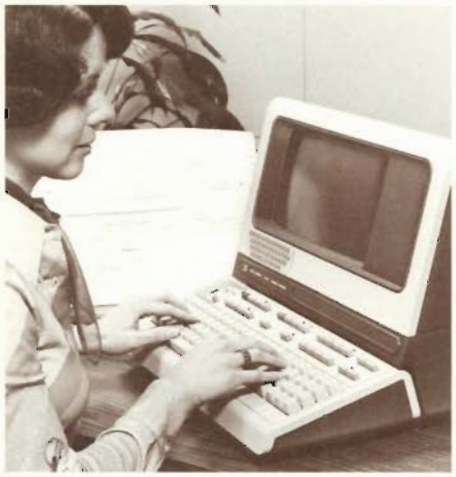
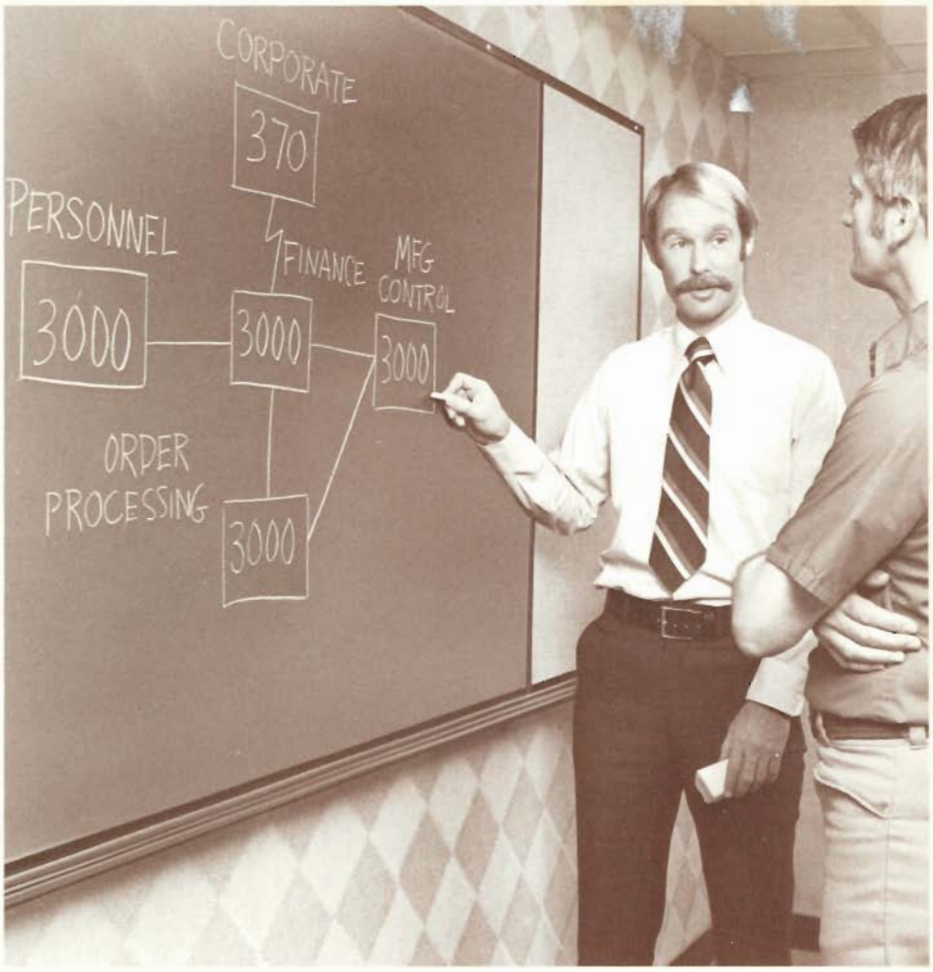
HP Distributed Systems Network

HEWLETT PACKARD

DS/3000

Reference Manual





HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

HP Distributed Systems Network

DS/3000

Reference Manual

HEWLETT PACKARD

5303 STEVENS CREEK BLVD., SANTA CLARA, CALIFORNIA 95050

Part No. 32190-90001
Product No. 32190A

Printed in U.S.A. 3/77

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the most recent date on which the technical material on any given page was altered. If a page is simply re-arranged due to a technical change on a previous page, it is not listed as a changed page. Within the manual, changes are marked with a vertical bar in the margin.

Pages	Effective Date
Title	March 1977
ii to ix	March 1977
1-1 to 1-7	March 1977
2-1 to 2-90	March 1977
3-1 to 3-15	March 1977
4-1 to 4-49	March 1977
5-1 to 5-33	March 1977
A-1 to A-33	March 1977
B-1 to B-11	March 1977
C-1 to C-4	March 1977
D-1 to D-2	March 1977
E-1 to E-4	March 1977
F-1 to F-6	March 1977
G-1	March 1977

PRINTING HISTORY

New editions incorporate all update material since the previous edition. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date on the title page and back cover changes only when a new edition is published. If minor corrections and updates are incorporated, the manual is reprinted but neither the date on the title page and back cover nor the edition change.

First Edition . . . . March 1977

PREFACE

This manual documents the Hewlett Packard Distributed Systems Network for the HP 3000 Computer System. It explains how an HP 3000 user can communicate with several HP 3000 computer systems by establishing a DS/3000 communications link.

The manual is a reference and tutorial text for new HP DS/3000 users. A new user should be familiar with the basic operating principles of the HP 3000 Series II Computer System and knowledgeable of the following manuals:

- HP 3000 Series II Computer System, Commands Reference Manual (30000-90009).
- HP 3000 Series II Computer Systems, Intrinsics Reference Manual (30000-90010).
- HP 3000 Series II Computer Systems, System Manager/ System Supervisor Reference Manual (30000-90014).
- HP 3000 Series II Computer Systems, Console Operators Reference Manual (30000-90013).

CONTENTS

Chapter 1 - INTRODUCING DS/3000 1-1

Chapter 2 - THE COMMUNICATIONS LINK

What is a Communications Link? 2-1
Opening a Hardwired Line 2-3
Specifying a Line 2-5
The DSLINE Command 2-7
Multiple Users 2-8
The REMOTE HELLO Command 2-15
Opening Multiple Lines 2-21
Line Opening Failures 2-29
Opening a Telephone Line 2-33
Specifying a Line 2-36
The DSLINE Command 2-39
Dialing the Remote Computer 2-43
ID Sequences 2-44
Multiple Users 2-45
The REMOTE HELLO Command 2-66
Opening Multiple Lines 2-73
Line Opening Failures 2-81
Closing a Line 2-87
Examples 2-89

Chapter 3 - REMOTE SESSIONS

Issuing Remote Commands 3-2
Using The Remote Subsystem From a Batch Job 3-7
The BREAK Key 3-8
The Control Keys 3-11
Issuing Local Commands 3-12
Terminating a Remote Session 3-13
From the Local Session 3-13
From the Remote Session 3-15

Chapter 4 - REMOTE FILE ACCESS

Command Access 4-2
Programmatic Access 4-32
Example 4-48

Chapter 5 - PROGRAM-TO-PROGRAM COMMUNICATIONS

The POPEN Intrinsic 5-9
The PREAD Intrinsic 5-15
The PWRITE Intrinsic 5-17
The PCONTROL Intrinsic 5-19
The PCLOSE Intrinsic 5-21
The GET Intrinsic 5-23

CONTENTS (continued)

The ACCEPT Intrinsic 5-25
The REJECT Intrinsic 5-27
The PCHECK Intrinsic 5-29
Example 5-30

Appendix A - CONFIGURATION DIALOGUE A-1

Appendix B - ERROR CODES AND MESSAGES

Communication Link Errors B-1
Line Opening Failures B-3
Errors on REMOTE HELLO B-7
Remote File Access Errors B-8
Program-To-Program Errors B-9

Appendix C - DSLINE CONSOLE OPERATOR COMMAND C-1

Appendix D - DS/3000 TRACE FACILITY

Invoking The Trace Facility D-1
Dumping The Trace File D-2
Example D-2

Appendix E - SYSTEM VERIFICATION TEST

Diagnostic Mode E-1
Normal Mode E-3
Example E-4

Appendix F - MODEM OPTIONS F-1
Appendix G - HP CHARACTER SET G-1

ILLUSTRATIONS

1-1. HP 3000 to HP 3000 Example 1-2
1-2. Initiating the Local Session 1-3
1-3. Initiating the Remote Session 1-4
2-1. DS/3000 Communications Link 2-2
2-2. DS/3000 Line Buffer Example 2-4
2-3. Sample I/O Device Table 2-6
2-4. The DSLINE Command 2-7
2-5. HSI Multiple User Example 2-9
2-6. HSI Multiple User Example 2-10
2-7. HSI Exclusive Option Example 2-11
2-8. HSI EXclusive Option Example 2-12
2-9. HSI Exclusive Option Example 2-13
2-10. HSI Exclusive Option Example 2-14
2-11. The REMOTE HELLO Command 2-16
2-12. Multiple Line Example (HSI Lines) 2-22
2-13. Initiating the Local Session (HSI Example) . 2-24
2-14. Establishing the Link With System B 2-26
2-15. Establishing the Link With System C 2-28
2-16. DS/3000 Line Buffer Example 2-34
2-17. Sample I/O Device TAble 2-38
2-18. The DSLINE Command 2-39
2-19. SSLC Multiple User Example 2-46
2-20. SSLC Multiple User Example 2-48
2-21. SSLC Multiple User Example 2-50
2-22. SSLC Multiple User Example 2-52
2-23. SSLC Multiple User Example 2-54
2-24. SSLC Multiple User Example 2-56
2-25. SSLC Multiple User Example 2-58
2-26. SSLC Multiple User Example 2-60
2-27. SSLC Multiple User Example 2-62
2-28. SSLC Multiple User Example 2-64
2-29. The REMOTE HELLO Command 2-67
2-30. Multiple Line Example (SSLC Lines) 2-74
2-31. Initiating the Local Session (SSLC Example) . 2-76
2-32. Establishing the Link With System B 2-78
2-33. Establishing the Link With System C 2-80
2-34. The DSLINE Command 2-87
4-1. MPE FILE Command Syntax For New or Old Files 4-3
4-2. MPE File Command Syntax For User Pre-Defined (Back-Referenced) Files 4-4
4-3. MPE FILE Command Syntax For System-Defined Files 4-5
4-4. MPE FILE Command Parameters 4-6
4-5. Remote Off-Line Listing Example 4-18
4-6. SORT Remote File Access EXample 4-22
4-7. FCOPY Remote File Access Example 4-25
4-8. COBOLGO Remote File Access Example 4-28
4-9. COBOL Remote File Access EXample 4-31
4-10. MPE FOPEN Intrinsic Syntax 4-33
4-11. MPE FOPEN Intrinsic Parameters 4-34
5-1. Interprogram Communication Using Remote File Access 5-2

ILLUSTRATIONS (continued)

5-2. POPEN Activity 5-10
5-3. PREAD Activity 5-15
5-4. PWRITE Activity 5-17
5-5. PCONTROL Activity 5-19

TABLES

5-1. Master Program-to-Program Intrinsics 5-4
5-2. Slave Program-to-Program Intrinsics 5-5
5-3. Single System/Distributed System Comparison . 5-6
F-1. 201A3 Options and Recommendations F-1
F-2. 201B3 Options and Recommendations F-2
F-3. 201C Options and Recommendations F-3
F-4. 208A Options and Recommendations F-4
F-5. 208B Options and Recommendations F-5
F-6. 209A Options and Recommendations F-6

INTRODUCING DS/3000

SECTION I

The Hewlett-Packard Distributed Systems Network is a combination of hardware and software products that make it possible for Hewlett-Packard computer systems to communicate with one another. The connections can be made over hardwired (coaxial cable) lines or over the public telephone (switched) facility, in any mixture. This capability, coupled with our proven remote entry capability to IBM computer systems, provides a total solution to large company EDP needs.

But just what exactly does this overall capability mean? It means that a large multi-divisional corporation can have a truly coordinated world-wide network of computer systems. Coordinated in the sense of tieing together the various commercial/industrial functions within each division and factory and coordinated in the larger sense of tieing together the various divisions and factories at the corporate level.

For example, imagine a large corporation which has factories in the United States, Canada, France, and West Germany. Within each factory there are HP 3000 computer systems performing such functions as inventory control, factory data collection, and operations management. With a Hewlett-Packard Distirbuted Systems Network these manufacturing information systems can be tied into an HP 3000 system which handles the factory's administrative functions (such as finance and accounting). The administrative systems of each factory can, in turn, be connected both to one another and (via remote job entry) to a large computer facility at corporate headquarters. This overall networking capability makes it possible to perform financial analysis and control at a group and corporate level as well as for the individual factories.

This manual describes how an HP 3000 user can communicate with several HP 3000 computers by establishing a DS/3000 communications link. DS/3000 is that part of the HP Distributed Systems Network in which several HP 3000 computer systems are connected to one another.

Imagine, if you will, that you are in the same room with the HP 3000 we have labeled "System A" in figure 1-1 and that the HP 3000 we have labeled "System B" resides in another part of the building. The two computers are connected to one another by a coaxial cable and a pair of communications interfaces (called Hardwired Serial Interfaces, or HSIs for short). By virtue of DS/3000 you can use the processing capability of both machines and pass data back and forth between them by entering commands through a single terminal.

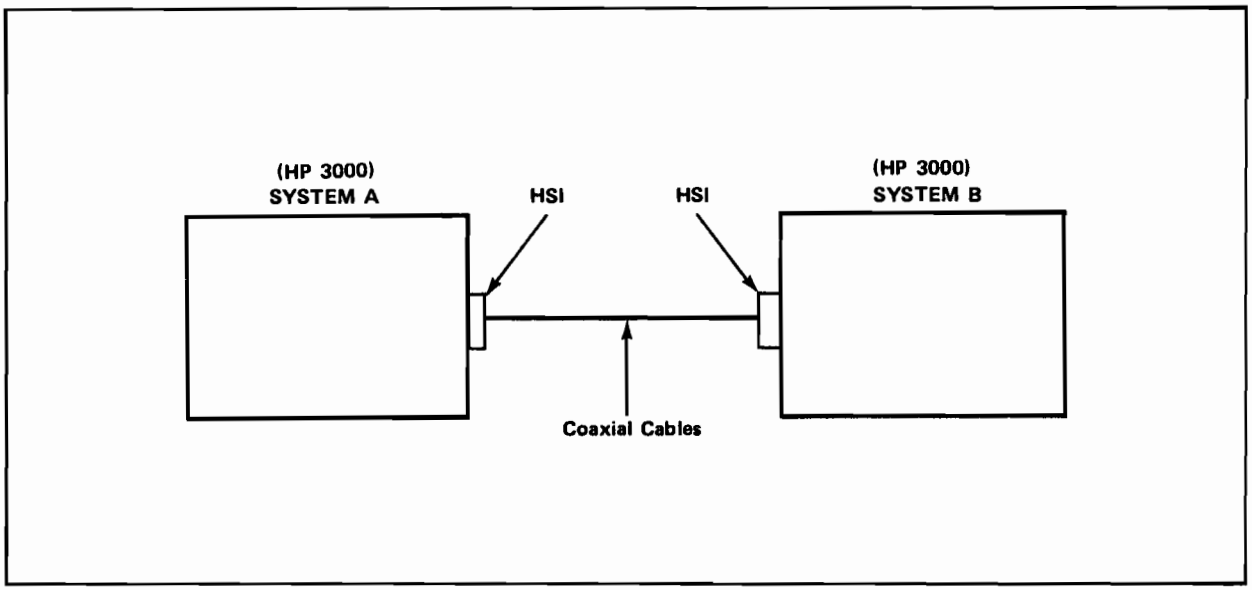


Figure 1-1. HP 3000 to HP 3000 Example

1. First let's sit down at a terminal connected to System A and initiate a session.

```
carriage return
:HELLO USER,ACCOUNT

 SESSION NUMBER = #S49
 FRI, MAY 9, 1976, 9:05 AM
 HP32002A.00.A1

WELCOME TO SYSTEM A.
:
```

Within the context of DS/3000, such a session is referred to as a local session because it is active within the HP 3000 to which our terminal is directly connected. Don't let this terminology confuse you. In actual fact all we have done so far is initiate a standard MPE session. At this point we have the situation illustrated in figure 1-2.

