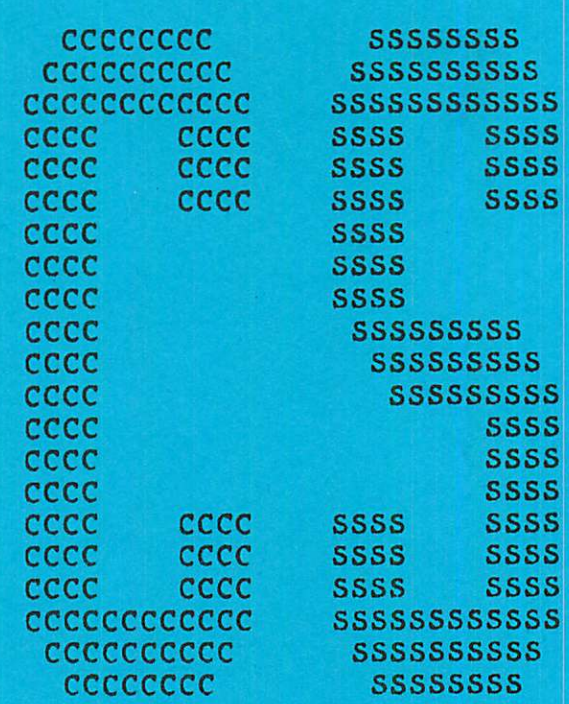

CS/3000

External Reference Specification

December 18, 1978

CONTENTS

1.1 TERMINOLOGY

1.2 Data Communications System

Viewed in its most elementary aspect, a data communications system consists of (1) a central computer (CPU) and associated transmission control equipment, (2) remote stations, and (3) the communications lines that connect the remote stations to the CPU.

Typically, the transmission control equipment is called a transmission control unit (TCU). TCU's are generally attached directly to a channel of the CPU. In the case of a 3000 CPU, examples of a TCU are the HSI, SSLC, and SCCP boards (described later).

The equipment constituting a remote station can be either a terminal or another CPU. A terminal consists of a control unit and one or more input/output devices, each of which is called a component of that terminal.

It is the method of connection to the CPU, rather than the distance from the CPU, that determines whether a station is classified as remote. A station is considered remote if it is connected to the CPU through a TCU. A station connected directly to a data channel of a CPU is termed a local station. In general, data communications software concerns itself with remote stations only.

1.3 Switched/Non-Switched

A non-switched (dedicated) line is one that continuously links the stations associated with it, regardless of the amount of time it is in use for message traffic. This type of line could connect a terminal physically located near the CPU, but usually it is a line provided by a common carrier on a contractual basis in order to service a distant station. A switched line is one in which an electrical connection between the CPU and a remote station is established by dialing. As in the public telephone network, the actual communication path for a given

transmission via a switched line is not fixed but is automatically selected from a variety of possible paths by common carrier switching equipment.

Each remote station on a switched line is continuously connected to a common carrier switching center (exchange) by an access line in the same way as a telephone. Similarly, each TCU at the central computer is connected to an exchange by access lines. Usually, a TCU has several access lines, each with its own telephone number.

Common carriers usually charge for switched lines on a time-used basis rather than a contractual basis.

1.4 Point-to-Point/Multi-Point

A line is called point-to-point if it connects the computer to a single remote station; it is multi-point if more than one remote station is connected to the line. (Multi-point is often called multi-drop.)

See Figure 1.1 for a general overview of line and station configurations.

1.5 Duplex/Half-Duplex/Simplex

The term duplex (or full-duplex) is applied to a communications line that can accommodate data transmission in both directions at once. Half-duplex lines permit transmission in only one direction at a time; the line may be used to both transmit and receive data, but not simultaneously. To allow the direction of data flow to be reversed, all traffic is stopped for some period of time. This quiescent interval is called the turn-around time. Duplex and half-duplex lines accommodate bidirectional communication; a simplex line is unidirectional.

Switched lines which use a one phone connection are constrained to be half duplex if they're operating at line speeds in excess of 1200 bits/second because the lines are two-wire and do not have sufficient bandwidth to allow full duplex operation. Two phone connections allow full duplex operation in a dial up mode. Nonswitched lines are usually full duplex.

FIGURE 1.1 Line and Station Configurations

1.6 Synchronous/Asynchronous Transmission

Basically, there are two distinct methods by which data are transmitted over communications lines:

a) synchronous communication is used for high-speed data transmission between the CPU and another remote computer or high-speed terminal.

b) asynchronous communication is used for data transmission at lower speeds between the CPU and remote terminals of various types.

Asynchronous transmission uses start and stop bits to separate one character from another. Synchronous transmission does not use start-stop bits and thereby permits more information to be passed over a circuit during a given time interval.

Although there are exceptions, a synchronous station is generally another CPU or a page-mode terminal which may have high speed components like a card reader, printer, magnetic tape drive, etc. On the other hand, an asynchronous terminal is usually a low-speed device with a keyboard.

1.7 Two-Way Alternate/Two-Way Simultaneous

A Two-Way Alternate (TWA) protocol can only manage data transmission in one direction at a time. A TWA protocol may be used on either a half or full duplex channel, but data will never be sent and received simultaneously. TWA protocol performance is better when used on a full duplex channel because the turn-around time is minimized.

A Two-Way Simultaneous (TWS) protocol manages simultaneous data transmission and reception. A full duplex communications channel is necessary when using a TWS protocol.

For two-way alternate operation, IBM defined a generalized transmission technique called binary synchronous communication (BSC). BSC is currently the de-facto industry standard for half-duplex synchronous communication. BSC accommodates a variety of transmission codes (EBCDIC, USASCII, Six-Bit Transcode) as well as a transparency feature that allows transmission of raw binary data which has no graphic or control significance. BSC is predicated on the assumption that transmission errors can occur; hence, much of BSC is concerned with error detection and correction.

For two-way simultaneous operation, an HP developed protocol called HPDLC I is used. This is based on "Recommendation X.25: Interface between DTE and DCE for terminals operating in the Packet Mode on

ublic Data Networks" as approved by CCITT (International Telephone & Telegraph Consultative Committee). This basis was chosen since it is an approved standard by an international body, and is a subset of both ANSI's current ADCCP recommendation and ISO HDLC recommendation.

.8 Modes

Any given BSC line in a communications network operates in one of two modes:

a) Contention Mode

For point-to-point lines (switched and non-switched), the two stations are often symmetrical with neither having control of the line or the other station. The two stations contend for the line by bidding to become the transmitter. To provide for the situation in which both stations bid simultaneously, one of the stations is denoted the primary while the other is the secondary station. The primary station will persist in bidding while the secondary relents.

Contention mode is never applicable to multi-point lines.

) Supervised Mode

This mode is required to operate on either switched or non-switched multi-point lines. On a line operating in supervised mode, there is always one control station; each of the remaining stations is a tributary station. The control station (normally a CPU) is the master, and the tributaries are slaves.

The control station "polls" the tributary to solicit input, and it "selects" the tributary for output.

HPDLC I makes use of two-way simultaneous communications with unbalanced, symmetrical stations operating in Asynchronous Response Mode. "Symmetrical, unbalanced stations" is a term denoting the fact that each station contains a primary function responsible for sending data and control and a secondary function responsible for receiving data and control. The secondary function also makes positive and negative acknowledgements of the data received. The primary function in one station transmits to the secondary function in the other station. "Asynchronous Response Mode" means that a secondary function can transmit at the earliest opportunity without any prompting by its corresponding primary function. Since this is a two-way simultaneous protocol, this opportunity is always available.

1.9 Addressing

a) Components.

A component (I/O device) of a station is either input-only, output-only, or input/output. A station may have multiple components.

On all contention lines and most modern multipoint lines, the components of a multi-component station are uniquely identified via component addresses within the transmitted text. However on supervised lines containing relatively old tributaries, such as IBM 2780 RJE terminals, the components are directly addressed in the polling and selection sequences.

b) Terminal Groups.

Most modern multipoint tributary terminals are attached to the line in groups. This enables a hardware cost savings since only one modem and one set of polling/selection decoding electronics are needed per group. Switched lines may have more than one group; however, all of the members of any particular group must share the same line. Nonswitched line may have an unlimited number. In general polling or selecting tributary groups requires a three field sequence:

1. group ID
2. station ID
3. component ID

CS logically assumes that all multipoint lines always use the full three field sequence. However it is quite rare that this would actually happen. Usually the line will fall into one of three possible configurations. Their characteristics and CS conventions are:

(1) Only station IDs are used.
    All stations logically belong to one group whose group ID is null. Each station has one component whose ID is null.
(2) Only group and station IDs are used.
    Each station has one component whose ID is null.
(3) Only station and component IDs are used.
    All stations logically belong to one group whose group ID is null.

An additional CS convention is that when configuring the line, the station ID is not explicitly specified. It must be embedded into each of the station's component IDs.

There are two additional advantages that terminal groups offer:

1. General Poll.

   The control station may invoke general poll (sometimes called "group poll") by sending:
   (a) group ID
   (b) general poll ID

   This allows any or all terminals in the group, having data to send, to respond. The electronics within the group insures that the polling proceeds in an orderly manner; that is, only one tributary at a time actually sends text and after the last tributary has finished, a special sequence is sent to the control station notifying it that there are no more terminals having data to send. Tributaries are required to place their station IDs at the beginning of their sent text so that the control station may identify the origin of the incoming data.

   The advantage of general poll is that it eliminates the overhead and delay of having to sequentially poll each terminal.

2. General select.

   This facility, sometimes called "group broadcast", enables the control station to broadcast a text block to all terminals attached to a group. No acknowledgement to the text is returned by the group or any of its terminals. The HP 2645 terminal also has line select which broadcasts text to all terminals on the line in a similar fashion.

1.10 Identification Sequences

To provide validation and security for the use of both switched and nonswitched lines, ID sequence verification is used. (This facility is not used by all protocols; for example, it is used by the BSC protocol, but not by the HPDLC I protocol.) To use this facility, each remote station that is permitted to access the computer over the line must have an ID sequence that it automatically sends immediately after the line connection has been established. The program compares the received sequence against a list of permissible sequences. If a match is found, message transmission can proceed. Otherwise, the connection will be broken.

1.11 Concurrent vs. Non-concurrent Input/Output

In data communications, there is a requirement for multiple lines to be controlled by a single application or process. This requirement, in turn, dictates the need for I/O requests to return to the caller without waiting for completion. To satisfy this requirement, CS provides two basic forms of I/O support:

1) I/O requests with an implicit wait. This mode will be called non-concurrent input/output (NCIO), because I/O is not done concurrently with the execution of the user's program.

2) I/O requests without wait, requiring the user to subsequently issue an explicit wait request. This mode will be called concurrent input/output (CIO), because I/O is performed concurrently with the execution of the user's program.

1.12 ABBREVIATIONS

The following abbreviations are used throughout the remainder of this document:

BSC - Binary Synchronous Communication
CIO - Concurrent Input/Output
CS - MPE Communications System (CS/3000)
DCE - Data Communications Equipment
The equipment installed at the user's site which (a) provides all the functions required to establish, maintain, and terminate a connection, (b) the signal conversion coding between the data terminating equipment (DTE) and the common carrier's line, e.g., data set, modem. DCE is also used to refer to the network side of a network/terminal interface on a PACKET MODE PUBLIC DATA NETWORK.
DTE - Data Terminating Equipment
Any piece of equipment at which a communications path begins or ends, e.g., a terminal or computer. DTE is also used to refer to the terminal side of a network/terminal interface on a PUBLIC MODE PUBLIC DATA NETWORK.
FS - MPE File System
HPDLC I- HP Data Link Control (a TWS protocol)
HSI - HP 30360A Hardwired Serial Interface
ID - ID sequence. Additionally it refers to a field of a polling or selection sequence for a multipoint tributary.
MP - Multi-point (also called multi-drop)
NCIO - Non-Concurrent Input/Output
NSW - Non-Switched

P - Point-to-Point
SCCP -HP 30010A (or 30020A) Single Channel Communications Processor
SDLC - Synchronous Data Link Control
SSLC - HP 30055A Synchronous Single Line Controller
SW - Switched
TCU - Transmission Control Unit (an HSI, SSLC, or SCCP)
TWA - Two way alternate transmission
TWS - Two way simultaneous transmission

CS/3000 provides the MPE user with a basic and flexible set of tools for the control of communications lines and terminals. The system is designed to satisfy the communications I/O requirements of both the ordinary user and the more sophisticated user. These requirements are met by providing a flexible set of user-callable intrinsics, each of which performs a specific communications function.

If CS is installed and communication lines are configured, a minimum of 64K words of main memory is required. In addition, CS requires at least one of the following hardware components:

1) HP 30055A Synchronous Single Line Controller (SSLC)
2) HP 30360A Hardware Serial Interface (HSI)
3) HP 30010A (or 30020A) Single Channel Communications Processor (SCCP)

The SSLC is a control unit which accommodates one synchronous line and is the mechanism for communication with other CPU's or synchronous terminals using the BSC or Conversational BSC protocol. The HSI is a control unit which provides a means of high speed communication between HP CPUs. The SCCP is a control unit which supports both synchronous and asynchronous lines, and provides a variety of user-specified protocols.

CS/3000 consists of a set of drivers and user-callable intrinsics. As such, CS is not a runnable subsystem but rather a set of primitives which facilitate the development of data communications applications.

CS will support SSLC, HSI, and SCCP lines; the MPE file system (FS) supports all of the standard peripherals plus a special form of asynchronous terminal support for log-on terminals. The device types supported by CS are

```
type 17 - SCCP   permanently allocated to CS
type 18 - SSLC   permanently allocated to CS
type 19 - HSI    permanently allocated to CS
```

All of the standard MPE peripherals will be permanently assigned to FS; SSLC, HSI, and SCCP lines will be permanently assigned to CS.

2.1 INTRINSICS

CS intrinsics begin with the letter "C" and, with few exceptions, are analogous in both name and function to FS intrinsics. The primary distinction between FS and CS is that FS presents a device-independent interface to the application program whereas the CS interface is device-dependent. The same set of CS intrinsics applies to all device types supported by CS.

Depending upon whether a device is permanently assigned to CS or FS, the method of access is restricted to either the "C" or "F" intrinsics, respectively, never a mixture.

FOPEN parameters may be supplied (or overridden) from an external source at execution time via the :FILE command. CS provides the analogous facility for the COPEN intrinsic with the :CLINE command. Similarly, CS provides a :CRESET command which is analogous to the :RESET command of the file system.

The FOPEN of FS returns to the caller a file number used to identify the opened file in subsequent FS intrinsic calls. The COPEN of CS returns to the caller a line number used to identify the opened line in subsequent CS intrinsic calls. Within a single process, FS file numbers and CS line numbers are unambiguous.

Unlike the FS, which permits multiple FOPENs of a single file, CS will permit only one COPEN for a given line.

2.2 CAPABILITY

Due to the limited number of hardware resources supported by CS, the user (not the program) must have a special capability in order to COPEN a CS line. The user attribute "CS" has been added to the :NEWACCT, :ALTACCT, :NEWUSER, and :ALTUSER commands. This capability will be reflected by a "1" in bit (13:1) of word 0 in the CAPABILITY parameter of the WHO intrinsic.

.3 S/3000 HARDWARE COMPONENTS

.4 HP 30360A Hardwired Serial Interface (HSI)

One HSI board occupies one slot on the multiplex channel (30035). An HSI cannot be attached to a selector channel (30030).

Each HSI board can accommodate up to 4 distinct lines, only one of which may be open at any given time. Physically, each of these four lines must be configured as hardwired point-to-point. The data transfer rate is a function of cable length; the maximum transfer rate is 2.5 megabits per second for cables less than 1000 feet in length.

NOTE: Although the HSI is a high-speed device, it is actually an asynchronous device because each data word is surrounded by start-stop bits (20 bits are transmitted per data word).

The HSI requires a full-duplex line.

.5 HP 30055A Synchronous Single Line Controller (SSLC)

One SSLC board occupies one slot on the multiplex channel (30035). An SSLC cannot be attached to a selector channel (30030).

Each SSLC board accommodates one synchronous line, which is configured as a point-to-point contention station or as a a control station of a multipoint line.

Physically, the SSLC line can either be nardwired or be connected via a modem. If a modem is used, the line may be either switched or non-switched, and the data transfer rate is limited by the modem. If the SSLC line is hardwired, the maximum data transfer rate is 19200 baud for cables less than 1000 feet in length.

The SSLC can operate on either a half-duplex or a full-duplex line.

.6 HP 30010A (or 30020A) Single Channel Communications Controller (SCCP)

One HP 30010A SCCP board occupies two slots on the multiplex channel (30035) on the HP 3000 Series II; an SCCP cannot be attached to a

selector channel (30030). The HP 30020A SCCP board is compatible with the HP Interface Bus (HPIB), used on the Toothpick/Amigo systems.

Each SCCP board accommodates one synchronous or asynchronous line, which may be configured as a BSC point-to-point contention station, a control station of a multipoint line, a conversational BSC station, or as a network DTE or DCE in an HPDLC I environment.

Physically, the SCCP line can either be hardwired or be connected via a modem. If a modem is used, the line may either be switched or non-switched, and the data transfer rate is limited by the modem, with a maximum speed of 56 Kbaud. If the SCCP line is hardwired, the data transfer rate is 56 Kbaud for cables less than 4000 feet in length.

The SCCP board can operate on either a half-duplex or full-duplex line.

MAXIMUM DATA TRANSFER RATES

Component	Baud
SCCP (xxxxx) (8 bits/byte) (10 bits/byte) (11 bits/byte)	******* *******
HSI (30360)(10 bits/byte) Hardwired	2,500,000
SSLC (30055)(8 bits/byte)	
Bell 201A Modem (SW)	2,000
Bell 201B Modem (NSW)	2,400
Bell 208A Modem (NSW)	4,800
Bell 208B Modem (SW)	4,800
Bell 209A Modem (NSW)	9,600

2.7 DRIVERS AND LINE PROTOCOLS

Each type of line supported by CS requires a different driver. The hardware board and the line protocol dictate how the driver is written.

Lines using either the SSLC or the HSI will employ the BSC protocol, which is widely used by the manufacturers of other CPUs and terminals. Since the BSC protocol may also be used with the SCCP board, the user's application program can be relatively independent of whether a line is SSLC, HSI, or SCCP.

A full enumeration of the protocol drivers provided by CS are:

A BSC driver for use with contention SSLC lines,
A BSC driver for HSI contention lines,
A BSC driver for SCCP contention lines,
A driver to enable the 3000 to control a multipoint SSLC line consisting of BSC tributaries such as the HP 2645.
A driver to enable the 3000 to emulate a HASP Workstation (MRJE) using the SSLC or SCCP.
A driver to enable the 3000 to emulate an IBM 3270 workstation using the SCCP.
A driver to enable the 3000 to operate as a DTE or DCE in a full-duplex mode, using the HPDLC I protocol on SCCP.

2.8 SUPPORTED LINE CONFIGURATIONS

For all CS line types (SSLC, HSI, and SCCP), the following line configurations will be supported:

1) non-switched, point-to-point lines

NOTE: Each of the four HSI ports may be physically connected to a different remote station, but only one may be selected by any given COPEN. In other words, no more than one HSI port should be configured per logical device number for point-to-point operation.

For the SSLC and SCCP line types the following line configurations will be supported:

1) switched, point-to-point lines

2) non-switched, multipoint lines in which the 3000 is the control station.

3) switched, multipoint lines in which the 3000 is the control station.

The SCCP board will also support the following line configurations:

1) non-switched, multipoint lines in which the 3000 is a tributary (secondary) station

2) switched, multipoint lines in which the 3000 is a tributary (secondary) station

3) switched lines in which the 3000 is acting as a DTE in a TWS environment

4) non-switched lines in which the 3000 is acting as a DTE in a TWS environment

5) switched lines in which the 3000 is actin as a DCE in a TWS environment

6) non-switched lines in which the 3000 is acting as a DCE in a TWs environment

3.1 MISCELLANEOUS COMMAND EXTENSIONS

The following standard MPE commands have been extended to accommodate CS:

:NEWACCT
:ALTACCT
:NEWUSER
:ALTUSER

The user attribute "CS" may be specified in the <capabilitylist> of the above commands.

3.2 USER COMMANDS

Two user commands are defined for CS, namely :CLINE and :CRESET. A user must have CS capability to use these commands; a capability error will result otherwise.

3.3 :CLINE Command

The purpose of the :CLINE command is to:

1) Enable programs to reference lines without knowledge of their actual logical device numbers.

2) Enable a user to make detailed line specifications at run time, as overrides to any corresponding specifications that may have been made in COPEN intrinsic calls at compile time.

Within a program, a line may be referenced indirectly in a COPEN intrinsic call via a "formal line designator" which is a name associated at run time with a specific :CLINE command.

For user pre-defined lines:

```
:CLINE formaldesignator =*formaldesignator1
```

For new lines:

```
:CLINE formaldesignator
  [;DEV=device]
  [;BUF=[numbuffers][,buffsize]]
                n
                BSC
  [;PROTO=      MRJE    ]
                HPDLCI

               n
               SENSE
  [;CODE=      EBCDIC   ]
               ASCII

               HIGH
  [;DUAL=             ]
               LOW

               R
               W
  [;DIAL=      RW   ]
               NO

                n
  [;LMODE=      PRI   ]
                SEC
                MPCNT
                MPSEC
                DTE
                DCE

  [;DRIVER=drivername]
  [;DOPTIONS=n]
  [;SPEED= inspeed[,outspeed]]
  [;LOCID=idsequence]
  [;REMID=idsequence[,idsequence]...]
  [;PHLIST=phonenumber[,phonenumber]...]
```

```
[;MISC=c,i[[,i]...],[c,i[[,i]...]],...]

[;DOWNFILE=filename]      ***** not implemented yet ****

 ;  TRACE [=[ALL][,[mask][,[numentries][,WRAP] ]]  ]
    NOTRACE

 ;  ID
    NOID

 ;  TIMEOUTS
    NOTIMEOUTS

[;SUPLIST=  general suplist  ]        ** not implemented **

[;POLLIST= pollentry [,pollentry]...]  * not implemented *
```

where

* id sequence= A "string" / E / O (d[,d]...) / H

* string - string of characters with "" representing ".

* d - octal (or hexadecimal) specification of 1 byte (0<=d<= 377 / FF)

(A=ASCII, E=EBCDIC, O=octal, H=hexadecimal)

* "n" is an integer whose value is the encoded representation of the corresponding field in the AOPTIONS, COPTIONS, or DOPTIONS parameter of COPEN.

* c, i - integers where

c is an integer defining the parameter type
i is an integer specifying the value of the parameter.

See the COPEN intrinsic specification for the permissible values of c.

* Filename is the name of the file which contains the default protocol driver information and the download records for the SCCP protocol driver. This parameter is ignored for SSLC and HSI devices.

* "ALL" will generate trace records for all CREAD, CWRITE, and CCONTROL requests. If "ALL" is not specified, then trace records will be written only when a CREAD, CWRITE or CCONTROL completes with a transmission error.

* Mask may be a decimal integer or an octal integer preceded by a percent sign.

* WRAP specifies that succeeding trace entries may be written over previous entries if there is no more space in the trace record. The default is NOWRAP.

* Numentries must be a decimal integer.

* general suplist=mplinetype [,LSEL=ss] [,GRP=gdesc] [,GRP=gdesc] ...

* mplinetype is a positive integer specifying the type of stations on a multipoint line. If no other field appears in the SUPLIST parameter, then "self-configuring" mode is assumed (see description of byte 0 of the SUPLIST parameter in the COPEN intrinsic for a more detailed explanation).

  LSEL is the line's line selection sequence.

  Gdesc describes the attributes of a multipoint terminal group.

```
                ps                  ps
gdesc=[ID= ps,ss ,] [GEN= ps,ss ,] STA=sdesc [,STA=sdesc]...
             ,ss                 ,ss
```

  ID is the group polling/selecting ID.

  GEN is the general poll/selection ID.

  Ps is the polling sequence ID.

  Ss is the selection sequence ID.

```
                A  "string"
ps              E
          =
ss              O  (d[,d]...)
                H
```

  Sdesc describes the attributes of a station attached to a multipoint terminal group.

```
sdesc=[type], [buffsize], cdesc [,cdesc]...
```

Type is an integer whose value may be from 0 to 255.

Buffsize is the station's output blocksize. If this parameter is omitted, then the line's buffsize will be used.

Cdesc describes the polling and selection IDs of a component attached to a multipoint tributary station.

```
cdesc=[ps][,ss]]
```

Note that CS assigns successive logical group numbers to each group defined in the SUPLIST parameter, beginning with one for the first group defined. In a similar manner, logical station numbers are assigned to each station specified. Logical station numbers are unique within the line. Logical component numbers are successively assigned to each component on a station, beginning with zero for the first component of a station. Logical component numbers are unique only within a particular station.

Pollentry is either a terminal group or a station/component.

```
            G=logical group number
Pollentry=  logical station number
            logical station number/logical component number
```

The various :CLINE parameters correspond to COPEN intrinsic call parameters as follows:

:CLINE Parameter	COPEN Parameter
formaldesignator	contents of array FORMALDESIG
device	contents of array DEVICE
numbuffers	NUMBUFFERS
buffsize	BUFFSIZE
PROTO=	
BSC	AOPTIONS (0:8) = 1
MRJE	= 2
HPDLCI	= 3
ID	COPTIONS (1:1) = 0
NOID	= 1
TIMEOUTS	COPTIONS (0:1) = 0
NOTIMEOUTS	COPTIONS (1:1) = 1
CODE=	
SENSE	COPTIONS (10:6) = 1
ASCII	= 2
EBCDIC	= 3
DUAL=	
LOW	COPTIONS (4:2) = 1
HIGH	= 2
DIAL=	
W	AOPTIONS (12:2) = 0
R	= 1
RW	= 2
NO	= 3
LMODE=	
PRI	COPTIONS (6:4) = 1
SEC	= 2
MPCNT	= 3
MPSEC	= 4
DTE	= 5
DCE	= 6

drivername	contents of array DRIVERNAME
DOPTIONS=n	DOPTIONS
TRACE=...	CTRACEINFO
inspeed	INSPEED
outspeed	OUTSPEED
LOCID=idsequence	local id sequence in array IDLIST
REMID=idsequence,...	remote id sequences in array IDLIST
PHLIST=phonenumber,...	phonenumbers in array PHONELIST
MISC=c,i,c,i,...	c: MISCARRAY parameter type code i: parameter value
DOWNFILE=filename	Contents of the array DWNLDFILE.
SUPLIST=...	SUPLIST parameter
POLLIST=...	POLLIST parameter

3.4 :CRESET Command

This command is the CS analog to the file system :RESET command. It allows the user to nullify any previous :CLINE command referencing the formal line designator. The format of the :CRESET command is

```
            formaldesignator
:CRESET
            @
```

formaldesignator - The formal line designator to be reset.

@ - An indication that the formal line designators referenced in all prior :CLINE commands in the job/session are to be reset.

(Either the formaldesignator or @ must be entered.)

3.5 USER MESSAGES

The :CLINE command may cause the following CS-specific errors:

Error #	Message
800	CLINE EQUATION TABLE FULL
801	BACK CLINE REFERENCE NOT FOUND
802	TOO MANY BACK CLINE REFERENCES
803	INVALID CLINE DESIGNATOR
805	NAME MISSING
806	EMBEDDED SPECIALS IN PROPER NAME
807	NAME BEGINS WITH NUMERIC
808	NAME LONGER THAN 8 CHARACTERS
809	EQUAL SIGN EXPECTED
810	VALUE NOT OPTIONAL
812	BINARY FAILED ON INTEGER
813	INTEGER EXCEEDS 377
814	COLON EXPECTED
815	KEYWORD EXPECTED
816	DEVICE NAME EXCEEDS 8 CHARACTERS
817	REDUNDANT KEYWORD
818	INTEGER EXCEEDS 63

819	INTEGER EXCEEDS 15
820	DRIVER NAME LONGER THAN 8 CHARACTERS
821	INTEGER EXCEEDS 127
822	LINE NAME REQUIRED
823	ACTUAL LINE DESIGNATOR REQUIRED
824	BACK REFERENCE WITH PARAMETERS IS ILLEGAL

The following warning message may also appear in conjunction with the :CRESET command:

804	CLINE EQUATION NOT FOUND
401	CRESET PARAMETER ERROR

4.1 GENERAL DESCRIPTION

Within a user's program, the accessing of and communication with remote devices is accomplished through CS intrinsic calls. The remote/line relationship is somewhat analogous to the file/device relationship existing in the file system. The line is the primary link between the user and CS; each line referenced is first opened through the COPEN intrinsic call. Then, other operations such as reading, writing, and line control can be performed with other line intrinsic calls. The line is closed through the CCLOSE intrinsic call, issued by the user's process or by MPE/3000 when the user's process terminates.

