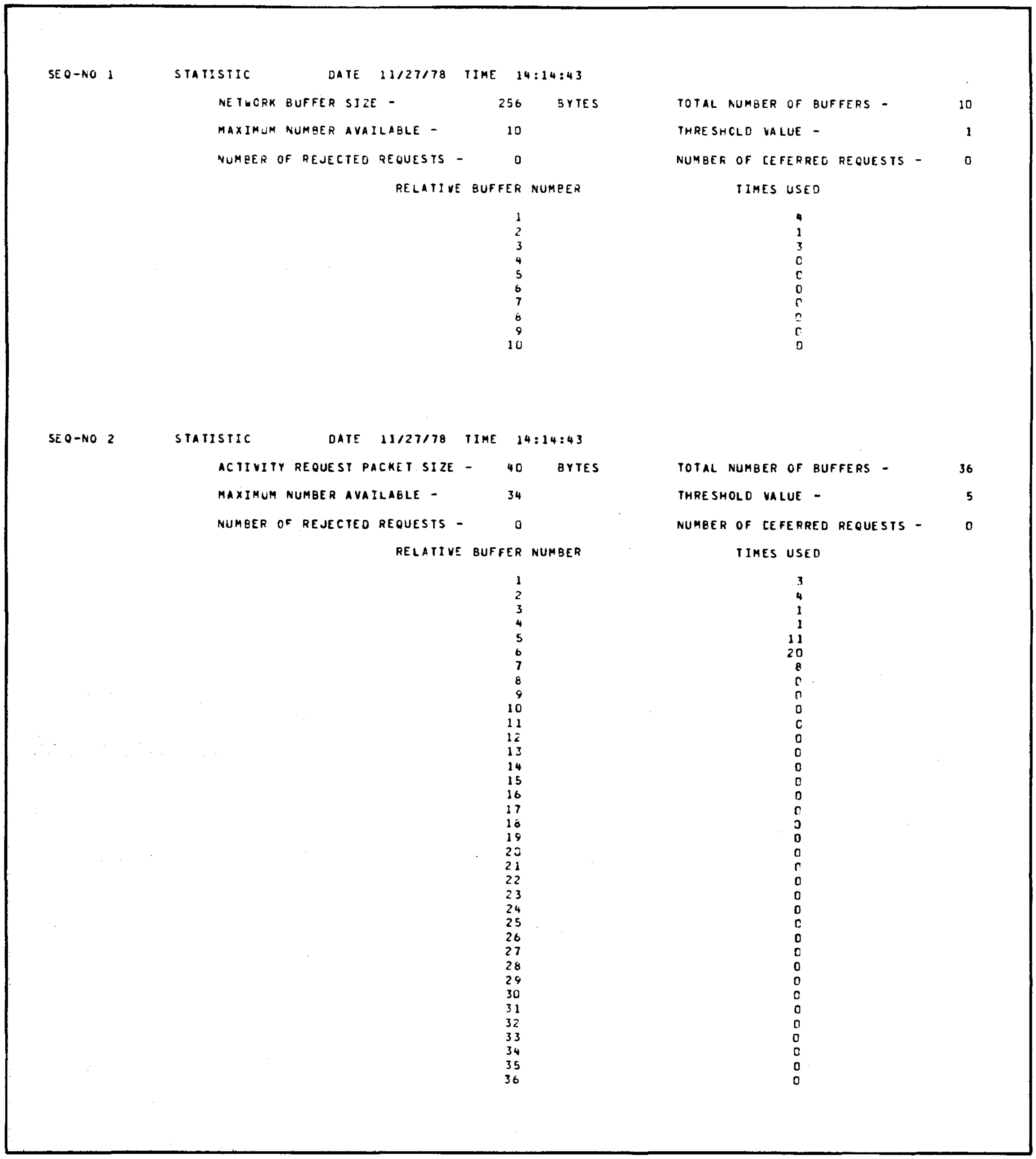
```
// JOB SELECT,,,7ØØØ
// DVC 2Ø // LFD PRNTR
// DVC 1ØØ  // VOL S1Ø3Ø9 // LFD INPUT1
// OPTION JOBDUMP
// EXEC JUST
/$
 SELECT
/*
// FIN
```