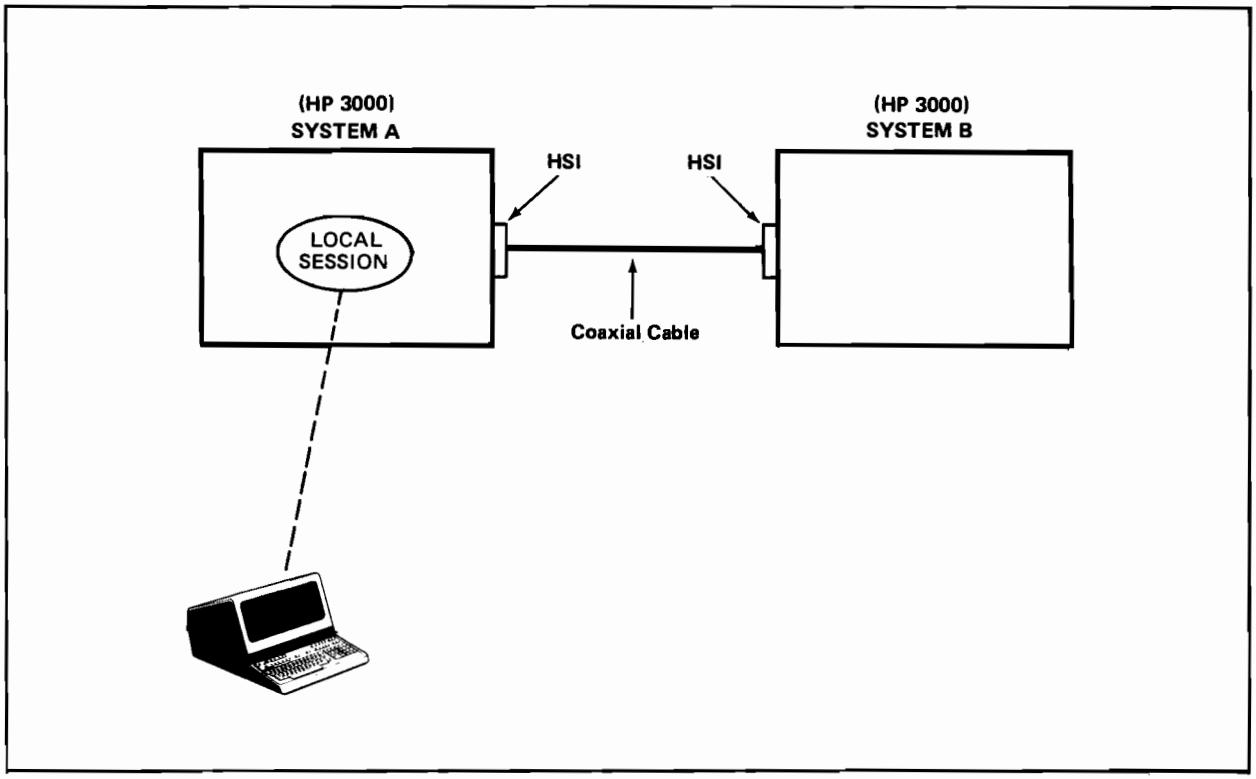


Figure 1-2. Initiating the Local Session

2. Now let's open a communications line between System A and System B. We do this by entering a DSLINE command.

```
:DSLINE HDS2

DS LINE NUMBER = #L3

:
```

In this example HDS2 is the device class name established during system configuration (in System A) for the Hardwired Serial Interface connected to the particular line we wish to use. DS/3000 opens the line and then assigns us a line number (3 in this example). This line number is analagous to the file number returned to you by the MPE File System when you open a file programmatically using the FOPEN intrinsic. Within your local session it uniquely identifies the particular line that you have opened. This becomes significant only if you must open more than one communications line during a session.

3. Now that we have acquired access to a communications line between System A and System B, let's initiate a session in System B (from our local log-on terminal). We do this by entering a REMOTE HELLO command.

```
:REMOTE HELLO RUSER.RACCOUNT

 SESSION NUMBER = #S11
 FRI, MAY 9, 1976, 9:08 AM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```

Within the context of DS/3000, this type of session is referred to as a remote session because it is active within the HP 3000 that is connected indirectly to our log-on terminal by way of a communications line and our local HP 3000. We now have two distinct sessions in progress concurrently, one in System A (under the user and account names USER.ACCOUNT) and one in System B (under the user and account names RUSER.RACCOUNT). It is important to keep in mind that within System A our local session is operating under the capabilities and security restrictions defined (by the accounting structure of System A) for USER.ACCOUNT while within System B our remote session is operating under the capabilities and security restrictions defined (by the accounting structure of System B) for RUSER.RACCOUNT. At this point we have the situation illustrated in figure 1-3. As will be seen in the next few steps, we can alternate freely between the two sessions.

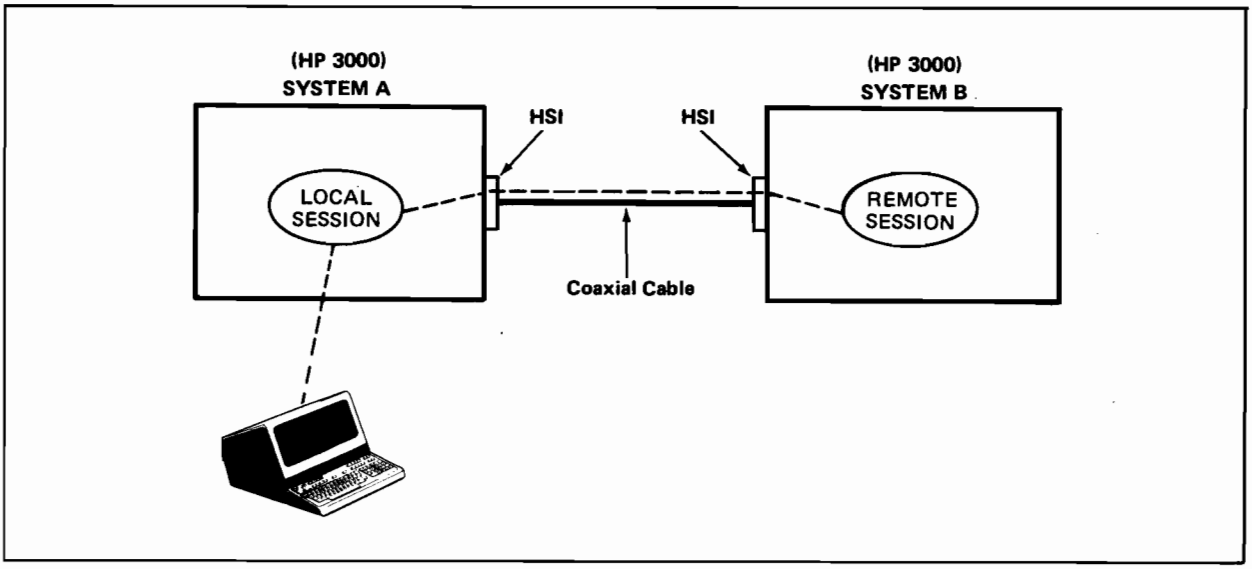


Figure 1-3. Initiating the Remote Session

4. Now let's see what files reside in the public group of the ACCOUNT account in System A.

```
:LISTF

FILENAME

DATA1     DATA3     FILE1     SOURCE2     SOURCE5
:
```

We can do the same for the public group of the RACCOUNT account in System B by entering the following command through the same terminal:

```
:REMOTE LISTF

FILENAME

DATA1     DATA5     DATA6     FILE3     SOURCE1
:
```

Notice that in both cases we entered the same command but in the latter case we used the prefix REMOTE. The presence or absence of that prefix is what determines whether a command is to be executed in the local session or in the remote session.

5. As a result of the LISTF and REMOTE LISTF displays we discover that a source file, named SOURCE1, exists in System B but not in System A. We wish to modify one of the statements in that program. To do that we will use the text editor in System B. This time, instead of prefixing our remote commands with REMOTE , let's try something a little different. We enter the following:

```
:REMOTE
#
```

This construct gets us into the remote session in such a way that all commands can be entered in their normal form (without the prefix REMOTE). The # is the prompt character issued by DS/3000 (in place of the usual MPE colon prompt). In all other respects it will seem to you as though you are executing a normal MPE interactive session.

6. Now let's invoke the text editor, copy the content of SOURCE1 (which is a file in System B) into the editor's work file, display the content of the work file, modify the desired statement, and store the altered source code back in SOURCE1.

```
#EDITOR

HP32201A.5.00 EDIT/3000  FRI, MAY 9, 1976, 9:11 AM
(C) HEWLETT-PACKARD CO. 1976
/SET SHORT
/TEXT SOURCE1
/LIST ALL
    1    $CONTROL USLINIT,SOURCE
    2     IDENTIFICATION DIVISION.
    3     PROGRAM-ID. COBOL-TEST1.
    4     AUTHOR. JOE DOKES.
    5     ENVIRONMENT DIVISION.
    6     DATA DIVISION.
    7     WORKING-STORAGE SECTION.
    8     77  EDIT-FIELD      PIC $Z,ZZ9.99.
    9     77  TOTAL-COST      PIC 999V99.
   10     77  COST-OF-SALE    PIC 99V99.
   11     77  TAX             PIC 99V99.
   12     77  Y-N             PIC X.
   13
   14     PROCEDURE DIVISION.
   15     ENTER-ROUTINE.
   16         MOVE ZEROS TO TOTAL-COST.
   17         DISPLAY SPACE.
   18         DISPLAY "ENTER COST OF SALE".
   19         ACCEPT COST-OF-SALE.
   20         COMPUTE TAX = COST-OF-SALE * .06.
   21         ADD COST-OF-SALE, TAX TO TOTAL-COST.
   22         MOVE TOTAL-COST TO EDIT-FIELD.
   23         DISPLAY "TOTAL COST = " EDIT-FIELD.
   24         DISPLAY "ARE YOU FINISHED? (Y OR N)".
   25         ACCEPT Y-N.
   26         IF Y-N = "N" GO TO ENTER-ROUTINE.
   27         STOP-RUN.
/MODIFY 18
MODIFY   18
    DISPLAY "ENTER COST OF SALE".
                                  I (NO DECIMAL POINT)
    DISPLAY "ENTER COST OF SALE (NO DECIMAL POINT)".
/KEEP SOURCE1
PURGE OLD? YES
/EXIT
CLEAR? YES

 END OF SUBSYSTEM
#
```

7. Our work in System B is now completed, so let's terminate the remote session and return control to our local session.

```
#BYE

CPU (SEC) = 4
CONNECT (MIN) = 7
FRI, MAY 9, 1976, 9:15 AM
END OF SESSION
#:
:
```

Note that we are now back in the local session in System A (signified by the colon prompt). The remote session no longer exists, but the communications line is still open. We could, if we wanted, initiate another remote session over the line by issuing another REMOTE HELLO command. To close the communications line we enter the following variation of the DSLINE command:

```
:DSLINE ;CLOSE
:
```

Finally we terminate the local session.

```
:BYE

CPU (SEC) = 1
CONNECT (MIN) = 11
FRI, MAY 9, 1976, 9:16 AM
END OF SESSION
```

THE COMMUNICATIONS LINK

SECTION II

WHAT IS A COMMUNICATIONS LINK?

Within the context of DS/3000, a "communications link" consists of the following elements:

- a normal interactive session in progress in an HP 3000 computer.
- a physical communications line between that HP 3000 and a remote HP 3000 computer.
- an interactive session in progress in the remote HP 3000 (initiated over the physical communications line from your local session).

Note that your local terminal is the log-on terminal for both the local session and the remote session. (Refer to figure 2-1.)

A communications link can be established over a hardwired communications line or over the public telephone network. The remainder of this chapter tells you first how to establish one or more links over a hardwired line, then how to establish one or more links over the public telephone network, and finally how to terminate one or more communications links.

This chapter treats hardwired and telephone lines separately merely to avoid confusion. In an actual DS/3000 configuration you may have both types of lines available and in use concurrently.

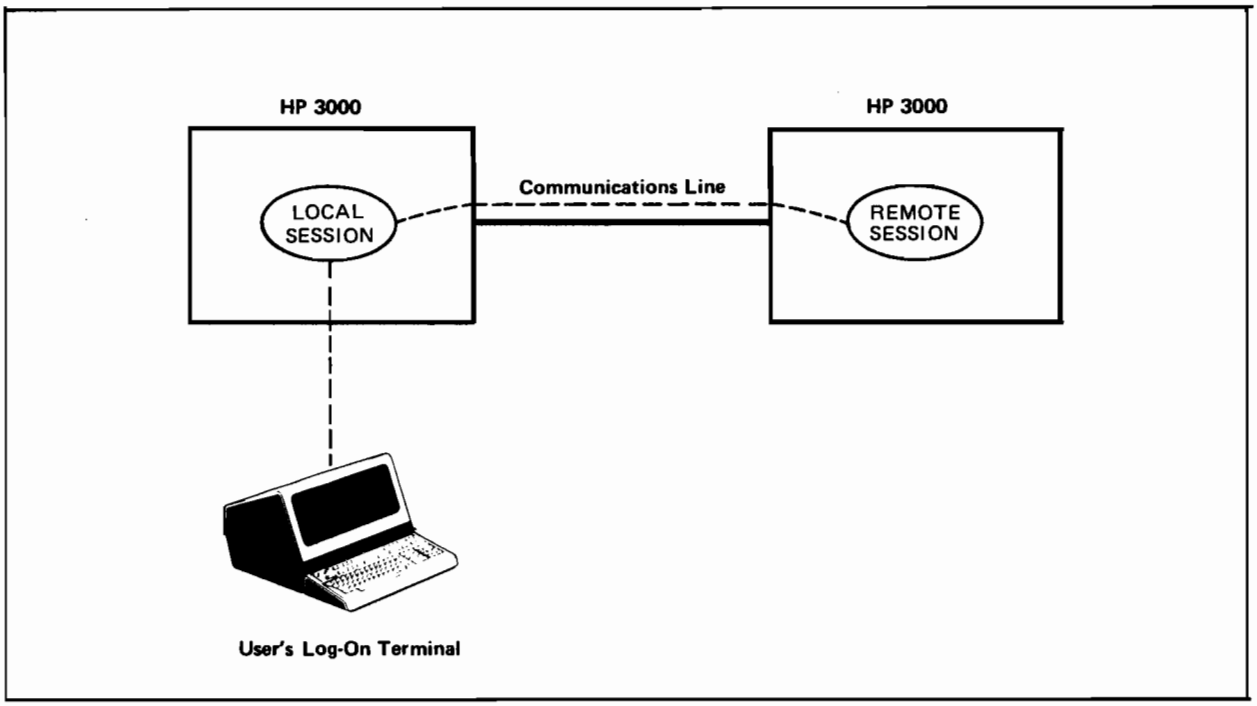


Figure 2-1. DS/3000 Communications Link (HP 3000 to HP 3000)

OPENING A HARDWIRED LINE

What is a hardwired line? In the general field of data communications there are two types of lines commonly referred to as "hardwired". The first type is a dedicated path on the public telephone network that is leased from the telephone company for the private use of a computer-to-computer configuration. Such a line serves as a permanent connection between the two computers and is interfaced to them by way of modems. Modem is a contraction of MOdulator-DEModulator. A modem is a device that translates digital signals (electrical impulses) generated by a computer into analog signals (tones) that can be transmitted over telephone lines, and vice versa. The modem, if present, is connected between the communications I/O interface of the computer and the telephone line. The other type of hardwired line is a coaxial cable that is connected directly to the communications I/O interfaces of the two computers without the use of modems. Within the context of DS/3000, "hardwired" always refers to the latter.

The coaxial cable is interfaced to each HP 3000 by way of the HP 30360A Hardwired Serial Interface (HSI). An HSI occupies one card slot in the Multiplexer Channel. Each HSI can accomodate up to four hardwired lines and each line may be connected to a separate remote computer. Only one of the four lines, however, is available for use at any given time. The console operator of your HP 3000 controls which HSI line is currently available.

In the case of a hardwired line it is relatively straightforward to obtain access to the line. All you are required to do is identify the particular HSI you wish to use. You do this by specifying the device class name or logical device number associated during system configuration with the desired HSI. In the example at the end of chapter 1 we used the DSLINE command for this purpose, as follows:

```
:DSLINE HDS2
```

In the DSLINE command you may also wish to specify the size of the DS/3000 line buffer to be used in conjunction with the line. The size of this buffer determines the maximum sized block that can be sent or received in a single physical transmission over the line. Note that a transmission as you

normally think of it (sending or receiving all or part of a file) may in fact consist of many physical transmissions. This buffer size in essence defines a blocking factor for the line. Refer to figure 2-2. A default buffer size is established during system configuration and in most cases (as in the example at the end of chapter 1) you will find it satisfactory to let this default value prevail.

When you execute a DSLINE command, DS/3000 attempts to give you access to the specified communications line and, if successful, informs you of the assigned DS line number by displaying the following message at your terminal:

```
DS LINE NUMBER = #Lx
```

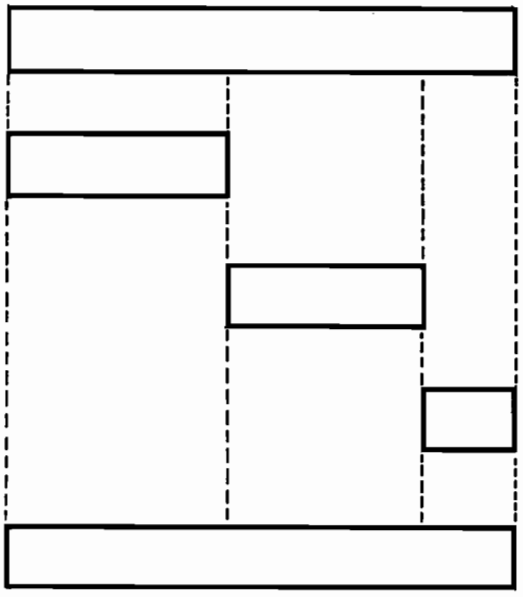
where x is the assigned DS line number. In the example at the end of chapter 1 we were assigned the DS line number "3". The DS line number is significant only if you open and use more than one communications line concurrently within a single local session (see "Opening Multiple Lines" later in this chapter).

At this point you have acquired a physical communications line but the communications link does not yet exist. The actual communications link between the two computers is established by initiating a remote session over the line. You do this by executing a REMOTE HELLO command. In the example at the end of chapter 1 we used a REMOTE HELLO command containing the minimum parameters required (a username and an accountname), as follows:

```
:REMOTE HELLO RUSER.RACCOUNT
```

The communications link between the two HP 3000 computers now exists.

Assume that the DS/3000 line buffer size is 512 words and that the user has initiated the transmission of a block of data 1200 words in length from his HP 3000 to a remote HP 3000. The block of data would actually be sent in three separate physical transmissions, as follows.



1200 words of data to be transmitted

First physical transmission: 512 words

Second physical transmission: 512 words

Third physical transmission: 176 words

1200 words of data received at remote computer.

Figure 2-2. DS/3000 Line Buffer Example

Specifying a Line

In order to open a hardwired communications line, we have seen that you must specify a device class name or logical device number identifying the particular HSI you wish to use. But how do you figure out which name or number to specify? The remainder of this topic will seem, particularly at first reading, a little complex and tedious. Don't let that frighten you. In actual practice, once the hardware and software configuration is installed and usable, most DS/3000 sites will post a simple notice defining all of the available communications lines and the proper device class names and logical device numbers for each. In that case all of the detective work described in the following paragraphs is already done for you.

For each HSI there is a pair of associated drivers. First there is the actual HSI driver that directly controls the operation of the interface board. In addition there is a DS/3000 communications driver that controls the operation of the HSI driver. The names of these drivers are as follows:

CSHBSC0 (HSI driver)

IODS0 (DS/3000 communications driver)

Now look at the sample I/O device table (produced during system configuration) in figure 2-3. If you look at the shaded items in the column labeled "DRIVER NAME" you will notice that we have four HSI lines (CSHBSC0) configured into the system as logical devices 12 through 15. In addition you will notice that for each of these lines there is a DS/3000 communications driver (IODS0) also configured into the system. Each IODS0 entry is related to the proper HSI entry by the number specified in the column labeled "DRT" (the # prefix indicates a back reference to a previously defined logical device number). In figure 2-3 logical devices 50 through 53 are paired with logical devices 12 through 15, respectively. It is the device class name or logical device number of the appropriate IODS0 entry that you use to specify the desired line. (Refer to Appendix A).

One or more pseudo-terminal drivers (IODSTRM0) also should be configured into the system. The IODSTRM0 entry allows another system to be logged on to this system and regulates the number of remote Session Main Processes (SMP) that can be assigned to a given line. Each IODSTRM0 entry is related to the proper HSI entry by the number specified in the column labeled "DRT". Figure 2-3 shows logical devices 60 through 65 are paired with logical devices 11 through 15.

Notice that the HSI board entries (logical devices 11 through 15) look the same except for the PORTMASK. The PORTMASK specifies which port on the board is to be used. There are also pseudo-terminals (logical devices 60 through 65) referencing back to logical device 12.

Since only one port on the HSI board can be opened at a time, only one block of pseudo-terminal entries are needed for that board. As each port is opened individuallly by specifying the corresponding DS entry in the console operator DSLINE command, the system will automatically reallocate the block of dummy terminal entries to that HSI board entry. This reallocation will not, however show up in the I/O configuration table.

LOG DEV #	DRT #	UNIT	CHAN	TYPE	SUB TYPE	TERM TYPE	TERM SPEED	REC WIDTH	OUTPUT DEV	MODE		DRIVER NAME	DEVICE CLASSES
1	4	0	0	0	6			128	0			IOMDISC1	SPOOL SYSDISK
2	5	0	0	0	3			128	0			IOMDISC0	DISC
5	13	0	0	8	0			40	LP	JA	S	IOCDRD0	CARD
6	14	0	0	32	2			66	0		S	IOLPRT0	LP
7	6	0	0	24	0			128	LP			IOTAPE0	TAPE
8	6	1	0	24	0			128	LP			IOTAPE0	TAPE
9	6	2	0	24	0			128	LP			IOTAPE0	TAPE
10	6	3	0	24	0			128	LP	JA	S	IOTAPE0	BATAPE
11	20	0	0	34	0			128	0			IOPTPN0	PTPUNCH
12	16	0	0	19	3			0	0			CSHBSC0	HSI1
13	16	0	0	19	3			0	0			CSHBSC0	HSI2
14	16	0	0	19	3			0	0			CSHBSC0	HSI3
15	16	0	0	19	3			0	0			CSHBSC0	HSI4
20	7	0	0	16	0	10	??	40	20	JAID		IOTERM0	CONSEL
21	7	1	0	16	0	11	??	40	21	JAID		IOTERM0	TERM
22	7	2	0	16	0	11	??	40	22	JAID		IOTERM0	TERM
23	7	3	0	16	0	11	??	40	23	JAID		IOTERM0	TERM
24	7	4	0	16	0	11	??	40	24	JAID		IOTERM0	TERM
25	7	5	0	16	0	11	??	40	25	JAID		IOTERM0	TERM
50	#12	0	0	41	0			128	0			IODS0	HDS1
51	#13	0	0	41	0			128	0			IODS0	HDS2
52	#14	0	0	41	0			128	0			IODS0	HDS3
53	#15	0	0	41	0			128	0			IODS0	HDS4
60	#12	0	0	16	0	??	??	36	60	J	ID	IODSTRM0	DSTERM
61	#12	1	0	16	0	??	??	36	61	J	ID	IODSTRM0	DSTERM
62	#12	2	0	16	0	??	??	36	62	J	ID	IODSTRM0	DSTERM
63	#12	3	0	16	0	??	??	36	63	J	ID	IODSTRM0	DSTERM
64	#12	4	0	16	0	??	??	36	64	J	ID	IODSTRM0	DSTERM
65	#12	5	0	16	0	??	??	36	65	J	ID	IODSTRM0	DSTERM

LDN	PM	PRT	LCL MOD	TC	RCV TMOUT	LCL TMOUT	CON TMOUT	MODE	TRANSMIT SPEED	TM	BUFFER SIZE	D C	DRIVER OPTIONS
12	8	1	1	1	20	60	900	C	250000	1	1024	N	0
13	4	1	1	1	20	60	900	C	250000	1	1024	N	0
14	2	1	1	1	20	60	900	C	250000	1	1024	N	0
15	1	1	1	1	20	60	900	C	250000	1	1024	N	0

Figure 2-3. Sample I/O Device Table

The DSLINE Command

The format of the DSLINE command is presented in figure 2-4 below. In addition to opening a hardwired line, this command can be used for opening a telephone line or for closing one or more communications lines. For completeness all of the available parameters are shown. The shaded parameters apply to opening a hardwired line; the others (non-shaded) apply to opening a telephone line or closing one or more communications lines and will be described when those topics are discussed later in this chapter.

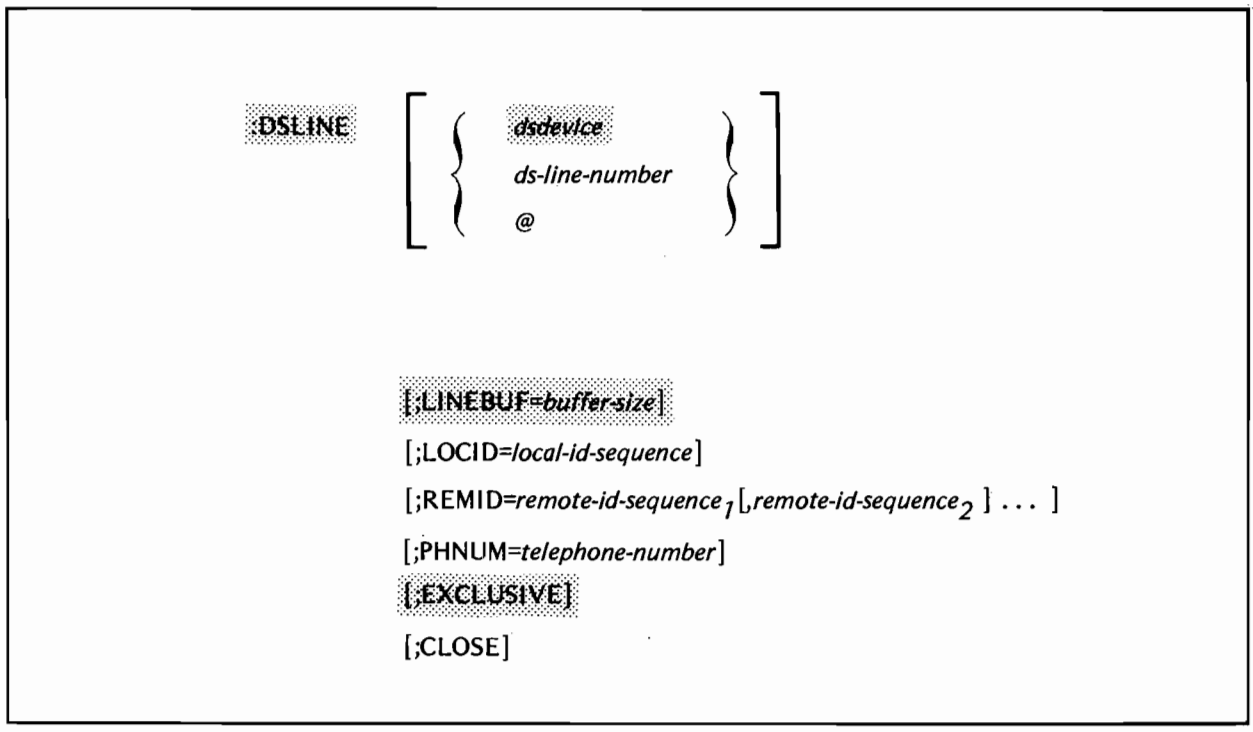


Figure 2-4. The DSLINE Command

The parameters that pertain to opening a hardwired communications line are as follows:

dsdevice — The device class name or logical device number assigned to the DS/3000 communications driver IODS0 during system configuration. This parameter specifies what physical line you wish to use.

(Required parameter.)

buffer-size — A decimal integer specifying the size (in words) of the DS/3000 line buffer to be used in conjunction with the particular communications line. The integer must be within the range 304 < buffer-size < 4096. The default value is the buffer size entered in response to the PREFERRED BUFFER SIZE prompt during system configuration.

(Optional parameter.)

EXCLUSIVE — This parameter, if present, specifies that you want exclusive use of the particular communications line. If the requested line is already open and you have specified the exclusive option, DS/3000 will deny you access to the line (see "Line Opening Failures" later in this chapter). Opening an EXCLUSIVE line requires the user to have CS capability.

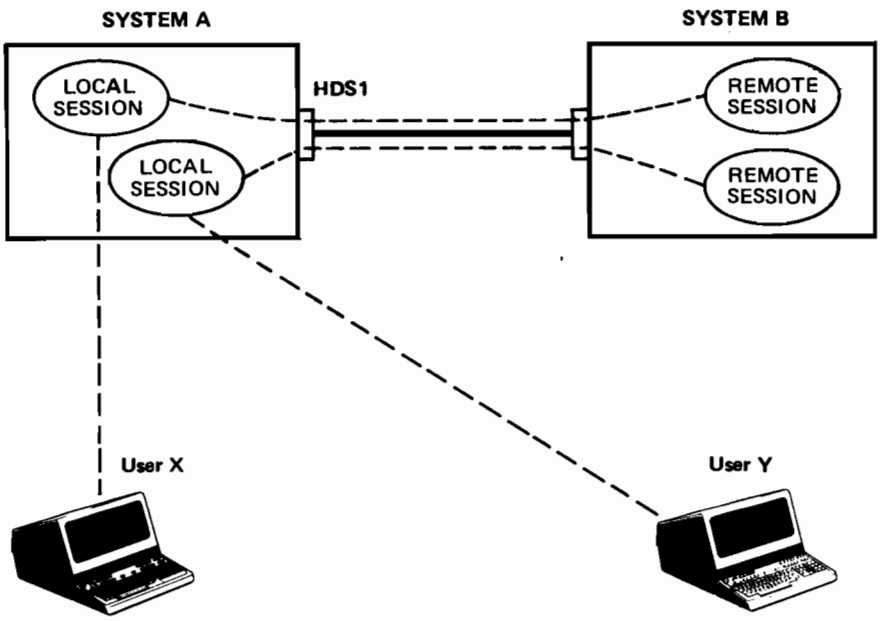
(Optional parameter.)

Multiple Users

Within a DS/3000 environment, it is possible for several users at either end of the line to share access to the same physical communications line or for a single user at one end of the line to obtain exclusive access to the line.

As you probably remember from the presentation of the DSLINE command above, the EXCLUSIVE parameter can be used to obtain exclusive access to the specified physical communications line. If you specify this parameter (and if access to the line is granted), no other user in either computer will be permitted to open that line until you close it. Exclusive access significantly increases your throughput over the line (this is even true when compared with being the only user of the line without the exclusive option). If you ask for exclusive access to a particular line and that line is already in use, DS/3000 denies your request (see "Line Opening Failures" later in this chapter).

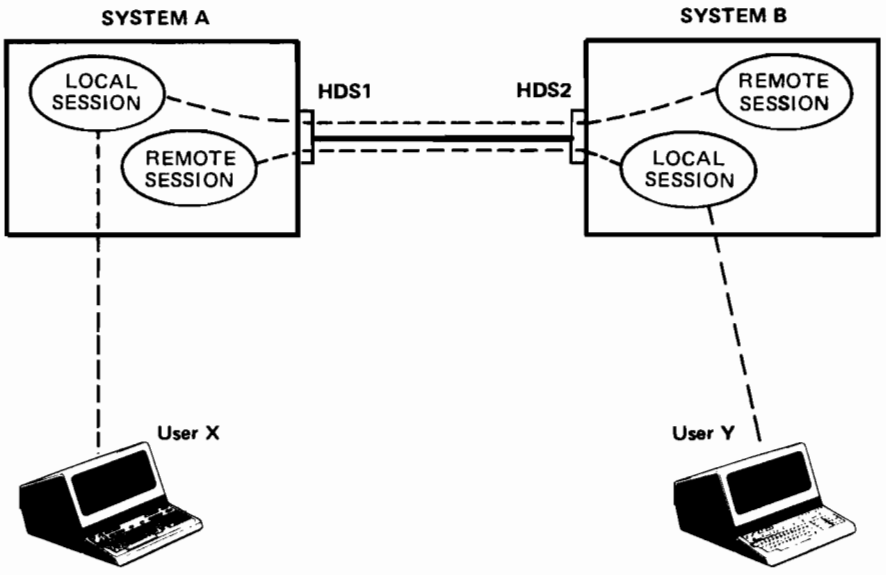
For an HSI line, multiple users at either end of the line can specify the same physical line in DSLINE commands and obtain access to that line as long as none of them requests exclusive access. In such a case the user's data are multiplexed so that each user's access to the line appears to be completely independent of all others. Figures 2-5 through 2-10 present annotated examples illustrating successful and unsuccessful attempts by different users to obtain access to the same HSI line.