In the COPEN intrinsic call, the user references a particular line by its formal line designator. When the COPEN intrinsic is executed, it returns to the user's process a line number by which the system uniquely identifies the line. This line number is also unique with respect to file numbers as managed by the file system. The line number, rather than the line designator, is used by subsequent intrinsics in referencing the line.

The condition codes returned to the user's program by the CS intrinsics have the following general meanings. The specitic meanings, of course, depend on the intrinsic:

Condition Code	Meaning
CCE	The function requested by the intrinsic call was completed successfully.
CCG	Variable, intrinsic-dependent.
CCL	The function requested by the intrinsic call terminated with an error or an abnormal condition; corrective action may, in some cases, be taken. (By issuing a CCHECK intrinsic call, the user can have a more detailed error description returned to his process.)

It should be noted that DB must point to the user's process' stack for all COPEN and CGETINFO intrinsic calls.

For the intrinsic descriptions to follow, all parameters are required unless otherwise noted; all bit fields marked for future use should be set to zero by the user.

4. OPENING LINES

The COPEN intrinsic opens a data communications line. Before a user's process can communicate with a remote device, the process must initiate access to the line, of which the remote device is a part, by opening it through a COPEN intrinsic call. The intrinsic returns to the user's process a line number which is used to identify the line in subsequent CS intrinsic calls.

If the line is opened successfully (i.e., the CCE condition code is returned), the line number returned is a positive integer ranging from 1 to 255. If the line cannot be opened (i.e., the CCL condition code is returned), the line number returned is zero.

In opening a line, COPEN establishes a communication link between the line and the user's program by

1. Allocating exclusively to the user's process the desired line. Once allocated, the line may not be opened or accessed by any other process in the user's job/session or by any other program until the opening process has issued a close request (CCLOSE intrinsic call). Thus, exclusive access to the line is always assumed. The process causing the open must not attempt to open the line more than once unless corresponding close requests have been issued.

2. Constructing the control blocks required by MPE/3000 for this particular access to the line. The information in these blocks is derived by merging specifications from the following sources, listed below in descending order of precedence:

   a. The non-overridable information contained in CONFIGURATOR - generated CS tables. This information overrides that from any other source.

   b. The parameter list of a previous :CLINE command referencing the same formal line designator name in the COPEN call, if such a command was issued in this job or session. This information overrides that from the three sources below.

   c. The parameter list of the COPEN call.

   d. The overridable information contained in CONFIGURATOR- generated CS tables.

When information in one of these sources conflicts with that in another, preempting takes place according to the order of precedence shown above. To determine the specifications actually taking effect, the user can call the CGETINFO intrinsic described later in this section.

The COPEN intrinsic declaration is as follows:

```
INTEGER PROCEDURE COPEN (FORMALDESIG, DEVICE, COPTIONS, AOPTIONS,
   DOPTIONS, NUMBUFFERS, BUFFSIZE, IDLIST, SUPLIST, POLLIST,
   DWNLDFILE, YYY, PHONELIST, INSPEED, OUTSPEED, MISCARRAY,
   DRIVERNAME, CTRACEINFO);
VALUE COPTIONS, AOPTIONS, DOPTIONS, NUMBUFFERS, BUFFSIZE, YYY,
   INSPEED, OUTSPEED, CTRACEINFO;
BYTE ARRAY FORMALDESIG, DEVICE, IDLIST, SUPLIST, PHONELIST, DWNLDFILE,
   DRIVERNAME;
LOGICAL COPTIONS, AOPTIONS, DOPTIONS, CTRACEINFO;
INTEGER NUMBUFFERS, BUFFSIZE, YYY;
LOGICAL ARRAY POLLIST, MISCARRAY;
DOUBLE INSPEED, OUTSPEED;
OPTION VARIABLE, EXTERNAL;
```

This intrinsic returns (as the value of COPEN) an integer line number used to identify the opened line in other CS intrinsic calls. If the line was not opened successfully, a zero is returned. It should be noted that within a single process, CS line numbers and file system file numbers are unique.

The condition codes for the COPEN intrinsic are as follows:

CCE - Request granted; the line was opened.

CCG - Not returned (reserved for future use).

CCL - Request denied.

The COPEN request will fail if the user does not have CS capability.

The COPEN intrinsic parameters specify the elements shown below. Either FORMALDESIG or DEVICE must be present (they may both be present); all other parameters are optional.

FORMALDESIG a byte array containing a string of ASCII characters, interpreted as a formal line designator. This string must begin with a letter, contain alphanumeric characters, and terminate with any non-alphanumeric character. Excluding the delimiter, the string must be eight characters or less in length.

If present, this string is used to search for the corresponding FORMALDESIG in :CLINE commands. If no corresponding :CLINE command has been supplied, then the DEVICE parameter is used to determine the device actually opened. If a corresponding :CLINE command is found, its DEV parameter (if specified) overrides the COPEN DEVICE parameter. This parameter may be omitted if the DEVICE

parameter is present.

If both FORMALDESIG and DEVICE are present, then FORMALDESIG is first used to locate a corresponding :CLINE command, which may override the COPEN DEVICE parameter.

DEVICE — A byte array containing a string of ASCII characters terminating with any non-alphanumeric character. The string may specify class name (up to eight alphanumeric characters beginning with a letter) or a logical device number (up to three numeric characters).

This parameter may be omitted if FORMALDESIG is present. However, if it is omitted and no :CLINE command has been supplied, the COPEN will fail.

COPTIONS — A word describing the communications options associated with the line. This parameter may be omitted, in which case the default value for each field is zero. The fields included are:

- Transmission code
- Mode
- Speed select
- CLINE override
- CS trace
- ID sequence verification
- Inhibit timeouts

These fields are described on page 4-10.

AOPTIONS — A word describing the access options associated with the line. This parameter may be omitted, in which case the default value for each field is zero. The fields included are:

- Wait mode
- CLINE buffer override
- Dial type
- Inhibit operator console error messages
- Protocol

These fields are described on page 4-12.

DOPTIONS — A word describing driver-related options, with the default value for each field being zero.
The format of DOPTIONS for the BSC SSLC driver and the BSC SCCP driver is described on page 4-13;
the format for the HSI driver is on page 4-15;

the format for the HPDLC I SCCP driver is on page 4-15.

NUMBUFFERS This integer specifies the total number of buffers to be assigned to the line or, if buffering is not desired, the number of outstanding read and write requests to be queued. If this parameter is omitted, one buffer will be used for lines not using HPDLC I protocol, and two buffers will be used for lines using HPDLC I protocol.

A positive value specifies the total number of buffers to be used. The maximum number of CS buffers allowed for a line using the SSLC or the HSI driver is two; for a line using the SCCP board it is fourteen, thus allowing read and write requests to be queued. For the SCCP board, the maximum number of requests of either type to be outstanding is seven, with the maximum total number being fourteen.

A negative value specifies that no buffers are to be assigned, but that read and write requests may be queued. The depth of the queue is specified by the absolute value of NUMBUFFERS. The maximum number of read requests or write requests to be queued is seven; the maximum number of total requests to be queued is fourteen.

A value of zero specifies that no buffering and no queueing is to be done.

BUFFSIZE This integer specifies the size of each buffer. A positive number denotes words; a negative number denotes bytes.

This parameter represents the maximum data transfer size. There exist three related default values in the CS system:

(1) a system-defined upper bound of 4096 words on any transfer
(2) an HPDLC I protocol-defined upper bound of 512 words on any transfer
(3) a configured, preferred buffer size for the line.

If BUFFSIZE exceeds the upper bound, the upper bound will be used in place of BUFFSIZE. If BUFFSIZE is not specified, the configured preferred buffer size will be used as a default value.

For a line operating in multipoint control mode, a non-zero BUFFSIZE specifies the maximum block size for a single transmission. A request to send a block of data

that exceeds the BUFFSIZE parameter will be reformatted into a series of requests with data blocks having length not exceeding BUFFSIZE. If data is received in blocks, an error will occur unless BUFFSIZE is non-zero, which permits the reception of data in blocks. (Specification that data may be read in blocks may also be made using the block size portion of the station field in the SUPLIST parameter.)

IDLIST — A byte array containing ID sequences for both the local and the remote stations. At most 255 ID sequences may be specified. Note: If no remote ID sequences are specified, then any received ID sequence is ignored. This array is described on page 4-16.

**** This parameter is inapplicable to, and ignored by, HPDLC I.

SUPLIST — This is a byte array describing the maximum set of groups, stations, and components which will be recognized on a multipoint line. For a line operating in multipoint control mode, this list describes the stations which will bepolled/selected by the 3000. For a line operating in multipoint secondary mode, this list describes the station or stations which will be simulated by the 3000 when polled/selected by the control station.

This array is used to generate a default POLLIST if the POLLIST parameter is omitted. This array is described on page 4-18.

**** This parameter is valid only for lines using multipoint control or multipoint secondary mode and is ignored for all other line modes.

POLLIST — A word array which defines the poll list for a multipoint line. When the local station is the control station, this list specifies which stations shall be polled and their order of polling. When the local station is a secondary station, this list specifies the current set of groups and stations to be simulated by the 3000. (See description of the POLLIST parameter of the CPOLLIST intrinsic for the format of the pollist.) CS will use this list whenever the STATION parameter of a CREAD is absent. If this parameter is omitted then a default list will be generated consisting of component zero of each station on the line. The default list will be optimized to the extent that a logical group number entry will be used to poll terminal groups which support general poll.

**** This parameter is valid only for lines operating in multipoint control mode and is ignored for all other lines.

DWNLDFILE — A byte array containing a string of ASCII characters, interpreted as the formal file designator of the file containing the default protocol driver information and the download records for the SCCP protocol driver. This string must begin with a letter, contain alphanumeric characters, slashes, or periods and terminate with any non-alphanumeric character except a slash or a period. If this parameter is omitted, then the default d ownload file name of SCCPLD01.PUB.SYS" is used. This parameter is ignored for SSLC and HSI devices.

YYY — (Reserved for future use.)

PHONELIST — A byte array specifying one or more remote phone numbers for a switched line, and is described on page 4-18. This array is ignored for non-switched lines.

INSPEED — A double-word integer specifying the line input speed in characters per second. If the line's hardware is such that its output and input speeds are always the same, then INSPEED will be used to specify both speeds.

OUTSPEED — A double-word integer specifying the line output speed in characters per second.

MISCARRAY — A logical array containing miscellaneous CS information. The fields included are:

Receive timout
Local timeout
Connect timeout
Response timeout
Line bid timeout
Number of error retries
Clear-to-send delay
Data-set-ready delay
Transmission mode
MMSTAT trace facility
Poll loop delay
Number of poll repeats
Poll entry delay

This array is described on page 4-20.

DRIVERNAME — A byte array containing a string of ASCII alphanumeric characters which are interpreted as the name of the

desired driver for the line. The name consists of up to eight characters, beginning with a letter, and followed by any alphanumeric characters, and terminated by a non-alphanumeric character. (The delimiter is not counted as one of the eight characters.)

This parameter may be omitted, in which case the default driver for the line will be used. When specified, the driver must be one which was included at configuration time in the set of available CS drivers.

The protocol driver must be compatible with the line configuration with respect to the following:

1. linetype (see CGETINFO).
2. local mode COPTIONS.(6:4).
3. protocol AOPTIONS.(0:8).
4. transmission code. COPTIONS.(10:6).

The driver used for the SCCP board on the 3000 is "IOSCCPX"; on Series 33 it is "TPDUMCS". If another driver name beginning with the letters "CS" is specified for the SCCP, the specified name is used to indicate the protocol driver to download, and not the mainframe driver.

CTRACEINFO A word used to control the CS trace facility. This parameter is ignored if the CS trace facility bit of the COPTIONS word is not set. See page 4-23 for the format of CTRACINFO.

COPTIONS

The fields of COPTIONS are defined as follows:

(0:1) Inhibit timeouts

= 0 allow timeouts
= 1 disable all timeouts
**** (This field does not apply to and is ignored by HPDLC I.)

(1:1) ID Sequence Verification

= 0 allow the use of ID sequences (both user-supplied and configured defaults)
= 1 inhibit the use of ID sequences. Any user-supplied or configured default ID sequences will be ignored (applies to both local and remote ID sequences).
**** (This field does not apply to, and is ignored by, HPDLC I.)

(2:1) CS trace

= 0 do not invoke CS trace facility
= 1 invoke CS trace facility.
(Refer to CTRACINFO parameter on page 4-21)

(3:1) :CLINE override

= 0 allow CLINE command override
= 1 prohibit CLINE command override.

(4:2) Speed Select (European modems only)

= 0 use configured default setting
= 1 set speed to low speed
= 2 set speed to high speed
= 3 reserved for future use. Specification of this value will cause a COPEN error.

(6:4) Local mode ("local" means the station at your end of the line)

= 0 use configured default setting
= 1 local is a primary contention station
= 2 local is a secondary contention station
= 3 local is a control station on a multipoint line.
= 4 local is a secondary station on a multipoint line.
= 5 local is a HPDLC I system acting as DTE
= 6 local is a HPDLC I system acting as DCE
= 7-15 reserved for HP use

A COPEN error will result if local mode is not compatible with either COPEN parameters or configured line values.

(10:6) Transmission code

= 0 use configured default setting
= 1 use automatic code sensing feature of driver
= 2 ASCII
= 3 EBCDIC
= 4-63 reserved for HP use
**** (Data is transparent to HPDLC I, and this field is ignored when using the HPDLC I protocol.)

AOPTIONS

The fields of AOPTIONS are defined as follows:

(0:8) Protocol

= 0 use configured default protocol
= 1 use BSC protocol
= 2 use MRJE protocol
= 3 use HPDLC I protocol
= 4-255 reserved for HP use

A COPEN error will result if protocol is not compatible with configured line specifications or driver capabilities.

(8:3) (Reserved for future use.)

(11:1) Inhibit console operator error message

= 0 allow CS to print hardware error messages at the operator console
= 1 inhibit CS from printing hardware error messages at the operator console

(12:2) Dial type

= 0 dial on write connect; answer on read connect
= 1 answer on write connect; dial on read connect
= 2 dial on write connect; dial on read connect
= 3 answer on write connect; answer on read connect.

(14:1) :CLINE buffer override

= 0 allow :CLINE override for NUMBUFFERS and BUFFSIZE
= 1 disallow buffer information override.

(15:1) Wait mode

= 0 perform all I/O using NCIO.
= 1 perform all I/O using CIO.

Note: Only users executing in privileged mode may open a line with CIO and no buffering.

DOPTIONS

The meaning of DOPTIONS is protocol driver-dependent.

For the SSLC drivers and the BSC SCCP drivers, the format of DOPTIONS is as follows:

(0:1) Reserved for future use

(1:1) Delay sequence wait
0 = Wait on received WACK/TTD sequences.
1 = Do not wait on received WACK/TTD sequences.

(2:1) Poll termination sequence
0 = Before switching between stations, an RVI is transmitted to return the line to control mode.
1 = Before switching between stations, an EOT is transmitted to return the line to control mode.

(3:1) Control state listen mode
0 = While in control state and between user requests the driver will listen for any control sequences from the remote. Receipt of a line bid will cause the line to be placed into text state.
1 = While in control state and between user requests the driver will ignore any control sequences from the remote.

(4:2) Ending sequence:
0 = Use BSC default (NSW=send EOT;
SW=send DLE EOT)
1 = send DLE EOT
2 = send EOT

(6:1) 0 = the remote will not send leading graphics
1 = expect leading graphics from the remote

(7:1) Value of USASCII block check character (bcc)
0 = VRC/LRC (non-transparent mode or transparent with header)
CRC-16 (transparent mode with no header)
1 = VRC/CRC-16 (non-transparent mode)
CRC-16 (transparent mode)

(8:1) 0 = automatic generation of WACK
1 = no WACK will be sent

(9:1) 0 = automatic generation of TTD
1 = no TTD will be sent

(10:1) ITB Sequences
0 = Do not expect to receive ITB sequences from the remote station
Note: If an ITB sequence is received, the driver will require a retransmission to properly receive the message. The driver then sets this bit to a one.
1 = Expect ITB sequence from the remote station.

(11:2) Message Format Word (MFW). Refer to Section 4.13 for a description of the MFW.
0 = MFW will not be placed into received text or expected in sent text. CS will use an implicit MFW of 000000 for sent text.
1 = MFW will not be placed into received text or expected in sent text. CS will use an implicit MFW of 100000 for sent text.
2 = MFW will be placed into received and expected in sent text.
3 = reserved for future use.

(13:1) Reserved for future use.

(14:2) Number of leading SYNs
0 = send four leading SYNs
1 = send eight leading SYNs
2 = send twelve leading SYNs
3 = send sixteen leading SYNs

For the HSI driver, the format of DOPTIONS is as follows:

(0:1) Reserved for future use

(1:1) Delay sequence wait

(2:1) Ignored.

(3:1) Control state listen mode
0 = While in control state and between user requests the driver will listen for any control sequences from the remote. Receipt of a line bid will cause the line to be placed into text state.
1 = While in control state and between user requests the driver will ignore any control sequences from the remote.

(4:2) Ending sequence:
0 = Use BSC default (NSW=send EOT;
SW=send DLE EOT)
1 = send DLE EOT
2 = send EOT

(7:1) ignored

(8:1) Automatic generation of WACK sequences:
0 = enable
1 = disable

(9:1) Automatic generation of TTD sequences:
0 = enable
1 = disable

(10:3) Ignored

(13:1) Reserved for future use

(14:2) Ignored

For the HPDLC I SC3 driver, the format of DOPTIONS is as follows:

(0:8) Reserved for future use

(8:8) Maximum number of outstanding frames. This is parameter K in the HPDLC I protocol. Valid values are one through seven. The default is seven.

IDLIST

IDLIST is a byte array with the following format:

byte	0	total number of ID sequences (including the zero-length IDs)
	1	length of local ID sequence in bytes (may be zero)
	2-n	local ID sequence (null if length =0)
	n+1	length of remote ID sequence in bytes (may be zero)
	(n+2)-p	remote ID sequence (null if length =0).

Repeat the last two fields for each remote ID.

The length of an ID sequence may vary from zero (meaning a null ID sequence) to sixteen bytes. If a remote station is not capable of sending an ID sequence, then the local ID sequence length should be zero; if none of the remote stations can transmit an ID sequence, the remote ID sequence should be zero.

PHONELIST

Phonelist is a byte array with the following format:

byte	0	number of phone numbers (binary)
	1	length of first phone number (binary)
	2-n	first phone number (ASCII numeric with embedded "dashes")

Each successive phone number is specified by repeating the last two fields. A phone number consists of from 1 to 20 characters (ASCII numerics with embedded "dashes").

This parameter is ignored if the line is non-switched. When operating in a switched full-duplex environment, there may be two phones. If there are two phones, CS will access the phone list in pairs, i.e., two dial messages will be issued when necessary. The specification of two phones is done at system configuration time.

LIST

LIST is a byte array with the following format:

ield one = Line field. This must be the first field of the SUPLIST.

Byte 0 — Multipoint Line Type
(This byte reserved for future use.)

Byte 1 — Line Descriptor
(0:5) Reserved for future use.
(5:2) ID Verification
0 = No ID verification will be performed.
1 = Initiate the sending of ID sequences and, when the remote's ID sequence has been successfully verified, send an EOT sequence.
2 = Allow the remote to initiate the sending of ID sequences.
3 = This value is undefined. CS will reset the field to a zero.
(7:1) Line selection
0 = The configuration does not support line selection.
1 = The configuration has the line selection facility.

Byte 2 — This byte is ignored.

Byte 3 — Number of remote groups.

Byte 4 — Length of line selection sequence.
Note: This parameter and the next one should be present if and only if bit (7:1) of the line descriptor specifies line selection.

Byte 5-n — Line selection sequence (null if length=0).

ield two = Group field. Repeated for each group attached to the line. COPEN assigns successive logical group numbers to each group defined for the line, beginning with 1 for the first group specified.

Byte 0 — Group descriptor
(0:6) Reserved for future use.
(6:1) 0 = No general select facility
1 = This group has the general select capability
(7:1) 0 = No general poll facility
1 = This group has the general poll facility

Byte 1 — Number of stations in this group.

Byte 2 — Length of the group ID portion of the polling sequence. A value of zero is valid.

te 3-k — Group ID portion of the polling sequence (null if

length=0).

Byte k+1 Length of the group ID portion of the selection sequence. A value of zero is valid.

Byte (k+2)-m Group ID portion of the selection sequence (null if length=0).

Byte (m+1) Length of general poll portion of the polling sequence.
Note: This parameter and the next one should be present if and only if bit (7:1) of the group descriptor specifies general polling.

Byte (m+2)-n General poll sequence (null if length = 0).

Byte (n+1) Length of general selection portion of the selection sequence.
Note: This parameter and the next one should be present if and only if bit (6:1) of the group descriptor specifies general selection.

Byte (n+2)-p General selection sequence (null if length = 0).

Field three = Station field. Repeated for each station in the group. COPEN assigns successive logical station numbers to each station defined for the line, beginning with 1 for the first station specified. Note that a station number is unique for the line, not just for the station's group.

Byte 0 Station type
(0:1) = Ignored.
(1:1) 0 = The terminal's input blocksize value shall be the same as the line's BUFFSIZE value.
1 = The terminal's input blocksize value is contained in bytes 2-3 of this field.
(2:6) 0 = HP 2645 terminal
1-40 = Reserved for HP use.
41-63 = Reserved for user defined terminal types.

Byte 1 Number of components attached to this station.

Bytes 2-3 Integer value of the terminal's fixed-length block size. This number specifies a word length if positive, and a byte length if negative.
Note: This parameter should only be present if and only if bit (1:1) of the station descriptor is set to a one.

Field four = Component field. Repeated for each component of the station. COPEN assigns logical component numbers for each component defined for the station, beginning with zero for the first component specified in the station.

Byte 0 Length of component ID portion of the polling sequence. May be set to zero.

Byte 1-k Component's polling ID. This field is omitted if the

length is zero.

Byte (k+1) Length of the component ID portion of the selection sequence. May be set to zero.

Byte (k+2)-m Component's selection ID. This field is omitted if the length is zero.

MISCARRAY

MISCARRAY is a word array with the following format:

word 0	number of words of parameter information following this word.
1	parameter type
2-n	parameter

Repeat the last two fields for each parameter type to be specified.

Each parameter type will define the meaning and size of the parameter which follows it. Currently, the parameter associated with each parameter type consists of a single word quantity. The defined parameter types are

0 - receive timeout in seconds.*
(default=20)
**** (This value is inapplicable to, and ignored by, HPDLC I.)

1 - local timeout in seconds.*
(default=60)
**** (This value is inapplicable to, and ignored by, HPDLC I.)

2 - connect timeout in seconds.*
(default=900)

3 - response timeout.*
(default=3)
**** For the HPDLC I protocol this is parameter T1. Configured default values vary with the transmission speed, as specified below.

SPEED(cps)	HPDLC I RESPONSE TIMEOUT (sec)
5600	
5000	
4080	
1920	** to be defined **
960	
480	
240	

4 - line bid timeout.*
(default=60)
**** (This value is inapplicable to, and ignored by, HPDLC I.)

5 - number of error recovery retries by driver.
(Default=6)

6 - = 0, Clear-to-Send delay shall be the default (which is determined by the modem for the SSLC board).
<>0, Clear-to-send delay value in tenths of seconds.

IOn the SCCP, clear-to-send delay defines the amount of time the driver will await the expected clear-to-send change before deciding that the modem is broken. The default value is 300 msecs; a non-zero value will override this default.

7 - = 0, No time will be allowed for the Data Set Ready signal to stabilize.
= 1, Wait 100 milliseconds for the Data Set Ready signal to stabilize after it first goes true.

8 - = 0, Set the line's transmission mode to full-duplex.
= 1, Set the line's transmission mode to half-duplex.
**** (This value is inapplicable to, and ignored by, HPDLC I.)

9 - = 0, Disable MMSTAT trace facility.
<>0, Enable MMSTAT trace facility to trace the driver's state transitions. When using the SSLC or the HSI interfaces, MMSTAT is used to trace the interface driver's state transitions. When using the SC3 interface, MMSTAT is
*** yet to be defined ***
This facility is described in the MPE Memory Manager documentation.

**** The following three parameters are used only for lines in multipoint control mode, and are otherwise ignored.

10 - Poll loop delay. Parameter is the number of hundredths of seconds of delay between iterations through a wrap (circular) polling list.

11 - Poll repeat. Parameter specifies the maximum number of iterations through the polling list. A value of zero implies an unlimited number of passes through the list. A value other than one makes the list a wrap (circular) list. Polling is terminated when either of the following conditions occurs: a) a station responds to a poll or b) the specified number of passes through the polling list has been made with no station responding.

12 - Poll entry delay. Parameter is the delay in milliseconds between polling each entry in the polling list. If this

parameter is omitted then the default value will be taken from
a. the configuration dialogue for pre-configured lines
b. from the driver for all other cases.

* A value of zero will disable the timeout.

CTRACEINFO

CTRACEINFO is used to control the CS trace facility; its fields are as follows:

(8:8) Number of trace entries. This value represents the number of CS trace entries which will be accumulated, for each I/O request, before wrap around occurs. If this field is zero, a driver-dependent default will be used.

(2:6) CS trace mask.
This field is a mask which will indicate to the driver which types of information are to be traced. If this field is zero, a driver-dependent default will be used.

For the drivers using the Bisync protocol, the format of the trace mask is as follows:

```
bit 2 generate STN entries
    3 generate OPR and EDT entries  *
    4 generate RCT entries  *
    5 generate RTX entries  *
    6 generate SCT, POL, SEL entries  *
    7 generate STX entries  *
```

* default
NOTE: CMP entries are automatically generated.

For the HPDLC I SCCP driver, the format of the trace mask is as follows:

*** to be defined ****

(1:1) Trace entry fill type.
This value specifies the action to be taken when the wrap-around condition occurs. Setting the field to a one causes previous entries to be overlayed, whereas a value of zero will cause new entries to be discarded.

(0:1) CS trace type.
= 0 trace on transmission errors only
= 1 trace all CREAD, CWRITE, and CCONTROL requests.

See Chapter 6 for a detailed discussion of the CS TRACE facility.

4.3 CLOSING LINES

To terminate access to a line, a user's process must close the line through a CCLOSE intrinsic call. This intrinsic de-allocates the line and deletes the control blocks and CS supplied buffers (if any) through which communication with the remote devices attached to the line was accomplished. If the user does not issue CCLOSE calls for all lines opened by his process, the system will issue such calls when the process terminates. If this request is made to an SCCP device which reports a RAM system failure, a dump of the SCCP RAM is attempted via a CCONTROL 53 with a paramter of zero.

The CCLOSE intrinsic declaration is as follows:

```
PROCEDURE CCLOSE (LINENUMBER);
   VALUE LINENUMBER;
   INTEGER LINENUMBER;
   OPTION EXTERNAL;
```

The CCLOSE intrinsic closes a CS line. If necessary, it will prepare the line for disconnection and then disconnect it.

The condition codes for the CCLOSE intrinsic are as follows:

CCE - The line was closed successfully.

CCG - (This condition code is not returned.)

CCL - The request was rejected because <linenumber> is invalid.

The CCLOSE parameter is

LINENUMBER An integer specifying the line number of the line to be closed. This is a required parameter. If the line was opened with the CIO specification and an I/O operation is pending, CCLOSE will abort the I/O operation and close the line in an orderly fashion.