Figure B—5. Typical SELECT Job Control Stream



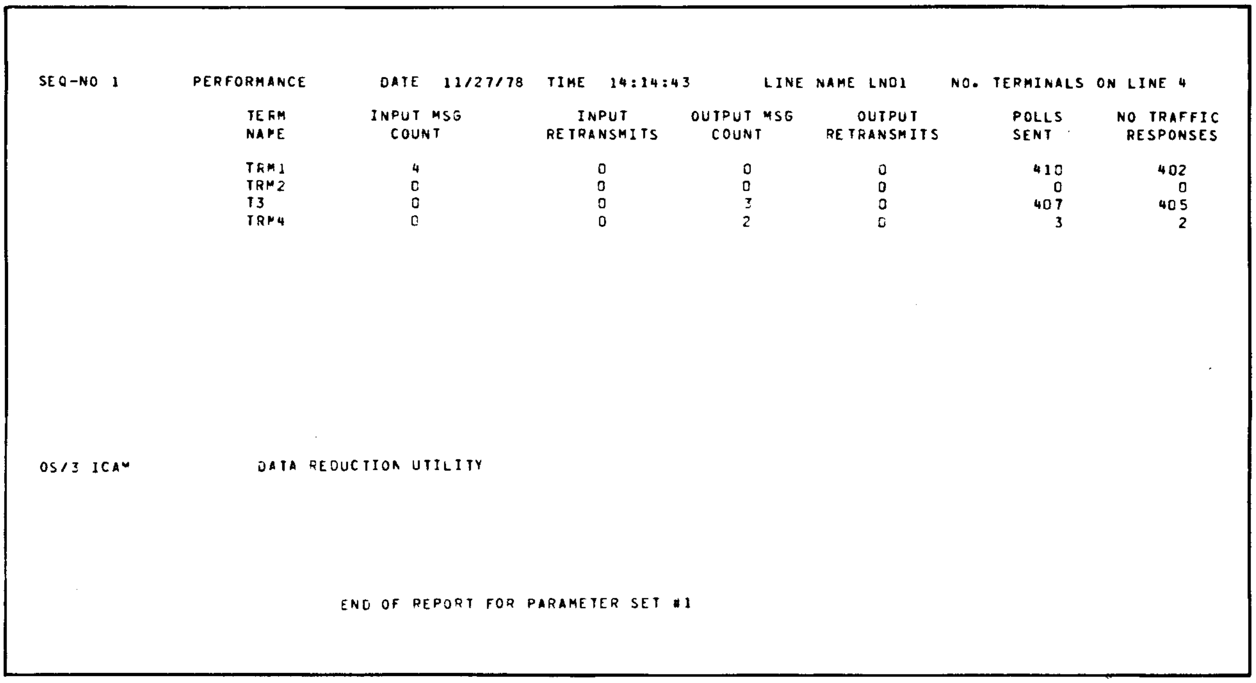
```
OS/3 ICAM           DATA REDUCTION UTILITY

 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 ALTD    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 ALTD    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 B—6. SELECT Report (Part 1 of 3)



```
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
```

Figure B—6. SELECT Report (Part 2 of 3)



```
SEQ-NO 1        PERFORMANCE       DATE  11/27/78  TIME  14:14:43       LINE NAME LNO1    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 B—6. SELECT Report (Part 3 of 3)

NOTES:

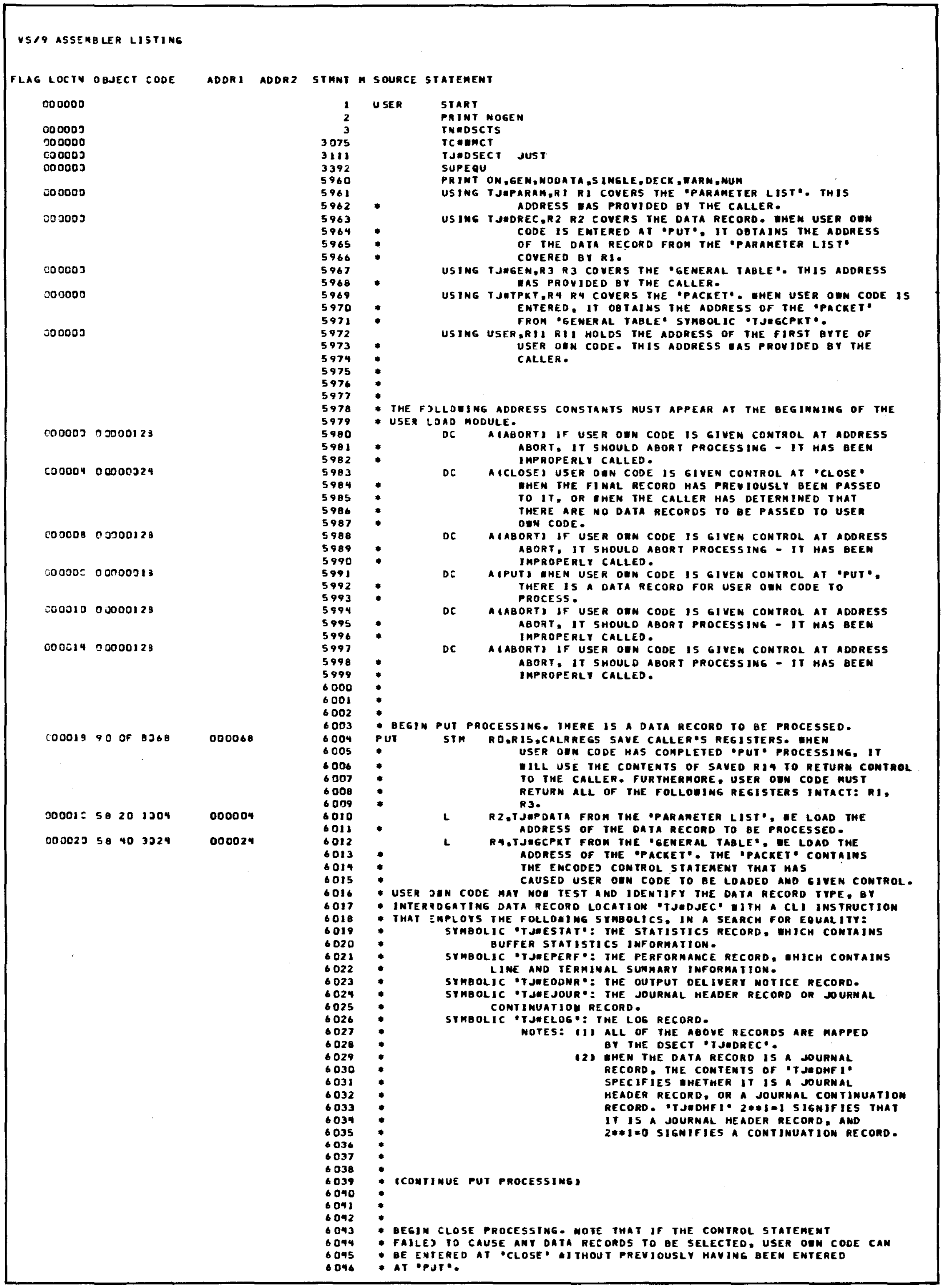
1. *The journal header and continuation and output delivery notification records make up the first part of the report. When INITIAL SEGMENT appears, it indicates the message is continued until FINAL SEGMENT appears in the journal record.*