```
:HELLO USER.X                    :HELLO USER.Y
:DSLINE HDS1                     :DSLINE HDS1
:REMOTE HELLO USER.X             :REMOTE HELLO USER.Y
```

In this example User X initiates a local session in System A, obtains access to the hardwired communications line that connects System A to System B, and initiates a remote session in System B. User Y subsequently initiates a local session in System A, obtains access to the same communications line, and initiates a remote session in System B. The request by User Y for the particular communications line is granted by DS/3000 because neither user asked for exclusive access to the line.

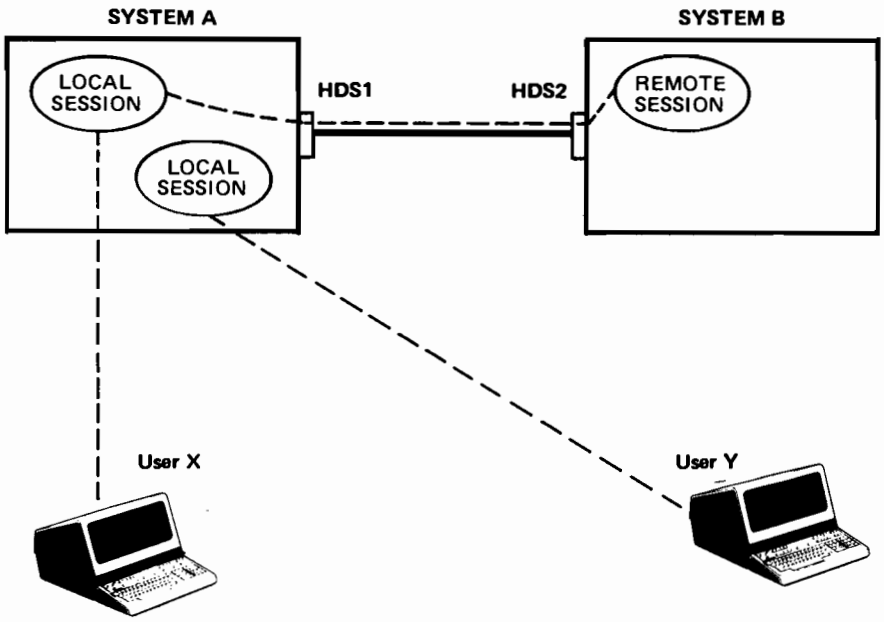
Figure 2-5. HSI Multiple User Example




```
:HELLO USER.X                    :HELLO USER.Y
:DSLINE HDS1                     :DSLINE HDS2
:REMOTE HELLO USER.X             :REMOTE HELLO USER.Y
```

In this example User X initiates a local session in System A, obtains access to the hardwired communications line that connects System A to System B, and initiates a remote session in System B. User Y subsequently initiates a local session in System B, obtains access to the same communications line, and initiates a remote session in System A. The request by User Y for the particular communications line is granted by DS/3000 because neither user asked for exclusive access to the line.

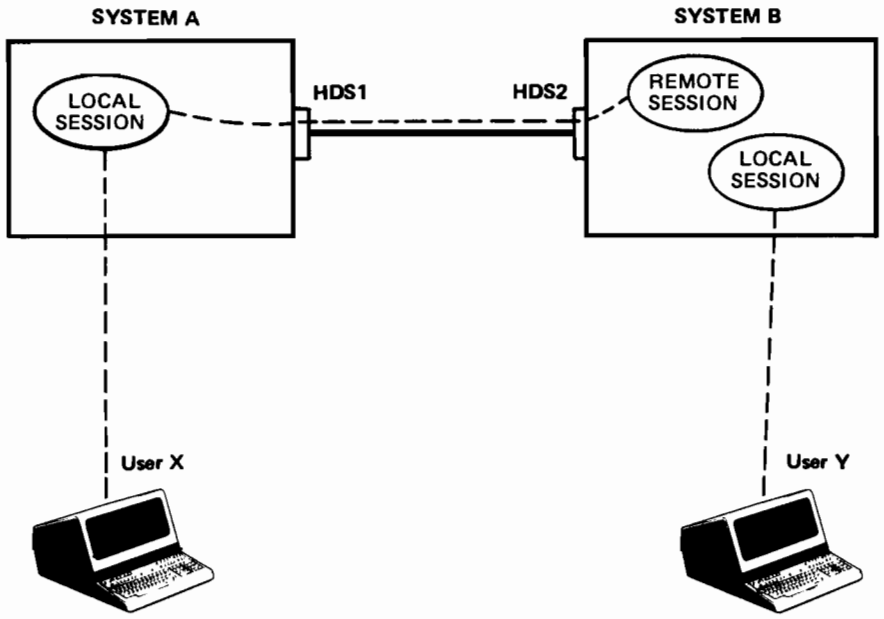
Figure 2-6. HSI Multiple User Example




```
:HELLO USER.X                          :HELLO USER.Y
:DSLINE HDS1 ;EXCLUSIVE                :DSLINE HDS1
:REMOTE HELLO USER.X
```

In this example User X initiates a local session in System A, obtains exclusive access to the hardwired communications line that connects System A to System B, and initiates a remote session in System B. User Y subsequently initiates a local session in System A and requests access to the same communications line. The request is denied by DS/3000 because User X already has exclusive access to the specified line.

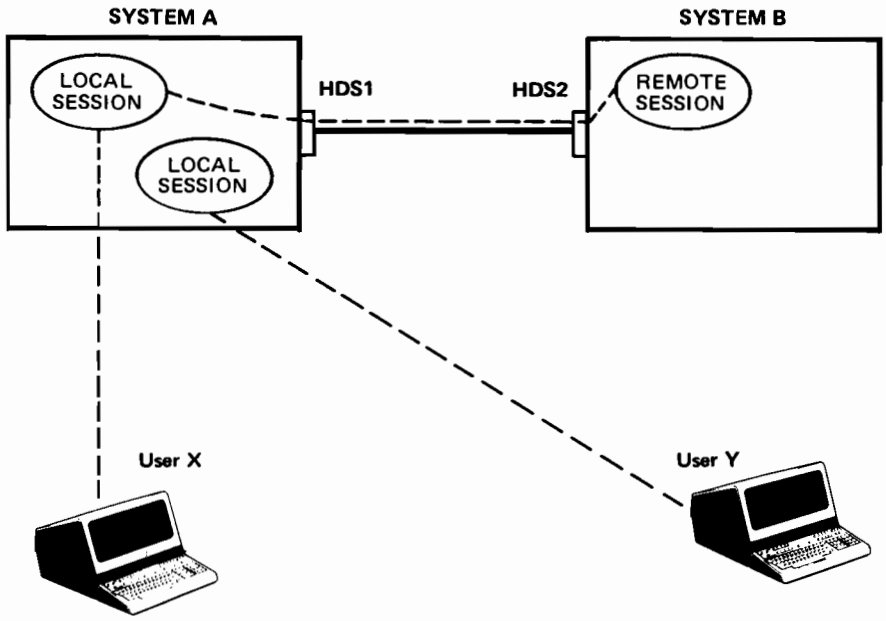
Figure 2-7. HSI Exclusive Option Example




```
:HELLO USER.X                          :HELLO USER.Y
:DSLINE HDS1 ;EXCLUSIVE                :DSLINE HDS2
:REMOTE HELLO USER.X
```

In this example User X initiates a local session in System A, obtains exclusive access to the hardwired communications line that connects System A to System B, and initiates a remote session in System B. User Y subsequently initiates a local session in System B and requests access to the same communications line. The request is denied by DS/3000 because User X already has exclusive access to the specified line.

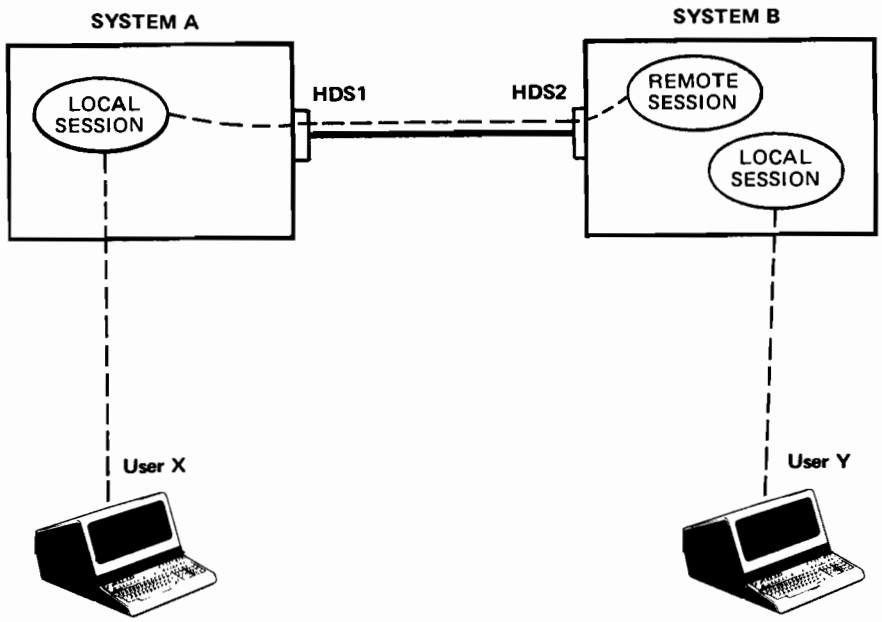
Figure 2-8. HSI Exclusive Option Example




```
:HELLO USER.X                          :HELLO USER.Y
:DSLINE HDS1                           :DSLINE HDS1 ;EXCLUSIVE
:REMOTE HELLO USER.X
```

In this example User X initiates a local session in System A, obtains access to the hardwired communications line that connects System A to System B, and initiates a remote session in System B. User Y subsequently initiates a local session in System A and requests exclusive access to the same communications line. The request is denied by DS/3000 because the specified line is already in use and therefore cannot be assigned exclusively to User Y.

Figure 2-9. HSI Exclusive Option Example




```
:HELLO USER.X                          :HELLO USER.Y
:DSLINE HDS1                           :DSLINE HDS2 ;EXCLUSIVE
:REMOTE HELLO USER.X
```

In this example User X initiates a local session in System A, obtains access to the hardwired communications line that connects System A to System B, and initiates a remote session in System B. User Y subsequently initiates a local session in System B and requests exclusive access to the same communications line. The request is denied by DS/3000 because the specified line is already in use and therefore cannot be assigned exclusively to User Y.

Figure 2-10. HSI Exclusive Option Example

The REMOTE HELLO Command

Once you have obtained access to a physical communications line using the DSLINE command, you use the REMOTE HELLO command to actually establish the communications link. The REMOTE HELLO command initiates a remote session on your behalf in the HP 3000 connected to the other end of the communications line.

The format of the REMOTE HELLO command is presented in figure 2-11 below. You will notice that, except for the three shaded items, it has exactly the same format as the standard MPE HELLO command.

Because the REMOTE HELLO command is initiating a session for you in a remote HP 3000, the parameters in that command specify information which pertains to the operating environment of the remote HP 3000 (not your local one). More specifically you must keep the following in mind:

- sessionname (if present) identifies the remote session and has no relationship to your local session.
- username, accountname, groupname, and their passwords (if any) must all be valid as defined by the accounting structure of the remote HP 3000.
- cpusecs (if present) refers to central-processor time in the remote system.
- BS, CS, DS, ES, inputpriority, and HIPRI (if present) all specify priorities for the remote session within the remote system.
- termtype (if present) has no meaning and is ignored because output from the remote session is directed to the communications line instead of to a terminal. The termtype parameter for your local session implicitly defines your log-on terminal type for any remote sessions that you initiate.

```
:REMOTE HELLO    [sessionname,]username[/userpasw].acctname[/acctpasw]
                 [,groupname[/grouppasw]]

                 [;TERM=termtype]

                 [;TIME = cpusecs]

                 [;PRI = { BS }
                         { CS }
                         { DS }
                         { ES } ]

                 [ ;INPRI = inputpriority ]
                 [ ;HIPRI                 ]

                 [;DSLINE=dsdevice]
```

sessionname — Arbitrary name used in conjunction with *username* and *acctname* parameters to form a fully-qualified session identity. Contains from 1 to 8 alphanumeric characters, beginning with a letter. Default: null session name. (Optional parameter.)

NOTE

A *fully-qualified session identity* consists of:

[*sessionname,*]*username.acctname*

and furnishes the minimum information required for log-on. Embedded blanks are forbidden in *username.acctname* combination.

username — A user name, established by Account Manager, that allows you to log-on under this account. This name is unique within the account. Contains from 1 to 8 alphanumeric characters, beginning with letter. (Required parameter.)

Figure 2-11. The REMOTE HELLO Command

userpasw — Your user password, optionally assigned by Account Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter. Separated from *username* by slash with no surrounding blanks, as in *username/userpasw*. (Required if assigned.)

acctname — Name of your account, as established by System Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter.

NOTE

Must be preceded by period as a delimiter.

(Required parameter.)

acctpasw — Account's password, optionally assigned by System Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter. Separated from *acctname* by slash with no surrounding blanks, as in *acctname/acctpasw*. (Required if assigned.)

groupname — Name of file group to be used for local file domain and central-processor time charges, as established by Account Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter. Default: Your home group if you are assigned one by Account Manager. (*Optional* if you have a home group; *Required* if you do not.)

grouppasw — Group's password, optionally assigned by Account Manager. Contains from 1 to 8 alphanumeric characters, beginning with a letter. Separated from *groupname* by slash with no surrounding blanks, as in *groupname/grouppasw*. (Not needed when you log-on under home group. Otherwise, required if assigned.)

termtype — Ignored. The TERM=termtype parameter of the HELLO command that initiated the local session also implicitly defines the log-on terminal type for any remote sessions initiated from the local session.

Figure 2-11. The REMOTE HELLO Command (Continued)

cpusecs

Maximum central-processor time that your session can use, entered in seconds. When this limit is reached, session is aborted. Must be value from 1 to 32767. To specify no limit, enter question mark or omit this parameter.

Default: no limit. (Optional parameter.)

BS
CS
DS
ES

The execution priority class that the Command Interpreter uses for your session, and also the default priority for all programs executed within the session. BS is highest priority; ES is lowest. If you specify a priority that exceeds the highest permitted for your account or user name by the system, MPE assigns the highest priority possible below BS. Default: CS.

NOTE

DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged.

(Optional parameter.)

inputpriority

Relative input priority used in checking against access restrictions imposed by the *job fence,* if one exists. Takes effect at log-on time. Must be a value from 1 (lowest priority) to 13 (highest priority). If you supply a value less than or equal to current job fence set by Console Operator, session is denied access.

Default: 8 if logging of session/job initiation is enabled, 13 otherwise. (Optional parameter.)

HIPRI

Request for maximum session-selection input priority, causing session to be scheduled regardless of current job fence or execution limit for sessions.

NOTE

You can specify this parameter only if you have System Manager or Supervisor Capability.

(Optional parameter.)

Figure 2-11. The REMOTE HELLO Command (Continued)

dsdevice	The device class name or logical device number assigned to the DS/3000 communications driver IODS0 during system configuration. This parameter, if present, specifies which hardwired line (HSI) you wish to use. (Optional parameter if a line is already open; otherwise it is required.)

Figure 2-11. The REMOTE HELLO Command (Continued)

So far we have been talking entirely about the DSLINE and REMOTE HELLO commands being used in conjunction with one another: the DSLINE command obtaining access to a physical line and the REMOTE HELLO command actually establishing the communications link by initiating a remote session over the acquired line. As you may have guessed from the above parameter definitions, the DSLINE parameter of the REMOTE HELLO command gives us a new, and simpler, way to obtain a line and establish a communications link. If you are satisfied to use the default DS/3000 line buffer size and you do not need exclusive use of the line, you can acquire a line and initiate a remote session over that line by using a single command: a REMOTE HELLO command with the DSLINE parameter. If you open a line in this way, however, it remains open only for the duration of the particular remote session (when the remote session is terminated the line is automatically closed). If, on the other hand, you use the DSLINE command to open a line, the line remains open for the duration of the local session (or until you explicitly close the line).

To illustrate this, let's take another look at the example at the end of chapter 1. In that example we used the DSLINE command to obtain access to the hardwired line HDS2 and the REMOTE HELLO command to initiate a remote session over the line:

```
:DSLINE HDS2

DS LINE NUMBER = #L3

:REMOTE HELLO RUSER.RACCOUNT

 SESSION NUMBER = #S11
 FRI, MAY 9, 1976, 9:08 AM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```

< Note >

In this case the acquired line remains open when the remote session is terminated.