READING FROM REMOTE DEVICES

To read from a remote device, a user process issues a CREAD intrinsic call specifying the line to which the remote device is attached.

The CREAD intrinsic can perform two separate functions with respect to data transfer, the choice of which is line dependent. If the line to be accessed was opened with NCIO, CREAD will return control to the user process after completion of the data transfer from the remote station.

If the line was opened with CIO, CREAD will cause the read operation to be initiated and then immediately return control to the calling process. A subsequent IOWAIT intrinsic call will wait for the read operation to be completed and data transferred into the user's stack. All CREAD intrinsic calls to a line opened with CIO must be followed at some point by an IOWAIT intrinsic call. Reads may be issued until the CREAD is rejected due to a lack of CS line buffers before it is necessary to issue an IOWAIT. It is guaranteed that reads will complete in the order in which they were requested.

An IOWAIT intrinsic call must not be issued after a read request for a line opened with NCIO.

If buffering is not used, the data read will be transferred directly into the user's stack causing the stack to be frozen until completion of the read. The use of buffering does not require that the stack be frozen during the I/O transfer. Thus, more efficient memory management is obtained when buffering is used. This is particularly true for data communications I/O which is relatively slow and often depends on human interaction at the remote site.

If this request is made to an SCCP device which reports a RAM system failure, a dump of the SCCP RAM is attempted via a CCONTROL 53 with a parameter of zero.

Within each data transfer function, the user can specify driver related operations to allow him complete control of the communications link between his program and the remote device.

The CREAD intrinsic declaration is as follows:

```
INTEGER PROCEDURE CREAD (LINENUMBER, INBUF, INCOUNT, STATION,
                         READSTATION);
   VALUE LINENUMBER, INCOUNT, STATION;
   INTEGER LINENUMBER, INCOUNT;
   LOGICAL STATION, READSTATION;
   ARRAY INBUF;
   OPTION VARIABLE, EXTERNAL;
```

This intrinsic returns (as the value of CREAD) a non-zero integer specifying the length of the received message for a line using NCIO if no transmission errors incurred; else zero is returned.

A positive value is always returned for the message length. If the INCOUNT parameter was negative, the procedure value represents bytes; if INCOUNT was positive, the procedure value represents words.

The condition codes for the CREAD intrinsic are as follows:

a) AOPTIONS of COPEN specified CIO:

CCE - Read initiated.
CCG - (This condition code is not returned.)
CCL - Read not initiated, an error occurred.

NOTE: A CCL will be returned if CS has no line buffer for the request (i.e., the line buffers were all allocated for previous requests; an IOWAIT should be issued to check for completion of these requests).

b) AOPTIONS of COPEN specified NCIO:

CCE - Read completed.
CCG - End of Transmission sequence received.
*** This condition code is never returned when using HPDLC I
CCL - An error or an abnormal condition occurred.

The CREAD parameters are

LINENUMBER — an integer specifying the line number of the line. This parameter is always required.

INBUF — a word pointer specifying the DB-relative address of the user's input buffer. This parameter is not required and is ignored when performing CIO with buffering or if INCOUNT has been set to zero or omitted.

INCOUNT — an integer specifying the length of the data to be transferred. A positive value denotes words; a negative value denotes bytes. This value must not exceed the resultant BUFFSIZE as determined by COPEN. If this parameter is omitted, its value will be set to zero.

STATION — a word identifying the logical station or logical group

number of the remote tributary for a line in control mode.

Station format
(0:4) Set to zero.
(4:4) Logical component number
(8:8) Logical station number. A station number of zero is invalid.

Group format
(0:1) Set to one.
(1:7) Set to zero.
(8:8) Logical group number. A group of zero or 255 is invalid.

A value of minus one for STATION specifies the currently accessed station. If STATION is absent or zero, the poll list will be used to solicit input. If the STATION parameter is specified and nonzero, the poll list is circumvented. Note: Logical group numbers may only be used for groups that support general poll.

**** This parameter is meaningful only when the local station is a control station and is ignored for any other type of line mode.

READSTATION If the local station is a control station, READSTATION is a word to which is returned the logical station and component numbers of the responding tributary station. If STATION was specified, then READSTATION = STATION. The READSTATION parameter format is identical to the station format of the STATION parameter.

**** This parameter is meaningful only for lines operating in control mode, and is otherwise zero.

4.5
WRITING TO REMOTE DEVICES

To write to a remote device, a user process issues a CWRITE intrinsic call specifying the line to which the remote device is attached.

Under the BSC line protocol, a conversational mode of communication can exist. This mode allows a transmission to be immediately followed by a receive on the same remote device. A user process can request such a conversational write in the call to CWRITE by specifying an input buffer. If the line has been opened with CIO and buffering, then only the INCOUNT parameter need be specified.

If the line to be written to has been opened with the CIO specification, the user must eventually issue an IOWAIT call to suspend his process and await the completion of one of the pending requests. Writes may be issued until the CWRITE is rejected due to a lack of line buffers before it is necessary to issue an IOWAIT. It is guaranteed that outstanding writes will complete in the order in which they were requested.

An IOWAIT intrinsic call must not be issued after a write to a line using NCIO.

If buffering is not used, the user's stack will remain frozen in memory until completion of the write request. Thus, in general, buffering should be used to allow more efficient memory management. The user should not attempt to access the write buffer specified in a CWRITE intrinsic call to a line using CIO if buffering is not used; data in the buffer may be destroyed before completion of the request by IOWAIT.

If this request is made to an SCCP device which reports a RAM system failure, a dump of the SCCP RAM is attempted via a CCONTROL 53 with a parameter of zero.

The CWRITE intrinsic declaration is as follows:

```
INTEGER PROCEDURE CWRITE (LINENUMBER, OUTBUF, OUTCOUNT, INBUF,
                          INCOUNT, STATION);
  VALUE LINENUMBER, OUTCOUNT, INCOUNT, STATION;
  INTEGER LINENUMBER, OUTCOUNT, INCOUNT;
  LOGICAL STATION;
  ARRAY OUTBUF, INBUF;
  OPTION VARIABLE, EXTERNAL;
```

This intrinsic returns (as the value of CWRITE) a non-zero integer only for requests doing NCIO which did not incur transmission errors. This integer specifies the length of the sent message if the write request did not specify an input buffer; it specifies the length of the received message for write requests which do specify an input buffer.

A positive value is always returned for the message length. If the INCOUNT parameter was negative, the procedure value represents bytes; if INCOUNT was positive, the procedure value represents words.

The condition codes for the CWRITE intrinsic are as follows:

a) AOPTIONS of COPEN specified CIO:

CCE Write initiated.
CCG (This condition code is not returned.)
CCL Write not initiated, an error occurred.

NOTE: This condition code will be returned if CS has no line buffers for the request (i.e., the line buffers are all allocated in previous requests; an IOWAIT should be issued to check for completion of these requests.)

b) AOPTIONS of COPEN specified NCIO:

CCE Write completed.
CCG End of Transmission sequence received.
**** This condition code is never returned when using HPDLC I.
CCL An error or abnormal condition occurred.

The CWRITE parameters are

LINENUMBER an integer specifying the line number of the line. This parameter is always required.

OUTBUF a word pointer specifying the DB-relative address of the user's output buffer. This parameter is ignored if OUTCOUNT is omitted or set to zero.

OUTCOUNT an integer specifying the length of the data to be output. A positive value denotes words; a negative value denotes bytes. This value must not exceed the resultant BUFFSIZE as determined by COPEN. If this parameter is omitted, its value will be set to zero.

INBUF a word pointer specifying the DB-relative address of the user's input buffer. This parameter is not required and, hence ignored when performing CIO with buffering. It is ignored if INCOUNT is omitted or set to zero.

**** Conversational Bisync (i.e. MRJE HASP) requires both INBUF and INCOUNT.

INCOUNT an integer specifying the length of the data to be input. A positive value denotes words; a negative value denotes bytes. This value must not exceed the resultant BUFFSIZE as determined by COPEN. If this parameter is omitted, its value will be set to zero.

STATION a word identifying the logical station or logical grou

number of the remote tributary for a line in control mode.

Station format
(0:4) Set to zero.
(4:4) Logical component number
(8:8) Logical station number. A station number of zero is invalid.

Group format
(0:1) Set to one.
(1:7) Set to zero.
(8:8) Logical group number. A group number of 0 or 255 is invalid.

A value of minus one for STATION specifies the currently accessed station. If STATION is absent or zero then all stations on the line are specified (line select).
Notes: 1) Line select may only be used for lines that support line select.
2) Logical group numbers may only be used for groups that support group select.

**** This parameter is meaningful only when the local station is a supervised control station and is ignored for any other type of line mode.

4.6 COMPLETION OF CONCURRENT I/O REQUESTS

If a CS line has been opened or CCONTROLed with the CIO specification, i.e., AOPTIONS.(15:1)=1, all read and write requests must be followed by an IOWAIT intrinsic call. This call initiates completion operations for the associated I/O request including data transfer into the user's stack if necessary. Multiple reads and writes to the same line may be issued before the IOWAIT call is made. This allows for maximum line utilization as well as I/O and processing overlap.

If this request is made to an SCCP device which reports a RAM system failure, a dump of the SCCP RAM is attempted via a CCONTROL 53 with a parameter of zero.

The IOWAIT intrinsic declaration is as follows:

```
INTEGER PROCEDURE IOWAIT (IONUMBER,INPUTBUFFER,RECORDLENGTH,CSINFO);
  VALUE IONUMBER;
  INTEGER IONUMBER, RECORDLENGTH;
  LOGICAL CSINFO;
  ARRAY INPUTBUFFER;
  OPTION VARIABLE, EXTERNAL;
```

This intrinsic returns (as the value of IOWAIT) an integer representing the line number for which the completion occurred. If no completion occurred, this number will be zero.

The IOWAIT intrinsic can return the following condition codes:

CCE - I/O completion occurred without errors (IOWAIT<>0).

CCG - End of Transmission sequence received for CS device (IOWAIT<>0).
**** This condition code is never returned when using HPDLC I.

CCL - Normal I/O completion did not occur, because of an error or abnormal completion.

1) no request(s) pending (IOWAIT=0)
2) parameter error (IOWAIT=0)
3) I/O error or abnormal condition (IOWAIT<>0)

The IOWAIT parameters are

IONUMBER An integer specifying a CS line number, an FS file

number, or zero. (NOTE: Within a single process, file numbers and line numbers are unambiguous.) If zero or not specified, IOWAIT will wait for the first I/O completion, without distinguishing between file system and CS. If IONUMBER is positive, then IOWAIT will wait for the completion of that particular line or file number.

INPUTBUFFER — A word pointer specifying the DB-relative address of the user's input buffer area to which the received message/record will be moved. This buffer area must be large enough to contain the input message/record. This parameter is required only if the COPEN specified buffering and the completed request was a read or a conversational write.

RECORDLENGTH — A word to which is returned a positive integer representing the length of the record/message received or transmitted. If the original request specified a byte count, the integer represents bytes; if the request specified words, the integer represents words. This parameter is required only for reads and conversational writes; it is ignored for all other writes.

CSINFO — A word to which is returned information about the completing request, with the following format:

(0:1) Group flag
- = 1 Logical group number is specified
- = 0 Logical station and component number is specified

(1:1) Type of completed request
- = 0 Read
- = 1 Write or Write Conversational

(4:4) Logical component number when local station is control station.
Zero otherwise

(8:8) Logical station number (if Group flag=0) or Logical group number (if Group flag=1) when local station is control station.
Else zero.

4.7 OBTAINING LINE INFORMATION

Once a user process opens a line, it can request information about that line from the CGETINFO intrinsic.

The CGETINFO intrinsic provides general information about the line as actually opened. The information returned includes 1) the parameters of the COPEN call as modified by :CLINE commands and CS (configured) default values, and 2) global information about the line such as its logical device number and line type.

The CGETINFO intrinsic declaration is as follows:

```
PROCEDURE CGETINFO (LINENUMBER, LDNUM, LINETYPE, COPTIONS, AOPTIONS,
                    DOPTIONS, NUMBUFFERS, BUFFSIZE, IDLIST,
                    IDLISTLEN, SUPLIST, SUPLISTLEN, POLLIST,
                    POLLISTLEN, DWNLDFILE, YYYYY, PHONELIST,
                    PHONELISTLEN, INSPEED, OUTSPEED, MISCARRAY,
                    MISCARRAYLEN, DRIVERNAME, CTRACEINFO);
  VALUE LINENUMBER, IDLISTLEN, SUPLISTLEN, POLLISTLEN, PHONELISTLEN,
                    MISCARRAYLEN;
  INTEGER LINENUMBER, NUMBUFFERS, BUFFSIZE, IDLISTLEN, SUPLISTLEN,
                    POLLISTLEN, YYYYY, PHONELISTLEN, MISCARRAYLEN;
  LOGICAL ARRAY POLLIST, MISCARRAY;
  LOGICAL LDNUM, LINETYPE, COPTIONS, AOPTIONS, DOPTIONS,
                    CTRACEINFO;
  DOUBLE INSPEED, OUTSPEED;
  BYTE ARRAY IDLIST, SUPLIST, DWNLDFILE, PHONELIST, DRIVERNAME;
  OPTION VARIABLE, EXTERNAL;
```

The condition codes for the CGETINFO intrinsic are as follows:

CCE - The request was granted.

CCG - The driver was unable to return the information requested.

CCL - The request was not granted because an error occurred.

All CGETINFO parameters are optional except for LINENUMBER. The parameters are defined as follows:

LINENUMBER — An integer specifying the line number of the line. This is a required parameter.

LDNUM — A word to which is returned the logical device number of the line.

LINETYPE — A word to which is returned three fields which define the type of line.

(0:4) device subtype
(4:6) device type
(10:6) a number which provides a more succinct description of the line:

1 = SSLC - SW
2 = SSLC - NSW
3 = SCCP - SW
4 = SCCP - NSW
5 = HSI - NSW

COPTIONS — A word to which is returned a bit string representing the communications options currently in effect. The format is the same as that of the COPTIONS parameter of COPEN.

AOPTIONS — A word to which is returned a bit string representing the access options currently in effect. The format is the same as that of the AOPTIONS parameter of COPEN.

DOPTIONS — A word to which is returned a bit string representing the driver options currently in effect. The format is the same as that of the DOPTIONS parameter of COPEN. Note that certain CCONTROL functions can change the value of DOPTIONS.

NUMBUFFERS — A word to which is returned an integer indicating the number of buffers currently assigned to the line if positive, or the depth of the request queue if negative. This is the same format as that of the NUMBUFFERS parameter of COPEN.

BUFFSIZE — A word to which is returned an integer indicating the size of each buffer assigned to the line. BUFFSIZE will be positive or negative in correspondence with the BUFFSIZE parameter specified in COPEN.

IDLIST — A byte array to which is returned an ID sequence list. The format is the same as that of the IDLIST parameter of COPEN. If IDLISTLEN is zero or omitted, then the entire list is returned to IDLIST; otherwise, up to IDLISTLEN bytes are returned to IDLIST. If no list was specified in COPEN or at configuration time, one byte will be returned (corresponding to the number of ID sequences) and its value will be zero.

IDLISTLEN — An integer specifying the length (in bytes) of the byte array IDLIST. If IDLISTLEN is zero or omitted, then it is assumed that IDLIST is sufficiently large to contain

the entire ID sequence list specified in COPEN or at configuration time. This parameter is ignored if IDLIST is omitted.

SUPLIST — A byte array to which is returned the list describing a supervised line. The format is the same as that of format #2 of the SUPLIST parameter of COPEN (exception - byte 2 is set to the total number of stations). If SUPLISTLEN is zero or omitted, then the entire list is returned to SUPLIST; otherwise, up to SUPLISTLEN bytes are returned to SUPLIST. If no list was specified by CLINE, COPEN, or at configuration time, then two bytes of zero are returned to SUPLIST.

SUPLISTLEN — An integer specifying the length (in bytes) of the byte array SUPLIST. If SUPLISTLEN is zero or omitted, then it is assumed that SUPLIST is sufficiently large to contain the entire list describing the supervised line (as specified by the SUPLIST parameter of COPEN or by the configurator dialog). This parameter is ignored if SUPLIST is omitted.

POLLIST — A word array to which is returned the polling list for the line. The format is the same as that of the POLLIST parameter of the CPOLLIST intrinsic.

If POLLISTLEN is zero or omitted, then the entire list is returned to POLLIST; otherwise, up to POLLISTLEN words are returned to POLLIST.

POLLISTLEN — An integer specifying the length (in words) of the word array POLLIST. If POLLISTLEN is zero or omitted, then it is assumed that POLLIST is sufficiently large to contain the entire polling list for the line. This parameter is ignored if POLLIST is omitted.

DWNLDFILE — A byte array to which is returned the formal file designator of the down load file. The maximum length of the file name is 28 bytes long. When the designator is returned, unused bytes will be filled with blanks. If there is no down load file then the whole array will be blank filled.

YYYYY — (Reserved for future use.)

PHONELIST — A byte array to which is returned the list of phone numbers for a switched line. The format is the same as that of the PHONELIST parameter of COPEN. If PHONELISTLEN is zero or omitted, then the entire list is returned to PHONELIST; otherwise, up to PHONELISTLEN bytes are returned to PHONELIST. If no list was specified by COPEN or via configurator dialog, then or

byte of zero will be returned to PHONELIST.

PHONELISTLEN An integer specifying the length (in bytes) of the byte array PHONELIST. If PHONELISTLEN is zero or omitted, then it is assumed that PHONELIST is sufficiently large to contain the entire phone list specified by COPEN or at configuration time. This parameter is ignored if PHONELIST is omitted.

INSPEED A double word to which is returned an integer specifying the current input speed of the line in characters per second.

OUTSPEED A double word to which is returned an integer specifying the current output speed of the line in characters per second.

MISCARRAY A word array to which is returned miscellaneous information defined at the time the line was opened. If MISCARRAY was not explicitly specified at COPEN time, it nevertheless exists in the form of default values. The format is the same as that of the MISCARRAY parameter of COPEN. If MISCARRAYLEN is zero or omitted, then all such information is returned to MISCARRAY; otherwise, up to MISCARRAYLEN words are returned. The returned information is ordered according to increasing parameter-type values.

MISCARRAYLEN An integer specifying the length (in words) of the array MISCARRAY. If MISCARRAYLEN is zero or omitted then it is assumed that MISCARRAY is sufficiently large to contain all pertinent information.

DRIVERNAME A byte array to which is returned the name of the driver for the line. The array must be at least 8 bytes long, and the name is left-justified with blank fill in the unused byte positions.

CTRACEINFO A word to which is returned CS trace facility information. The format is the same as that of the CTRACEINFO parameter of COPEN. A zero will be returned if the CS trace facility is not invoked for this line.

4.8 OBTAINING MULTIPOINT GROUP INFORMATION

This intrinsic supplies detailed information about a remote group attached to a multi-point line.

**** This intrinsic is valid only for a multipoint line.

The CGROUPINFO intrinsic declaration is:

```
CGROUPINFO(LINENUMBER, LGNUM, GROUPDESC, FIRSTA,  NUMSTA, POLLID,
           POLLIDLEN, SELID, SELIDLEN, GPOLLID, GPOLLIDLEN,
           GSELID, GSELIDLEN);
  VALUE LINENUMBER, LGNUM;
  INTEGER LINENUMBER, LGNUM, FIRSTA, NUMSTA, POLLIDLEN, SELIDLEN,
          GPOLLIDLEN, GSELIDLEN;
  LOGICAL GROUPDESC;
  BYTE ARRAY POLLID, SELID, GPOLLID, GSELID;
  OPTION VARIABLE, EXTERNAL;
```

The condition codes for the CGROUPINFO intrinsic are as follows:

CCE - Request granted.

CCG - (This condition code is not returned.)

CCL - Request denied because an error occurred.

All CGROUPINFO parameters are optional except for LINENUMBER. The parameters are defined as follows:

LINENUMBER	An integer specifying the line number of the line. This is a required parameter.
LGNUM	An integer specifying the logical group number of the remote group. If this parameter is omitted, its value will be set to one.
GROUPDESC	A word to which is returned in bits (8:8) the group's group descriptor. The format is the same as the group descriptor byte in the COPEN SUPLIST array.
FIRSTA	A word to which is returned the logical station number of the first station attached to the group. The remaining stations attached to this group have successive logical station numbers.
NUMSTA	A word to which is returned the total number of stations attached to this group.

POLLID — A byte array to which is returned the group's polling ID. The byte array must be 8 bytes long. When the actual ID is returned, unused bytes in the array are null-filled to the right.

POLLIDLEN — A word to which is returned the length of the group's polling ID.

SELID — A byte array to which is returned the group's selection ID ID. The byte array must be 8 bytes long. When the actual ID is returned, unused bytes in the array are null-filled to the right.

SELIDLEN — A word to which is returned the length of the group's selection ID.

GPOLLID — A byte array to which is returned the group's general poll ID. The byte array must be 8 bytes long. When the actual ID is returned, unused bytes in the array are null-filled to the right.

GPOLLIDLEN — A word to which is returned the length of the group's general poll ID.

GSELID — A byte array to which is returned the group's general selection ID. The byte array must be 8 bytes long. When the actual ID is returned, unused bytes in the array are null-filled to the right.

GSELIDLEN — A word to which is returned the length of the group's general selection ID.

4.9 OBTAINING MULTIPOINT STATION INFORMATION

This intrinsic supplies detailed information about a remote station (terminal) attached to a multi-point line.

**** This intrinsic is valid only for a multipoint line.

The CSTATIONINFO intrinsic declaration is:

```
PROCEDURE CSTATIONINFO(LINENUMBER, STANUM, COMPNUM, GROUPNUM,
              STATIONDESC, NUMCOMP, BLKSIZE, POLLID, POLLEN, SELID,
              SELEN);
  VALUE LINENUMBER, STANUM, COMPNUM;
  INTEGER LINENUMBER, STANUM, COMPNUM, GROUPNUM, NUMCOMP, BLKSIZE,
   POLLEN, SELEN;
  LOGICAL STATIONDESC;
  BYTE ARRAY POLLID, SELID;
  OPTION VARIABLE, EXTERNAL;
```

The condition codes for the CSTATIONINFO intrinsic are as follows:

CCE - Request granted.

CCG - (This condition code is not returned.)

CCL - Request denied because an error occurred.

All CSTATIONINFO parameters are optional except for LINENUMBER. The parameters are defined as follows:

LINENUMBER	An integer specifying the line number of the line. This is a required parameter.
STANUM	An integer specifying the logical station number of the remote terminal. If this parameter is omitted, then its value is set to one.
COMPNUM	An integer specifying the logical component number of a component attached to the terminal. If this parameter is omitted, its value will be set to zero.
GROUPNUM	A word to which is returned the logical group number of the group to which the station is attached.
STATIONDESC	A word to which is returned in bits (8:8) the terminal's station descriptor. The format is the same as the station descriptor byte in format #2 of the COPEN SUPLIST array.

NUMCOMP — A word to which is returned the number of components attached to the station.

BLKSIZE — A word to which is returned the maximum block size which can be sent to the station.

POLLID — A byte array to which is returned the polling ID of the logical component specified by COMP. The byte array must be 8 bytes long. When the actual ID is returned, unused bytes in the array are null-filled to the right.

POLLEN — A word to which is returned the length of the component's polling ID.

SELID — A byte array to which is returned the selection ID of the logical component specified by COMP. The byte array must be 8 bytes long. When the actual ID is returned, unused bytes in the array are null-filled to the right.

SELEN — A word to which is returned the length of the component's selection ID.

4.10 OBTAINING LINE ERROR AND STATUS INFORMATION

When a CS intrinsic returns a condition code indicating an error, the programmer can request additional information about the error by issuing a CCHECK intrinsic call. This intrinsic accepts zero as a legal line parameter value. When zero is specified, the returned error code reflects the status of the last COPEN or unsuccessful IOWAIT.

The CCHECK intrinsic provides error information as well as last-event status information for the line. Under non-error conditions, a CCHECK intrinsic call will provide the user with additional line information complementary to that provided by CGETINFO. This information reflects the state of various lists as well as message and error counters for the last (completed) I/O operation on the line.

The CCHECK intrinsic declaration is:

```
PROCEDURE CCHECK (LINENUMBER, ERRORCODE, IDLISTINDEX, POLLISTINDEX,
                  PHONELISTINDEX, MSGSENT, MSGRECV, RECOVERRORS,
                  IRRECOVERRORS);
  VALUE LINENUMBER;
  INTEGER LINENUMBER, ERRORCODE, IDLISTINDEX, POLLISTINDEX,
                  PHONELISTINDEX, RECOVERRORS, IRRECOVERRORS;
  DOUBLE MSGSENT, MSGRECV;
  OPTION VARIABLE, EXTERNAL;
```

The condition codes for the CCHECK intrinsic are as follows:

CCE - Request granted.

CCG - (This condition code is not returned.)

CCL - Request denied because an error occurred.

All CCHECK parameters are optional except for LINENUMBER. for those parameters returning an index, the index should be used as a subscript to the corresponding array returned by CGETINFO. The parameters are defined as follows:

LINENUMBER — An integer specifying the line number of the line. This parameter is required. If LINENUMBER has a value of zero, the returned error code reflects the status of the last COPEN or unsuccessful IOWAIT, whichever is the most recent.

ERRORCODE — A word to which is returned a 16-bit code, specifying the type of error that occurred. If the previous operation was successful, all bits are set to zero.

In the 16 bits returned to the word specified by the ERRORCODE parameter, the low-order eight bits contain the error-type code that shows what kind of error occurred. The high-order eight bits specify the last recoverable line-error that occurred on the last I/O access. Error codes are listed later in this section.

IDLISTINDEX — A word to which is returned an integer whose value denotes the (byte) subscript position of the most recently used entry in the list of ID sequences. A value of zero indicates that there is no "current" entry.

POLLISTINDEX — A word to which is returned an integer whose value denotes the (word) subscript position of the most recently used entry in the polling list. A value of zero indicates that there is no "current" entry.

PHONELISTINDEX — A word to which is returned an integer whose value denotes the (byte) subscript position of the current entry in the list of phone numbers. A value of zero indicates that there is no "current" entry.

MSGSENT — A double word to which is returned an integer indicating the number of messages that have been sent successfully over the line since the line was last connected. Note that SSLC transmissions which end in ETB, as well as ETX, are counted as messages.

MSGRECV — A double word to which is returned an integer indicating the number of messages that have been received successfully over the line since the line was last connected. Note that SSLC transmissions which end in ETB, as well as ETX, are counted as messages.

RECOVERRORS — A word to which is returned an integer indicating the number of recoverable errors that have occurred on the line since the line was last connected.

IRRECOVERRORS — A word to which is returned an integer indicating the number of irrecoverable errors that have occurred on the line since the line was last connected.

4.11 CCHECK Irrecoverable Error Codes [ERRORCODE (8:8)]

The CCHECK irrecoverable error codes codes have been divided into six subsets.

Range	Description
0	request completed successfully
1 - 40	an error was found by the COPEN intrinsic
41 - 50	the request was not initiated because of an error found by the CS Intrinsics (including COPEN)
51 - 100	the request was not initiated because of an error found by the CS Intrinsics (except COPEN)
101 - 150	a hardware error occurred
151 - 200	an error or exceptional condition which resulted in the line's being disconnected (driver dependent)
201 - 250	an error or exceptional condition occurred which which did not result in the line's being disconnected (driver-dependent)

The following tables describe the defined error codes.

NOTE: The heading "BSC P-P" includes both conversational and non-conversational BSC.