2. *The STAT record makes up the second part of the report and lists the buffer and ARP pool statistical data. This report corresponds to the BSTAT report shown in Figure B—2 except that no utilization percentage is listed.*

3. *The PERF record makes up the final part of the report and lists the line activity. This report corresponds to the SUM report shown in Figure B—4 except the items are not totaled.*

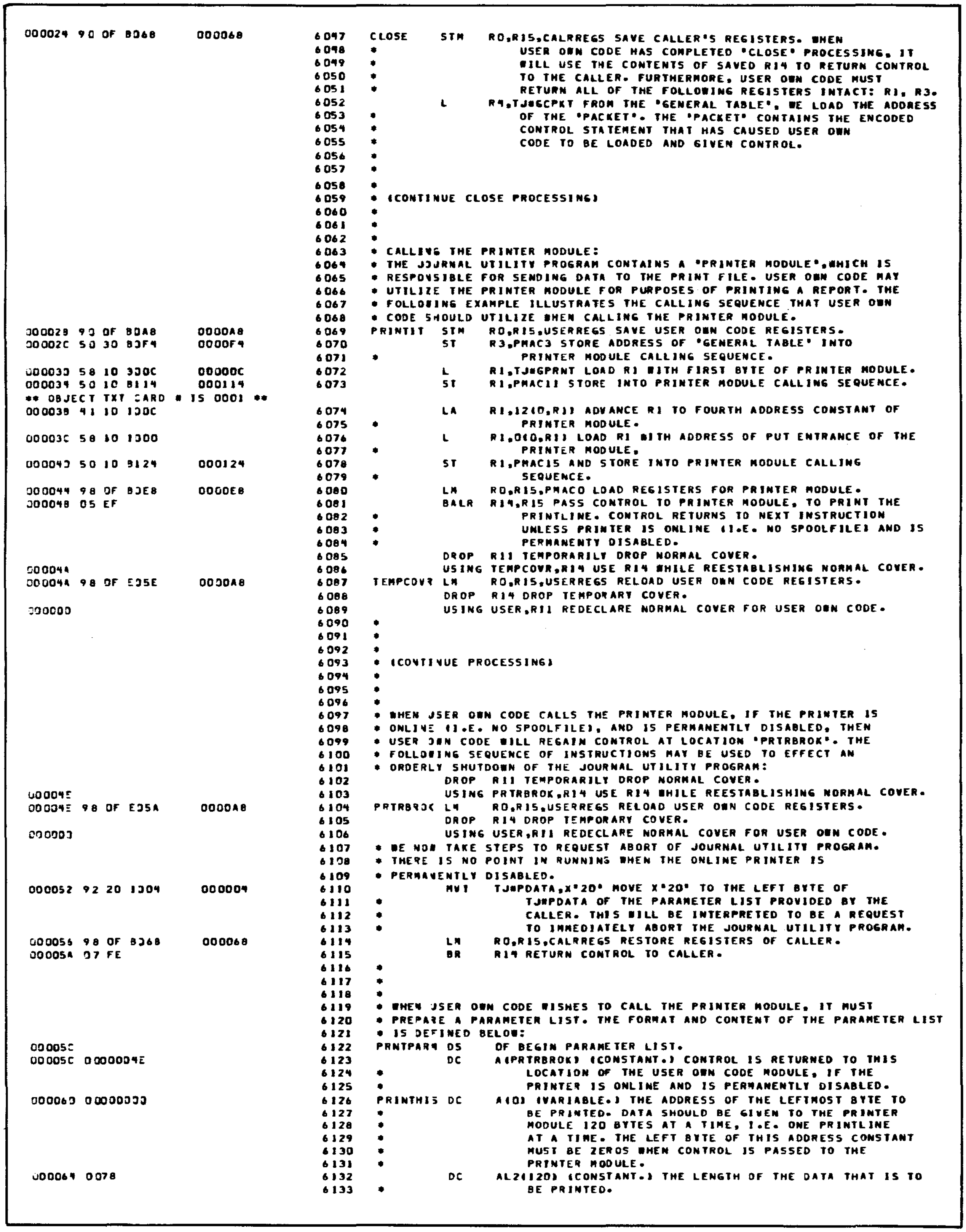
B.6. USER STATEMENT PROGRAM CODE

Figure B-7 lists the USER statement source code that must be incorporated into the user-own-code to create the user's data-reduction routine. Comments are provided in the listing that indicate where the user-own-code is to be inserted. Once the data-reduction routine is assembled and linked, the load module is to be placed in the system load library ($Y$LOD).