By including the DSLINE parameter in the REMOTE HELLO command we could perform essentially the same operations using a single command, as follows:

```
:REMOTE HELLO RUSER.RACCOUNT;DSLINE=HDS2

DS LINE NUMBER = #L3

 SESSION NUMBER = #S11
 FRI, MAY 9, 1976, 9:08 AM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```

< Note >

In this case the acquired line is closed when the remote session is terminated.

Opening Multiple Lines

Within your local session you may open more than one physical communications line and you may have remote sessions active concurrently over all of the opened lines. You are limited, however, to one remote session per physical line at any given time.

If it is able to obtain access to the specified line, DS/3000 responds to each DSLINE command by displaying a DS line number at your log-on terminal. This line number is roughly analagous to the file number returned by the MPE FOPEN intrinsic in that it is an arbitrary number that uniquely identifies (within your local session) your current access to a particular communications line. It has no relationship to the logical device number or any other configuration parameter associated with the line. DS line numbers are meaningful only if you have more than one line open concurrently within a single local session. In that case you are assigned a separate DS line number for each line you have opened and you subsequently use these numbers to specify which line you wish to use for a given remote command (or sequence of remote commands) or to close a particular line without closing the others.

Figure 2-12 illustrates a situation where a user has established two communications links concurrently from within a single local session. Let's take a closer look at that situation and examine the sequence of commands that was used to create it.

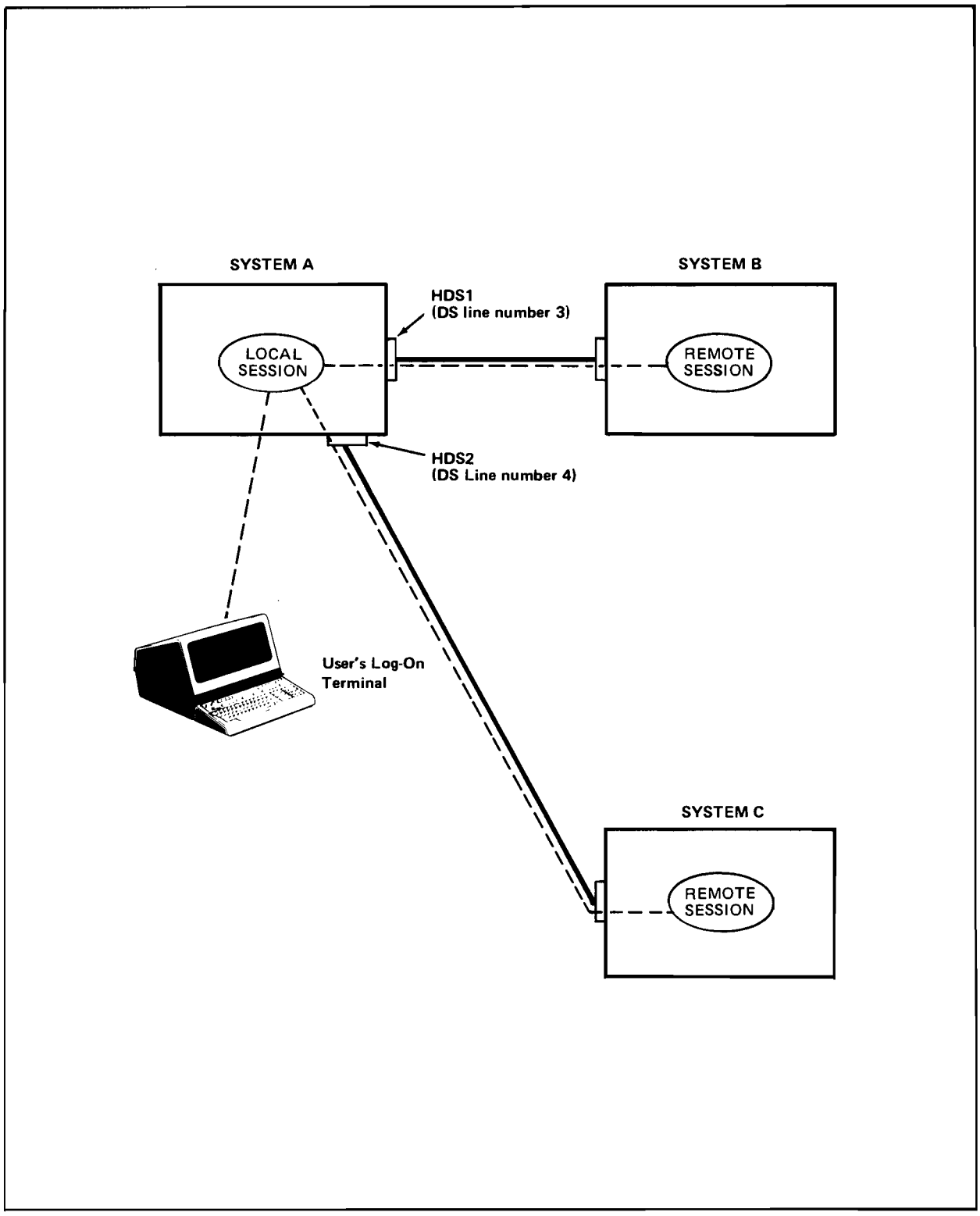


Figure 2-12. Multiple Line Example (HSI Lines)

First the user sat down at a terminal connected to System A and initiated a local session:

```
:HELLO USER.ACCOUNT

 SESSION NUMBER = #S13
 FRI, MAY 7, 1976, 1:37 PM
 HP32002A.00.a1

WELCOME TO SYSTEM A.
:
```

USER and ACCOUNT are valid user and account names, respectively, as defined by the accounting structure of System A.

At this point we have the situation illustrated in figure 2-13. You will notice that so far no communications link exists between any of the three systems.

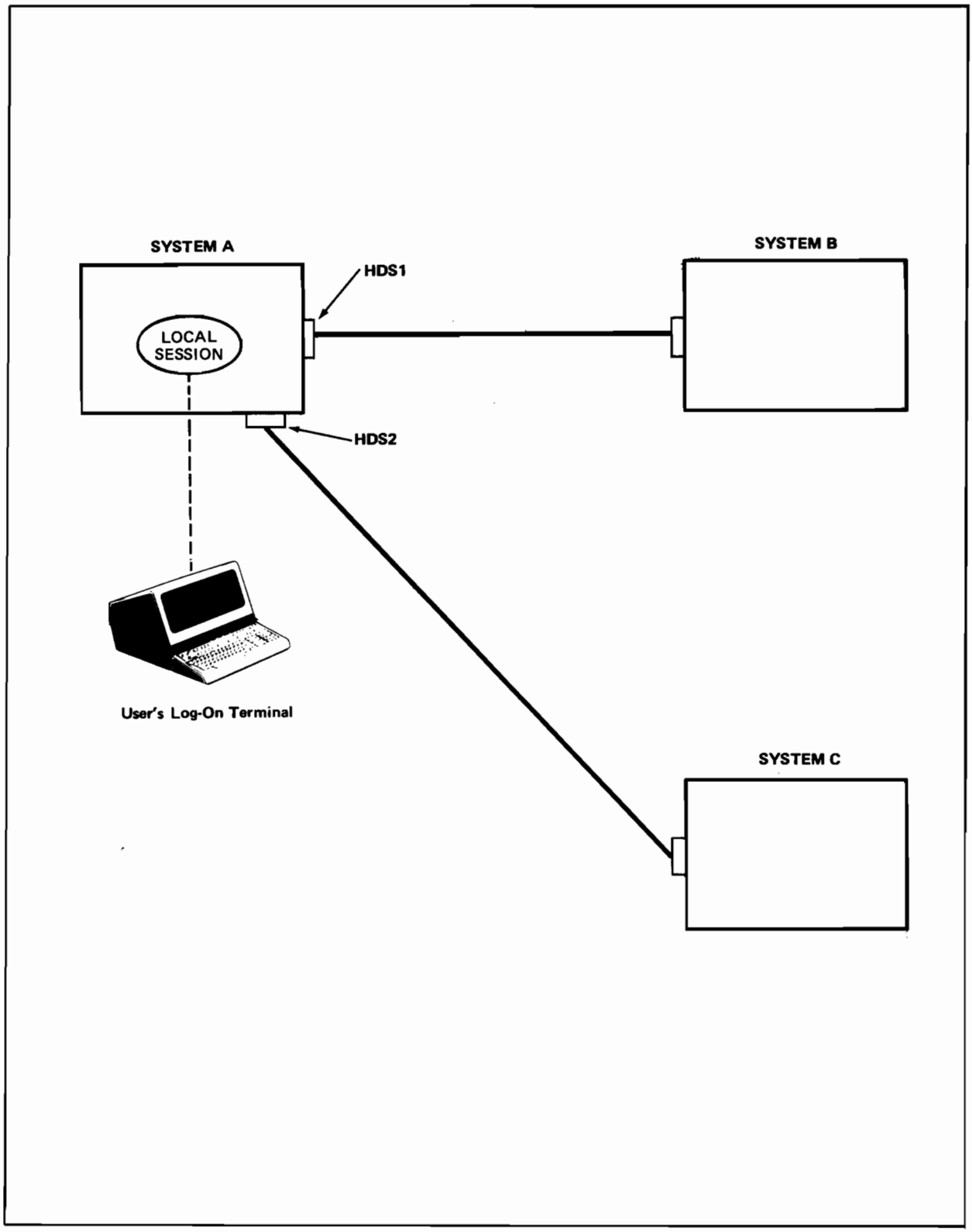


Figure 2-13. Initiating the Local Session (HSI Example)

Next the user acquired access to an HSI line between Systems A and B and initiated a remote session in System B:

```
:DSLINE HDS1

DS LINE NUMBER = #L3

:REMOTE HELLO RUSER.RACCOUNT

 SESSION NUMBER = #S35
 FRI, MAY 7, 1976, 1:38 PM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```

HDS1 is the device class name (as defined within System A) associated with the particular HSI. RUSER and RACCOUNT are valid user and account names, respectively, as defined by the accounting structure of System B.

Now we have the situation illustrated in figure 2-14.

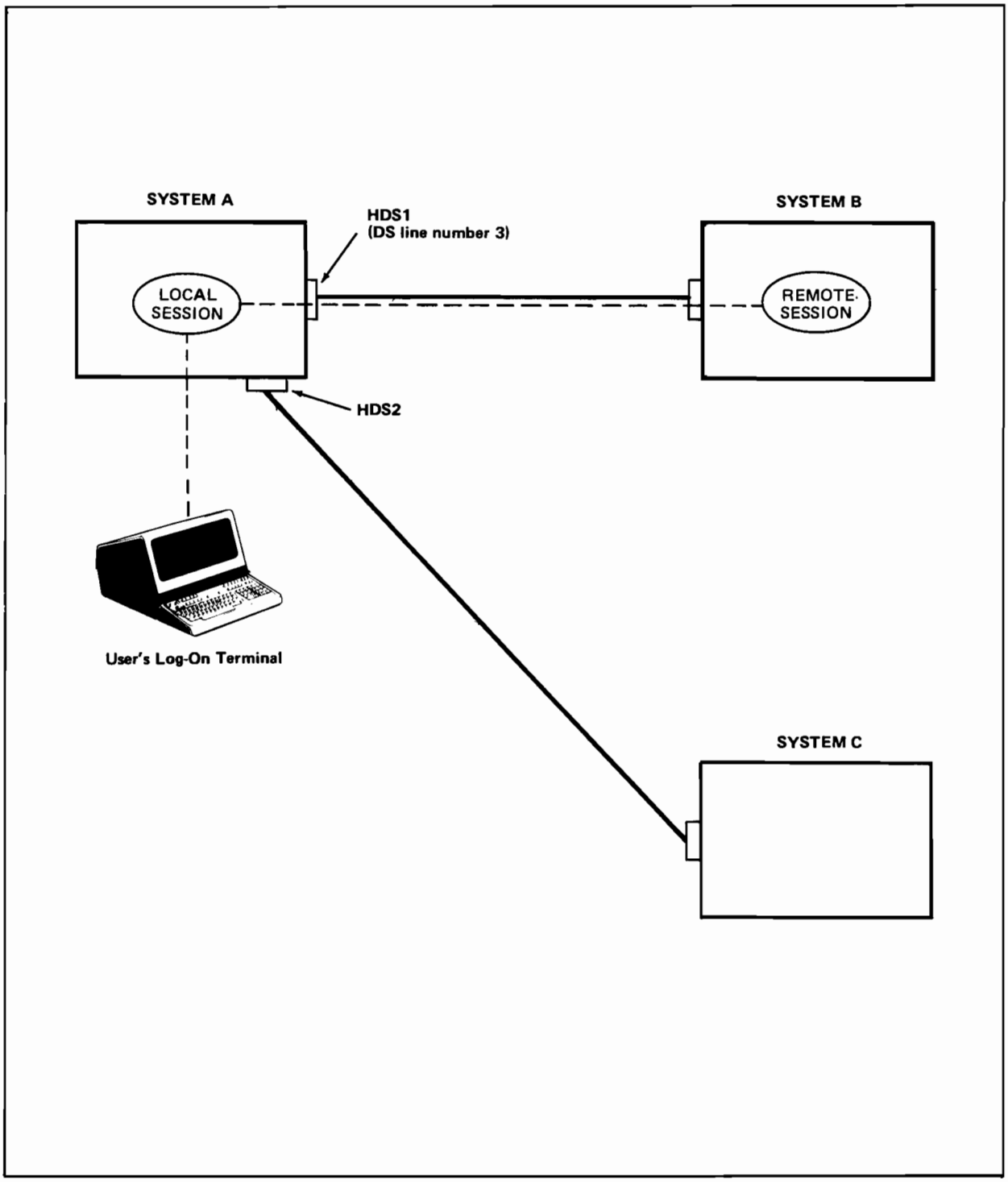


Figure 2-14. Establishing the Link With System B (HSI Example)

Finally the user acquired access to an HSI line between Systems A and C and initiated a remote session in System C:

```
:DSLINE HDS2

DS LINE NUMBER = #L4

:REMOTE HELLO SUSER.SACCOUNT

 SESSION NUMBER = #S21
 FRI, MAY 7, 1976, 1:39 PM
 HP32002A.00.A1

WELCOME TO SYSTEM C.
:
```

HDS2 is the device class name (as defined within System A) associated with the particular HSI. RUSER and RACCOUNT are valid user and account names, respectively, as defined by the accounting structure of System C.

We end up with the situation illustrated in figure 2-15, which you will notice is identical to figure 2-12 which started this example.