COPEN INTRINSIC - IRRECOVERABLE ERRORS

NUM	DESCRIPTION	BSC P-P	BSC MP	HPDLC I
0	Request completely successfully	X	X	X
1	none or too many groups		X	
2	none or too many stations		X	
3	none or too many components		X	
4	invalid poll/selection sequence length		X	
5	not enough stack space for COPEN to process	X	X	X
6	invalid driver name	X	X	X
7	driver not found	X	X	X
8	driver not compatible with line's attributes	X	X	X
9	driver not changeable	X	X	X
10	undefined device	X	X	X
11	device not available	X	X	X
12	not cs device	X	X	X
13	cs device in use	X	X	X
14	invalid id sequence length	X	X	
15	invalid buffer size	X	X	X
17	invalid phone number length	X	X	X
18	illegal character in phone number	X	X	X
19	not compatible with line type	X	X	X
20	invalid information type in MISCARRAY	X	X	X
21	invalid information value in MISCARRAY	X	X	X
23	invalid entry in the poll list		X	
24	could not open trace file; issue FCHECK(0) to find specific File System error code	X	X	X
25	invalid trace entry size	X	X	X
26	invalid user capability	X	X	X
27	invalid line designator	X	X	X
28	no designator or device specified	X	X	X
29	too many files/lines	X	X	X
31	insufficient memory space	X	X	X
32	driver failed to open line	X	X	X
33	local mode was control station, but SUPLIST parameter was not specified		X	

ALL INTRINSICS - IRRECOVERABLE ERRORS

NUM	DESCRIPTION	BSC P-P	BSC MP	HPDLC I
48	no virtual memory available for trace and/or buffering	X	X	X
49	db not pointing at stack	X	X	X

NON-COPEN INTRINSICS - IRRECOVERABLE ERRORS

NUM	DESCRIPTION	BSC P-P	BSC MP	HPDLC I
51	invalid line number - no such line	X	X	X
52	invalid parameter value	X	X	X
53	I/O error on trace file	X	X	X
54	not used			
55	buffer or count not specified	X	X	X
56	invalid buffer count parameter	X	X	X
57	no answer to dial attempt	X	X	X
58	no phonelist for dial attempt	X	X	X
59	bad dial msg. - system problem	X	X	X
60	invalid array length parameter	X	X	X
61	bad ccontrol code	X	X	X
63	no I/O in progress for abort	X	X	X
64	abort ignored	X	X	X
65	logical group number is invalid		X	
66	logical station number is invalid		X	
67	logical component number is invalid		X	
73	parameter bounds violation	X	X	X
76	buffer parameter required but not specified	X	X	X
77	maximum number of I/O requests are already outstanding; must first issue IOWAIT	X	X	X
78	no I/O-s pending for any file/line	X	X	X
79	no I/O pending for specified file/line	X	X	X
115	SCCP RAM system failure	X	X	X
116	Mainframe driver time-out error	X	X	X

HARDWARE RELATED IRRECOVERABLE ERRORS

NUM	DESCRIPTION	BSC P-P	BSC MP	HPDLC I
83	SCCP system failure	X	X	X
85	SCCP parity error	X	X	X
87	SCCP received bad self-test control character	X	X	X
88	SCCP DMA self-test error	X	X	X
89	SCCP microprocessor (MC2) failure	X	X	X
90	SCCP ROM faiure	X	X	X
91	SCCP RAM failure	X	X	X
92	SCCP USART transmitter overrun	X	X	X
93	SCCP USART parity error	X	X	X
94	SCCP USART self-test receive error	X	X	X
95	SCCP USYNRT self-test transmitter underrun	X	X	X
96	SCCP USYNRT self-test receive error	X	X	X
97	SCCP USYNRT self-test receive overrun	X	X	X
98	SCCP USYNRT self-test receive aborted	X	X	X
99	SCCP USART self-test received no data	X	X	X
101	Non-responding device	X	X	X
102	Transfer error	X	X	X
103	Data set not ready	X	X	X
104	Carrier loss	X	X	X
105	Data overrun	X	X	

DRIVER DEPENDENT IRRECOVERABLE ERRORS RESULTING IN DISCONNECTION

NUM	DESCRIPTION	BSC P-P	BSC MP	HPDLC I
151	Connect timeout	X	X	X
153	Remote rejected the connection	X		X
154	Power failure occurred	X	X	
155	Local timeout	X	X	
156	An internal error was detected by the driver	X	X	X
157	Remote protocol error	X	X	
158	Remote sent shutdown sequence and disconnected	X		X
159	Remote sent shutdown sequence and disconnected before the I/O request was issued.	X		X
160	An internal error was detected by MPE	X	X	X

OTHER DRIVER DEPENDENT IRRECOVERABLE ERRORS

NUM	DESCRIPTION	BSC P-P	BSC MP	HPDLC I
201	Operation aborted	X	X	X
202	Invalid user request	X	X	X
203	Remote is not ready to accept line bid	X		
204	Remote rejected the line bid	X		
205	Remote primary station bid for the line while local user was also bidding	X		
206	Remote has requested to send (an RVI sequence was received)	X		
207	Retry count exhausted	X	X	X
208	Unexpected text was received	X	X	
209	Receive timeout	X	X	
210	Remote sent end-of-transmission	X		
211	Remote sent end-of-transmission sequence and disconnected before the I/O request was issued.	X		
212	During the execution of a CWRITE conversational with output buffer specified to also be the input buffer; the remote requested a resend of the output buffer but its contents had been modified while receiving from the remote.	X		
213	Remote sent an ACK sequence in response to local CREAD acknowledgement.	X		
214	Remote sent a NAK sequence in response to local CREAD acknowledgement.	X		
215	Remote sent an RVI sequence in response to local CREAD acknowledgement.	X		
216	Remote requested a download sequence be initiated	X		
217	No line bid was received from the remote, local timed out	X		
218	Remote sent a delay sequence instead of the expected text/response	X		
219	The entries in the pollist were polled the required number of times, and no station responded		X	
220	An EOT was received from the remote before the last block of a multiblock transmission was sent	X		
221	After an RVI was sent to the remote, the remote responded with text instead of the (expected) EOT	X		
222	Pollentry down or Pollist down		X	
223	Too much data was transmitted by the remote; part of the data was lost.		X	

2 CCHECK Recoverable Error Codes (ERRORCODE (0:8))

The following table describes the defined recoverable error codes.

RECOVERABLE ERRORS

NUM	DESCRIPTION	BSC P-P	BSC MP	HPDLC I
0	No recoverable error occurred	X	X	X
1	Invalid ID sequence received	X		
2	Received unintelligible sequence	X	X	X
3	Block check characteror field check sequence error	X	X	X
4	Response timeout	X	X	X
5	Received incorrect acknowledgement	X	X	
6	Remote attempted to bid for the line	X		
7	Remote did not respond to local's line bid	X		
8	Received unintelligible sequence after sending text	X	X	
9	Received enquiry character after sending text	X	X	
10	Remote requested a resend of local's last response	X	X	
11	Remote requested resend of last text block	X	X	X
12	Received end-of-transmission character while in control state	X		
13	Received text overflow	X	X	X
14	Data overrun occurred on SIO multiplexer	X	X	X
15	Transfer error occurred on the SIO multiplexor	X	X	X
16	Data overrun on the SCCP interface board			X
17	Data underrun on the SCCP interface board			X
18	Host sent invalid data to 3270 station	X		

4.13 DISPLAYING LINE INFORMATION

The CS user may request a display of all relevant information about any CS line by invoking the PRINT'LINE'INFO intrinsic.

The PRINT'LINE'INFO intrinsic declaration is as follows:

```
PROCEDURE PRINT'LINE'INFO (LINENUMBER,OUTFILE);
   VALUE LINENUMBER,OUTFILE;
   INTEGER LINENUMBER,OUTFILE;
   OPTION EXTERNAL,VARIABLE;
```

PRINT'LINE'INFO is simply a utility intrinsic, having no special privileges, which formats and writes to OUTFILE the information returned by the CS intrinsics CGETINFO, CCHECK, and certain options of CCONTROL. Figure 4-1 is a sample output of PRINT'LINE'INFO.

The parameter LINENUMBER is the CS line number returned by the COPEN intrinsic. If LINENUMBER=0 or is not specified then only the CCHECK information is displayed. If OUTFILE=0 or is not specified then $STDLIST is used.

The condition codes returned by PRINT'LINE'INFO are as follows:

CCE - PRINT'LINE'INFO completed successfully
CCG - an end-of-file was detected while writing to OUTFILE
CCL - an error was detected while writing to OUTFILE

```
**************************************************
*-L-I-N-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y*
**************************************************
*  LINE NUMBER:  6      LOGICAL DEV. NUMBER:  54   *
*  DEV. TYPE:  19       SUBTYPE:  3  VER: A.00.03  *
*            0123456789012345                      *
*  COPTIONS: 0010000010000010                      *
*  AOPTIONS: 0000000100001100                      *
*  DOPTIONS: 0000000100000000                      *
*  NUMBUFFERS: 1           BUFFSIZE: 1000 (WORDS)  *
*  INSPEED: 2500           OUTSPEED: 2500   SECS.  *
*  MISCARRY:         RECEIVE TIMEOUT: 30    SECS.  *
*                      LOCAL TIMEOUT: 900   SECS.  *
*                    CONNECT TIMEOUT: 900   SECS.  *
*                   RESPONSE TIMEOUT: 3     SECS.  *
*                  NO. ERROR RETRIES: 7            *
*                 CLEAR-TO-SEND DELAY: 00.0  SECS  *
*                DATA-SET-READY DELAY: DISABLED    *
*                  TRANSMISSION MODE : DUPLEX      *
*               MMSTAT TRACE FACILITY: ENABLED     *
*                    POLL LOOP DELAY :       MSECS.*
*                        POLL REPEAT :             *
*                    POLL ENTRY DELAY :      MSECS.*
*  DRIVERNAME: CSSBSC1    LINESTATE: UNCONNECTED   *
*  CTRACEINFO:   ENTRIES=25      MASK=111111       *
*                TYPE OF TRACE = ALL, WRAP         *
*  POLLIST:     ENTRIES=6          INDEX=2         *
*      1 STA=  1 CMP= 0      2 STA=  2 CMP= 0      *
*      3 GRP=  2             4 STA=  4 CMP=12      *
*      5 STA= 11 CMP=  2     6 GRP=  1             *
*  PHONELIST:   ENTRIES=0          INDEX=0         *
*  IDLIST:      ENTRIES=0          INDEX=0         *
*  SUPLIST: GROUPS=1                  DESC=0111000 *
*           LINE SEL=A1 A1                         *
*   GRP 1   STATIONS=1                 DESC=0011001 *
*           POLL ID =41 41                         *
*           SEL  ID =61 61                         *
*         GEN POLL ID=22 22                        *
*         GEN SEL  ID=F6 F6                        *
*    STA 1  COMPONENTS= 1  UP         TYPE=0       *
*     CMP 0      POLL=46 46                        *
*                SEL =46 46                        *
*  ERRORCODE:  RECOVERABLE=0   IRRECOVERABLE=0     *
*  MSGSENT: 22              MSGRECV: 9             *
*  RECOVERRORS: 3           IRRECOVERRORS: 0       *
**************************************************
```

Figure 4-1. PRINT'LINE'INFO Sample Output

LEGEND: 1) COPTIONS, AOPTIONS, DOPTIONS, and Trace MASK are in binary.

2) Poll and Selection ID sequences are in hexadecimal

3) All other numbers are in decimal.

DIRECTING LINE CONTROL OPERATIONS

The user can perform various control functions to modify the operation of a line by issuing a CCONTROL intrinsic call. These functions include list and list index modifications, I/O request cancellation, and line configuration changes. List modification provides a mechanism for list entry insertion and deletion. List index modification allows the user to alter the normal (i.e., sequential) scanning of CS list structures.

The CCONTROL intrinsic provides a means for cancelling any outstanding I/O requests. An intrinsic call for this purpose may be issued at any time (after the line is opened). All other CCONTROL functions require that any outstanding CIO requests be quiesced via the IOWAIT intrinsic before the desired operation is performed. A CCONTROL error will result if there is any outstanding I/O when the CCONTROL is issued.

The CCONTROL intrinsic declaration is as follows:

```
PROCEDURE CCONTROL (LINENUMBER, CONTROLCODE, PARAMETER);
  VALUE LINENUMBER, CONTROLCODE;
  INTEGER LINENUMBER, CONTROLCODE;
  LOGICAL PARAMETER;
  OPTION EXTERNAL;
```

The condition codes for the CCONTROL intrinsic are as follows:

CCE - Request granted.

CCG - (This condition code is not returned.)

CCL - Request denied because an error occurred.

NOTE: In CIO mode with CONTROLCODE <>0, a CCL will be returned if any I/O requests are outstanding.

The CCONTROL parameters are

LINENUMBER — An integer specifying the line number of the line.

CONTROLCODE — An integer defining the specific operation to be performed.

PARAMETER — A word whose interpretation is a function of the particular CONTROLCODE.

The following is the definition of the various control codes and the corresponding parameters:

4.15 CONTROLCODE Functions

0 cancels an outstanding request (or requests) on a line.

A request that has not been physically initiated is cancelled. A request that has been physically initiated is allowed to complete with one exception:

If a polling operation is in progress and no affirmative response is received from the polled component, then the request is cancelled without completing the polling list.

PARAMETER specifies which request or requests to abort. 0 - abort all outstanding I/O -n - abort the n'th oldest read request (1 being the oldest) +m - abort the m'th oldest write request

NOTE: This CONTROLCODE request is meaningful only for lines opened with the CIO specification.

1 Reset the line from text state to control state.

An end-of-transmission sequence (EOT) is sent to the remote.

NOTE: a. If the current state is control then an EOT will still be sent.
b. If the current state is unconnected then this CCONTROL is ignored.

2 Disconnect the line

A disconnect sequence is sent to the remote and the connection is broken. The type of disconnect sequence (EOT or DLE EOT) is a function of DOPTIONS.(4:2) or a previous CCONTROL with controlcode of 258.

NOTE: If the line is already disconencted, then this CCONTROL is ignored.

32 changes the index into the phone list to the value PARAMETER, which is an integer denoting a byte subscript position into the phone list array.

A condition code of CCL will be returned if PHONELIST(PARAMETER) does not point to a valid phone list entry.

33 inserts an entry into the poll list for a multi-point line.

PARAMETER is the entry to be inserted at the bottom of the poll list. PARAMETER has the same format as a pollist entry in in the CPOLLIST intrinsic.

A condition code of CCL will be returned if the entry does not have the correct format.

**** This control code is only valid for multi-point lines.

34 deletes an entry from the poll list for a multi-point line. If the entry appears more than once in the POLLIST, then all instances are deleted.

PARAMETER is the entry to be deleted, and has the same format as a pollist entry in the CPOLLIST intrinsic.

A condition code of CCL will be returned if the entry to be deleted does not exist in the list. **** This control code is only valid for multi-point lines.

35 Change POLLIST index.

changes the index into the poll list to the value PARAMETER which is an integer denoting a word subscript position into the poll list array.

**** This control code is only valid for multi-point lines.

36 change dialing convention.

PARAMETER

= 0 dial on write connects; answer on read connects
= 1 answer on write connect; dial on read connect
= 2 dial on write connect; dial on read connect
= 3 answer on write connect; answer on read connect

A condition code of CCL is returned if PARAMETER is negative or greater than 3.

37 change the value of the receive time out.

If PARAMETER is a positive, nonzero number then it will be the new value of the receive timeout (seconds). A value of zero will cause the timeout to be disabled. Setting parameter to a minus one will set the timeout to the default value of 20 seconds. Any other values of parameter are invalid.

**** This control code is not valid for lines using HPDLC I.

38 change the value of the local timeout.

If PARAMETER is a positive, nonzero number then it will be the new value of the local timeout (seconds). A value of zero will cause the timeout to be disabled. Setting parameter to a minus one will set the timeout to the default value of 60 seconds. Any other values of parameter are invalid.

**** This control code is not valid for lines using HPDLC I.

39 change the value of the connect timeout.

If PARAMETER is a positive, nonzero number then it will be the new value of the connect timeout (seconds). A value of zero will cause the timeout to be disabled. Setting parameter to a minus one will set the timeout to the default value of 900 seconds. Any other values of parameter are invalid.

40 change the number of error recovery retry attempts to be made by the driver.

The COPEN default value (see MISCARRAY parameter of COPEN) is driver-dependent.

PARAMETER is the new number of retry attempts.
PARAMETER must be a positive integer.

41 change local station mode for a contention line.

PARAMETER = 1 set local station to a primary contention station
= 2 set local station to a secondary contention station
= 3 set local station to a control station
= 4 set local station to a 3270 secondary station
= 5 set local station to DTE
= 6 set local station to DCE

A condition code of CCL will be returned if PARAMETER is negative or greater than

6.

42 read line state

PARAMETER is a word to which is returned the current state of the line.

```
PARAMETER (0.8) = reserved
          (8:8) = 0 = unconnected
                = 1 = control
                = 2 = text
```

**** This control code is not valid for lines using HPDLC I.

43 Disable Tracing

Parameter is ignored.

The core-resident trace area in deleted and if this is the only line accessing the trace file, the trace file is closed. If the line is not tracing thru this CCONTROL is ignored.

44 Enable Tracing

```
parameter(0) = same meaning as the CTRACEINFO
               parameter of COPEN
PARAMETER(1) = first word of the array describing
               the trace file. The type of array
               is defined by PARAMETER(1).(0:8):
     alpha   = the array contains a formal file
               designator
     numeric = the array contains an ASCII number
               specifying the trace file's logical
               device number
     special = the array contains a device class
               name starting in PARAMETER(1).(8:8)
```

If the line already has a tracing it is first disabled (see CCONTROL with OPCODE = 43). A main memory trace area is created for this line and, if the trace file is not currently being accessed, the trace file is opened.

45 Return current remote ID sequence

Parameter = word address of a nine word array to which is returned:

```
   word 0     - length of current remote ID (set to
                if there is no current remote ID) current
   words 1-8  - remote ID sequence
```

46 Change concurrent I/O specification

Parameter (15:1) = 0 - NCIO
1 = CIO

NOTE: CIO without buffering is allowed only when the user is operating in privileged mode.

47 Change the value of the response time out.

Parameter is the value of the response timeout (seconds). Parameter must be a positive value. A value of zero will disable the timeout.

For HPDLC I protocol, this is parameter T1, the maximum amount of time the local will wait for a response to a transmitted Information frame.

48 Change the value of the line bid time out.

Parameter is the value of the line bid timeout (seconds). Parameter must be a positive value. A value of zero will disable the timeout.

49 Enable/Disable MMSTAT Driver Trace

PARAMETER = 0 Disable trace
1 Enable trace

50 Current ID in Configured ID List

Determines if the current ID is in the configured ID list. Values returned in PARAMETER are:

-1 Current ID is not in configured list
0 Zero length current ID not in configured list
1 Zero length current ID is in configured list
2 Non-zero length current ID is in configure list

51 Set all components in pooll list up.

Resets any components in the poll list that were marked down due to lack of response to up.

**** This control code is only valid for multi-point lines.

52 Return last transmission log

Returns the transmission log of the last CS I/O event returned to user.

53 Dump SCCP RAM

The SCCP RAM contents, mainframe driver information, and CS I/O request information is stored in a file. An attempt is made to open the file to receive the dump information as new; if successful, the dump proceeds. If an old file with that name is found, the dump is not initiated. Thus, if there are successive attempts to dump to the same file name, only the first attempt which opens the file new will actually dump the contents of the SCCP.

PARAMETER is the first word of the array containing the desired dump file name. If zero, the name "CDUMP" with the logical device number appended will be used.

A message is written is written to the operator console when an SCCP RAM dump is initiated and successfully completed. These messages are described in section 9.4.

256 changes the automatic generation of WACK and/or TTD sequences by the BSC drivers (both SSLC and HSI).

PARAMETER (15:1) = 1, disable automatic generation of WACK sequences.
= 0, enable automatic generation of WACK sequences. (default: enabled).

(14:1) = 1, disable automatic generation of TTD sequences.
= 0, enable automatic generation of TTD sequences. (default: enabled).

257 change the USASCII block cneck character algorithm for the SSLC (BSC) driver. This CONTROLCODE is ignored by the HSI (BSC) driver.

PARAMETER = 0, VRC/LRC (non-transparent).
CRC-16 (transparent)
= 1, VRC/CRC-16 (non-transparent)
CRC-16 (transparent).

The default mode for the SSLC (BSC) driver is equivalent to PARAMETER=0.

258 change the disconnect sequence for the BSC drivers (both SSLC and HSI).

PARAMETER = 0, send EOT for non-switched and DLE EOT for switched.

= 1, always send DLE EOT.
= 2, always send EOT.

The default mode for the BSC drivers is equivalent to PARAMETER=0.

259 inform the SSLC BSC driver whether or not to expect intermediate text block characters (itb's) to be received from the remote station. This CONTROLCODE is ignored by the HSI BSC driver.

PARAMETER = 0, do not expect itb's from the remote station.

NOTE: The driver will accept itb's from the remote station in this mode and will format them as described in paragraph 4.8. However, it will be necessary for the remote to resend the first text block which contains ITBs.

= 1, expect to receive itb's from the remote station.

NOTE: The driver will treat received text which contains no itb's as a degenerate case of itb reception. That is, MFW (1:1) will be set and the received text will be prefixed by a length (in bytes) and suffixed by a delimiter of all ones (see paragraph 4.8).

This CCONTROL function in no way affects transmitted text. The default mode for the SSLC driver is equivalent to PARAMETER=0. Receipt of a text block containing itb's will cause the driver to automatically enter the mode equivalent to PARAMETER=1.

260 Modify the action of the next CREAD

Parameter = 0 = the CREAD will be executed in the normal manner.
1 = The next CREAD will send an RVI acknowledgement (instead of an ACK0 or ACK1) then read the next text block.
2 = The next CREAD will send a negative acknowledgement then read the last text block sent by the remote.

All other Parameter values are invalid.

NOTE: 1. This CCONTROL only applies to the next CREAD. Subsequent CREAD's are performed in the normal manner.

2. If at the time the next CREAD is initiated the line state is not text then the CREAD will not

be modified.

261 Change the Message Format Word (MFW) convention for SSLC Driver.
parameter = 0 MFW will not be placed into received text nor expected in sent text. The implicit MFW to be used to format sent text will be 000000.
1 = MFW will not be placed into received text nor expected in sent text. The implicit MFW to be used to format sent text will be 100000.
2 = MFW will be placed into word 0 of the user's input buffer. It will be expected to be in word 0 of the user's output buffer.

262 Change Download Mode

PARAMETER = 0 = disable download mode
1 = enable download mode.
A CREAD in unconnected or control state will interpret a NAK (instead of an ENQ) as a request for download.

263 Read Download Byte

The received download byte is returned to PARAMETER.(8:8). PARAMETER(0:8) is cleared.

Note: This CCONTROL applies only to the SSLC Driver, the Hsi Driver will return a binary zero.

264 Change Control state listen mode
PARAMETER = 0 = While in control state and between user requests the driver will listen for control sequences from the remote. Receipt of a line bid from the remote will cause the line to be placed into text state.
1 = While in control state and between user requests the driver will ignore any control sequences issued from the remote.

265 Reserved for 3270 use.

266 Reserved for 3270 use.

Some CCONTROL controlcodes are serviced completely by the driver, some by the intrinsic without involving the driver, and some by both the intrinsic and the driver. For these reasons, controlcodes are assigned using the following convention:

0-255 Controlcodes whose execution involves both the intrinsic and the driver.

256-16383 controlcodes executed by the driver.

6 Modifying Polling Lists

A user process can create or update a poll list, after the line to which the poll list refers has been opened, by issuing a CPOLLIST intrinsic call. In addition, this intrinsic allows other poll-related information to be changed by the user. In contrast to the CCONTROL intrinsic, which will add or delete a single station to or from a poll list, this intrinsic provides a facility for respecification of an entire list.

**** This intrinsic is only valid for multi-point lines

The CPOLLIST intrinsic declaration is as follows:

```
PROCEDURE CPOLLIST (LINENUMBER, POLLIST, POLLDELAY,
  POLLREPEAT, ENTRYDELAY);
  VALUE LINENUMBER, POLLDELAY, POLLREPEAT, ENTRYDELAY;
  INTEGER LINENUMBER, POLLDELAY, POLLREPEAT, ENTRYDELAY;
  LOGICAL ARRAY POLLIST;
  OPTION VARIABLE, EXTERNAL;
```

All parameters to CPOLLIST are optional except LINENUMBER. If a parameter is not specified, its existing value will remain in effect; each parameter will have a default assigned at either configuration or COPEN time.

The CPOLLIST intrinsic can return the following condition codes:

CCE Request granted.

CCG (This condition code is not returned.)

CCL Request denied, because an error occurred.

NOTE: In CIO mode, a CCL will be returned if any requests are pending on the line (i.e., no IOWAIT has been issued for the outstanding I/O).

The parameters are defined as follows:

LINENUMBER An integer specifying the line number of the line. This parameter is required.

POLLIST A word array defining the groups, stations, and components to be polled and the order of polling:

word 0 number of entries in the list

1 first entry of the list

Station format
(0:4) Set to zero.
(4:4) Logical component number
(8:8) Logical station number.

Group format
(0:1) Set to one.
(1:7) Set to zero.
(8:8) Logical group number. Group numbers of 0 and 255 are invalid.

2 second entry of the list

.
.
.

n nth entry of the list

NOTES: 1) A poll list need not specify the total set of stations physically present on the line. Those that are not specified are never accessed. If POLLIST specifies one or more stations that are DOWN, these stations remain in the list but are not accessed.

2) The poll list index is reset to zero when POLLIST is present. (See POLLISTINDEX parameter of CCHECK).

3) Group format is permitted only for terminal groups that have the general poll facility.

POLLDELAY Number of hundredths of seconds of delay between iterations through a wrap (circular) polling list.

POLLREPEAT Specifies the maximum number of iterations through the polling list. A value of zero implies an unlimited number of passes through the list. A value other than one makes the list a wrap (circular) list. Polling is terminated when either of the following conditions occurs: a) a station responds to a poll or b) the specified number of passes through the polling list has been made with no station responding.

ENTRYDELAY Specifies the delay in milliseconds between polling each entry in the POLLIST.

5.1 REMOTE FILE TRANSFER PROGRAM (CODED IN SPL/3000)

This example shows a remote file transfer application program (named RFT) that transmits an existing MPE disc file from one HP 3000 Computer System to another. It is assumed that RFT is being executed simultaneously at both ends of the line and that the overall operation has been coordinated by a voice telephone conversation.