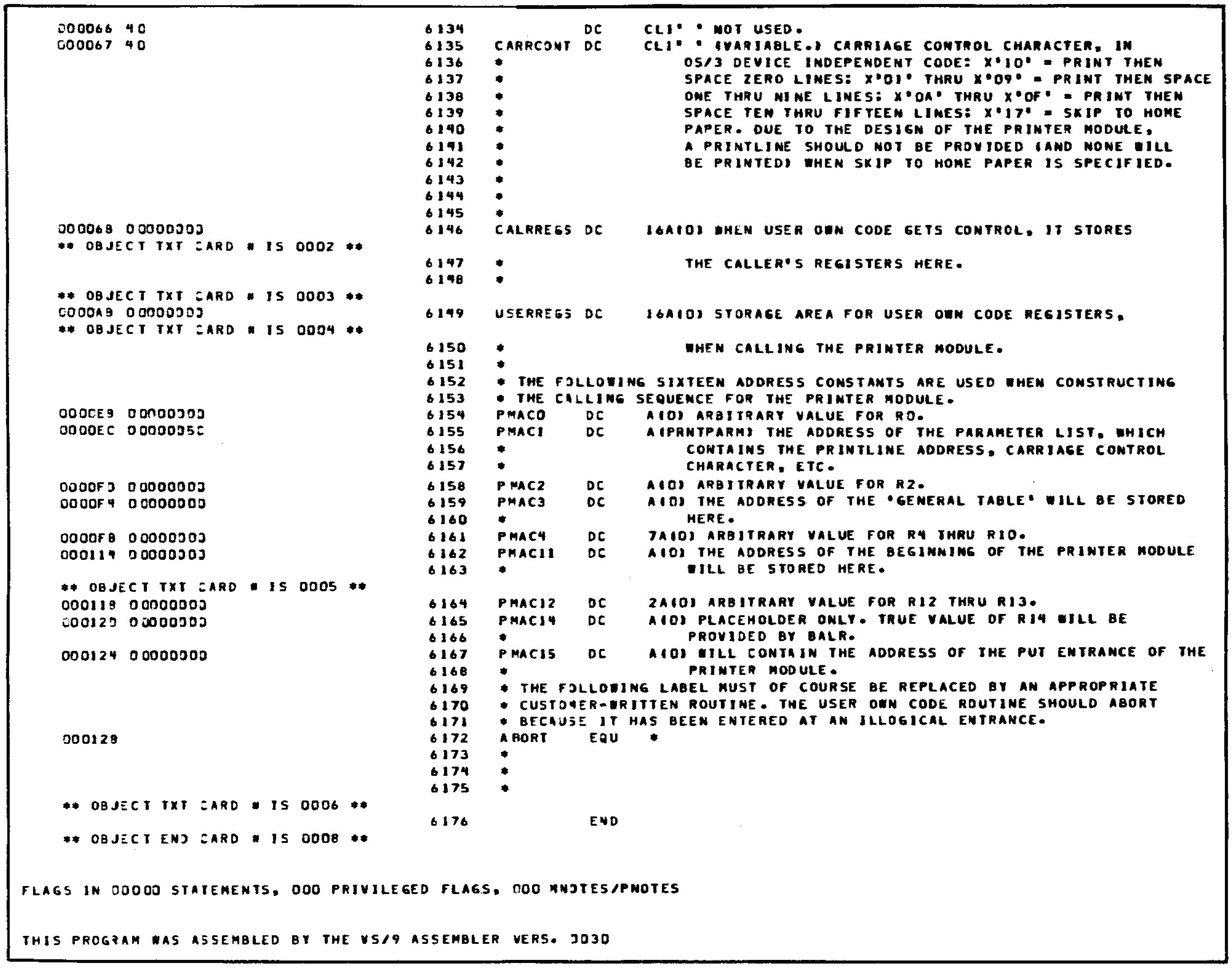
```
VS/9 ASSEMBLER LISTING

FLAG LOCTN OBJECT CODE    ADDR1  ADDR2  STMNT M SOURCE STATEMENT

     000000                                   1   USER     START
                                              2            PRINT NOGEN
     000000                                   3            TN#DSCTS
     000000                                3075            TC#WMCT
     000000                                3111            TJ#DSECT  JUST
     000000                                3392            SUPEQU
                                           5960            PRINT ON,GEN,NODATA,SINGLE,DECK,WARN,NUM
     000000                                5961            USING TJ#PARAM,R1 R1 COVERS THE 'PARAMETER LIST'. THIS
                                           5962   *                  ADDRESS WAS PROVIDED BY THE CALLER.
     000000                                5963            USING TJ#DREC,R2 R2 COVERS THE DATA RECORD. WHEN USER OWN
                                           5964   *                  CODE IS ENTERED AT 'PUT', IT OBTAINS THE ADDRESS
                                           5965   *                  OF THE DATA RECORD FROM THE 'PARAMETER LIST'
                                           5966   *                  COVERED BY R1.
     000000                                5967            USING TJ#GEN,R3 R3 COVERS THE 'GENERAL TABLE'. THIS ADDRESS
                                           5968   *                  WAS PROVIDED BY THE CALLER.
     000000                                5969            USING TJ#TPKT,R4 R4 COVERS THE 'PACKET'. WHEN USER OWN CODE IS
                                           5970   *                  ENTERED, IT OBTAINS THE ADDRESS OF THE 'PACKET'
                                           5971   *                  FROM 'GENERAL TABLE' SYMBOLIC 'TJ#GCPKT'.
     000000                                5972            USING USER,R11 R11 HOLDS THE ADDRESS OF THE FIRST BYTE OF
                                           5973   *                  USER OWN CODE. THIS ADDRESS WAS PROVIDED BY THE
                                           5974   *                  CALLER.
                                           5975   *
                                           5976   *
                                           5977   *
                                           5978   * THE FOLLOWING ADDRESS CONSTANTS MUST APPEAR AT THE BEGINNING OF THE
                                           5979   * USER LOAD MODULE.
     000000 00000128                       5980            DC    A(ABORT) IF USER OWN CODE IS GIVEN CONTROL AT ADDRESS
                                           5981   *                  ABORT, IT SHOULD ABORT PROCESSING - IT HAS BEEN
                                           5982   *                  IMPROPERLY CALLED.
     000004 00000324                       5983            DC    A(CLOSE) USER OWN CODE IS GIVEN CONTROL AT 'CLOSE'
                                           5984   *                  WHEN THE FINAL RECORD HAS PREVIOUSLY BEEN PASSED
                                           5985   *                  TO IT, OR WHEN THE CALLER HAS DETERMINED THAT
                                           5986   *                  THERE ARE NO DATA RECORDS TO BE PASSED TO USER
                                           5987   *                  OWN CODE.
     000008 00000128                       5988            DC    A(ABORT) IF USER OWN CODE IS GIVEN CONTROL AT ADDRESS
                                           5989   *                  ABORT, IT SHOULD ABORT PROCESSING - IT HAS BEEN
                                           5990   *                  IMPROPERLY CALLED.
     00000C 00000018                       5991            DC    A(PUT) WHEN USER OWN CODE IS GIVEN CONTROL AT 'PUT',
                                           5992   *                  THERE IS A DATA RECORD FOR USER OWN CODE TO
                                           5993   *                  PROCESS.
     000010 00000128                       5994            DC    A(ABORT) IF USER OWN CODE IS GIVEN CONTROL AT ADDRESS
                                           5995   *                  ABORT, IT SHOULD ABORT PROCESSING - IT HAS BEEN
                                           5996   *                  IMPROPERLY CALLED.
     000014 00000128                       5997            DC    A(ABORT) IF USER OWN CODE IS GIVEN CONTROL AT ADDRESS
                                           5998   *                  ABORT, IT SHOULD ABORT PROCESSING - IT HAS BEEN
                                           5999   *                  IMPROPERLY CALLED.