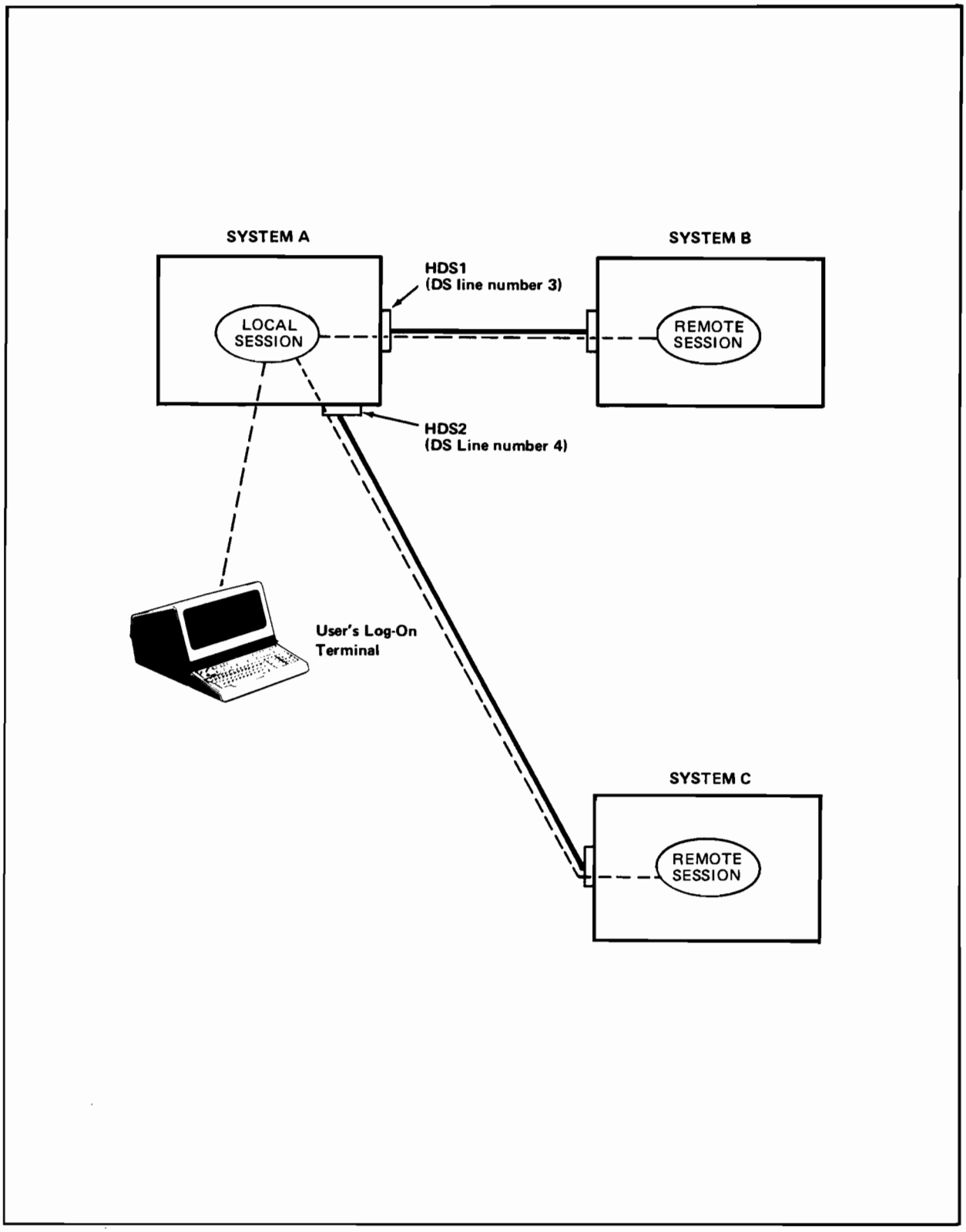


Figure 2-15. Establishing the Link With System C (HSI Example)

Line Opening Failures

There are several reasons why a DSLINE command for opening a hardwired line might be rejected by DS/3000, some of which have already been illustrated earlier in this chapter. The following list summarizes all of the likely causes of a line open failure.

- You made a syntax error in the DSLINE command.
- You gave an erroneous line specification (dsdevice) in the the DSLINE command. (There is no HSI with the specified device class name or logical device number.)
- Someone already has exclusive access to the specified line.
- You asked for EXCLUSIVE access to a line which was already in use.
- DS/3000 detected a hardware problem (the HSI board is not responding correctly).

The various error numbers and messages for line opening failures are as follows:

Error Number	Message	Meaning	Corrective Action
1	UNKNOWN COMMAND	The command entered was not recognized as a legal command. You probably mistyped the command name (DSLINE or REMOTE HELLO).	Re-enter the command and be sure that the command name is entered correctly.

Error Number	Message	Meaning	Corrective Action
20,x	SYNTAX ERROR	A delimeter in the parameter list is not permitted in the command, or is not permitted at the point where it was detected. When a qualifying number is shown (x), the bad delimeter is adjacent to the indicated parameter element (usually following it).	Re-enter the command correctly.
22,x	ILLEGAL PARAMETER	A parameter in the command was not recognized as legal. The qualifying number (x) indicates which parameter was bad.	Check the parameter list, and re-enter the command correctly.
26,x	ILLEGAL KEYWORD	A keyword in the command was not recognized as legal. The qualifying number (x) indicates which keyword was bad.	Check the keywords, and re-enter the command correctly.
27,x	DUPLICATE KEYWORD	The command was rejected because the same keyword appeared twice in the parameter list. The qualifying number (x) indicates the duplicate keyword.	Re-enter the command without the duplicate keyword.

Error Number	Message	Meaning	Corrective Action
30,x	INVALID NUMBER	The buffer-size in the LINEBUF parameter is not decimal or is out of range (304 to 4096).	Check the parameter list, and re-enter the command correctly.
60,55	DS/3000 ERROR	Someone already has exclusive access to to the line. *** OR *** You requested exclusive access to the line and someone is already using it. *** OR *** The specified HSI has been shut down by the console operator.	None. You must wait until the line is available. Either wait until the line is available or re-enter the command without the EXCLUSIVE parameter. Re-enter the command specifying a different HSI or get the console operator to bring up the particular HSI.

Error Number	Message	Meaning	Corrective Action
60,56	DS/3000 ERROR	You gave an erroneous line specification (dsdevice). There is no HSI with the specified device class name or logical device number.	Re-enter the command using a correct dsdevice parameter.
60,204	DS/3000 ERROR	You used line opening parameters (LINEBUF= or EXCLUSIVE) in a DSLINE ;CLOSE command.	
60,243	DS/3000 ERROR	The remote computer did not respond within 16 attempts to establish the line (48 seconds).	
60,244	DS/3000 ERROR	There is a software problem in DS/3000 at either end of the line.	
60,252	DS/3000 ERROR	DS/3000 detected a hardware problem with the particular HSI interface board.	
60,254	DS/3000 ERROR	The requested HSI was not configured properly during system configuration.	

OPENING A TELEPHONE LINE

A DS/3000 communications link can also be established over the public telephone network. In such a case the information passed back and forth between the two computers travels over the same lines that are used for normal voice traffic. Each computer is interfaced to the telephone lines by way of a modem. Modem is a contraction of MOdulator-DEModulator. A MODEM is a device that translates digital signals (electrical impulses) generated by a computer into analog signals (tones) that can be transmitted over telephone lines, and vice versa.

The MODEM is connected to the HP 3000 by way of the HP 30055A Synchronous Single Line Controller (SSLC). An SSLC occupies one card slot in the Multiplexer Channel. Each SSLC controls one modem (such as a Bell System Type 201 or 208 modems) and is capable of both initiating and accepting a telephone connection with a remote computer over the public telephone network or private line.

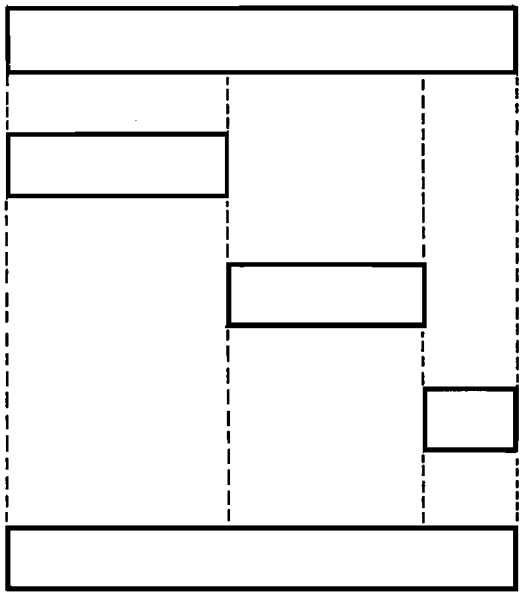
It is a little more complex to obtain access to a telephone line than to a hardwired line.

First you must identify the particular SSLC you wish to use. You do this by specifying the device class name or logical device number associated during system configuration with the desired SSLC. You can use the DSLINE command for this purpose, as follows:

```
:DSLINE SDS1
```

In the DSLINE command you may also wish to specify the size of the DS/3000 line buffer to be used in conjunction with the line. The size of this buffer determines the maximum sized block that can be sent or received over the line. Note that a transmission as you normally think of it (sending or receiving all or part of a file) may in fact consist of many physical transmissions. This buffer size in essence defines a blocking factor for the line. Refer to figure 2-16. A default buffer size is established during system configuration and in most cases (as in the example at the end of chapter 1) you will find it satisfactory to let this default value prevail.

Assume that the DS/3000 line buffer size is 512 words and that the user has initiated the transmission of a block of data 1200 words in length from his HP 3000 to a remote HP 3000. The block of data would actually be sent in three separate physical transmissions, as follows.



1200 words of data to be transmitted

First physical transmission: 512 words

Second physical transmission: 512 words

Third physical transmission: 176 words

1200 words of data received at remote computer.

Figure 2-16. DS/3000 Line Buffer Example

Next you may wish to supply a set of identification sequences to be used in verifying that the desired pair of computers are connected to one another. This is discussed under "ID Sequences" later in this chapter. Briefly, however, you may supply an ID sequence that identifies your HP 3000 and one or more ID sequences that identify those remote computers with which your HP 3000 may validly be connected. When a telephone connection is established between your HP 3000 and a remote HP 3000, the two computers exchange ID sequences and the validity of them determines whether or not the connection is to remain in effect. You use the DSLINE command to supply ID sequences, as follows:

```
:DSLINE SDS1 ;LOCID="SYSTEM A" &
             ;REMID="SYSTEM X"
```

where SYSTEM A is the ID sequence identifying your HP 3000 and SYSTEM X is the ID sequence identifying the remote computer with which you want to establish a telephone connection.

Again there are default values that can be established during system configuration. In most cases, however, you will at least want to explicitly identify the desired remote HP 3000 to be certain that the proper connection is being established.

Now you must establish the physical connection between the two computers by dialing (at the modem) the telephone number of the remote computer. If you wish to have the console operator of your HP 3000 dial the number for you, you may supply the desired number in the DSLINE command and it will be displayed as part of a dial request message at the operator's console. In such a case you would supply the telephone number as follows:

```
:DSLINE SDS1 ;LOCID="SYSTEM A" &
             ;REMID="SYSTEM X" &
             ;PHNUM=555-1234
```

The various possibilities involved in establishing a telephone connection with a remote computer are discussed under "Dialing the Remote Computer" later in this chapter.

When you execute the DSLINE command, DS/3000 attempts to give you access to the specified SSLC and, if the telephone connection is successfully established, informs you of the assigned DS line number by displaying the following message at

your terminal:

```
DS LINE NUMBER = #Lx
```

where x is the assigned DS line number. In the example at the end of chapter 1 we were assigned the DS line number "3". The DS line number is significant only if you open and use more than one communications line concurrently within a single local session (see "Opening Multiple Lines" later in this chapter).

At this point you have acquired a physical communications line but the communications link does not yet exist. The actual communications link between the two computers is established by initiating a remote session over the line. You do this by executing a REMOTE HELLO command. In the example at the end of chapter 1 we used a REMOTE HELLO command containing the minimum parameters required (a username and an accountname), as follows:

```
:REMOTE HELLO RUSER,RACCOUNT
```

The communications link between the two HP 3000 computers now exists.

Specifying a Line

In order to open a telephone line, we have seen that you must first specify a device class name or logical device number identifying the particular SSLC you wish to use. But how do you figure out which name or number to specify? The remainder of this topic will seem, particularly at first reading, a little complex and tedious. Don't let that frighten you. In actual practice, once the hardware and software configuration is installed and usable, most DS/3000 sites will post a simple notice defining all of the available communications lines and the proper device class names and logical device numbers for each. In that case all of the detective work described in the following paragraphs is already done for you.

For each SSLC there is a pair of associated drivers. First there is the actual SSLC driver that directly controls the operation of the interface board. In addition there is a DS/3000 communications driver that controls the operation of the SSLC driver. The names of these drivers are as follows:

CSSBSC0 (SSLC driver)

IODS0 (DS/3000 communications driver)

Now look at the sample I/O device table (produced during system configuration) in figure 2-17. If you look at the shaded items in the column labeled "DRIVER NAME" you will notice that we have one SSLC (CSSBSC0) configured into the system as logical device 13. In addition you will notice that there is a DS/3000 communications driver (IODS0) also configured into the system. The IODS0 entry is related to the proper SSLC entry by the number specified in the column labeled "DRT" (the # prefix indicates a back reference to a previously defined logical device number). In figure 2-17 logical device 61 is paired with logical device 13. It is the device class name or logical device number of the appropriate IODS0 entry that you use to specify the desired line.

One or more pseudo-terminal drivers (IODSTRM0) also should be configured into the system. The IODSTRM0 entry allows another system to be logged on to this system and regulates the number of remote Session Main Processes (SMP) that can be assigned to a given line. Each IODSTRM0 entry is related to the proper SSLC entry by the number specified in the column labeled "DRT". Figure 2-12 shows logical devices 64 and 65 are paired with logical device 13.

If you have only one SSLC configured into your system, there is no question about which name or number to specify. If there is more than one SSLC, however, you must know (or ask someone who knows) which CSSBSC0 and IODSTRM0 entry pertains to the particular SSLC (and modem) you wish to use.

LOG DEV	DRT #	UNIT	CHAN	TYPE	SUB TYPE	TERM TYPE	TERM SPEED	REC WIDTH	OUTPUT DEV	MODE	DRIVER NAME	DEVICE CLASSES
1	4	0	1	0	6			128	0		*IOMDISC1	DISC SPOOL
6	10	0	0	32	0			66	0	S	IOLPRT0	LP
7	6	0	0	24	0			128	0		IOTAPE0	TAPE
8	6	1	0	24	0			128	0		IOTAPE0	TAPE
9	6	2	0	24	0			128	0		IOTAPE0	TAPE
10	6	3	0	24	0			128	LP	JA	IOTAPE0	JTAPE
11	16	0	0	19	3			0	0		CSHBSC0	HSI1
12	17	0	0	19	3			0	0		CSHBSC0	HSI2
13	18	0	0	18	0			0	0		CSSBSC0	SSLC
20	7	0	0	16	0	6	??	36	20	JAID	IOTERM0	CONSOLE
21	7	1	0	16	0	11	??	36	21	JAID	IOTERM0	T2644
22	7	2	0	16	0	11	??	36	22	JAID	IOTERM0	T2644
23	7	3	0	16	0	0	??	36	23	JAID	IOTERM0	TERM
24	7	4	0	16	0	0	??	36	24	JAID	IOTERM0	TERM
25	7	5	0	16	0	0	??	36	25	JAID	IOTERM0	TERM
26	7	6	0	16	0	0	??	36	26	JAID	IOTERM0	TERM
27	7	7	0	16	0	0	??	36	27	JAID	IOTERM0	TERM
28	7	8	0	16	0	0	??	36	28	JAID	IOTERM0	TERM
29	7	9	0	16	0	0	??	36	29	JAID	IOTERM0	TERM
30	7	10	0	16	0	0	??	36	30	JAID	IOTERM0	TERM
31	7	11	0	16	0	0	??	36	31	JAID	IOTERM0	TERM
32	7	12	0	16	0	0	??	36	32	JAID	IOTERM0	TERM
33	7	13	0	16	0	0	??	36	33	JAID	IOTERM0	TERM
34	7	14	0	16	0	0	??	36	34	JAID	IOTERM0	TERM
35	7	15	0	16	0	0	??	36	35	JAID	IOTERM0	TERM
55	#11	0	0	41	0			128	0		IODS0	HDS1
56	#12	0	0	41	0			128	0		IODS0	HDS2
57	#11	0	0	16	0	??	??	36	57	J ID	IODSTRM0	DSTERM
58	#11	0	0	16	0	??	??	36	58	J ID	IODSTRM0	DSTERM
59	#12	0	0	16	0	??	??	36	59	J ID	IODSTRM0	DSTERM
61	#13	0	0	41	0		??	128	0		IOD50	SDS1
64	#13	0	0	16	0	??	??	36	64	J ID	IODSTRM0	DSTERM
65	#13	0	0	16	0	??	??	36	65	J ID	IODSTRM0	DSTERm

Figure 2-17. Sample I/O Device Table

The DSLINE Command

The format of the DSLINE command is presented in figure 2-18 below. In addition to opening a hardwired or telephone line, this command can be used for closing one or more communications lines. For completeness all of the available parameters are shown. The shaded parameters apply to opening a telephone line; the others (non-shaded) apply to closing one or more communications lines and will be described when that topic is discussed later in this chapter.