The calling sequence (entered through the user's terminal) for the sending station is as follows:

```
:CLINE T;DEV=SSLC1

:RUN RFT

ENTER FILE NAME -

Carriage return
```

The calling sequence (entered through the user's terminal) for the receiving station is as follows:

```
:CLINE R;DEV=SSLC2

:RUN RFT1

ENTER FILE NAME -

Filename
```

At the receiving station, RFT opens (as a new ASCII file) a file with the specified name and then transmits the filename to the sending station. At the sending station, RFT opens the specified file, transmits its record size, and then transmits the file, record by record, until an end-of-file is detected. When this occurs the sending station transmits an EOT control character to the receiving station and the operation is finished.

```
$CONTROL USLINIT,NOWARN,CODE,MAP
$CONTROL MAIN=RFT
<<                                                                    >>
<<                                                                    >>
<<       THIS PROGRAM, RFT, TRANSFERS DISC FILES BETWEEN TWO          >>
<<       COMPUTERS.                                                   >>
<<                                                                    >>
<<       THE TRANSMITTER'S DIALOGUE IS AS FOLLOWS:                    >>
<<       ********************************************************     >>
<<       * :CLINE T;DEV=XXXXX                                  *     >>
<<       * :RUN RFT                                            *     >>
<<       * ENTER FILE NAME -                                   *     >>
<<       * (DEPRESS THE CARRIAGE RETURN KEY)                   *     >>
<<       ********************************************************     >>
<<                                                                    >>
<<       THE RECEIVER'S DIALOGUE IS AS FOLLOWS:                       >>
<<       ********************************************************     >>
<<       * :CLINE R;DEV=XXXXXX                                 *     >>
<<       * :RUN RFT                                            *     >>
<<       * ENTER FILE NAME -                                   *     >>
<<       * FILENAME                                            *     >>
<<       ********************************************************     >>
<<                                                                    >>
<<       THE RECORDS OF THE FILE, FILENAME, RESIDING IN THE SENDER'S  >>
<<       COMPUTER WILL BE SENT, A RECORD AT A TIME, TO THE            >>
<<       RECEIVER'S COMPUTER AND IS WRITTEN TO THE FILE WITH THE      >>
<<       SAME NAME THERE.                                             >>
<<                                                                    >>
<<                                                                    >>
BEGIN
INTEGER READCNT,LINENUM,FILENUM,ANYTHING,RECSIZE,RECORDCNT:=0;
BYTE ARRAY REMOTEABORTED(0:13):="REMOTE ABORTED";
BYTE ARRAY GETFILENAME(0:16):="ENTER FILE NAME -";
BYTE ARRAY RECTRANSFERRED(0:26):="RECORDS TRANSFERRED =       ";
BYTE ARRAY CDESIG1(0:17):="CSLINET CSCLASST ";
BYTE ARRAY CDESIG2(0:17):="CSLINER CSCLASSR ";
BYTE ARRAY DESIG(0:29):="                                ";
BYTE ARRAY BUFFER(0:255):=0,0;
BYTE ARRAY PHONELIST(0:39):=1,8,"257-1562";

EQUATE
   FSFOPTIONS1 =  000003,    << OLD  >>
   FSFOPTIONS2 =  000004,    << NEW,ASCII  >>
   FSAOPTIONS1 =  000000,    << READ ACCESS  >>
   FSAOPTIONS2 =  000001;    << WRITE ACCESS  >>

EQUATE
   SENDEOT    = 1;
```

```
TRINSIC READ,PRINT,ASCII,FOPEN,FCLOSE,FREAD,FWRITE,
        FGETINFO,TERMINATE,COPEN,CREAD,CWRITE,CCONTROL;

PROCEDURE PRINT'FILE'INFO(FILENUM);
VALUE FILENUM; INTEGER FILENUM; OPTION EXTERNAL;

PROCEDURE PRINT'LINE'INFO(LINENUM,OUTFILE);
VALUE LINENUM,OUTFILE; INTEGER LINENUM,OUTFILE;
OPTION VARIABLE,EXTERNAL;



SUBROUTINE ENDOFJOB;
   BEGIN
   IF FILENUM <> 0 THEN
      BEGIN
      FCLOSE(FILENUM,1,0);
      IF <> THEN PRINT'FILE'INFO(FILENUM);
      END;
   ASCII(RECORDCNT,10,RECTRANSFERRED(22));
   PRINT(RECTRANSFERRED,-27,0);
   TERMINATE;
   END;

BROUTINE CSPROBLEM;
   BEGIN
   IF > THEN
      PRINT(REMOTEABORTED,-14,0);
   ELSE
      PRINT'LINE'INFO(LINENUM);
   ENDOFJOB;
   END;

SUBROUTINE FSPROBLEM;
   BEGIN
   PRINT'FILE'INFO(FILENUM);
   ENDOFJOB;
   END;


PRINT(GETFILENAME,-17,0);
READCNT:=-READ(DESIG,-30);
IF > THEN ENDOFJOB;
IF READCNT = 0 THEN
   BEGIN     <<SEND EXISTING FILE>>
   LINENUM:=COPEN(CDESIG1,CDESIG1(8));
   IF <> THEN CSPROBLEM;
   READCNT:=CREAD(LINENUM,DESIG,-28);    <<GET FILE NAME FROM RECEIVER>>
   IF <> THEN CSPROBLEM;
```

```
   FILENUM:=FOPEN(DESIG,FSFOPTIONS1,FSAOPTIONS1);
   IF <> THEN FSPROBLEM;
   FGETINFO(FILENUM,,,,RECSIZE);
   RECSIZE:=IF RECSIZE > 0 THEN -2*RECSIZE ELSE RECSIZE;
   CWRITE(LINENUM,RECSIZE,1);     <<SEND FILE'S RECORD SIZE>>
   IF <> THEN CSPROBLEM;
   DO
      BEGIN     <<TRANSFER CONTENTS OF THE FILE TO THE REMOTE>>
      FREAD(FILENUM,BUFFER,RECSIZE);
      IF < THEN FSPROBLEM;
      IF = THEN
         BEGIN
         RECORDCNT:=RECORDCNT+1;
         CWRITE(LINENUM,BUFFER,RECSIZE);
         IF <> THEN CSPROBLEM;
         END;
      END UNTIL >;     <<SENSED END-OF-FILE>>
   CCONTROL(LINENUM,SENDEOT,ANYTHING);
   IF < THEN CSPROBLEM;
   END
ELSE
   BEGIN     <<RECEIVE NEW FILE>>
   LINENUM:=COPEN(CDESIG2,CDESIG2(8),,,,,,,,,,PHONELIST);
   IF <> THEN CSPROBLEM;
   <<SEND FILE NAME AND GET RECORD SIZE OF THE FILE>>
   CWRITE(LINENUM,DESIG,READCNT-1,RECSIZE,1);
   IF <> THEN CSPROBLEM;
   FILENUM:=FOPEN(DESIG,FSFOPTIONS2,FSAOPTIONS2,RECSIZE);
   IF <> THEN FSPROBLEM;
   DO
      BEGIN     <<RECEIVE CONTENTS OF THE FILE>>
      READCNT:=CREAD(LINENUM,BUFFER,RECSIZE);
      IF < THEN CSPROBLEM;
      IF = THEN
         BEGIN
         RECORDCNT:=RECORDCNT+1;
         FWRITE(FILENUM,BUFFER,-READCNT,0);
         IF <> THEN FSPROBLEM;
         END;
      END UNTIL >;     <<EOT RECEIVED>>
   END;
ENDOFJOB;
END.
```

5.2 MULTIPOINT TRANSACTION PROCESSING PROGRAM (CODED IN SPL/3000)

This example shows a transaction processing program that accepts data from HP 2645 terminals on one multipoint line. Normally the line's terminal configuration would already have been specified via SYSDUMP or INITIAL. However in this example both the CLINE SUPLIST and the COPEN SUPLIST array have been specified to illustrate their use. In this example the multipoint line type of 64 was arbitrarily chosen to indicate a line composed entirely of HP 2645 terminals.

```
:CLINE CSLINE                                                          &
: SUPLIST=64,LSEL=0(125,125,125,125),                                  &
:          GRP =ID=A"AA",A"aa",GEN=0(42,42),0(42,42),                  &
:               STA=0,512,A"AA",A"aa",                                 &
:               STA=0,512,A"BB","bb",                                  &
:          GRP =ID=A"BB","bb",                                         &
:               STA=0,512,A"AA",A"aa",                                 &
:               STA=0,512,A"BB",A"bb",                                 &
:              STA=0,512,A"CC",A"cc";
```

```
EQUATE
   HP2645        = 0,
   OUTLENGTH     = 100,
   INLENGTH      = 100;

BEGIN
BYTE ARRAY SUPLIST(0:63):=
   64,                              <<HP 2645 STATIONS>>
   1,                               <<LINE SELECT FACILITY>>
   0,2,                             <<TWO GROUPS>>
   4, 125, 125, 125, 125,           <<LINE SELECTION SEQUENCE>>
      7,2,                          <<LOGICAL GROUP 1, NUMBER OF STATIONS>>
      2,"AA",2,"aa",                <<GROUP I.D.>>
      2, 42, 42,2, 125, 125,        <<SUFFIXES FOR GENERAL POLL/SELECT>>
         HP2645,512,1,2,"AA",2,"aa",<<LOGICAL STATION 1>>
         HP2645,512,1,2,"BB",2,"bb",<<LOGICAL STATION 2>>
      7,3                           <<LOGICAL GROUP 2, NUMBER OF STATIONS>>
      2,"BB",2,"bb",                <<GROUP I.D.>>
      2, 42, 42,2, 125, 125,        <<SUFFIXES FOR GENERAL POLL/SELECT>>
         HP2645,512,1,2,"AA",2,"aa",<<LOGICAL STATION 3>>
         HP2645,512,1,2,"BB",2,"bb",<<LOGICAL STATION 4>>
         HP2645,512,1,2,"CC",2,"cc";<<LOGICAL STATION 5>>
ARRAY
```

```
   INBUF(0:99),OUTBUF(0:99);
BYTE ARRAY
   LINEDESIG(0:6):="CSLINE ";
BYTE ARRAY
   PARTIALSUPLIST(0:3);
INTEGER
   LINENUM,INLENGTH,OUTLENGTH;
LOGICAL
   STATION,FATALERROR;
DEFINE
   NUMSTATIONS         = INTEGER(PARTIALSUPLIST(2)#;

PROCEDURE INITIALIZE(NUMSTATIONS);
VALUE NUMSTATIONS;
INTEGER NUMSTATIONS;
OPTION EXTERNAL;

LOGICAL PROCEDURE ANALYZECSERROR(LINENUM);
VALUE LINENUM;
INTEGER LINENUM;
OPTION EXTERNAL;

INTEGER PROCEDURE PROCESSTRANSACTION(INBUF,INLENGTH,
   OUTBUF,STATION);
VALUE INLENGTH,STATION;
ARRAY INBUF,OUTBUF;
INTEGER INLENGTH;
LOGICAL STATION;
OPTION EXTERNAL;

INTRINSIC
   COPEN,CREAD,CWRITE,QUIT,CGETINFO,PRINT'LINE'INFO;

LINENUM:=COPEN(LINEDESIG,,,,,,,,SUPLIST);
IF <> THEN
   BEGIN
   PRINT'LINE'INFO;
   QUIT(0);
   END;

CGETINFO(LINENUM,,,,,,,,,,PARTIALSUPLIST,4);
INITIALIZE(NUMSTATIONS);

DO
   BEGIN  <<READ AND PROCESS ONE TRANSACTION>>
   INLENGTH:=CREAD(LINENUM,INBUF,INBUFLEN,,STATION);
   IF = THEN
      BEGIN
      OUTLEN:=PROCESSTANSACTION(INBUF,INLENGTH,OUTBUF,STATION);
      IF OUTLEN <> 0 THEN
```

```
        BEGIN
        CWRITE(LINENUM,OUTBUF,OUTLEN,,,-1);
        IF <> THEN FATALERROR:=ANALYZECSERROR(LINENUM);
        END;
     END;
  ELSE
     FATALERROR:=ANALYZECSERROR(LINENUM);
  END UNTIL FATALERROR;
END;
```

THE CS/3000 TRACE FACILITY — Section VI

Each CS/3000 intrinsic call, when performed by the driver, consists of a series of actions, operations, and events. An action or operation is something that the driver does and an event is something that the driver must react to. If you write an application program that includes CS/3000 intrinsics and the program doesn't work as it is supposed to, it can be extremely difficult to pinpoint the problem without a record of the actions, operations, and events that occurred. The CS/3000 trace facility provides such a record.

The trace facility is invoked only at the user's request. Tracing can be invoked for any or all communications lines that your program uses. The trace request is made either in the COPEN intrinsic call at the time a line is opened, in a :CLINE command prior to running your application program, or with a CCONTROL.

Once it has been invoked for a particular communications line, the trace facility continues to record line activity until the line is closed or a CCONTROL with CONTROLCODE of 43 is executed. The trace facility keeps track of actions, operations, and events in the form of trace entries. The trace entries are grouped into trace records, each composed exclusively of CS/3000 intrinsic entries, protocol driver entries, or interconnect driver entries. The trace records are stored permanently in a file known as the trace file. The contents of a CS/3000 trace file can be formatted and printed through the use of a trace dump utility program that is described later in this section.

6.1 DEFINING THE TRACE FILE

The Trace Facility uses the file "CSTRAldn" as the trace file if no alternate file is specified by the user, where ldn is the logical device number of the line to be traced.

PARAMETER(1) of CCONTROL 44 is used to specify an alternate trace file designator. This trace file designator will have the logical device number of the line appended to it to ensure a unique trace file for each line. The ldn is appended after the last character in the designator name if the name is less than five characters in length. If the designator name is

greater than or equal to five characters in length, the device number will replace characters 6 through n of the designator name.

As an example, if the trace file designator "CSTRACE" is specified for the line with ldn of 148, then the designator used will be "CSTRA148".

6.2 INVOKING THE TRACE FACILITY

To invoke the CS/3000 trace facility for a particular communications line, do one of the following:

A. Set bit (2:1) of COPTIONS to "1" in the COPEN intrinsic call that opens the line.

B. Include the TRACE= parameter in a :CLINE command for the line prior to running your application program.

C. execute a CCONTROL with CCONTROLCODE of 44.

The trace facility will then be invoked for the line (when the line is opened for either steps A or B of the above).

Note: If a COPEN call in your application program requests tracing and you wish to suppress tracing for that line at run time, include the NOTRACE parameter in a :CLINE command for that line prior to running your application program. Conversely, if a COPEN call in your application program does not request tracing and you wish to request tracing for that line at run time, include the TRACE= parameter in a :CLINE command for that line prior to running your application program.

In addition to invoking the trace facility, you can specify other trace-related information by using the CTRACEINFO parameter of the COPEN and CCONTROL intrinsic call. The format of the CTRACEINFO parameter is as follows:

(0:1) 0 = trace I/O errors only
1 = trace all activities

(1:1) 0 = if the trace table is full, succeeding entries will be flushed.
1 = if the trace table is full, succeeding entries will overlay the prior entries.

(2:6) 0 = use driver default trace mask
>0 = trace mask indicating what types of trace entries are to be generated. The format of the trace mask for the BSC and MRJE protocols is as follows:

```
                bit 2 = generate STN entries
                    3 = generate OPR and EDT entries  *
                    4 = generate RCT entries  *
                    5 = generate RTX entries  *
                    6 = generate SCT, POL, SEL entries  *
                    7 = generate STX entries  *
                Note:  CMP entries are automatically generated

                *  default

    (10:1)  0 = do not generate SCCP interconnect driver entries.
            1 = generate SCCP interconnect driver entries.

    (11:5)  0 = use driver default maximum number of trace entries per
                record
           >0 = maximum number of trace entries per trace record / 8
                (ie: to specify 8 entries/record, use 1)
```

The CTRACEINFO parameters shown above can be specified instead (or altered) at run time by including the TRACE= parameter in a : CLINE command for the particular line prior to running your application program.

The trace facility produces a series of trace records, each consisting of a series of trace entries. If you specify that only I/O errors are to be traced, the trace facility will deposit in the trace file only those records in which an I/O error occurred.

The types of trace entries mentioned under the CTRACEINFO parameter above (STN, CMP, OPR, interconnect, etc.) are described in detail later in this section. At this point it is sufficient to say that you can suppress the various types of trace entries, if you so desire.

Trace entries are deposited in a trace record in a circular manner. For example, if you specify that there be a maximum of 35 trace entries per trace record, trace entries beyond the 35th will overlay the first, second, third, etc., trace entries in the record. If this happens, trace entries that have been overlayed will be missing from the listing and a warning message will appear at the start of the record in the listing telling you that the particular number of entries are missing.

If the CTRACEINFO parameter is omitted from the COPEN intrinsic call, and if the corresponding parameters are not supplied in a :CLINE command, the drivers use the following default specifications:

- o trace I/O errors only
- o flush overflow trace entries
- o generate all entry-types except STN
- o do not generate SCCP interconnect driver entries

- maximum of 25 trace entries per trace record

6.3 THE TRACE FILE

Each line opened by your process that requires tracing will have their trace information written to the trace file specified, or to "CSTRAldn" if no name was specified.

If tracing has been requested and the trace file is not yet open, the CS/3000 trace facility issues an FOPEN intrinsic call with the following parameters:

Parameter		Value	
formal file designator		CSTRACE	
FOPTIONS	(14:2)	11	(old file)
	(13:1)	0	(binary file)
	(10:3)	0	(use actual file designator)
	(8:2)	0	(fixed length records)
	(7:1)	0	(no carriage control)
	(6:1)	0	
	(5:1)	1	(disallow file equation)
	(0:5)	0	
AOPTIONS	(12:4)	4	(input/output access)
	(11:1)	0	(no multi-record option)
	(10:1)	0	(disallow dynamic locking/unlocking)
	(8:2)	0	(exclusive access)
	(0:8)	0	
blockfactor		1	

If the trace file cannot be opened because it does not exist, then a new file will be opened in the system domain.

If an error occurs when trying to open the trace file, the particular COPEN intrinsic call will fail.

When the line is closed, the CS/3000 trace facility issues an FCLOS intrinsic call with the following parameters:

Parameter	Value
filenum	trace file number
disposition	1 (save)
seccode	0 (unrestricted access)

6.4 TRACE DUMP PROGRAM

There is a CS/3000 trace dump utility program (CSDUMP) in the syste library that will format and print the contents of trace files generated b the CS/3000 trace facility. The trace dump program requires a trace fil and a list file.

6.5 List File

The formal file designator of the list file is LIST. The list file may b defined as a CRT terminal, a line printer, or an old disc file. To defin the list file, enter an MPE :FILE command prior to invoking the trace dum program. Some typical examples are as follows:

:FILE LIST;DEV=LP	(LP is assumed to be the device class nam for one or more line printers)
:FILE LIST;DEV=filename	(filename is assumed to be the name of an ol job or system disc file)

If a list file is not defined by a :FILE command, the trace dump program uses $STDLIST as the list file. If a :FILE command is used to define the list file, the file is opened as an old file. Should the file not exist, then it is opened as a new file in the system domain. After the dump program has run, the file is closed as a permanent file with unlimited access. The contents of this file may be examined by the HP/3000 editor, EDIT/3000.

6.6 Invoking the Trace Dump Program

After the trace and list files have been defined , enter the following command:

```
:RUN CSDUMP.PUB.SYS;PARM=x
```

The trace dump program uses the trace file as input and produces a formatted trace listing on the list file. The format of the trace file listing is specified by the PARM parameter of the RUN command:

PARM	Format
0,1	List all trace file entries by time of occurance.
2	List all CS/3000 intrinsic entries by time, with driver entries listed by time under their associated intrinsic entry.
3	List all CS/3000 intrinsic entries by time.

The default for PARM is 0. A sample trace listing is shown in Figure 6-1.

Figure 6-1. Sample Trace Listing.

```
CS TRACE ANALYZER (A.02.00)       MON, MAR  6, 1978, 12:45 AM

TRACE FILE IS CSTRACE.PUB.MP
ALL ENTRIES DUMPED BY TIME

LAST OPENED ON MON, MAR  6, 1978, 12:01 AM

SYSTEM ID=32.17


*********************************
* BEGIN TRACING FOR DEVICE 32 *
*********************************

*****************************************************
*-L-I-N-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y-*
*****************************************************
*  LINE NUMBER: 4       LOGICAL DEV. NUMBER: 32    *
*  DEV. TYPE:  19       SUBTYPE:  3  VER: A.02.00  *
*            0123456789012345                       *
*  COPTIONS: 0010000010000001                       *
*  AOPTIONS: 0000000100001100                       *
*  DOPTIONS: 0000000000000000                       *
*  NUMBUFFERS: 1           BUFFSIZE: 512  (WORDS)   *
*  INSPEED: 300            OUTSPEED: 300            *
*  MISCARRAY:        RECEIVE TIMEOUT: 7200  SECS.   *
*                      LOCAL TIMEOUT: 7200  SECS.   *
*                    CONNECT TIMEOUT: 7200  SECS.   *
*                   RESPONSE TIMEOUT: 3     SECS.   *
*                   LINE BID TIMEOUT: 60    SECS.   *
*                  NO. ERROR RETRIES: 7             *
*                CLEAR-TO-SEND DELAY: 00.0  SECS.   *
*               DATA-SET-READY DELAY: DISABLED      *
*                  TRANSMISSION MODE: HALF DUPLEX.  *
*              MMSTAT TRACE FACILITY: DISABLED.     *
*  DRIVERNAME: CSSBSC0     LINESTATE: UNCONNECTED   *
*  CTRACEINFO:   ENTRIES=25      MASK=011111        *
*                TYPE OF TRACE = ALL, NOWRAP        *
*  PHONELIST:   ENTRIES=0          INDEX=0          *
*  IDLIST:      ENTRIES=0          INDEX=0          *
*  ERRORCODE:  RECOVERABLE=0    IRRECOVERABLE=0     *
*  MSGSENT: 0                MSGRECV: 0             *
*  RECOVERRORS: 0            IRRECOVERRORS: 0       *
*****************************************************
```

Figure 6-1. Sample Trace Listing.

```
*************************************************************************
* CREAD                              REQUEST ID = 000505                *
* CALLER: SEGMENT=PRG  000           ADDRESS= 000122                    *
* STATE:  LINE STATE=DISCONNECT      COPTIONS= 020201  DOPTIONS= 000000 *
* INPUT:  IN BUF= 000076 LENGTH=-28      STATION #=0   COMPONENT #=0    *
* OUTPUT: TRANSMISSION LOG=-4            STATION #=0   COMPONENT #=0    *
*************************************************************************

0         1.722 POPR REQUEST ID= 000505(!0145)
                    WAIT FOR CONNECTION THEN RECEIVE CONTROL SEQ
                    TIMEOUT= 7200.000
                    IN BUFR= 0.021145 LENGTH=-16

1         6.611 PRCT REQUEST ID= 000505(!0145)
                    205.377 377.377


2         6.612 PEDT REQUEST ID= 000505(!0145)
                    RECV ENQUIRY      XLOG=0

3         6.616 POPR REQUEST ID= 000505(!0145)
                    SEND CONTROL SEQ THEN RECEIVE TEXT
                    SEND SEQ=ID ACK    TIMEOUT= 7200.000
                    OUT BFR= 0.000000 LENGTH=0
                    IN BUFR= 1.000044 LENGTH=-28

4         6.617 PSCT REQUEST ID= 000505(!0145)
                    020.260
                        0

5         6.725 PRTX REQUEST ID= 000505(!0145)
                    020.002 120.060 070.040 020.203 026.240 377.000
                            P   0   8
                    377.377


6         6.727 REDT REQUEST ID= 000505(!0145)
                    RECV TEXT          XLOG=-6
                    200.000 120.060 070.040
                            P   0   8

7         6.730 PCMP REQUEST ID= 000505(!0145)
                    ERROR CODE=0      LAST RECOVERABLE ERROR CODE= 0
                    #MSG SENT=0       #MSG RECV=1       STATE=TEXT
                    # RECOVERABLE ERR=0       # IRRECOVERABLE ERR=0


*************************************************************************
* CCLOSE                             REQUEST ID = 000777                *
* STATE:  LINE STATE=TEXT            COPTIONS= 020201  DOPTIONS= 000000 *
*************************************************************************
```

figure 6-1. Trace Dump Listing

```
0         8.912 POPR REQUEST ID= 000777(!01FF)
                    UNCONDITIONAL CLEAR OF ANY CURRENT OPERATION

1         8.914 POPR REQUEST ID= 000777(!01FF)
                    SEND CONTROL SEQ THEN DISCONNECT
                    SEND SEQ=EOT       TIMEOUT=     3.000

2         8.915 PSCT REQUEST ID= 000777(!01FF)
                    026.004


3         8.975 PCMP REQUEST ID= 000777(!01FF)
                    ERROR CODE=0      LAST RECOVERABLE ERROR CODE= 0
                    #MSG SENT=0       #MSG RECV=1       STATE=DISCONNECT
                    # RECOVERABLE ERR=0       # IRRECOVERABLE ERR=0


**********************************
* END OF TRACE FOR DEVICE  32   *
**********************************

*****************************************************
*-L-I-N-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y-*
*****************************************************
*  LINE NUMBER: 4       LOGICAL DEV. NUMBER: 32    *
*  DEV. TYPE:  19       SUBTYPE:  3  VER: A.02.00  *
*            0123456789012345                       *
*  COPTIONS: 0010000010000010                       *
*  AOPTIONS: 0000000100001100                       *
*  DOPTIONS: 0000000000000000                       *
*  NUMBUFFERS: 1            BUFFSIZE: 512  (WORDS)  *
*  INSPEED: 300             OUTSPEED: 300           *
*  MISCARRAY:         RECEIVE TIMEOUT: 7200  SECS.  *
*                       LOCAL TIMEOUT: 7200  SECS.  *
*                     CONNECT TIMEOUT: 7200  SECS.  *
*                    RESPONSE TIMEOUT: 3     SECS.  *
*                    LINE BID TIMEOUT: 60    SECS.  *
*                  NO. ERROR RETRIES: 7             *
*                CLEAR-TO-SEND DELAY: 00.0  SECS.   *
*               DATA-SET-READY DELAY: DISABLED      *
*                  TRANSMISSION MODE: HALF DUPLEX.* 
*              MMSTAT TRACE FACILITY: DISABLED.     *
*  DRIVERNAME: CSSBSC0    LINESTATE: UNCONNECTED    *
*  CTRACEINFO:   ENTRIES=25      MASK=011111        *
*                TYPE OF TRACE = ALL,NOWRAP         *
*  PHONELIST:   ENTRIES=0          INDEX=0          *
*  IDLIST:      ENTRIES=0          INDEX=0          *
*  ERRORCODE:  RECOVERABLE=0   IRRECOVERABLE=0      *
*  MSGSENT: 0               MSGRECV: 1              *
*  RECOVERRORS: 0           IRRECOVERRORS: 0        *
*****************************************************
```

figure 6-1. Trace Dump Listing

```
END OF JOB.
```

6.7 Trace Listing Header Message

```
CS TRACE ANALYZER (A.02.00)      MON, MAR  6, 1978, 12:32 AM

TRACE FILE IS CSTRACE.PUB.MP
ALL ENTRIES DUMPED BY TIME

LAST OPENED ON MON, MAR  6, 1978, 12:01 AM

SYSTEM ID=02.66
```

At the start of the trace listing is a header message telling the date and time-of-day when the listing was printed, the fully-qualified name of the trace file being used, and the format of the trace listing. The meanings of the two remaining items in the header message are as follows:

LAST OPENED ON etc.	This tells you the date and time-of-day when the trace was performed.
SYSTEM ID=xx.yy	This tells you the version number (xx) and fix level (yy) of the MPE/3000 operating system that was being used when the trace was performed.

6.8 "Begin Tracing" Message

```
*********************************
* BEGIN TRACING FOR DEVICE 32 *
*********************************

*****************************************************
*-L-I-N-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y-*
*****************************************************
*  LINE NUMBER: 4       LOGICAL DEV. NUMBER: 32    *
*  DEV. TYPE:  19       SUBTYPE:  3  VER: A.02.00  *
*            0123456789012345                       *
*  COPTIONS: 0010000010000001                       *
*  AOPTIONS: 0000000100001100                       *
*  DOPTIONS: 0000000000000000                       *
*  NUMBUFFERS: 1           BUFFSIZE: 512  (WORDS)   *
*  INSPEED: 300            OUTSPEED: 300            *
*  MISCARRAY:        RECEIVE TIMEOUT: 7200  SECS.   *
*                      LOCAL TIMEOUT: 7200  SECS.   *
*                    CONNECT TIMEOUT: 7200  SECS.   *
*                   RESPONSE TIMEOUT: 3     SECS.   *
*                   LINE BID TIMEOUT: 60    SECS.   *
*                  NO. ERROR RETRIES: 7             *
*                CLEAR-TO-SEND DELAY: 00.0  SECS.   *
*               DATA-SET-READY DELAY: DISABLED      *
*                  TRANSMISSION MODE: HALF DUPLEX.  *
*              MMSTAT TRACE FACILITY: DISABLED.     *
*  DRIVERNAME: CSSBSC0    LINESTATE: UNCONNECTED    *
*  CTRACEINFO:   ENTRIES=25     MASK=011111          *
*                TYPE OF TRACE = ALL, WRAP          *
*  PHONELIST:   ENTRIES=0         INDEX=0            *
*  IDLIST:      ENTRIES=0         INDEX=0            *
*  ERRORCODE:  RECOVERABLE=0   IRRECOVERABLE=0      *
*  MSGSENT: 0              MSGRECV: 0               *
*  RECOVERRORS: 0          IRRECOVERRORS: 0         *
*****************************************************
```

The message "BEGIN TRACING FOR DEVICE etc." appears at each point in the listing where a line to be traced is opened. The message tells you the decimal logical device number of the line (32 in the above example). It indicates that the particular line's activities are now being monitored by the trace facility. It is followed by the PRINT'LINE'INFO "tombstone" describing the line's state when tracing started.