                                           6000   *
                                           6001   *
                                           6002   *
                                           6003   * BEGIN PUT PROCESSING. THERE IS A DATA RECORD TO BE PROCESSED.
     000018 90 0F B068     000068          6004   PUT      STM   R0,R15,CALRREGS SAVE CALLER'S REGISTERS. WHEN
                                           6005   *                  USER OWN CODE HAS COMPLETED 'PUT' PROCESSING, IT
                                           6006   *                  WILL USE THE CONTENTS OF SAVED R14 TO RETURN CONTROL
                                           6007   *                  TO THE CALLER. FURTHERMORE, USER OWN CODE MUST
                                           6008   *                  RETURN ALL OF THE FOLLOWING REGISTERS INTACT: R1,
                                           6009   *                  R3.
     00001C 58 20 1004     000004          6010            L     R2,TJ#PDATA FROM THE 'PARAMETER LIST', WE LOAD THE
                                           6011   *                  ADDRESS OF THE DATA RECORD TO BE PROCESSED.
     000020 58 40 3024     000024          6012            L     R4,TJ#GCPKT FROM THE 'GENERAL TABLE', WE LOAD THE
                                           6013   *                  ADDRESS OF THE 'PACKET'. THE 'PACKET' CONTAINS
                                           6014   *                  THE ENCODED CONTROL STATEMENT THAT HAS
                                           6015   *                  CAUSED USER OWN CODE TO BE LOADED AND GIVEN CONTROL.
                                           6016   * USER OWN CODE MAY NOW TEST AND IDENTIFY THE DATA RECORD TYPE, BY
                                           6017   * INTERROGATING DATA RECORD LOCATION 'TJ#DJEC' WITH A CLI INSTRUCTION
                                           6018   * THAT EMPLOYS THE FOLLOWING SYMBOLICS, IN A SEARCH FOR EQUALITY:
                                           6019   *        SYMBOLIC 'TJ#ESTAT': THE STATISTICS RECORD, WHICH CONTAINS
                                           6020   *              BUFFER STATISTICS INFORMATION.
                                           6021   *        SYMBOLIC 'TJ#EPERF': THE PERFORMANCE RECORD, WHICH CONTAINS
                                           6022   *              LINE AND TERMINAL SUMMARY INFORMATION.
                                           6023   *        SYMBOLIC 'TJ#EODNR': THE OUTPUT DELIVERY NOTICE RECORD.
                                           6024   *        SYMBOLIC 'TJ#EJOUR': THE JOURNAL HEADER RECORD OR JOURNAL
                                           6025   *              CONTINUATION RECORD.
                                           6026   *        SYMBOLIC 'TJ#ELOG': THE LOG RECORD.
                                           6027   *                   NOTES: (1) ALL OF THE ABOVE RECORDS ARE MAPPED
                                           6028   *                              BY THE DSECT 'TJ#DREC'.
                                           6029   *                          (2) WHEN THE DATA RECORD IS A JOURNAL
                                           6030   *                              RECORD, THE CONTENTS OF 'TJ#DHF1'
                                           6031   *                              SPECIFIES WHETHER IT IS A JOURNAL
                                           6032   *                              HEADER RECORD, OR A JOURNAL CONTINUATION
                                           6033   *                              RECORD. 'TJ#DHF1' 2**1=1 SIGNIFIES THAT
                                           6034   *                              IT IS A JOURNAL HEADER RECORD, AND
                                           6035   *                              2**1=0 SIGNIFIES A CONTINUATION RECORD.
                                           6036   *
                                           6037   *
                                           6038   *
                                           6039   * (CONTINUE PUT PROCESSING)
                                           6040   *
                                           6041   *
                                           6042   *
                                           6043   * BEGIN CLOSE PROCESSING. NOTE THAT IF THE CONTROL STATEMENT
                                           6044   * FAILED TO CAUSE ANY DATA RECORDS TO BE SELECTED, USER OWN CODE CAN
                                           6045   * BE ENTERED AT 'CLOSE' WITHOUT PREVIOUSLY HAVING BEEN ENTERED
                                           6046   * AT 'PUT'.
```