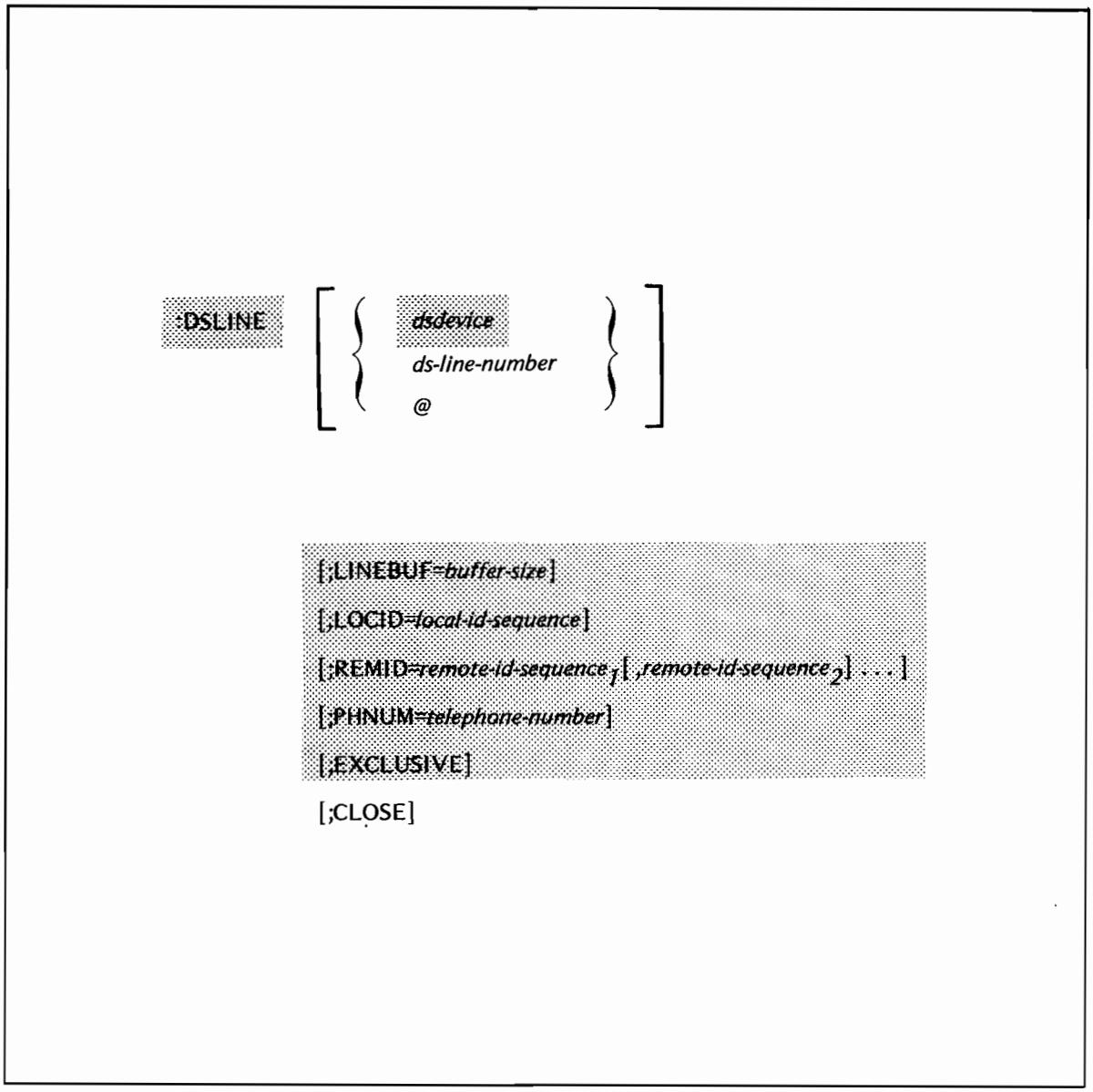


Figure 2-18. The DSLINE Command

The paramaters that pertain to opening a telephone line are as follows:

dsdevice	The device class name or logical device number assigned to the DS/3000 communications driver IODS0 during system configuration. This parameter specifies what SSLC (and MODEM) you wish to use. (Required parameter.)
buffer-size	A decimal integer specifying the size (in words) of the DS/3000 line buffer to be used in conjunction with the communications line. The integer must be within the range 304 < buffer-size < 4096. The default value is the buffer size entered in response to the PREFERRED BUFFER SIZE prompt during system configuration. (Optional parameter.)
telephone-number	A telephone number consisting of digits and dashes. The maximum length permitted (including both digits and dashes) is 20 characters. Provided that YES was entered in response to the DIAL FACILITY prompt during system configuration, this telephone number will be displayed at the operator's console of your HP 3000 and the operator will then establish the telephone connection by dialing that number at the MODEM. The default telephone number is the first one entered in response to the PHONE NUMBER prompt during system configuration. (Optional parameter.)

local-id-sequence

A string of ASCII characters contained within quotation marks or a string of octal numbers separated by commas and contained within parentheses. If you wish to use a quotation mark within an ASCII string, use two successive quotation marks. In the case of an octal sequence, each octal number represents one byte and must be within the range 0-377. The maximum number of ASCII characters or octal numbers allowed in the string is 16.

The supplied string of ASCII characters or octal numbers defines the ID sequence that will be sent from your HP 3000 to the remote HP 3000 when you attempt to establish the telephone connection. If the remote HP 3000 does not recognize the supplied ID sequence as a valid one, the telephone connection is terminated. The default value is the ASCII or octal string entered in response to the LOCAL ID SEQUENCE prompt during system configuration.

(Optional parameter.)

remote-id-sequence

Same format as local-id-sequence.

The supplied strings of ASCII characters or octal numbers define those remote HP 3000 ID sequences that will be considered valid when you attempt to establish the telephone connection. If the remote HP 3000 does not send a valid ID sequence, the telephone connection is terminated. The default set of remote ID sequences consists of the ASCII and octal strings entered in response to the REMOTE ID SEQUENCE prompt during system configuration.

(Optional parameter.)

EXCLUSIVE	This parameter, if present, specifies that you want exclusive use of the particular communications line. If the specified SSLC is already open and you have specified the exclusive option, DS/3000 will deny you access to the line (you cannot open it). Opening an EXCLUSIVE line requires the user to have CS capability. (Optional parameter.)

Dialing the Remote Computer

In the DSLINE command you may supply a telephone number to be dialed at the modem connected to the specified SSLC. If you supply a telephone number, DS/3000 displays a message on the system console telling the operator to dial that number. The operator, after dialing the specified number, enters YES or NO through the system console =REPLY command to let DS/3000 know whether or not the telephone connection was successfully made. If the operator enters YES, DS/3000 proceeds with the exchanging of ID sequences. If the operator enters NO, your DSLINE request is denied (you cannot open the line).

If you do not supply a telephone number, the sequence of events is as described in the above paragraph except that DS/3000 uses by default the first telephone number in the PHONELIST established during system configuration.

If you do not supply a telephone number and no PHONELIST was established during system configuration, DS/3000 merely waits (up to 15 minutes) until the SSLC driver senses a signal from the modem indicating that a telephone connection exists. No message is displayed at the system console and no operator response is expected. If 15 minutes elapses without a telephone connection being made, the DSLINE command is rejected. This method might be used when you will dial the remote HP 3000 yourself. It is recommended, however, that you always supply a phone number in the DSLINE command so that you can let DS/3000 know immediately (by entering NO through the system console) if no telephone connection can be made.

ID Sequences

Once a telephone connection exists between the specified SSLC and a remote HP 3000, the two computers exchange ID sequences with one another. As you may remember from the description of the DSLINE command above, within the context of DS/3000 an ID sequence is a string of up to 16 ASCII characters or octal numbers that identifies a particular computer.

During system configuration each HP 3000 can be assigned a local ID sequence and a list of remote ID sequences. The local ID sequence identifies the particular HP 3000 in which it is established; the remote ID sequences identify those remote computers with which a communications link can be established over the public telephone network.

In the DSLINE command you may supply a local ID sequence and one or more remote ID sequences to be used instead of those established during system configuration.

When a telephone connection is established between your HP 3000 and a remote HP 3000, the local ID sequence supplied in your DSLINE command is transmitted to the remote system. The remote system compares that ID against its list of remote ID sequences. If that ID sequence is found to be valid, the remote system transmits its local ID sequence over the telephone line to your HP 3000. The received ID sequence is then compared against the remote ID sequence(s) supplied in your DSLINE command. If that ID sequence is found to be valid, the telephone connection is considered successful and DS/3000 grants you access to the line. If the ID sequence received at either end of the line is not considered valid, your DSLINE request is denied (you cannot open the line).

If you do not supply any ID sequences, DS/3000 uses those established during system configuration. If no ID sequences were established during system configuration and you do not supply any, no local ID sequence is transmitted from your HP 3000 to the remote system and any remote ID sequence received is considered valid.

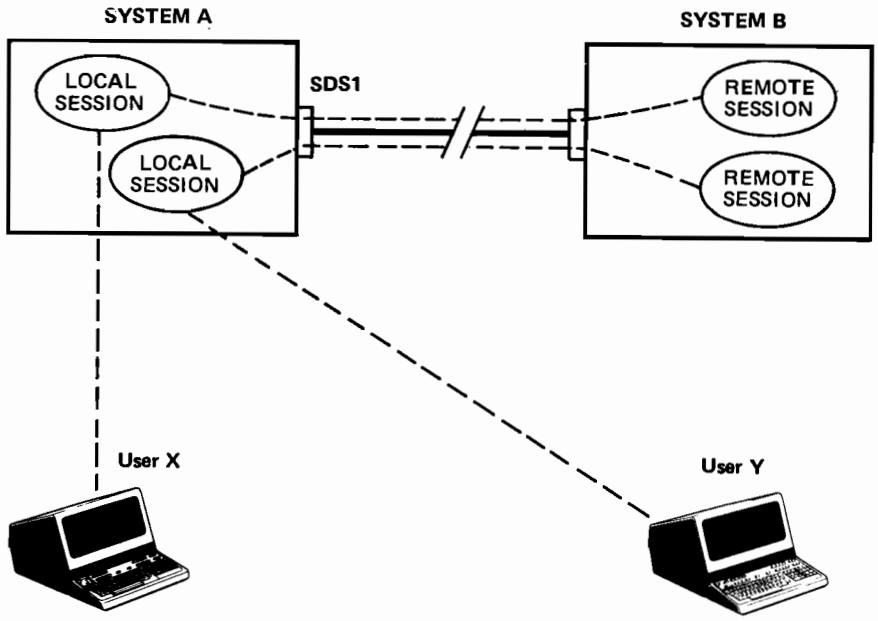
Multiple Users

Within a DS/3000 environment, it is possible for several users at either end of the line to share access to the same physical communications line or for a single user at one end of the line to obtain exclusive access to the line.

As you probably remember from the presentation of the DSLINE command above, the EXCLUSIVE parameter can be used to obtain exclusive access to the specified physical communications line. If you specify this parameter (and if access to the line is granted), no other user in either computer will be permitted to open that line until you close it. If you ask for exclusive access to a particular line and that line is already in use, DS/3000 denies your request (you cannot open the line).

For a telephone line, multiple users at either end of the line can specify the same physical line in DSLINE commands and obtain access to that line as long as none of them requests exclusive access and all users other than the one who originally opened the line specify (explicitly or by default) the currently active remote ID sequence. Figures 2-19 through 2-28 present annotated examples illustrating successful and unsuccessful attempts by different users to obtain access to the same telephone line.

```
Configured Local ID:  A          Configured Local ID:  B
Configured Remote IDs:  B,C      Configured Remote IDs:  A,C
```




```
:HELLO USER.X                    :HELLO USER.Y
:DSLINE SDS1 &                   :DSLINE SDS1 &
        ;PHNUM=555-1234 &                ;PHNUM=555-1234 &
        ;REMID="B"                       ;REMID="B"
:REMOTE HELLO USER.X             :REMOTE HELLO USER.Y
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the modem connected to the

(continued)

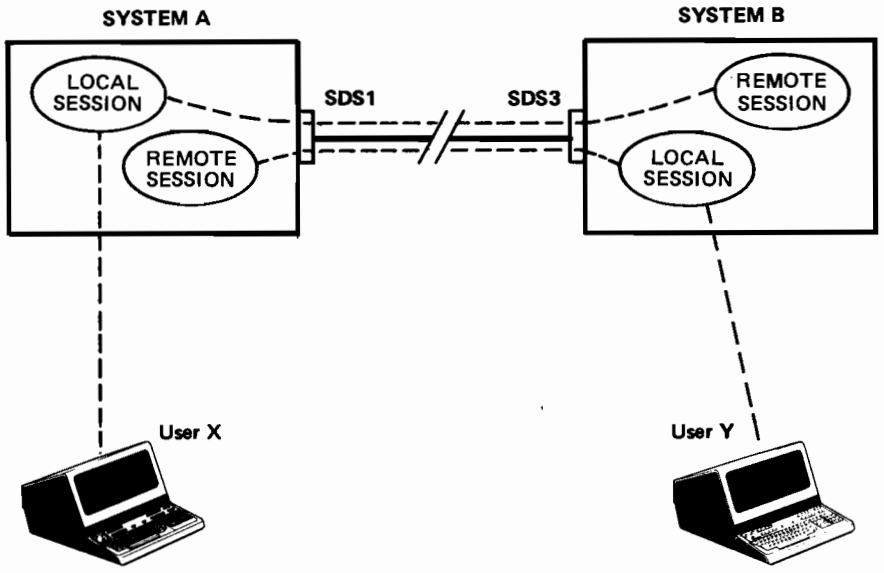
Figure 2-19. SSLC Multiple User Example

particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System A and requests access to the same SSLC (SDS1). Since that SSLC is already open, DS/3000 ignores the supplied telephone number (no message is displayed at the system console). Access to the currently opened line is granted to User Y because neither user requested exclusive access and User Y specified the currently active remote ID sequence (REMID="B") in his DSLINE command.

Figure 2-19. SSLC Multiple User Example (Continued)

```
Configured Local ID:  A          Configured Local ID:  B
Configured Remote IDs:  B,C      Configured Remote IDs:  A,C
```




```
:HELLO USER.X                    :HELLO USER.Y
:DSLINE SDS1 &                   :DSLINE SDS3 &
        ;PHNUM=555-1234 &                ;PHNUM=777-4321 &
        ;REMID="B"                       ;REMID="A"
:REMOTE HELLO USER.X             :REMOTE HELLO USER.Y
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the modem connected to the

(continued)

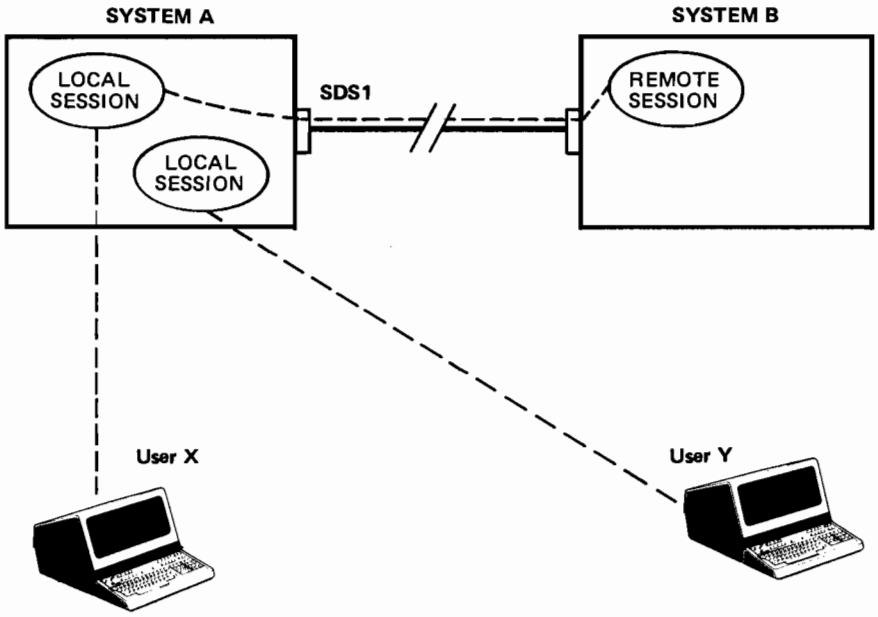
Figure 2-20. SSLC Multiple User Example

particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System B and requests access to the SSLC identified by the device class name SDS3. Since that SSLC is already open, DS/3000 ignores the supplied telephone number (no message is displayed at the system console). Access to the currently opened line is granted to User Y because neither user requested exclusive access and User Y specified the currently active remote ID sequence (REMID="A") in his DSLINE command.