6.9 "End of Trace" Message

```
*********************************
* END OF TRACE FOR DEVICE  32   *
*********************************

*****************************************************
*-L-I-N-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y-*
*****************************************************
*  LINE NUMBER: 4       LOGICAL DEV. NUMBER: 32    *
*  DEV. TYPE:  19       SUBTYPE:  3  VER: A.02.00  *
*            0123456789012345                       *
*  COPTIONS: 0010000010000010                       *
*  AOPTIONS: 0000000100001100                       *
*  DOPTIONS: 0000000000000000                       *
*  NUMBUFFERS: 1           BUFFSIZE: 512  (WORDS)   *
*  INSPEED: 300            OUTSPEED: 300            *
*  MISCARRAY:        RECEIVE TIMEOUT: 7200  SECS.   *
*                      LOCAL TIMEOUT: 7200  SECS.   *
*                    CONNECT TIMEOUT: 7200  SECS.   *
*                   RESPONSE TIMEOUT: 3     SECS.   *
*                   LINE BID TIMEOUT: 60    SECS.   *
*                  NO. ERROR RETRIES: 7             *
*                CLEAR-TO-SEND DELAY: 00.0  SECS.   *
*               DATA-SET-READY DELAY: DISABLED      *
*                  TRANSMISSION MODE: HALF DUPLEX.  *
*              MMSTAT TRACE FACILITY: DISABLED.     *
*  DRIVERNAME: CSSBSC0    LINESTATE: UNCONNECTED    *
*  CTRACEINFO:   ENTRIES=25      MASK=000101         *
*                TYPE OF TRACE = ERROR, NOWRAP      *
*  PHONELIST:   ENTRIES=0          INDEX=0          *
*  IDLIST:      ENTRIES=0          INDEX=0          *
*  ERRORCODE:  RECOVERABLE=0    IRRECOVERABLE=0     *
*  MSGSENT: 0              MSGRECV: 1               *
*  RECOVERRORS: 0          IRRECOVERRORS: 0         *
*****************************************************
```

The message "END OF TRACE FOR DEVICE etc." appears at each point in the listing where a line that is being traced is closed. The message tells you the decimal logical device number of the line (32 in the above example) and indicates that the particular line's activities are no longer being monitored by the trace facility. It is followed by the PRINT'LINE'INFO "tombstone" showing the line's state just before tracing was stopped.

6.10 CS/3000 Intrinsic Entry Header Message

```
*************************************************************************
* CREAD                              REQUEST ID = 000505                *
* CALLER: SEGMENT=PRG  000           ADDRESS= 000122                    *
* STATE:  LINE STATE=DISCONNECT      COPTIONS= 020201  DOPTIONS= 000000 *
* INPUT:  IN BUF= 000076 LENGTH=-28      STATION #=0   COMPONENT #=0    *
* OUTPUT: TRANSMISSION LOG=-4            STATION #=0   COMPONENT #=0    *
*************************************************************************
```

The trace listing is organized into a series of CS/3000 intrinsic entries and driver entries related to each intrinsic entry. The format of the listing is specified when the dump program is invoked(see section 6.6).

The trace record header message specifies a request id used to identify driver entries related to the intrinsic call, the type of intrinsic, the location of the user's program where the intrinsic was invoked, the line's state, and the user's calling parameters.

6.11 "Missing Entries" Message

```
MISSING PROTOCOL ENTRIES
MISSING INTERCONNECT ENTRIES
```

If the message "MISSING PROTOCOL ENTRIES" or "MISSING INTERCONNECT ENTRIES" appears in the listing it means that the specified record was not large enough to accomodate all of the trace entries and some entries were lost. If the tracing was performed with NOWRAP then the missing entires were at the end just before the PCMP entry or the IDX entry. Otherwise they are missing at the beginning. If the missing entries are crucial, do as follows:

a. Purge the trace file.

b. Change the "number of entries per record" specification by way of the TRACE=,,numentries parameter in a :CLINE command for the particular line.

c. Re-run the application program.

6.12 Driver Trace Entry Format

```
8          8.852 POPR REQUEST ID= 000505(!0145)
                  SEND CONTROL SEQ THEN RECEIVE CONTROL SEQ
                  SEND SEQ=WACK      TIMEOUT= 7200.000
                  IN BUFR= 0.021145 LENGTH=1
```

All driver entries in a trace listing contain a prefix consisting of four fields:

1. an entry number (8 in the above example)
2. a "time stamp" in seconds and thousandths of seconds (8.852 in the above example)
3. an entry-type mnemonic (POPR in the above example)
4. a request id

The first entry of each trace record is numbered "0" and successive entries in the particular trace record are numbered consecutively in ascending order.

The "time stamp" makes it possible for you to determine the elapsed time in seconds and milliseconds between one trace entry and another. Note: time stamp resolution for entries generated by an SCCP associated protocol driver is one hundredth of a second.

The entry-type mnemonic tells you what type of trace entry you are examining. There are ten types of protocol driver trace entries and seventeen types of interconnect driver trace entries; they are summarized in Tables 6-1 and 6-2.

The body of each trace entry tells you the pertinent information for the particular activity that has happened or is about to happen. This part of the trace entry is described in detail for each entry-type on the remaining pages of this section.

Table 6-1. Protocol Driver Trace Entry Type Mnemonic Definitions

MNEMONIC	ENTRY TYPE	DEFINITION
POPR	Operation	This type of trace entry is generated each time the physical driver is called upon to perform an operation. The POPR trace entry tells what operation is to be performed.
PSTN	State Transition Entry	This type of trace entry is generated each time the driver transfers from one internal state to another. The PSTN trace entry tells what event just happened and what action is about to be performed.
PEDT	Editor Entry	This type of trace entry is generated each time a text message or control character sequence is received from the remote station. In the case of a text message, the PEDT trace entry shows the first 13 (for HSI) or 14 (for SSLC) words of the user's buffer; control characters, pad characters, and CRC parity sequences are omitted. In the case of a control character sequence, the PEDT trace entry supplies a mnemonic phrase telling what was received.
PRCT	Receive Control Sequence Entry	This type of trace entry is generated each time a control character sequence is received from the remote station. The PRCT trace entry shows (in octal) byte-for-byte exactly what was received.
PSCT	Send Control Sequence Entry	This type of trace entry is generated each time the driver sends a control character sequence to the remote station. The PSCT trace entry shows (in octal) byte-for-byte exactly what was sent.

Table 6-1. Protocol Driver Trace Entry Type Mnemonic Definitions

MNEMONIC	ENTRY TYPE	DEFINITION
PPOL	Send Polling Sequence Entry	This type of trace entry is generated each time the driver sends a polling sequence. The PPOL sequence shows byte-for-byte the polling sequence. All BSC control characters (such as SYNs, EOT, and ENQ) are not shown.
PSEL	Send Selection Sequence Entry	This type of trace entry is generated each time the driver sends a selection sequence. The PSEL sequence shows byte-for-byte the selection sequence. All BSC control characters (such as SYNs, EOT, and ENQ) are not shown.
PRTX	Receive Text Entry	This type of trace entry is generated each time a text message is received from the remote station. The PRTX trace entry shows (in octal) byte-for-byte exactly what was received.
PSTX	Send Text Entry	This type of trace entry is generated each time the driver sends a text message to the remote station. The PSTX entry shows (in octal) byte-for-byte exactly what was sent.
PCMP	User Request Completed	This type of trace entry is generated each time a user request (i.e., a CREAD, CWRITE, driver-performed CCONTROL, or CCLOSE intrinsic call) is completed. The PCMP trace entry summarizes the number of text messages sent and received, the number of errors that have occurred, etc.

Table 6-2 Interconnect Driver Trace Entry Mnemonic Definitions

MNEMONIC	ENTRY TYPE	DEFINITION
IDC	Driver Called	This entry is generated whenever the driver is called to perform an operation.
IDX	Driver Exited	This entry is generated whenever the driver completes an execution of the main control routine.
IADQ	Add to Queue	This entry is generated whenever the driver adds a request to one of its internal queues.
IRFQ	Remove from Queue	This entry is generated whenever the driver removes a request from one of its internal queues.
IDF	Data Frozen	This entry is generated whenever the driver requests a target data segment to be frozen in memory or to check if a previous request to freeze a data segment has been completed.
IUNF	Unfreeze Data	This entry is generated whenever driver wishes to unfreeze a previously frozen data segment or to insure that a data segment associated with a request is not frozen by the driver.
INR	New Request	This entry is generated each time the driver begins processing a new request.
IPR	Process Request	This entry is generated whenever the driver processes a request which may be completed immediately(ie: requires no I/O to SCCP) or whenever a request requires some preprocessing before I/O is to be done.
IAR	Abort Request	This entry is generated whenever a request is to be hard aborted.

Table 6-2 Interconnect Driver Trace Entry Mnemonic Definitions

STO	Start Timeout	This entry is generated whenever the driver starts a software timeout on a request.
ISS	Set Status	This entry is generated whenever the request completion status is set.
ICR	Complete Request	This entry is generated whenever a request has been fully completed by the driver and is released to the request initiator.
ICC	Check Completion	This entry is generated whenever the driver calls the physical driver to check I/O completion status and to check for software timeout completions.
IPM	Process Message	This entry is generated whenever the driver receives a message from SCCP.
IPFR	Power Fail Recovery	This entry is generated each time the power fail recovery routine is called.
ICD	Call Driver	This entry is generated each time the physical driver is called to perform an operation.
IDIO	Do I/O	This entry is generated each time the driver wishes to do an operation which sends a message to SCCP or moves data to/from sccp or moves data between requests.
IRB	Illogical Condition	This entry is generated whenever the driver detects an illogical internal condition or receives an erroneous or illogical message from SCCP.

6.13 POPR (Operand) Trace Entries

```
3          6.616 POPR REQUEST ID= 000505(!0145)
                  SEND CONTROL SEQ THEN RECEIVE TEXT
                  SEND SEQ=ID ACK    TIMEOUT=    30.000
                  OUT BFR= 0.000000 LENGTH=0
                  IN BUFR= 1.000044 LENGTH=-28
```

A POPR trace entry is generated each time the physical driver is called upon to perform an operation.

The meanings of the various items are as follows:

SEND CONTROL SEQ etc.	This item tells you what operation is being performed.
SEND SEQ=	This item tells you what control character sequence, if any, is about to be sent to the remote station (ID ACK in the above example).
TIMEOUT= etc.	This item tells you the starting value of the applicable timeout timer. In the above example the driver sends an ID ACK to the remote station and then waits for a character response. "TIMEOUT= 30.000" specifies that the response timeout timer will be activated and set to 30 seconds.
OUT BFR=x.xxxxxx	This item specifies bank and bank offset of the user/CS output buffer.
IN BFR=x.xxxxxx	This item specifies the bank and bank offset of the user/CS input buffer.
LENGTH=y	These two items specify the size (+=words;-=bytes) of the CS output and input buffers, respectively.

6.14 PSTN (State Transition) Trace Entries

```
16        3.167 PSTN REQUEST ID= 000505(!0145)
                EVENT IS RECEIVED WACK SEQUENCE
                ACTION IS  SEND ENQ, GET TEXT/RESPONSE ( 033)
                RETRY COUNTER=0    STATE= 0640
                PENDING EVENT= 000000 EVENT MASK= 024000
```

An PSTN trace entry is generated each time the driver transfers from one internal state to another.

The meanings of the various items are as follows:

EVENT IS etc.	This line tells you what event just occurred. Look up the event phrase (RECEIVED WACK SEQUENCE in the above example) in Table 6-2 for a detailed description.
ACTION IS etc.	This line tells you what action the driver is about to perform. The action number (033 in the above example) is of no concern to you and should be ignored. Look up the action phrase (SEND ENQ, GET TEXT/RESPONSE in the above example) in Table 6-3 for a detailed description.
RETRY=	This item tells you the current value of the retry counter. For certain error conditions, the driver increments the retry counter and then re-executes the operation in which the error occurred.
STATE=	This item tells you what driver state is to be entered next.
PENDING EVENT=	This item tells you the current contents of bits 0-7 of the word EXTERNAL'EVENT in the driver.
EVENT MASK=	This item tells you the current contents of bits 8-15 of the word EXTERNAL'EVENT in the driver.

6.15 PEDT (Editor) Trace Entries

```
6          6.727 PEDT REQUEST ID= 000505(!0145)
                 RECV TEXT          XLOG=-6
                 200.000 120.060 070.040
                         P   0   8
```

PEDT trace entries are generated in conjunction with received text messages and received control character sequences.

In the case of a received text message, the body of an EDT trace entry shows in octal the first 13 (for HSI) or 14 (for SSLC) words of the text message. Regardless of how long the text message is, only one PEDT trace entry will be generated per received text message. The text message is shown as it appears in the user's buffer: control characters, pad characters, and CRC parity sequences are omitted. XLOG specifies the total number of words or bytes (+=words;-=bytes) deposited into the user's buffer. If the MFW (1 word or 2 bytes) is used then it is also included in XLOG.

In the case of a received control character sequence, the PEDT trace entry includes a mnemonic phrase telling what control character sequence was received. If the control character sequence was accompanied by an ID sequence, the ID sequence is shown in octal below the mnemonic phrase. XLOG will normally be zero, except in the case when an ID sequence was also received. In the latter case, XLOG specifies the length of the ID sequence (+=words;-=bytes).

Wherever possible, the trace dump program converts the received octal codes to a character and displays the character beneath its code. The translation from code to character is performed properly for EBCDIC transmissions as well as ASCII transmissions.

In the example above (RECV TEXT), the body of the PEDT trace entry is interpreted as follows:

> A text message was received from the remote station. The total number of data characters received was 4 (XLOG=-4). The 4 data characters received are
>
> 120 = P 060 = 0
>
> 070 = 8 040 = space

6.16 PRCT (Receive Control Sequence) Trace Entries

```
6          1.026 PRCT REQUEST ID= 000505(!0145)
                  020.260 377.377
                     0
```

An PRCT trace entry is generated each time a control character sequence is received from the remote station. The body of an PRCT trace entry shows you byte-for-byte exactly what was received (i.e., the actual contents of the driver's internal buffer).

Control character sequences are terminated by a trailing pad character (377 octal). When interpreting the body of an PRCT trace entry, ignore anything following the 377 code.

In the above example, an ACK0 control character sequence was received. The octal codes are interpreted as follows:

020 = DLE

260 = 0 (060 with parity bit set) ACK0 sequence

377 = trailing pad character

Wherever possible, the trace dump program converts the octal codes to a character and displays the character beneath its code. The translation from code to character is performed properly for EBCDIC transmissions as well as ASCII transmissions.

Note: The trace facility will not generate PRCT trace entries if the HSI driver is being used. In such a case, to trace received control character sequences PEDT trace entries must be generated.

6.17 PSCT (Send Control Sequence) Trace Entries

```
5          0.956 PSCT REQUEST ID= 000777(!01FF)
                       205.377
```

An PSCT trace entry is generated each time the driver sends a contro character sequence to the remote station. The body of an PSCT trace entr shows you byte-for-byte exactly what was sent to the remote station (i.e. the actual contents of the driver's internal buffer).

In the above example, an ENQ control character was sent. The octal code are interpreted as follows:

205 = ENQ (005 with parity bit set)

377 = PAD (This is the trailing pad character)

Wherever possible, the trace dump program converts the octal codes to character and displays the character beneath its code. The translatio from code to character is performed properly for EBCDIC transmissions a well as ASCII transmissions.

6.18 PPOL (Send Polling Sequence) Trace Entries

```
6          3.807 PPOL REQUEST ID= 000777(!01FF)
                       301.301 302.302
                       A   A   B   B
```

A PPOL trace entry is generated each time the driver sends a polling sequence to the remote group/station. The body of a PPOL trace entry shows you byte-for-byte the polling sequence sent on the line. It does not show the BSC control characters that accompany the polling sequence such as SYN, EOT, and ENQ.

In the above example, the polling sequence was the ascii string, AABB.

Wherever possible, the trace dump program converts the octal codes to a character and displays the character beneath its code. The translation from code to character is performed properly for EBCDIC transmissions as well as ASCII transmissions.

6.19 PSEL (Send Selection Sequence) Trace Entries

```
3          4.096 PSEL REQUEST ID= 000777(!01FF)
                       103.103 304.304
                       C   C   D   D
```

A PSEL trace entry is generated each time the driver sends a selection sequence to the remote group/station. The body of a PSEL trace entry shows you byte-for-byte the selection sequence sent on the line. It does not show the BSC control characters that accompany the selection sequence such as SYN, EOT, and ENQ.

In the above example, the selection sequence was the ascii string, CCDD.

Wherever possible, the trace dump program converts the octal codes to a character and displays the character beneath its code. The translation from code to character is performed properly for EBCDIC transmissions as well as ASCII transmissions.

6.20 PRTX (Receive Text) Trace Entries

```
5          6.725 PRTX REQUEST ID= 000777(!01FF)
                  020.002 120.060 070.040 020.203 026.240 377
                          P   0   8
```

PRTX trace entries are generated each time a text message is received from the remote station. The body of an PRTX trace entry shows you byte-for-byte exactly what was received (i.e., the actual contents of the driver's internal buffer).

Text messages are terminated by a two-byte CRC (cyclic redundancy check) parity sequence followed by a trailing pad character (377 octal). When interpreting the body of an PRTX trace entry, ignore anything following the 377 code.

In the above example, the octal codes are interpreted as follows:

020 = DLE	002 = STX
120 = P	060 = 0
070 = 8	040 = space
020 = DLE	203 = ETX (003 with parity bit set)
026 =	240 = CRC parity sequence
	377 = trailing pad character

Each PRTX trace entry can show a maximum of 32 bytes. If a text message exceeds this length, as many successive PRTX trace entries will be generated as are necessary.

Wherever possible, the trace dump program converts the octal codes to a character and displays the character beneath its code. The translation from code to character is performed properly for EBCDIC transmissions as well as ASCII transmissions.

Note: The trace facility will not generate PRTX entries if the HSI driver is being used. In such a case, to trace received text messages PEDT trace entries must be generated.

.21 PSTX (Send Text) Trace Entries

```
13        1.037 PSTX REQUEST ID= 000505(!0145)
                020.002 120.060 070.040 020.203 026.240 377
                          P   0   8
```

PSTX trace entries are generated each time the driver sends a text message to the remote station. The body of an PSTX trace entry shows you byte-for-byte exactly what was sent to the remote station (i.e., the actual contents of the driver's internal buffer).

In the above example, the octal codes are interpreted as follows:

020 = DLE	122 =R
002 = STX	040 = space
120 = P	060 = 0
070 = 8	040 = space
020 = DLE	203 = ETX (003 with parity bit set)
026 =	240 = CRC parity sequence

Each PSTX trace entry can show a maximum of 32 bytes. If a text message exceeds this length, as many successive PSTX trace entries are generated as are necessary. The above example is such a case and was continued into another PSTX trace entry.

Wherever possible, the trace dump program converts the octal codes to a character and displays the character beneath its code. The translation from code to character is performed properly for EBCDIC transmissions as well as ASCII transmissions.

Note: The Trace Facility will only trace the first 32 bytes of sent text if the HSI driver is being used.

6.22 PCMP (I/O completion) Trace Entries

```
22     3.332 PCMP REQUEST ID= 000505(!0145)
                 ERROR CODE=210   LAST RECOVERABLE ERROR CODE= 0
                 #MSG SENT=0      #MSG RECV=0      STATE=CONTROL
                 # RECOVERABLE ERR=0      # IRRECOVERABLE ERR=0
```

A PCMP trace entry is generated each time a user request (i.e., a CREAD, CWRITE, driver-performed CCONTROL, or CCLOSE intrinsic call) is completed. The meanings of the various items are as follows:

Error Code=X	xxx is the error code of the request's most recent recoverable error. See Section 4.12 of this manual.
Last Recoverable Error Code=X	If a recoverable error occurred, x is its error code. See section 4.13 of this manual.
#MSG SENT=X	x specifies the total number of text messages that have so far been sent for this connection.
#MSG RECV=X	x specifies the total number of text messages that have been received so far for this connection.
#RECOVERABLE ERR=X	x specifies the total number of recoverable errors that have occurred so far for this connection.
#IRRECOVERABLE ERR=X	x specifies the total number of irrecoverable errors that have occurred so far for this connection.
STATE=	This item specifies what line state the line is in after the completion of the user request (CONTROL state in the above example).

.xx IADQ (Add Request to Queue) Trace Entries

```
3      7.075 IADQ REQUEST ID= 000505(!0145)   QUEUE=ACTIVE   QUEUE LEN=5
                 FUNCTION=READ                ERRORCODE=0
```

An IADQ trace entry is generated whenever the driver adds a request to one of its internal queues.

The meanings of the various items are as follows:

REQUEST ID= etc.	This item tells you the id of the request responsible for this driver action. Id values are given in octal and hex.
QUEUE= etc.	This item tells you the queue to which the request is added.
QUEUE LEN=y	This item specifies the number of requests queued to the above queue.
FUNCTION= etc.	This item tells you the SCCP function specified in the request.
ERRORCODE=X	This item specifies an irrecoverable error code as defined in Chapter 4.

x.xx IRFQ (Remove Request From Queue) Trace Entries

```
4      7.075 IRFQ REQUEST ID= 000505(!0145)   QUEUE=ACTIVE   QUEUE LEN=5
                 FUNCTION=READ                ERRORCODE=0
```

An IRFQ trace entry is generated whenever the driver removes a request from one of its internal queues.

The meanings of the various items are as follows:

REQUEST ID= etc.	This item tells you the id of the request responsible for this driver action. Id values are given in octal and hex.
QUEUE= etc.	This item tells you the queue to which the request is removed.
QUEUE LEN=y	This item specifies the number of requests queued to the above queue.
FUNCTION= etc.	This item tells you the SCCP function specified in the request.
ERRORCODE=X	This item specifies an irrecoverable error code as defined in Chapter 4.

.xx INR (New Request) Trace Entries

```
7       3.034 INR  REQUEST ID= 000777(!01FF)   QMISC= 000001   PCB#= 013
                   FUNCTION=INFO TRANSFER
```

An INR trace entry is generated each time the driver begins processing a new request.

The meaning of the various items are as follows:

REQUEST ID= etc.	This item tells you the id of the request responsible for this driver action. Id values are given in octal and hex.
QMISC= xxxxxx	This item tells you the miscellaneous parameter associated with the request.
PCB#= xxx	This item specifies the PCB number of the originator of the request.
FUNCTION= etc.	This item tells you the SCCP function specified in the request.

x.xx IPR (Process Request) Trace Entries

```
8       3.034 IPR  REQUEST ID= 000777(!01FF)   QMISC= 000000   PCB#= 013
                   FUNCTION=INFO TRANSFER
```

An IPR trace entry is generated whenever the driver processes a request which may be completed immediately(ie requires no I/O to SCCP) or whenever a request requires some preprocessing before I/O is to be done.

The meaning of the various items are as follows:

REQUEST ID= etc.	This item tells you the id of the request responsible for this driver action. Id values are given in octal and hex.
QMISC= xxxxxx	This item tells you the miscellaneous parameter associated with the request.
PCB#= xxx	This item specifies the PCB number of the originator of the request.
FUNCTION= etc.	This item tells you the SCCP function specified in the request.

.xx IAR (Abort Request) Trace Entries

```
9       8.173 IAR  REQUEST ID= 000505(!0145)
                   FUNCTION=READ                ERRORCODE=102
```

An IAR trace entry is generated whenever a request is to be hard aborted.

The meanings of the various items are as follows:

REQUEST ID= etc.	This item tells you the id of the request responsible for this driver action. Id values are given is octal and hex.
FUNCTION= etc.	This item tells you the SCCP function specified in the request.
ERRORCODE=X	This item specifies the irrecoverable error code as defined in Chapter 4.

x.xx ISTO (Start Timeout) Trace Entries

```
10      8.173 ISTO REQUEST ID= 000777(!01FF)   FUNCTION=WRITE
```

An ISTO trace entry is generated whenever the driver starts a software timeout on a request.

The meanings of the various items are as follows:

REQUEST ID= etc.	This item tells you the id of the request responsible for this driver action. Id values are given in octal and hex.
FUNCTION= etc.	This item tells you the SCCP function specified in the request.

xx IRB (Illogical Condition) Trace Entries

```
18      6.543 IRB  REQUEST ID=NONE
                   LOGICAL DRVR ERRORCODE=0    PHYSICAL DRIVER ERRORCODE=3
```

An IRB trace entry is generated whenever the driver detects an illogical internal condition or receives an erroneous or illogical message from SCCP.

The meanings of the various items are as follows:

LOGICAL DRVR ERRORCODE=X	This item tells you the code of the catastrophic error detected by the logical driver.
PHYSICAL DRVR ERRORCODE=X	This item tells you the code of the catastrophic error detected by the physical driver.

Table 6-2. Trace Entry EVENT Definitions

EVENT	DEFINITION
ABORT CURRENT USER REQUEST	The current intrinsic call must be aborted. ALL STATIONS DOWN No stations or groups can be polled or selected because errors have occurred on every station or group listed in the polling sequence or for the selected station.
BROADCAST TO ALL STATIONS	Broadcast text to either all stations on the line or all stations in a particular group (see STATION parameter in the CWRITE heading).
CARRIER LOSS	The communications link with the remote station has been broken.
CCLOSE	A CCLOSE intrinsic call has been encountered. The driver is to be closed.
CCONTROL	A CCONTROL intrinsic has been encountered. The driver is to perform the specified control function.
CONNECT TIMEOUT EXPIRED	The line was not physically connected within the maximum time allowed.
DATA OVERRUN OCCURRED	Before a received data word could be transfered from the TCU to main memory, it was overrwritten by the next received data word.
DATASET NOT RDY	The "Data Set Ready" (CC) signal from the MODEM changed from "set" to "clear".
HARDWARE FAILURE OCCURRED	A hardware error was detected. The failure occurred in the local station, not the remote.
INITIALIZE	A COPEN intrinsic call has been encountered. The driver is to be initialized.
INTERNAL TIMEOUT EXPIRED	The driver must transmit a WACK or TTD.
LAST ACTION WAS SUCCESSFUL	The action specified in the previous STN trace entry was performed successfully.

Table 6-2. Trace Entry EVENT Definitions (Cont.)

EVENT	DEFINITION
LAST ACTION WAS UNSUCCESSFUL	The action specified in the previous STN trace entry was not performed successfully.
LOCAL TIMEOUT EXPIRED	A CREAD or CWRITE intrinsic call was not issued within the maximum time allowed following a previous CREAD or CWRITE intrinsic call.
NO ID VERIFICATION	No ID verification is to be performed.
NON RESP DEVICE	Non-responding local TCU (i.e., HSI or SSLC).
POWER FAIL OCCURRED	A power failure was detected.
PUT REMOTE IN CONTROL STATE	Return station to control state so that another station can be polled or selected.
READ INTERRUPT	A CREAD is to be performed. An RVI is to be sent to the remote rather than an ACK.
READ (INPUT BUFFER)	A CREAD is to be performed.
READ (NO BUFFER)	A CREAD with no input buffer is to be performed.
READ REPEAT	A CREAD is to be performed. A NAK is to be sent to the remote instead of an ACK.
RECEIVE ID THEN SEND ID	ID verification is to be performed on this line. The local station is to first receive and verify the remote's ID. It then sends its ID to the remote.
RECEIVE TIMEOUT EXPIRED	Nothing (including TTD) was received from the remote station within the maximum time allowed during a read operation. Note that a TTD always resets the timer.
RECEIVED ACK0 SEQUENCE	An ACK0 control character sequence (positive acknowledgement) was received from the remote station.

Table 6-2. Trace Entry EVENT Definitions (Cont.)

EVENT	DEFINITION
RECEIVED ACK1 SEQUENCE	An ACK1 control character sequence (positive acknowledgement) was received from the remote station.
RECEIVED DLE EOT SEQUENCE	A DLE EOT control character sequence (disconnect sequence) was received from the remote station.
RECEIVED ENQ	An ENQ control character was received from tne remote station.
RECEIVED EOT	An EOT control character was received from the remote station.
RECEIVED GARBAGE CHARACTERS	Data was received from the remote station but was unrecognizable.
RECEIVED NAK SEQUENCE	A NAK control character sequence (negative acknowledgement) was received from the remote station.
RECEIVED RVI SEQUENCE	An RVI control character sequence ("reverse interrupt" positive acknowl- edgement) was received from the remote station.
RECEIVED TEXT	A text message (with no errors) was received from the remote station.
RECEIVED TEXT BLOCK	A text message, ending with an end-of-text block character was received from the remote. The remote wants to send more text.
RECEIVED TEXT IN ERROR	A text message was received from the remote station. However, a parity error was detected.
RECEIVED TEXT OVERFLOW	A text message was received from the remote station. However, the message length exceeded that specified by the INCOUNT parameter of the CREAD intrinsic call and overflow occurred. Note that in such cases the driver cannot perform parity error checking. The received text should be considered unreliable.