Figure B—7. Required USER Statement Program Code (Part 1 of 3)



```
000024 90 0F B068     000068      6047  CLOSE    STM   R0,R15,CALRREGS SAVE CALLER'S REGISTERS. WHEN
                                  6048  *                    USER OWN CODE HAS COMPLETED 'CLOSE' PROCESSING, IT
                                  6049  *                    WILL USE THE CONTENTS OF SAVED R14 TO RETURN CONTROL
                                  6050  *                    TO THE CALLER. FURTHERMORE, USER OWN CODE MUST
                                  6051  *                    RETURN ALL OF THE FOLLOWING REGISTERS INTACT: R1, R3.
                                  6052           L     R4,TJ#GCPKT FROM THE 'GENERAL TABLE', WE LOAD THE ADDRESS
                                  6053  *                    OF THE 'PACKET'. THE 'PACKET' CONTAINS THE ENCODED
                                  6054  *                    CONTROL STATEMENT THAT HAS CAUSED USER OWN
                                  6055  *                    CODE TO BE LOADED AND GIVEN CONTROL.
                                  6056  *
                                  6057  *
                                  6058  *
                                  6059  * (CONTINUE CLOSE PROCESSING)
                                  6060  *
                                  6061  *
                                  6062  *
                                  6063  * CALLING THE PRINTER MODULE:
                                  6064  * THE JOURNAL UTILITY PROGRAM CONTAINS A 'PRINTER MODULE',WHICH IS
                                  6065  * RESPONSIBLE FOR SENDING DATA TO THE PRINT FILE. USER OWN CODE MAY
                                  6066  * UTILIZE THE PRINTER MODULE FOR PURPOSES OF PRINTING A REPORT. THE
                                  6067  * FOLLOWING EXAMPLE ILLUSTRATES THE CALLING SEQUENCE THAT USER OWN
                                  6068  * CODE SHOULD UTILIZE WHEN CALLING THE PRINTER MODULE.
000028 90 0F B0A8     0000A8      6069  PRINTIT  STM   R0,R15,USERREGS SAVE USER OWN CODE REGISTERS.
00002C 50 30 B0F4     0000F4      6070           ST    R3,PMAC3 STORE ADDRESS OF 'GENERAL TABLE' INTO
                                  6071  *                    PRINTER MODULE CALLING SEQUENCE.
000030 58 10 300C     00000C      6072           L     R1,TJ#GPRNT LOAD R1 WITH FIRST BYTE OF PRINTER MODULE.
000034 50 10 B114     000114      6073           ST    R1,PMAC11 STORE INTO PRINTER MODULE CALLING SEQUENCE.
** OBJECT TXT CARD # IS 0001 **
000038 41 10 100C                 6074           LA    R1,12(0,R1) ADVANCE R1 TO FOURTH ADDRESS CONSTANT OF
                                  6075  *                    PRINTER MODULE.
00003C 58 10 1000                 6076           L     R1,0(0,R1) LOAD R1 WITH ADDRESS OF PUT ENTRANCE OF THE
                                  6077  *                    PRINTER MODULE,
000040 50 10 B124     000124      6078           ST    R1,PMAC15 AND STORE INTO PRINTER MODULE CALLING
                                  6079  *                    SEQUENCE.
000044 98 0F B0E8     0000E8      6080           LM    R0,R15,PMAC0 LOAD REGISTERS FOR PRINTER MODULE.
000048 05 EF                      6081           BALR  R14,R15 PASS CONTROL TO PRINTER MODULE, TO PRINT THE
                                  6082  *                    PRINTLINE. CONTROL RETURNS TO NEXT INSTRUCTION
                                  6083  *                    UNLESS PRINTER IS ONLINE (I.E. NO SPOOLFILE) AND IS
                                  6084  *                    PERMANENTY DISABLED.
                                  6085           DROP  R11 TEMPORARILY DROP NORMAL COVER.
00004A                            6086           USING TEMPCOVR,R14 USE R14 WHILE REESTABLISHING NORMAL COVER.
00004A 98 0F E05E     0000A8      6087  TEMPCOVR LM    R0,R15,USERREGS RELOAD USER OWN CODE REGISTERS.
                                  6088           DROP  R14 DROP TEMPORARY COVER.
000000                            6089           USING USER,R11 REDECLARE NORMAL COVER FOR USER OWN CODE.
                                  6090  *
                                  6091  *
                                  6092  *
                                  6093  * (CONTINUE PROCESSING)
                                  6094  *
                                  6095  *
                                  6096  *
                                  6097  * WHEN USER OWN CODE CALLS THE PRINTER MODULE, IF THE PRINTER IS
                                  6098  * ONLINE (I.E. NO SPOOLFILE), AND IS PERMANENTLY DISABLED, THEN
                                  6099  * USER OWN CODE WILL REGAIN CONTROL AT LOCATION 'PRTRBROK'. THE
                                  6100  * FOLLOWING SEQUENCE OF INSTRUCTIONS MAY BE USED TO EFFECT AN
                                  6101  * ORDERLY SHUTDOWN OF THE JOURNAL UTILITY PROGRAM:
                                  6102           DROP  R11 TEMPORARILY DROP NORMAL COVER.
00004E                            6103           USING PRTRBROK,R14 USE R14 WHILE REESTABLISHING NORMAL COVER.
00004E 98 0F E05A     0000A8      6104  PRTRBROK LM    R0,R15,USERREGS RELOAD USER OWN CODE REGISTERS.
                                  6105           DROP  R14 DROP TEMPORARY COVER.
000000                            6106           USING USER,R11 REDECLARE NORMAL COVER FOR USER OWN CODE.
                                  6107  * WE NOW TAKE STEPS TO REQUEST ABORT OF JOURNAL UTILITY PROGRAM.
                                  6108  * THERE IS NO POINT IN RUNNING WHEN THE ONLINE PRINTER IS
                                  6109  * PERMANENTLY DISABLED.
000052 92 20 1004     000004      6110           MVI   TJ#PDATA,X'20' MOVE X'20' TO THE LEFT BYTE OF
                                  6111  *                    TJ#PDATA OF THE PARAMETER LIST PROVIDED BY THE
                                  6112  *                    CALLER. THIS WILL BE INTERPRETED TO BE A REQUEST
                                  6113  *                    TO IMMEDIATELY ABORT THE JOURNAL UTILITY PROGRAM.
000056 98 0F B068     000068      6114           LM    R0,R15,CALRREGS RESTORE REGISTERS OF CALLER.
00005A 07 FE                      6115           BR    R14 RETURN CONTROL TO CALLER.
                                  6116  *
                                  6117  *
                                  6118  *
                                  6119  * WHEN USER OWN CODE WISHES TO CALL THE PRINTER MODULE, IT MUST
                                  6120  * PREPARE A PARAMETER LIST. THE FORMAT AND CONTENT OF THE PARAMETER LIST
                                  6121  * IS DEFINED BELOW:
00005C                            6122  PRNTPARM DS    0F BEGIN PARAMETER LIST.
00005C 0000004E                   6123           DC    A(PRTRBROK) (CONSTANT.) CONTROL IS RETURNED TO THIS
                                  6124  *                    LOCATION OF THE USER OWN CODE MODULE, IF THE
                                  6125  *                    PRINTER IS ONLINE AND IS PERMANENTLY DISABLED.
000060 00000000                   6126  PRINTHIS DC    A(0) (VARIABLE.) THE ADDRESS OF THE LEFTMOST BYTE TO
                                  6127  *                    BE PRINTED. DATA SHOULD BE GIVEN TO THE PRINTER
                                  6128  *                    MODULE 120 BYTES AT A TIME, I.E. ONE PRINTLINE
                                  6129  *                    AT A TIME. THE LEFT BYTE OF THIS ADDRESS CONSTANT
                                  6130  *                    MUST BE ZEROS WHEN CONTROL IS PASSED TO THE
                                  6131  *                    PRINTER MODULE.
000064 0078                       6132           DC    AL2(120) (CONSTANT.) THE LENGTH OF THE DATA THAT IS TO
                                  6133  *                    BE PRINTED.
```

Figure B—7. Required USER Statement Program Code (Part 2 of 3)



```
000066 40                        6134             DC    CL1' ' NOT USED.
000067 40                        6135   CARRCONT  DC    CL1' ' (VARIABLE.) CARRIAGE CONTROL CHARACTER, IN
                                 6136   *                   OS/3 DEVICE INDEPENDENT CODE: X'10' = PRINT THEN
                                 6137   *                   SPACE ZERO LINES: X'01' THRU X'09' = PRINT THEN SPACE
                                 6138   *                   ONE THRU NINE LINES: X'0A' THRU X'0F' = PRINT THEN
                                 6139   *                   SPACE TEN THRU FIFTEEN LINES: X'17' = SKIP TO HOME
                                 6140   *                   PAPER. DUE TO THE DESIGN OF THE PRINTER MODULE,
                                 6141   *                   A PRINTLINE SHOULD NOT BE PROVIDED (AND NONE WILL
                                 6142   *                   BE PRINTED) WHEN SKIP TO HOME PAPER IS SPECIFIED.
                                 6143   *
                                 6144   *
                                 6145   *
000068 00000000                  6146   CALRREGS  DC    16A(0) WHEN USER OWN CODE GETS CONTROL, IT STORES
** OBJECT TXT CARD # IS 0002 **
                                 6147   *                   THE CALLER'S REGISTERS HERE.
                                 6148   *
** OBJECT TXT CARD # IS 0003 **
0000A8 00000000                  6149   USERREGS  DC    16A(0) STORAGE AREA FOR USER OWN CODE REGISTERS,
** OBJECT TXT CARD # IS 0004 **
                                 6150   *                   WHEN CALLING THE PRINTER MODULE.
                                 6151   *
                                 6152   * THE FOLLOWING SIXTEEN ADDRESS CONSTANTS ARE USED WHEN CONSTRUCTING
                                 6153   * THE CALLING SEQUENCE FOR THE PRINTER MODULE.
0000E8 00000000                  6154   PMAC0     DC    A(0) ARBITRARY VALUE FOR R0.
0000EC 0000005C                  6155   PMAC1     DC    A(PRNTPARM) THE ADDRESS OF THE PARAMETER LIST, WHICH
                                 6156   *                   CONTAINS THE PRINTLINE ADDRESS, CARRIAGE CONTROL
                                 6157   *                   CHARACTER, ETC.
0000F0 00000000                  6158   PMAC2     DC    A(0) ARBITRARY VALUE FOR R2.
0000F4 00000000                  6159   PMAC3     DC    A(0) THE ADDRESS OF THE 'GENERAL TABLE' WILL BE STORED
                                 6160   *                   HERE.
0000F8 00000000                  6161   PMAC4     DC    7A(0) ARBITRARY VALUE FOR R4 THRU R10.
000114 00000000                  6162   PMAC11    DC    A(0) THE ADDRESS OF THE BEGINNING OF THE PRINTER MODULE
                                 6163   *                   WILL BE STORED HERE.
** OBJECT TXT CARD # IS 0005 **
000118 00000000                  6164   PMAC12    DC    2A(0) ARBITRARY VALUE FOR R12 THRU R13.
000120 00000000                  6165   PMAC14    DC    A(0) PLACEHOLDER ONLY. TRUE VALUE OF R14 WILL BE
                                 6166   *                   PROVIDED BY BALR.
000124 00000000                  6167   PMAC15    DC    A(0) WILL CONTAIN THE ADDRESS OF THE PUT ENTRANCE OF THE
                                 6168   *                   PRINTER MODULE.
                                 6169   * THE FOLLOWING LABEL MUST OF COURSE BE REPLACED BY AN APPROPRIATE
                                 6170   * CUSTOMER-WRITTEN ROUTINE. THE USER OWN CODE ROUTINE SHOULD ABORT
                                 6171   * BECAUSE IT HAS BEEN ENTERED AT AN ILLOGICAL ENTRANCE.
000128                           6172   ABORT     EQU   *
                                 6173   *
                                 6174   *
                                 6175   *
** OBJECT TXT CARD # IS 0006 **
                                 6176             END
** OBJECT END CARD # IS 0008 **

FLAGS IN 00000 STATEMENTS, 000 PRIVILEGED FLAGS, 000 MNOTES/PNOTES

THIS PROGRAM WAS ASSEMBLED BY THE VS/9 ASSEMBLER VERS. 0030
```