Table 6-2. Trace Entry EVENT Definitions (Cont.)

EVENT	DEFINITION
RECEIVED TTD SEQUENCE	A TTD control character sequence ("temporary text delay") was received from the remote station during a read operation.
RECEIVED WACK SEQUENCE	A WACK control character sequence ("wait before transmitting" positive acknowledgement) was received from the remote station during a write operation.
RESPONSE TIMEOUT EXPIRED	No acknowledgement was received from the remote station within the maximum time allowed. Note that a WACK control character always resets the timer.
SAME STATION AS BEFORE	Transmit data to or receive data from the same station as the last request.
SELECT THE STATION	Select a particular station on a multipoint line.
SEND DISCONNECT	A CCONTROL intrinsic call with CONTROLCODE=2 has been encountered. The driver is to perform a "Send Disconnect" operation.
SEND ID THEN RECEIVE ID	ID verification is to be performed on this line. The local station is to first send its ID to the remote and then receive the remote's ID.
SEND RESET	A CCONTROL intrinsic call with CONTROLCODE=1 has been encountered. The driver is to perform a "Send Reset" operation.
SEND TO DIFFERENT STATION	The station specified by the request's STATION parameter is different from the currently polled/selected station.
SYSTEM CONTROL REQUEST	A request for a START TRACE, STOP TRACE, or STOP FLUSH function has been encountered. The driver is to perform the specific function.

Table 6-2. Trace Entry EVENT Definitions (Cont.)

EVENT	DEFINITION
SYSTEM INFORMATION TRANSFER	A request for a CGETINFO, CCHECK, or SHOWCOM information transfer to/from the mainframe has been encountered. The driver is to perform the specific transfer.
TRANSFER ERROR	A local TCU (i.e., HSI or SSLC) transfer error was detected.
WRITE (CONVERSATIONAL)	A CWRITE is to be performed having an input and an output buffer.
WRITE (NO BUFFER)	A CWRITE is to be performed having no input or ouput buffer.
WRITE (OUT BUFFER)	A CWRITE is to be performed having on output buffer but no input buffer. operation.

Table 6-3. Trace Entry ACTION Definitions

ACTION	DEFINITION
ABORT THE REQUEST	The driver is to abort the current intrinsic call and process the next user request.
BID FOR THE LINE	The driver is to bid for the line (i.e., send an ENQ to the remote station and then receive a response; the expected positive response is an ACK0 sequence).
BREAK CONNECTION	A CCLOSE intrinsic call has been encountered. If the communications link with the remote station currently exists, the driver is to send either a DLE EOT sequence (switched line) or an EOT (non-switched line) to the remote station and then disconnect the line.
BROADCAST TO ALL TRIBUTARIES	Send text to all stations on the line.
DISCONNECT	Initiate disconnect sequence.
EXEC LAST REQ,RE=EOT IN CNTL	An EOT was detected in a control character sequence received from the remote station. The driver is to re-execute the latest physical driver request (i.e., the OPERATION specified in the previous OPR trace entry).
EXEC LAST REQ,RE=INVALID ID	An invalid ID sequence was received from the remote station. The driver is to re-execute the latest physical driver request (i.e., the OPERATION specified in the previous DVR trace entry).
EXEC LAST REQ,RE=RECV GARBGE	An unrecognizable text message or control character sequence was received from the remote station. The driver is to re-execute the latest physical driver request (i.e., the OPERATION specified in the previous DVR trace entry).

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
EXEC LAST REQ,RE=REQ RSP RSND	The remote station has asked the local station to resend its latest response. The driver is to re-execute the latest physical driver request (i.e., the OPERATION specified in the previous OPR trace entry).
EXECUTE CCONTROL REQUEST	A CCONTROL intrinsic call has been encountered. The driver is to perform the specified control function.
EXEC SYSTEM CONTROL REQUEST	A request for a START TRACE, STOP TRACE, or STOP FLUSH has been encountered. The driver will perform the specific function.
EXEC SYSTEM INFORMATION TRANSFER	A request for a CGETINFO, CCHECK, or SHOWCOM information transfer to/from the mainframe has been encountered. The driver will perform the specific transfer.
FREEZE I.D. LIST	The driver is to freeze the line's MISC Data Segment which contains the I.D. List.
EEZE THE BUFFER	The driver is to freeze the user's stack (NUMBUFFERS=0) or freeze the CS buffer (NUMBUFFERS>0) in preparation for a data transfer operation.
GENERATE CURRENT USER REQ	The driver is to recall the current intrinsic call.
GENERATE CURRENT WRITE REQ	The driver is to recall the current CWRITE intrinsic call.
GENERATE REQUEST OR ABORT	The driver is to recall the current intrinsic call, unless a soft abort was received, in which case an abort should be performed.
GET TYPE OF ID CHECKING	Test the kind of ID checking to be performed - read, write, or no checking.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
INC RETRY CT,RE=ATTMPT LBID	The local station is the primary station and has received an ENQ in response to its line bid. The driver is to increment the retry counter and then either retry the line bid (if the allowable number of retries has not been exceeded) or perform the "POST LINE BID PREEMPTED" action described later in this table.
INC RETRY CT,RE=BAD ACK	The local station has received the wrong positive acknowledgement (i.e., ACK0 when ACK1 was expected, or vice versa). The driver is to increment the retry counter and then either resend the latest text message (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.
INC RETRY CT,RE=ENQ TO TEXT	The local station received an ENQ as the response to a text message. The driver is to increment the retry counter and then either resend the latest text message (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.
INC RETRY CT,RE=GARBGE TO TX	The local station received a garbled response to a text message. The driver is to increment the retry counter and then either resend the latest text message (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
INC RETRY CT,RE=INVALID ID	An invalid ID sequence was received from the remote station. The driver is to increment the retry counter and then either retry the line bid (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.
INC RETRY CT,RE=NO RESP TO ENQ	The remote did not respond to the ENQ sent by the local.
INC RETRY CT,RE=RECV ENQ	The remote sent an ENQ.
INC RETRY CT,RE=RECV GARBAGE	A garbled text message or control character response was received from the remote station. The driver is to increment the retry counter and then either ask the remote station to resend the message or response (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.
INC RETRY CT,RE=REQ TXT RSND	The remote station has asked the local station to resend its latest text message. The driver is to increment the retry counter and then either resend the latest text message (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.
INC RETRY CT,RE=RESP TIMEOUT	No acknowledgement was received from the remote station within the maximum time allowed. The driver is to increment the retry counter and then either ask the remote station to resend its latest response (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
INC RETRY CT,RE=TEXT ERROR	The text received from the remote has a bad check character and the local must reject.
INC RETRY CT,RE=TXT OVERFLOW	Either the remote station sent text which was larger then the input buffer or the ETX/ETB delimiter was garbled. The driver is to increment the retry counter and then either ask the remote station to resend its text (if the allowable number of retries has not been exceeded) or perform the "POST MAX RETRIES" action described later in this table.
INITIALIZE THE DRIVER	A COPEN intrinsic call has been encountered. The driver is to be initialized.
LAST TRIBUTARY SENT TEXT	Text was received from the current station even though an attempt was made to put it back into control mode by sending it an RVI.
MRJE READ CONTINUE	Reset the ACK counters. (MRJE protocol uses only ACK0's.) Continue the last operation (Read).
NO OPERATION	No action is to be performed.
POLL	Poll the requested station(s) to complete the CREAD request.
POLL (START WITH CURRENT STA)	Poll the requested station(s) starting with the current station.
POST ALL STATIONS DOWN	An attempt has been made to read from the list of stations or groups specified in the current CREAD intrinsic or an attempt has been made to write to the station or to broadcast to the line as specified in the current CWRITE intrinsic. However, because all appropriate stations have been marked down in previous reads and writes, this request cannot be satisfied.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
POST BROADCAST DONE	The request to send text to all stations has been completed successfully.
POST EXHAUSTED POLLING	No stations have had data to send and the requested number of iterations through the poll list have been completed.
POST HARDWARE FAILURE	A hardware failure has occurred with the modem or TCU. The driver is to disconnect the line, report the specific hardware problem to the user program, and process the next user request.
POST INVALID REQUEST	The current intrinsic call is invalid.
POST LINE BID PREEMPTED	If the local station is the primary station, this means that the remote station continually responded with an ENQ when the local station was bidding for the line (i.e., the retry counter overflowed). If the local station is the secondary station, this means that the remote station responded with an ENQ when the local station tried to bid for the line. In either case, the driver is to process the next user request.
POST MAX RETRIES, DISCONNECT	The maximum number of allowable retries has been exceeded. The driver is to disconnect the line, post the error, and then process the next user request.
POST MAX RETRIES, SEND EOT	The maximum number of allowable retries has been exceeded. The driver is to send an EOT control character to the remote station, post the error, and then process the next user request.
POST NO LINE BID RECEIVED	The local expected to receive a line bid but none were received from the remote.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
POST OPERATION FINISHED	The current intrinsic call has been completed. The driver is to process tne next user request.
POST READ BUFFER OVERFLOWED	A text message was received from the remote station. However, the message length exceeded that specified by the INCOUNT parameter of the CREAD intrinsic call and overflow occurred. Note that in such cases the driver CANNOT perform parity error checking. The received text should be considered unreliable. The driver is to process the next user request.
POST READ REQUEST SUCCESSFUL	The current CREAD intrinsic call was performed successfully.
POST RECEIVE TIMEOUT	Nothing (including TTD) was received from the remote station within the maximum time allowed during execution of a read operation. Note that a TTD sequence always resets the timer. The driver is to process the next user request.
POST RECEIVED UNEXPECTED RVI	An RVI was received from the remote although the local is sending text in blocks and not all of the blocks have been sent.
POST RECEIVE UNEXPECTED TEXT	The remote sent text and the local user did not provide an input buffer to receive it. The driver is to process the next request.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
POST RECV UNEXPECTED EOT	An EOT was received although the local has not yet sent all of its text.
POST REMOTE NOT READY	The remote station sent a NAK when it was being selected or when sending ID sequences.
POST REM PROTCL ERR, DISCNCT	The remote station contempuously violated the BSC protocol conventions. The driver is to disconnect the line, post the error, and process the next user request.
POST REM PROTCL ERR, SND EOT	A protocol error was received from the remote - send an EOT.
POST REMOTE REJCTD CONNECTN	The remote station rejected the local station's attempt to establish a connection. The driver is to disconnect the line, post the error, and process the next user request.
POST REMOTE REJECTD LINE BID	The remote station responded with a NAK or EOT when the local station tried to bid for the line. The driver is to process the next user request.
POST REMOTE SENT DELAY SEQ	The remote sent TTD or WACK but it is not permitted in this context (see DOPTIONS *******)
POST REMOTE SENT EOT	An EOT was received from the remote station. The driver is to process the next user request.
POST REQUEST ABORTED	The current intrinsic call has been aborted. The driver is to process the next user request.
POST RETRY OVERFLOW	The number of retries allowed has been performed. Return to the calling program without taking any further action.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
POST RVI RECEIVED	The remote station sent an RVI sequence ("reverse interrupt" positive acknowledgement) in response to a text message. The driver is to process the next user request.
POST SENT ACK, RECEIVED ACK	The remote sent an ACK in response to our ACK. The driver is to process the next user request.
POST SENT ACK, RECEIVED NAK	The remote sent an NAK in response to our ACK. The driver is to process the next user request.
POST SENT ACK, RECEIVED RVI	The remote sent an RVI in response to our ACK. The driver is to process the next user request.
POST SYSTEM ERROR	While performing some function for the driver, MPE detected an internal error. The driver is to process the next user request.
POST WRITE 0 BUFF SUCCESSFUL	The current "Write 0 buff" request (CWRITE intrinsic call with no input or output buffer specified) was performed successfully. The driver is to process the next user request.
POST WRITE REQ SUCCESSFUL	The current CWRITE intrinsic call was performed successfully. The driver is to process the next user request.
READ CTRL SEQ,RE=RECV GARBGE	An unrecognizable control character sequence was received from the remote station. The driver is to retry the "read control" operation.
READ 0 BF: POST READ SUCCESS	The current "Read 0 buf" request (CREAD intrinsic call with no buffer specified) was performed successfully. The driver is to process the next user request.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
READ TEXT, RE=RECV GARBAGE	During a "read text" operation, an unrecognizable text message was received from the remote station. The driver is to retry the "read text" operation.
READ TEXT, RE=TEXT OVERFLOW	Either the remote sent a text block which was too large for our input buffer or the ETX/ETB character was garbled. The driver is to process the next request.
RECEIVE REMOTE'S LINE BID	The driver is to receive a line bid (ENQ) from the remote station.
RECORD CONNECT TIMEOUT	The line was not physically connected within the allowed time. The driver is to post the error and process the next user request.
RECORD LATENT DLE EOT	A DLE EOT sequence was received from the remote station in response to a WACK or TTD. The driver is to disconnect the line, post the error, and await the next user request.
RECORD LATENT EOT	An EOT was received from the remote station in response to a WACK or TTD. The driver is to post the error and await the next user request.
RECORD LATENT RECV TIMEOUT	No response was received from the remote station to a WACK or TTD within the allowed time. The driver is to post the error and await the next user request.
RECORD LOCAL T/O, DISCONNECT	A CREAD or CWRITE intrinsic call was not issued within the maximum time allowed following a previous CREAD or CWRITE intrinsic call. The driver is to disconnect the line, post the error, and await the next user request.
RECORD POWER FAIL, DISCNCT	A power failure occurred. The driver is to disconnect the line, post the error, and await the next user request.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
RECORD RECV T/O, DISCONNECT	Nothing (including TTD) was received from the remote station within the maximum time allowed during execution of a read operation. The driver is to disconnect the line, post the error, and await the next user request.
RECRD OUT BFR DIRTY, SND EOT	During execution of a "Write Conversational" request, the remote station asked the driver to resend its latest text message. However, the driver found that the text message in its buffer had been overlayed by a garbled response from the remote station. The driver is to send an EOT to the remote station.
REMOTE REQUESTED DOWNLOAD	While executing a CREAD from unconnected or control state, the remote sent a NAK (instead of an ENQ) which requests the local user to send the download file. The driver is to process the next request.
REMOTE SENT DLE EOT	A DLE EOT sequence was received from the remote station. The driver is to disconnect the line, post the error, and process the next user request.
REPEAT LAST PHYS DVR REQUEST	The driver is to repeat the latest physical driver request (i.e., the OPERATION specified in the previous OPR trace entry).
REQ RESEND OF LAST ACK	The driver is to send an ENQ to the remote station. This asks the remote station to resend its latest acknowledgement.
REQUEST RESEND OF TEXT	The text message received from the remote station (in conjunction with a "read text" or "write conversational" operation) was either garbled or contained parity errors and the local station is asking for it to be resent.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
RESET TRIBUTARY	The remote multipoint station must be put into control mode so that it will properly act upon the next select or poll request.
RESPOND ENQ TO WACK	A WACK sequence was received from the remote station. The driver is to respond by sending an ENQ.
RESPOND NAK TO TTD	A TTD sequence has been received from the remote station. The driver is to respond by sending a NAK.
SELECT TRIBUTARY	Select the requested station so that the user's CWRITE request can be completed.
SEND ACK, RECEIVE TEXT	The driver is to send either an ACK0 or ACK1 sequence (whichever is proper) to the remote station and then receive a text message.
SEND DELAY SEQ, GET RESPONSE	Send either a WACK or a TTD depending on whether reading or writing, respectively. Wait for a response from the remote. WACK is sent if the local is not ready to receive more data. TTD is sent if the local is not ready to transmit data.
SEND DISCONNECT SEQUENCE	The driver is to send either a DLE EOT sequence (switched line) or an EOT to the remote station, disconnect the line, and then process the next user request.
SEND ENQ, GET RESPONSE	The driver is to send an ENQ to the remote station and then receive a control character response.
SEND ENQ, GET TEXT/RESPONSE	The driver is to send an ENQ to the remote station and then receive either a text message or a control character response.
SEND EOT	The driver is to send an EOT to the remote station and then process the next user request.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
SEND ID-ACK, RECEIVE TEXT	The driver is to send an ID ACK0 sequence to the remote station and then receive a text message.
SEND ID-ENQ, RECEIVE ID-ACK	The driver is to send an ID ENQ sequence to the remote station and then receive an ID ACK0 sequence.
SND MRJE DELAY,RECV TXT/RESP	The driver is to send a special text message, with the MRJE wait'a'bit set. After the Resp/Text is received in reply, no further text will be transmitted until the next intrinsic call (CREAD, CWRITE). The handshaking ACK0 send by the CREAD/CWRITE clears this condition at the Host. The CHECK TEXT FLAG action clears this condition locally. This is sent only when an internal timeout occurs.
SEND NAK AND RECEIVE TEXT	Ask the remote to resend its text.
SEND RVI AND RECEIVE TEXT	The driver is to send an RVI sequence to the remote station and then receive a text message.
SEND TEXT AND RECV RESPONSE	The driver is to send a text message to the remote station and then receive a control character response.
SEND TEXT, RECV TEXT/RESP	The driver is to send a text message to the remote station and then receive either a text message or a control character response.
SET STATION DOWN	The station or group being addressed does not respond, or responds with unacceptable data. Note that this station or group is "down" and refuse to reference it unless the station is marked "up" by the intrinsics.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
SET TIMEOUT FOR NXT WACK/TTD	The driver is to reset the "WACK/TTD generator timer" to its starting value.
TEST FOR TEXT RECEIVED	Test the Text flag. This would have been set if text was received in response to the MRJE DELAY action.
TEST IF BLOCKING INPUT	An ETB was received at the end of this block of text - determine whether the user expects data to be blocked.
TEST IF IN DOWNLOAD MODE	While awaiting a line bid a NAK has been received instead of the expected ENQ. If the driver has been configured for download then the NAK indicates that the remote is now ready to be downloaded.
TEST IF LOCAL IS PRI OR SECN	The driver is to test whether the local station is the primary or secondary station.
TEST IF OUTPUT BUFR IS DIRTY	During execution of a "Write Conversational" request, the remote station asked the driver to resend its latest text message. The driver is to examine the contents of its buffer to see if the text message got overlayed by a garbled response from the remote station.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
TEST IF PROPER ACK RECEIVED	The driver is to test the ACK0 or ACK1 sequence received from the remote station to make sure it is the one expected.
TEST IF RSP T/O JUST OCCURRD	The wrong ACK sequence was received from the remote in response to a text message. The driver is to test to see if a response timeout also occurred in conjunction with the "lost" text block.
TEST IF SELECTION OR BRDCAST	Check the intrinsic call to see whether the user wants to talk to an individual station or to the entire line.
TEST IF SENT LAST BLOCK	Determine whether any more data for the current CWRITE remains to be sent.
TEST IF STATION IS NEW	A read or write is to be performed. Determine if the station referenced is the same as the previously referenced station.
TEST IF WILL WAIT FOR REMOTE	Determine if a delay request from the remote is allowed.
TEST RECEIVED ID SEQUENCE	Test to see if the ID sequence received is valid.
TEST SOFT ABORT FLAG	Test to see if a soft abort has been requested by the intrinsics.
TEXT RECEIVED, NO REQUEST	Text has been received in response to the MRJE DELAY action - the Text flag has been set.
TRANSFER TEXT TO READ	Text was received in response to a previous SEND MRJE DELAY action - transfer this text instead of reading from the line.
TRANSFER TEXT TO WRITE	Text was received in response to a previous SEND MRJE DELAY action - transfer this text instead of reading from the line for the write conversational.

Table 6-3. Trace Entry ACTION Definitions (Cont.)

ACTION	DEFINITION
UNEXPECTED EVENT, DISCONNECT	Something unexpected happened. The driver is to post the error and then await the next user request. If the communications link with the remote station currently exists, the driver will first send a DLE EOT sequence (switched line) or an EOT (non-switched line) to the remote station and disconnect the line.
WAIT FOR LINE BID	The current intrinsic call is either "Read Connect" or "Read Connect, Enquiry" (CREAD with OPCODE=0 or 2) and the driver is waiting for the communications link to be established and a line bid (ENQ) to be received from the remote station.
WAIT FOR PHYSICAL CONNECTION	The driver is waiting for the communications link to be established.
WAIT FOR RESEND OF RESPONSE	The control character response from the remote station (in conjunction with a "write text" operation) was either garbled or lost and the local station has asked for it to be resent. The local station is currently waiting for the retransmission of the control character response.
WRITE CONVR: TEXT ERROR	Text containing parity errors was received from the remote station as the response to a text message during execution of a "Write Conversational" request (CWRITE with output and input buffer specified). The driver has asked the remote station to resend its response.
WRTE CONV: POST READ SUCCSFL	The current "Write Conversational" request (CWRITE with output and input buffer specified) was performed successfully.

7.1 Configurator Dialog

A number of extensions have been made to the MPE configurator to accommodate CS. These extensions, concerned exclusively with I/O configuration changes, are necessary to add CS to a system or to alter an existing CS configuration.

The extensions are of one of two forms: 1) modified control paths through the dialog steps resulting from responses to existing (i.e., file-system related) questions, and 2) new dialog steps pertaining to CS. Both forms are described in the dialog changes given below. Two step numbers are given for each interrogation. The first refers to :SYSDUMP command dialog steps and the second (parenthesized) refers to the initiator dialog steps; both of which are taken from the MPE Version C ERS. Where existing dialog steps are discussed, only changes and extensions are described.

3.1(8) LIST I/O DEVICES?

If YES is entered, a list of all input/output devices will be printed. This list will include CS devices. For CS devices, only that information common to CS and FS will be printed.

If CS is known to exist on the system, the next step will be 3.1.1(8.1). Otherwise, the next step will be 3.2(9).

3.1.1(8.1) LIST CS DEVICES?

To print a list of CS device characteristics, enter YES.

To suppress the listing, enter NO.

The next step will be 3.2(9).

3.3(10) LOGICAL DEVICE #?

If a zero or carriage return is entered, the dialog will skip either to 3.3.13(22.1) if CS exists or has been added to the system or 3.4(23) otherwise.

3.3.1(11) DRT #?

To add a device, enter its DRT number. To remove a device and return to step 3.3(10), enter zero.

3.3.2(12) UNIT #?

Enter a zero for CS device types 17, 18, or 19.

3.3.2A CHANNEL #?

Enter a zero for CS device types 17, 18, or 19.

3.3.3(13) TYPE?

The following device types have been allocated to CS:

17 - Single Channel Communications Processor (SCCP)
18 - Synchronous Single Line Controller (SSLC)
19 - High speed Serial Interface (HSI)

3.3.4(14) SUB-TYPE?

If TYPE is 19, then SUB-TYPE must be 0

If TYPE is 17 or 18, then the following SUB-TYPE values are defined:

0 - Synchronous/Switched/Modem
1 - Synchronous/Non-switched/Modem
3 - Synchronous/Non-switched
7 - Asynchronous/Non-switched

If TYPE and SUB-TYPE indicate this is not a CS device, the next step will be 3.3.4.0(15). If TYPE=19, the next step will be 3.3.4.0(14.0). If TYPE=17 or 18, the next step will be 3.3.4.1(14.1).

3.3.4.0(14.0) PORT MASK?

This value will form a mask indicating which HSI channel will be used by this line. Permissable values are

8 - HSI channel 0
4 - HSI channel 1
2 - HSI channel 2
1 - HSI channel 3.

3.3.4.1(14.1) PROTOCOL?

The protocol number for the device should be entered. Allowable values are 1-255. Currently the only defined values are

1 = Bisync
2 = MRJE
3 = HPDLC I

3.3.4.2(14.2) LOCAL MODE?

The local mode for the station should be entered. Allowable values are 1-15. Currently defined values are:

1 = local is primary contention station
2 = local is secondary contention station
3 = local is multipoint control station.
4 = local is secondary station on a multipoint line
5 = local is an HPDLC I system acting as DTE
6 = local is an HPDLC I system acting as DCE

The next step is 3.3.4.3(14.3).

3.3.4.3(14.3) TRANSMISSION CODE?

The transmission code for the device should be entered. Allowable values are 1-63; the currently defined values are

1 - automatic code sensing if the connection is accomplished by a CREAD, ASCII if the connection is done with a CWRITE.
2 - ASCII
3 - EBCDIC

3.3.4.4(14.4) RECEIVE TIMEOUT?

The receive timeout value should be entered. This is the length of time that the local station will wait for a text block to be sent by the remote. If a carriage return is entered, a 20-second default will be provided.

3.3.4.5(14.5) LOCAL TIMEOUT?

The local timeout value should be entered. This is the amount of time that CS will allow between the completion of the previous I/O request and the initiation of the next one. Note that this only applies when the line is connected. If a carriage return is entered, a 60-second default will be provided. The next step is 3.3.4.7(14.7).

3.3.4.7(14.7) CONNECT TIMEOUT?

The connect timeout value should be entered. If a carriage return is entered, a 900-second default will be provided.

For non-switched lines, if SUB-TYPE implies connection via a modem, the next dialog step will be 3.3.4.11(14.11). If SUB-TYPE implies a hardwire connection the next dialog step will be 3.3.4.13(14.13). If SUB-TYPE implies a switched line, the dialog will proceed in sequence.

3.3.4.8(14.8) DIAL FACILITY?

A YES response will indicate that the line has the ability to dial out (either manually or automatically).

3.3.4.9(14.9) ANSWER FACILITY?

To specify that this device has an answering capability enter YES. If the response is NO, the dialog will skip to step 3.3.4.11(14.11).

4.9.1(14.9.1) AUTOMATIC ANSWER?

To specify that this device has automatic answer capabilities enter YES. A NO response will indicate manual answering is required.

3.3.4.11(14.11) DUAL SPEED?

To specify that the modem is dual speed, enter YES. If the response is NO, the dialog will skip to step 3.3.4.13(14.13).

3.3.4.12(14.12) HALF SPEED?

If the modem is to operate at half-speed enter YES. To specify that the modem is to operate at full speed enter NO or a carriage return. The next dialog step will be 3.3.4.14(14.14).

3.3.4.13(14.13) SPEED CHANGEABLE?

If the speed of the line is changeable, enter YES.

3.3.4.14(14.14) TRANSMISSION SPEED?

Enter the transmission speed of the line in characters per second.

3.3.4.15(14.15) TRANSMISSION MODE?

One of the following transmission modes should be entered:

0 = full duplex
1 = half duplex
2 = simplex - write only
3 = simplex - read only.

The HSI hardware requires full duplex. When using the Bisync protocol, the SSLC interface may require either half or full duplex, depending on the type of line and modems.

3.3.4.16(14.16) PREFERRED BUFFER SIZE?

Enter the buffer size (in words) most suitable to the device being configured. When using the HSI or SSLC boards, this value must be less than or equal to 4095. This value becomes the default COPEN BUFFSIZE.

3.3.4.17(14.17) DRIVER CHANGEABLE?

If a user program can change the driver for this device, enter YES.

3.3.4.18(14.18) DRIVER OPTIONS?

Enter the appropriate driver options value for the default driver. The next dialog step will be 3.3.11(21).

3.3.11(21) DRIVER NAME?

Enter the name of the default driver for this line. If the device is not a CS device (i.e., TYPE <> 18-xx), the dialog will skip to step 3.3.12(22). Otherwise, if SUB-TYPE implies a non-switched line then:

a. if the line is point to point the dialogue will return to step 3.3.11.4(21.4).
b. if the line is multipoint the dialog will skip to 3.3.11.6(21.6). If the line is switched, the dialog will proceed with step 3.3.11.2(21.2).

3.3.11.2(21.2) PHONELIST?