Figure B—7. Required USER Statement Program Code (Part 3 of 3)

B.7. JOURNAL UTILITY - COLD RESTART FACILITY

The cold restart facility of the journal utility is used to rebuild ICAM queues when a hardware failure occurs during ICAM processing. By selecting the RESTART statement, the cold restart facility reads the history file up to the point where the hardware failure occurred; then, the journal utility writes the messages to a disk file. After all the messages are written, the journal utility will go to normal end-of-job. After job termination, a warm restart of ICAM is possible via RESTART parameter on the NETREQ macroinstruction.

For ICAM to create a history file containing cold restart records, you must have specified the RESTART operand in the JRNINIT parameter on the CCA and JRNFILE macroinstructions to create the SAT disk file. Additional files are required to cold restart. These files must be defined and formatted in the same manner as they were for the ICAM disk queue processor in the previous ICAM session. Disk or tape input files must be standard labeled SAT/TSAT files, defined via job control. Input and output restart files must not be assigned the same file name.

NOTE:

Some messages may be lost between the time when the hardware failure occurred and the time when the last message was written to the history file.

RESTART

B.7.1. RESTART Statement

Function:

Restores the ICAM message queues through a cold restart of the network by reading the history file. The file is defined in the journal utility job control stream via the LFD filename INPUT1.

Format:

△OPERATION△	OPERAND
RESTART	$\left[\text{FILE=}\left\{\begin{array}{l}\text{file-name}\\ \text{(file-name}_1\text{, file-name}_2\text{, . . . file-name}_n\text{)}\end{array}\right\}\right]$

Parameter:

FILE
: Designates the file-name of the disk file to be created. Single or multiple files may be specified.

If omitted, all the disk queue files residing on the input file are created.

Example:

```
// JOB RESTART,,7000
// DVC 20  // LFD PRNTR
// DVC 50
// VOL 0531CM
// LBL HISTORY-FILE
// LFD INPUT1
// DVC 51
// VOL NEWFFIL
// LBL NEW-HISTORY
// LFD OUTFIL
// OPTION DUMP
// EXEC JUST
/$
RESTART FIXIT
/*
/&
// FIN
```

1. Main storage size specification on job card

2–6. Printer file and INPUT1 file specification

7–10. Output file specification

11. Desire dump upon abnormal termination

12. Execute the journal utility

13. Start of utility statements

14. User statement defining the user data-reduction routine

15. End of journal utility statements

16. End of job

17. End of card input

B.8. ERROR HANDLING

If the journal utility has encountered data errors in the SELECT report, the following message is listed in the system log file:

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

The number of data errors found in the report are totaled and printed with this message:

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

The user should send the report along with a software user report (SUR) to his local system analyst.

All other error messages are printed via canned message files whenever invalid statements or keywords are encountered. See system message manual, UP-8076 (current version).

Index

Comments concerning this manual may be made in the space provided below. Please fill in the requested information.

System:

Manual Title:

UP No: Revision No: Update:

Name of User:

Address of User:

Comments:

UT

FOLD

FIRST CLASS
PERMIT NO. 21
BLUE BELL, PA.

BUSINESS REPLY MAIL NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

SPERRY UNIVAC

P.O. BOX 500
BLUE BELL, PA.
19424

ATTN: SYSTEMS PUBLICATIONS DEPT.

FOLD

CUT