To provide a default phone number list, enter YES. If the response is NO, the dialog will skip to step 3.3.11.4(21.4).

3.3.11.3(21.3) PHONE NUMBER?

A phonenumber can be specified by entering a string composed of numeric characters and hyphens. The total string length must not exceed 20 characters. This dialog step will be repeated until a carriage return is entered.

3.3.11.4(21.4) LOCAL ID SEQUENCE?

The local ID sequence can be specified in one of the following two forms:

1. A
   - "string"
   E

2. O
   - (d,[,d]...)
   H

These are the same sequence specifications as those described for those of the :CLINE command (Section III). A maximum of 16 bytes is allowed.

If a carriage return is entered, a null local ID sequence will be assumed.

3.3.11.5(21.5) REMOTE ID SEQUENCE?

A remote ID sequence is specified in the same format as the local ID sequence.

This dialog step will be repeated until a carriage return is entered. If Local mode indicates a control station in supervised mode then the next dialogue step will be 3.3.11.6 (21.6). Otherwise the dialog returns to step 3.3(10).

3.3.11.6(21.6) POLL ENTRY DELAY?

Enter the number of milliseconds between polling each entry in the POLLIST. If a carriage return is entered, a 200 millisecond default will be provided.

3.3.11.7(21.7) NUMBER OF POLL REPEATS?

Enter the number of times the poll list is to be scanned per I/O operation. If this number is zero the poll list will be repeated until a terminal responds affirmitively. If a carriage return is entered, a value of zero will be used.

3.3.11.8(21.8) CIRCULAR POLL DELAY?

Enter the length of time, in hundredths of seconds, between successive passes through the poll list. If a carriage return is entered, a value of zero will be used.

3.3.11.9(21.9) MULTIPOINT LINE TYPE?

This is reserved for future use - enter a carriage return.

3.3.11.11(21.11) ID VERIFICATION?

If no ID verification is to be performed then enter NO or carriage return. Otherwise type SEND if the local station is to initiate sending of IDs or RECV if the remote staton is to wait for the remote to initiate ID sending.

3.3.11.12(21.12) LINE SELECTION SEQUENCE?

The line selection sequence is specified in the same format as the local ID sequence. A maximum of 8 bytes is allowed. If the line does not support this facility, a carriage return should be entered.

3.11.13(21.13) NUMBER OF GROUPS?

Enter the total number of terminal groups attached to the line. The permissable range is from 1 to 254. This number will determine the number of times the dialog step sequence 3.3.11.14 - 3.3.11.24 will be repeated. Following this the dialog returns to step 3.3(10).

3.11.14(21.14) GROUP g POLLING ID?

The polling ID is specified in the same format as the line selection sequence. A maximum of 8 bytes is allowed. If the group has no polling ID then a carriage return should be entered.

3.11.15(21.15) GROUP g SELECTION ID?

The selection ID is specified in the same format as the line selection sequence. A maximum of 8 bytes is allowed. If the group has no selection ID then a carriage return should be entered.

3.3.11.16(21.16) GROUP g GENERAL POLL ID?

The general poll ID is specified in the same format as the line selection sequence. Note that this is appended to the group polling ID to form the general poll sequence. The maximum length of the concatenated sequence is 8 bytes. If the group does not support this facility, then a carriage return should be entered.

3.3.11.17(21.17) GROUP g GENERAL SELECTION SEQUENCE?

The general selection ID is specified in the same format as the line selection sequence. Note that this is appended to the group selection ID to form the general selection sequence. The maximum length of the concatenated sequence is 8 bytes. If the group does not support this facility, then a carriage return should be entered.

3.3.11.19(21.19) NUMBER OF STATIONS IN GROUP g?

Enter the total number of stations (terminals) attached to the group. This number will determine the number of times the dialog step sequence 3.3.11.22 - 3.3.11.24 will be repeated. The maximum number of statons permitted on the line is 254.

3.3.11.20(21.20) STATION TYPE?

The allowable values are:

0 = Hp 2645
1 -40 = Reserved for future use
41-63 = user defined station type.

3.3.11.21(21.21) STATON INPUT BLOCKSIZE?

If the value is different from the line's PREFERED BUFFER SIZE, then enter the proper value. Otherwise enter a carriage return.

3.3.11.22(21.22) NUMBER OF COMPONENTS ON STATION s?

Enter the total number of components attached to the

station. This number will determine the number of times the dialog step sequence 3.3.11.23 - 3.3.11.24 will be repeated. The permissable number of components is from one to sixteen.

.3.11.23(21.23) COMPONENT c POLL ID?

The polling ID is specified in the same format as the line selection format. Note that this ID is appended to the group polling ID to form the component's polling sequence. The maximum length of the concatenated sequence is 8 bytes. If this component does not support polling then a carriage return should be entered.

.3.11.24(21.24) COMPONENT c SELECTION ID?

The selection ID is specified in the same format as the line selection format. Note that this ID is appended to the group selection ID to form the component's selection sequence. The maximum length of the concatenated sequence is 8 bytes. If this component does not support selection then a carriage return should be entered.

3.3.12(22)CLASS NAME?

Enter a list containing at least one device class name (up to eight alphanumeric characters, beginning with a letter). Class names are separated from each other by commas.

3.4(23) LIST I/O DEVICES?

If CS is not present in the system, then the dialog will skip to step 3.5(23.1).

.4.1(23.0.1) LIST CS DEVICES?

To print a list of CS device characteristics, enter YES.

To suppress the listing, enter NO.

If CS is not present on the system, then the dialog will skip to step 4(24).

3.6(23.8) ADDITIONAL CS DRIVER CHANGES?

Enter YES to add or delete drivers from the list of CS drivers which may be used but may not be attached to particular devices. Enter NO to skip these changes and proceed to step 4(24).

3.6.1(23.8.1) LIST ADDITIONAL DRIVERS?

Enter YES to list the additional driver names.

3.6.2(23.8.2) DELETE DRIVER?

To delete one or more drivers, enter a YES. Otherwise enter NO to skip to step 3.6.4(23.8.4).

3.6.3(23.8.3) DRIVER NAME?

Enter the name of the driver to be deleted. This question is repeated until a carriage return is entered.

3.6.4(23.8.4) ADD DRIVER?

To add one or more drivers, enter a YES. Otherwise enter NO to skip to step 3.6.6(23.8.6).

3.6.5(23.8.5) DRIVER NAME?

Enter the name of the program file containing the alternate driver. This question is repeated until a carriage return is entered.

3.6.6(23.8.6) LIST ADDITIONAL DRIVERS?

Enter YES to list the additional driver names.

8. CS Log Records

There are three different types of CS log records:

1) Line disconnection record (type 9)
2) Line close record (type 10)
3) Line errors detected by the driver (type 11)

Each of these record types may be selected/suppressed at system configuration time.

.1 Line Disconnection Record

This record type provides accounting information for the duration of the connection.

Record format:

word displacement	number of words	contents
0	1	record type = 9
1	1	record length = 43
2	3	time stamp of disconnect or close
5	1	job type, job number
6	1	logical device number
7	2	duration of the connection in milliseconds
9	2	number of output data transfers
11	2	number of input data transfers
13	1	number of recoverable line errors modulo 2**16
14	1	number of irrecoverable line errors modulo 2**16
15	9	local ID sequence
24	9	remote ID sequence
33	10	phone number of remote

NOTES: 1) time stamps are in CHRONOS/CLOCK format
2) ID sequences are significant for both SW and NSW lines. The first byte is the length of the sequence. The remainder is the actual ID sequence (up to 16 bytes, left-justified).
3) The phone number is significant only if the local 3000 performed the dialing for an outgoing call. The phone number is represented in ASCII (up to 20 characters, left-justified with trailing blanks).
4) CCLOSE of a connected line will emit one line disconnect log record followed by one line close log record.

8.2 Line Close Record

This record type provides accounting information about the allocation of a line (logical device).

Record format:

word displacement	number of words	Contents
0	1	record type = 10
1	1	record length = 14
2	3	time stamp of close
5	1	job type, job number
6	1	logical device number
7	3	time stamp of open
10	4	driver name

NOTES: 1) time stamps are in CHRONOS/CLOCK format.
2) the driver name may be up to eight ASCII characters in length, left-justified with trailing blanks.

8.3 CS I/O Errors

This record type provides information about instances of the following types of CS line errors:

1) Any hardware error (CCHECK error code values of 101-149)
2) retry count exhausted due to any of the following recoverable errors:

CCHECK Recoverable Error code	Description
2	received unintelligible sequence
3	block check character error
5	received incorrect acknowledgement
8	received unintelligible sequence after sending text
9	received enquiry character after sending text
11	remote requested resend of last text block
13	received text overflow
14	data overrun occurred on the SIO multiplexor channel
15	transfer error occurred on the SIO multiplexor channel

The record type and format is the same as an I/O Error (type=11). Please refer to the System Supervisor Capabilities Manual for the record's format. Within the format, the following data items have special meaning:

I/O Record Format Name	Interpretation for CS device
HARDWARE STATUS	Same as ERRORCODE parameter of CCHECK intrinsic
DRIVER DEFINED PARAMETER #1	Current number of recoverable errors during this connection
DRIVER DEFINED PARAMETER #2	Number of retries while executing the current I/O request

9.1 Dial Message

This message informs the operator that a manual dialing operation is required in order to physically connect a switched line. A reply to the message is required to inform the appropriate CS intrinsic if the phone number specified was successfully dialed. The message has a format number of 2; the <message> portion is

LDEV# ldn DIAL phonenumber. ANSWER(Y/N)?

where ldn = logical device number of the CS line

If the number is successfully dialed then a "Y" should be typed, else type "N".

9.2 Error Message

This message informs the operator when an irrecoverable error occurs on a line; no reply is required.

LDEV# ldn CS I/O ERROR: errorcode

where ldn = logical device number of the CS line
errorcode = is a CCHECK hardware error number.

This error message may be suppressed by setting AOPTIONS.(11:1) to "1" in the COPEN parameters.

9.3 System Failures

The following System Failures are generated by CS:

900 I/O request no longer associated with the user's process
902 Unable to freeze segment in main memory
903 Unable to lock segment in main memory

904 Unable to increase data segment size
905 Unable to decrease data segment size
906 Unable to unfreeze segment in main memory
907 Unable to unlock segment in main memory
909 Invalid pointer to pollist area

9.4 SCCP RAM Dumps

When an SCCP system failure is detected, CS attempts to dump the contents of the SCCP RAM to a disc file by issuing a CCONTROL call with the controlcode = 53 and a parameter value of 0 (zero). Upon initiating the SCCP RAM dump, the following message is printed on the operator console:

SCCP RAM DUMP - LDEVnnn

where nnn is the logical device number for the SCCP.
If the dump is successfully completed, the message

SCCP RAM DUMP COMPLETED ****

will appear. A dump file may have the information from the SCCP RAM even though the entire dump operation was not successfully completed.

More information concerning the SCCP RAM dump facility is included in the description of CCONTROL 53.

10.1 CS/3000 SHOWCOM Command

CS/3000 provides for the monitoring of communication system device status via the SHOWCOM operator's console command. The status information can be used to determine communications line activity and quality.

Communications device status provided via the SHOWCOM command includes:

- Number of messages sent and received.
- Last recoverable and irrecoverable errors.
- Number of recoverable and irrecoverable errors.
- Number of retransmissions, response timeouts, clear to send losses, and underruns.
- Number of BCC/CRC errors, receive timeouts, carrier losses, and overruns.
- Line state(closed, connected, or disconnected).

note: Clear to send losses and carrier losses will only be valid for comunication lines configured as full duplex.
Underruns will occur only when the communication line uses the HPDLC protocol.
Receive timeouts will not occur on communication lines using the HPDLC protocol.

When the line is in the open state, either connected or disconnected, the status SHOWCOM reports is a snapshot of the statistics generated from the last open of the line to the point in time the SHOWCOM command is invoked. When the line state is closed, SHOWCOM reports status obtained at the closure of the line reflecting the statistics generated over the last open/close sequence.

10.2 SHOWCOM Command Syntax

The format of the SHOWCOM operator's console command is

```
=SHOWCOM ldn[;ERRORS][;RESET]
```

PARAMETERS	DESCRIPTION
ldn	Logical device number of a communications system device.
ERRORS	A request for the full status list. If not specified an abbreviated list is reported(see 10.3).
RESET	A request to reset all status information to zero after reporting.

10.3 SHOWCOM Command Output

SHOWCOM produces the following report if the ERRORS parameter is not specifed.

```
                        LDN - XXX
MESSAGES SENT      XXXXX       MESSAGES RECEIVED   XXXXX
          LAST RECOVERABLE ERROR    XXX
          LAST IRRECOVERABLE ERROR  XXX
          LINE IS LLLLLLLLLL
```

When the ERRORS parameter is specified, SHOWCOM provides the following report.

```
     TRANSMIT          LDN - XXX       RECEIVE
MESSAGES SENT      XXXXX       MESSAGES RECEIVED   XXXXX
RETRANSMISSIONS    XXXXX       BCC/CRC ERRORS      XXXXX
RESPONSE TIMEOUTS  XXXXX       RECEIVE TIMEOUTS    XXXXX
UNDERRUNS          XXXXX       OVERRUNS            XXXXX
CLR TO SEND LOSSES XXXXX       CARRIER LOSSES      XXXXX
          # OF RECOVERABLE ERRORS   XXX
          LAST RECOVERABLE ERROR    XXX
          # OF IRRECOVERABLE ERRORS XXX
          LAST IRRECOVERABLE ERROR  XXX
          LINE IS LLLLLLLLLL
```

In the above examples, XXXXX indicates a positive integer, and LLLLLLLLL specifies either the closed, connected, or disconnected line state. The last irrecoverable and recoverable error codes are specified in sections 4.12 and 4.13.

SHOWCOM command syntax errors or parameter errors will produce one of the following error messages.

- TOO MANY PARAMETERS
- LDN MISSING
- ILLEGAL LDN : the ldn specified is > 255.
- NOT A CS DEVICE : the ldn specified does not represent a communications system device.
- INVALID DELIMITER
- ILLEGAL OPTION : a parameter other than ERRORS or RESET was specified.
- CS I/O ERROR XXX : in processing the SHOWCOM request, I/O error XXX was encountered.

CREAD

The execution of a CREAD will differ substantially depending upon protocol driver, line mode and line state.

A.1 CONTENTION DRIVER.

Unconnected line state

Line State: unconnected ----------> text

Function:
a. establish physical connection
b. receive the remote's line bid (optionally with ID verification)
c. send an acknowledgement to the remote's line bid (optionally with an ID)
d. receive the remote's first text block

Note: If INCOUNT is omitted or set to zero, then only steps a and b are executed. If there is a local ID sequence then it will be sent on the next CREAD request. The next I/O request should be a CREAD.

Control line state

Line state: control----------> text

Function:
a. receive the remote's line bid
b. acknowledge the remote's line bid
c. read the remote's text block

Note:
1. If INCOUNT is omitted or set to zero, then only step a is executed. The next I/O request should be a CREAD.
2. While the line is in control state the CS driver listens for a line bid from the remote. If the line bid occurrs then the line is automatically placed into text state. If bit 3 of the DOPTIONS is set to a one, then the driver will ignore remote line bids between requests.

Text line state

Line state: text----------> text

Function: a. acknowledge last text block
b. read the remote's next text block

A.2 MULTIPOINT CONTROL DRIVER.

Unconnected line state

Line State: unconnected ----------> text
or
unconnected ----------> control

Function:
a. Establish physical connection.
b. Optionally perform ID verification as specified by the Line Descriptor in SUPLIST.
c. If the STATION parameter specified a station then:
1) poll that station
2) if the station responds negatively then post the request finished (line state = Control)
3) else receive the text from the remote (line state = Text)
d. Else if the STATION parameter specifies the POLLIST then starting with the current entry in the POLLIST:
1) poll the station (or group)
2) if the station responds affirmatively then receive its text (line state = Text)
3) else if this is the last entry in the POLLIST and the required number of passes through the list have been done, then post the request done (line state = Control).
Else index to the next entry in the list and go back to step 1.

NOTE: If data can be received in blocks, then several blocks of data may be received with an acknowledgement to each block. Conceptually, however, a buffer of data is received as described above.

Control line state

Line State: control ----------> text
or
control ----------> control

Function: Same as steps c - d of the unconnected line state.

Text line state

Line state: text ----------> text
or
text ----------> control

Function:

a. If the STATION parameter specifies the current station then:
   1) send an acknowledgement of the remote's last text block
   2) receive the next text block from the station (line state=Text)

b. Else if the STATION parameter specifies another station then:
   1) if DOPTIONS.(2:1) specifies that the previous station must be reset by sending it an RVI, then send an RVI sequence and receive an EOT sequence. (See COPEN for description of DOPTIONS)
   2) poll the station
   3) receive either a negative response to the poll (line state=Control) or the remote's text block (line state=Text)

c. Else if the STATION parameter specifies the POLLIST then:
   1) if current POLLIST entry is a logical station entry
      (a) if DOPTIONS.(2:1) specifies that the previous station must be reset by sending it an RVI, then send an RVI sequence and receive an EOT sequence. (See COPEN for description of DOPTIONS)
      (b) starting with the next entry in the POLLIST:
          (1) poll the station (or group)
          (2) if the station responds affirmatively then receive its text (state=Text)
          (3) if this is the last entry in the POLLIST and the required number of passes through the list have been done, then post the request done (state = Control).
              Else index to the next entry in the list and go back to step (1).
   2) current POLLIST entry is a logical group entry
      (a) acknowledge the previous text block and attempt to read the next text block from the current group.
      (b) if an EOT is received instead of text, then poll as in case c.1.c of the above.

Note: If the STATION parameter is specified, then with one exception the pollist index remains unchanged:
1) text state has been previously entered through a CREAD specifying the POLLIST,
2) the STATION parameter specifies the current station,

3) and the remote terminal responds with an EOT instead of an acknowledgement or text.

A.3 MRJE DRIVER.

Unconnected line state (not currently used by MRJE)

Line State: unconnected ----------> text

Function:
a. Establish physical connection
b. Receive line bid
   (receive SOH,ENQ; send ACK0)
c. While waiting for the text block, answer remote's ACK0's with ACK0's
d. Read the remote's text block

Control line state (not defined for MRJE)

Text line state

Line state: text----------> text

Function:
a. acknowledge last text block with ACK0
b. while waiting for next text block, answer remote's ACK0s with ACK0s
c. read the remote's next text block

A.4 SCCP and SSLC BSC Driver Message Format Word (MFW)

All BSC control characters are deleted from the incoming text stream by the driver.

If DOPTIONS.(11:2) is set to two then The driver describes the received text's format via the message format word (MFW) which is contained in word 0 of the user's buffer. Any other value of DOPTIONS.(11:2) will cause the MFW to be omitted from the user's text and, hence, unavailable to the user. The MFW fields have the same meaning as in sent text.

MFW (0:1) -- 0= the received text was transparent. If the text contained ITBs then this bit means that at least one intermediate text block was transparent.
1= the received text was non-transparent.

MFW (1:1) -- 0= the received text contained no intermediate text blocks (itb's).

1= there were itb(s). The length of each itb immediately precedes its block. The length is a 16-bit quantity and always begins on a word boundary. The last itb is followed by a delimiter word set to all ones.

MFW (2:1) -- Note: This field is significant only when there were itb's in non-transparent text.

0= not all itb's began with an SOH or STX character.
1= each itb was begun with an SOH or STX character.

MFW (3:1) -- 0= the text block ended with an ETX character.
1= the text block ended with an ETB character.

NOTE: Regardless of whether a text block ends in ETB or ETX, its successful transmission or reception will increment its respective "message" counter (i.e. MSGSENT/MSGRECV in the CCHECK Intrinsic).

MFW (4:4) -- Set to zero.

MFW (8:8) -- Length of the header portion of the text block. A value of zero indicates that there was no header.

Nontransparent block, not the last block of text.	STX data ETB
Nontransparent block, the last or only block of text.	STX data ETX
Transparent block not the last block of text.	DLE STX data ETB
Transparent block, the last or only block of text.	DLE STX data ETX
A nontransparent block, the last or only block of text, with a heading.	SOH heading STX data ETX
A transparent block, the last or only block of text, with a heading.	SOH heading DLE STX data ETX
A heading all by itself.	SOH heading ETB
A nontransparent block to be ignored/aborted.	STX data ENQ
A transparent block to be ignored/aborted.	DLE STX data DLE ENQ

Figure A-1

Construction of Data Blocks Using Control Characters

execution of a CWRITE will vary substantially depending on protocol driver, line state, and on which parameters are specified.

A.5 CONTENTION DRIVER.

Unconnected line state

Line state: unconnected --------> text

Function:
a. establish physical connection
b. bid for the line (optionally with I/O verification)
c. send contents of output buffer
d. receive either an acknowledgement of the sent text block or a text block from the remote.

Note:
1. If the input buffer is omitted then only an acknowledgement is expected from the remote in step d.
2. If both the input and output buffers are not specified then only steps a and b will be performed.

Control line state

Line state: control --------> text

Function:
a. bid for the line
b. send contents of output buffer
c. receive either an acknowledgement of the sent text block or a text block from the remote.

Note:
1. If the input buffer is omitted then only an acknowledgement is expected from the remote in step c.
2. If both the input and output buffers are not specified then only step a is executed.
3. While the line is in control state the CS driver listens for a line bid from the remote. If the line bid occurrs then the line is automatically placed into text state. If bit 3 of the DOPTIONS is set to a one, then the driver will ignore remote line bids between requests.

Text line state

Line state: text --------> text

Function: a. send contents of output buffer
b. receive either an acknolwedgement of the sent text block or a text block from the remote.

Note: If the input buffer is omitted then only an acknowledgement is expected from the remote.

6 MULTIPOINT CONTROL DRIVER.

NOTE: If the text is to be sent in blocks, each block is acknowledged by the receiving station. Conceptually, however, the transmission of the entire buffer of data may be described as follows below.

Unconnected line state

Line state: unconnected ----------> text
or
unconnected ----------> control

Function: a. Establish physical connection.
b. Optionally perform ID verification as specified by the Line Descriptor in SUPLIST.
c. If the STATION parameter specified only one station then:
1) select that station and receive acknowledgement
2) send the contents of the output buffer
3) receive an acknowledgement of the sent text
4) final line state = Text
d. Else if the STATION parameter specifies general select or line select then send the contents of the output buffer. The final line state is Control.

Control line state

Line state: control ----------> text
or
control ----------> control

Function: Same as steps c - d of the unconnected line state.

Text line state

Line state: text ----------> text
or
text ----------> control

Function: a. If the STATION parameter specifies the current station and the previous operation was also a write then:
1) send the contents of the output buffer to the station
2) receive an acknowledgement of the sent text block
3) final line state = Text
b. Else if the STATION parameter specifies another station and the previous operation was also a write then:
1) select the station (includes sending an EOT sequence)
2) send the contents of the output buffer
3) receive an acknowledgement of the sent text block
5) final line state = Text
c. Else if the STATION parameter specifies a station and the previous operation was a read and DOPTIONS.(2:1) specifies that an RVI should be issued before changing stations, then:
1) send an RVI and receive an EOT response
2) perform the steps outlined in "b." above
c. Else if the STATION parameter specifies general or line select then:
1) send an EOT sequence
2) send the group or line select sequence followed by the contents of the output buffer
3) final line state = Control

Note: With this one exception the pollist index remains unchanged:
1) text state has been previously entered through a CREAD specifying the POLLIST,
2) the STATION parameter specifies the current station,
3) and the remote terminal responds with an EOT instead of an acknowledgement or text.

A.7 MRJE DRIVER

Unconnected line state:

Line state: unconnected --------> text

Function: a. establish phsyical connection
b. bid for the line using MRJE line bid (send SOH ENQ, receive ACK0)
c. send contents of output buffer
d. receive either an acknowledgement of the sent text block (always ACK0) or a block from the remote.

Note: CWRITE defined only for MRJE when specifying both input and output buffers.

Control line state: (not defined for MRJE)

Text line state:

Line state: text --------> text

Function:
a. send contents of output buffer
b. receive either an acknolwedgement of the sent text block (always ACK0) or a text block from the remote.

Note: CWRITE is defined for MRJE only when specifying both input and output buffers.

A.8 SCCP and SSLC BSC Driver Message Format Word (MFW)

All BSC control characters are inserted into the outgoing data stream by the driver. SYN and block check character (bcc) sequences are always generated. However, the remaining message format is specified by the message format word (MFW). The value of MFW is determined by DOPTIONS.(11:2). Refer to the DOPTIONS parameter in the COPEN description (section 4.2).

MFW (0:1) -- 0= the text block is to be sent in BSC's transparent mode.

1= the text block is to be sent in non-transparent mode. NOTE: It is the user's responsibility to ensure that his text block contains no BSC control characters.

MFW (1:1) -- 0= there are no intermediate text blocks (itb's) contained within the text block.

1= the text block contains itb's. The length of each itb (in + bytes) must precede its block. The length is a 16-bit quantity and must begin on a word boundary. The last itb must be followed by a delimiter word set to all ones.

MFW (2:1) -- This field is significant only if intermediate text blocks are to be sent in non-transparent mode.

0= no SOH/STX character will be inserted after every itb.

Exception: The STX character is always inserted after the header, even if this coincides with the end of an itb.

1= an SOH/STX character will be injected into the text stream after every itb.

MFW (3:1) -- 0= the text block shall end with an ETX character.
1= the text block shall end with an ETB character.

MFW (4:4) -- Reserved for future use; should be set to zero.

MFW (8:8) -- Length of the header portion of the text block (in + bytes). If this field is zero, then there is no header portion.

NOTE: it is the user's responsibility to ensure that there are no BSC control characters within the header.

Note that the MFW, itb lengths, and itb delimiter are for the driver's information and are not actually transmitted. The buffer's tcount parameter must include the MFW and the itb lengths and delimiter.

A.9 Message Formatting

As was previously mentioned, the HP-supplied CS drivers CSSBSC0 and CSSBSC1 (SSLC), and CSHBSC0 (HSI) employ the BSC protocol. Associated with the use of any protocol are the concepts of line control and message formatting. The HP drivers assume the burden of line control, but the user is responsible for message formatting.

In many communications systems (e.g., IBM's BTAM), the job of message formatting requires the user to insert the proper BSC control characters at the appropriate points in his text stream prior to issuing a write request, and he must remove them from the text received from a read request. This is not true of CS/3000.

The SCCP and SSLC drivers have adopted the convention of using a message format word (MFW) which the user optionally supplies as the first word of his output buffer on write requests and which governs the automatic insertion of the appropriate BSC control characters by the driver. On read requests, the driver automatically removes the BSC control characters from the text stream and constructs the appropriate MFW which optionally becomes the first word of the user's input buffer. The INCOUNT and OUTCOUNT parameters of CREAD/CWRITE always include the MFW, but the MFW is never actually transmitted across the line.

The HSI driver, on the other hand, does not use an MFW; all data in the output buffer is transmitted across the line. Because the user is guaranteed that he is communicating with another HP computer (3000 or 2100) when he uses the HSI, there is no need for drivers to insert/remove special BSC control characters in/from the text stream. Hence, there is no MFW interpretation or construction by the HSI drivers on the 3000 or 2100. Therefore, if the HSI user wants to send BSC headers or intermediate text blocks, he may do it using any convention that is agreeable to both sender and receiver.

A.10 User Program Compatibility Between SCCP, SSLC and HSI

The use of BSC as the protocol for both types of boards means that the line control mechanism (i.e., state transition) is identical for these devices. However, from the above discussion on message formatting, it is clear that there exists different message formatting requirements.

If line control and message formatting were identical for both the SSLC and the HSI, then a given CS application program could use either device interchangeably (without even recompiling) by simply supplying different :CLINE commands with each execution of the program.

The desirability of this capability suggests that 3000 users either:

1. Adopt the SCCP and SSLC BSC MFW convention for use with the HSI even though the HSI drivers on HP 3000 and HP 2100 do not even recognize an MFW.

2. Always use an implicit MFW with the SCCP and SSLC BSC (refer to DOPTIONS in section 4.2).

If users adopt either of these conventions, their programs will achieve a higher degree of flexibility.