Figure 2-20. SSLC Multiple User Example (Continued)

```
Configured Local ID:  A          Configured Local ID:  B
Configured Remote IDs:  B,C      Configured Remote IDs:  A,C
```




```
:HELLO USER.X                    :HELLO USER.Y
:DSLINE SDS1 &                   :DSLINE SDS1 &
        ;PHNUM=555-1234 &                ;PHNUM=555-1234 &
        ;REMID="B" &                     ;REMID="B"
        ;EXCLUSIVE
:REMOTE HELLO USER.X
```

In this example User X initiates a local session in System A and obtains exclusive access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone

(continued)

Figure 2-21. SSLC Multiple User Example

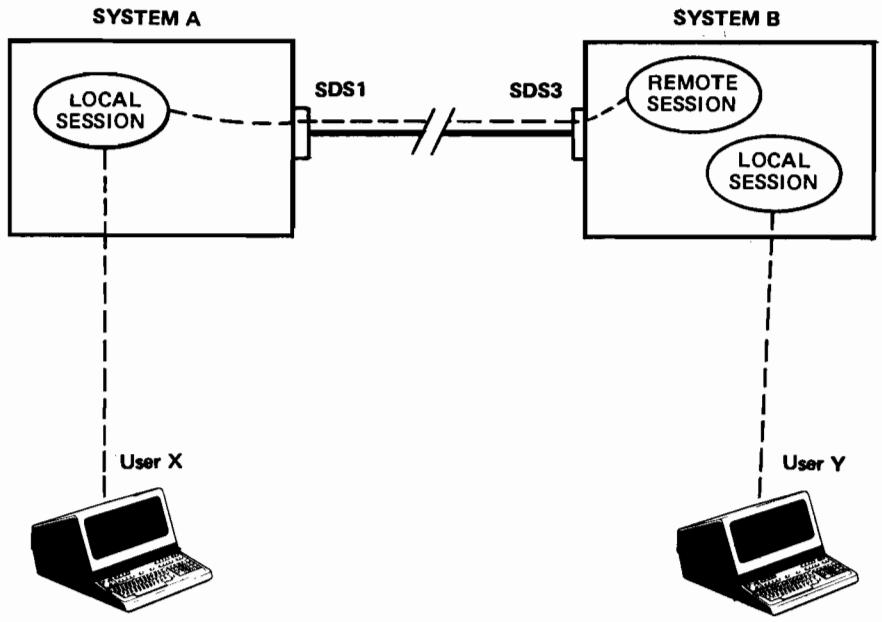
connection by dialing the number at the modem connected to the particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System A and requests access to the same SSLC (SDS1). The request is denied by DS/3000 because the specified SSLC is already open and User X has exclusive access to it.

Figure 2-21. SSLC Multiple User Example (Continued)

Configured Local ID: A
Configured Remote IDs: B,C

Configured Local ID: B
Configured Remote IDs: A,C




```
:HELLO USER.X
:DSLINE SDS1 &
        ;PHNUM=555-1234 &
        ;REMID="B" &
        ;EXCLUSIVE
:REMOTE HELLO USER.X
```

```
:HELLO USER.Y
:DSLINE SDS3 &
        ;PHNUM=777-4321 &
        ;REMID="A"
```

In this example User X initiates a local session in System A and obtains exclusive access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone

(continued)

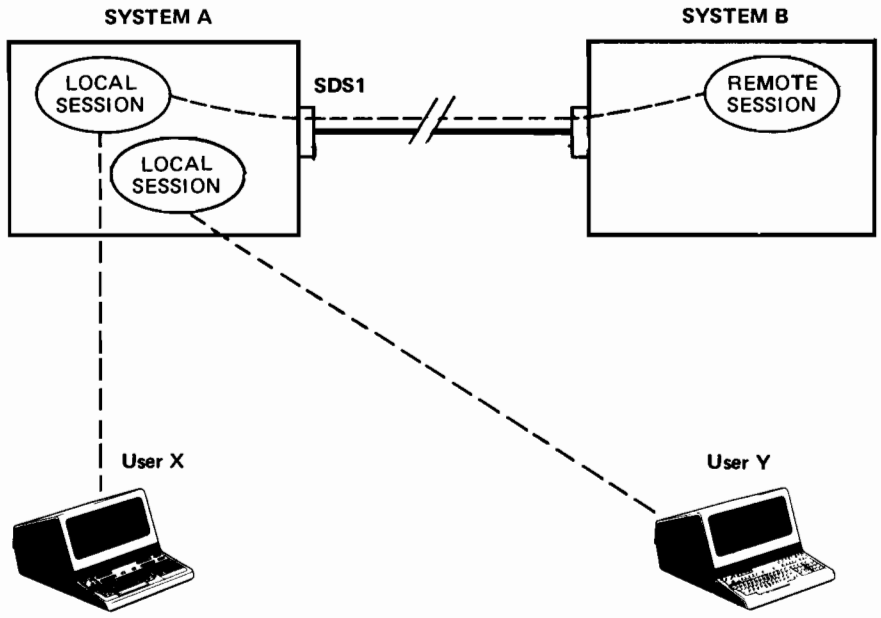
Figure 2-22. SSLC Multiple User Example

connection by dialing the number at the modem connected to the particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System B and requests access to the SSLC identified by the device class name SDS3. The request is denied by DS/3000 because the specified SSLC is already open and User X has exclusive access to it.

Figure 2-22. SSLC Multiple User Example (Continued)

```
Configured Local ID:  A                Configured Local ID:  B
Configured Remote IDs:  B,C            Configured Remote IDs:  A,C
```




```
:HELLO USER.X                          :HELLO USER.Y
:DSLINE SDS1 &                         :DSLINE SDS1 &
        ;PHNUM=555-1234 &                      ;PHNUM=555-1234 &
        ;REMID="B"                             ;REMID="B" &
:REMOTE HELLO USER.X                           ;EXCLUSIVE
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the MODEM connected to the

(continued)

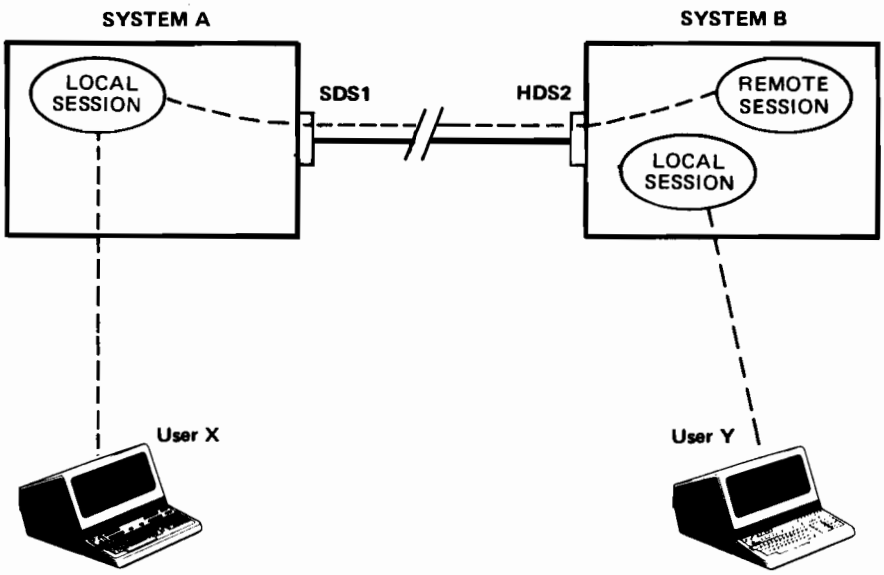
Figure 2-23. SSLC Multiple User Example

particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System A and requests exclusive access to the same SSLC (SDS1). The request is denied by DS/3000 because the specified SSLC is already open and therefore cannot be assigned exclusively to User Y.

Figure 2-23. SSLC Multiple User Example (Continued)

```
Configured Local ID:  A          Configured Local ID:  B
Configured Remote IDs:  B,C      Configured Remote IDs:  A,C
```




```
:HELLO USER.X                    :HELLO USER,Y
:DSLINE SDS1 &                   :DSLINE SDS3 &
        ;PHNUM=555-1234 &                ;PHNUM=777-4321 &
        ;REMID="B"                       ;REMID="A" &
:REMOTE HELLO USER.X             ;EXCLUSIVE
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the modem connected to the

(continued)

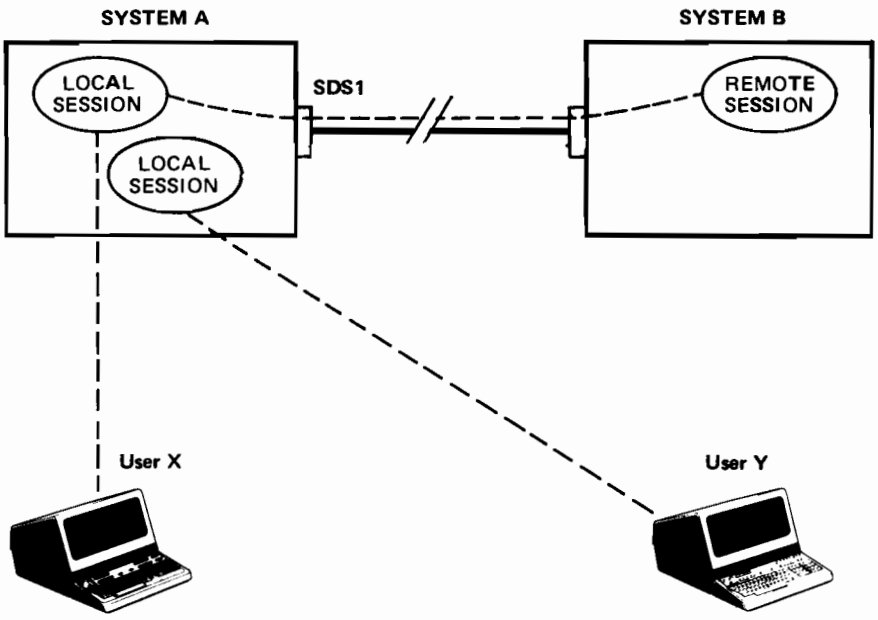
Figure 2-24. SSLC Multiple User Example

particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System B and requests exclusive access to the SSLC identified by the device class name SDS3. The request is denied by DS/3000 because the specified SSLC is already open and therefore cannot be assigned exclusively to User Y.

Figure 2-24. SSLC Multiple User Example (Continued)

```
Configured Local ID:  A          Configured Local ID:  B
Configured Remote IDs:  B,C      Configured Remote IDs:  A,C
```




```
:HELLO USER.X                    :HELLO USER.Y
:DSLINE SDS1 &                   :DSLINE SDS1 &
        ;PHNUM=555-1234 &                ;PHNUM=555-2001 &
        ;REMID="B"                       ;REMID="C"
:REMOTE HELLO USER.X
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the modem connected to the

(continued)

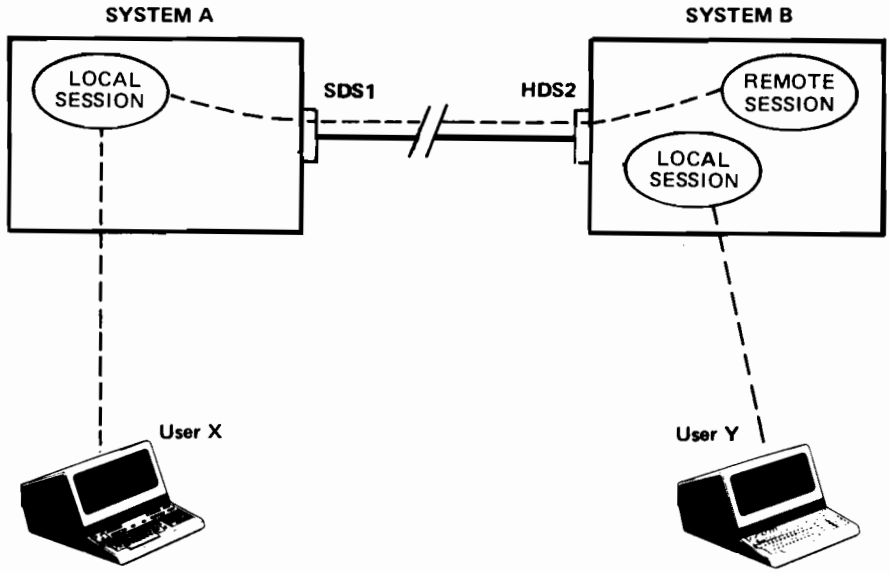
Figure 2-25. SSLC Multiple User Example

particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System A and requests access to the same SSLC (SDS1). The request is denied by DS/3000 because the specified SSLC is already open and User Y did not specify the currently active remote ID sequence (B) in his DSLINE command.

Figure 2-25. SSLC Multiple User Example (Continued)

```
Configured Local ID:  A          Configured Local ID:  B
Configured Remote IDs:  B,C      Configured Remote IDs:  A,C
```




```
:HELLO USER.X                        :HELLO USER.Y
:DSLINE SDS1 &                       :DSLINE SDS3 &
        ;PHNUM=555-1234 &                    ;PHNUM=555-2001 &
        ;REMID="B"                           ;REMID="C"
:REMOTE HELLO USER.X
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the modem connected to the

(continued)

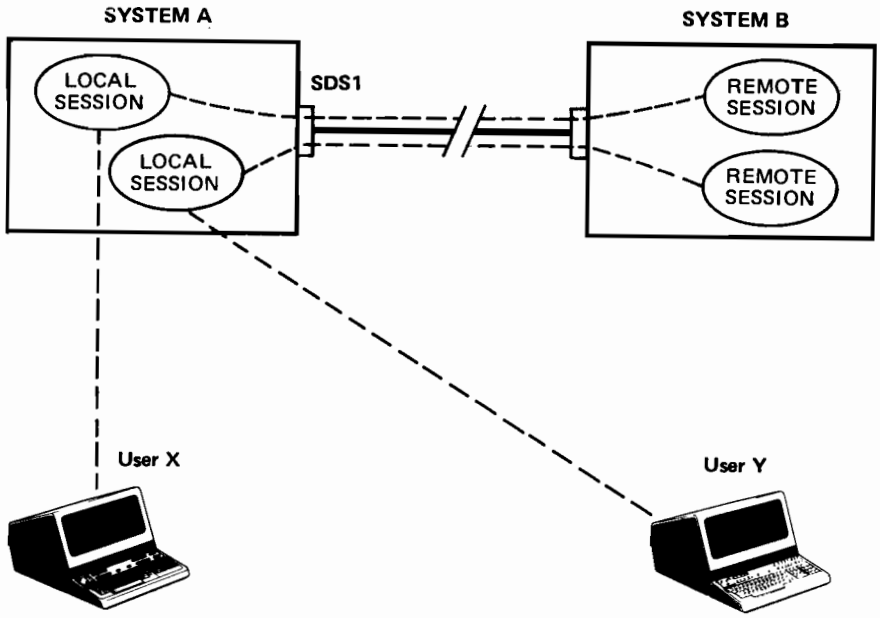
Figure 2-26. SSLC Multiple User Example

particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection was successfully made. The two computers exchange their configured local ID sequences. System A compares the received ID sequence (B) against the remote ID sequence specified by User X (REMID="B") and System B compares the received ID sequence (A) against its list of configured remote ID sequences (A,C). Since the received ID sequences are found to be valid at both ends of the line, the telephone connection is allowed to remain in effect. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System B and requests access to the SSLC identified by the device class name SDS3. The request is denied by DS/3000 because the specified SSLC is already open and User Y did not specify the currently active remote ID sequence (A) in his DSLINE command.

Figure 2-26. SSLC Multiple User Example (Continued)

```
Configured Local ID: (none)          Configured Local ID: (none)
Configured Remote IDs: (none)        Configured Remote IDs: (none)
```




```
:HELLO USER.X                        :HELLO USER.Y
:DSLINE SDS1 ;PHNUM=555-1234         :DSLINE SDS1 ;PHNUM=555-1234
:REMOTE HELLO USER.X                 :REMOTE HELLO USER.Y
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the modem connected to the particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection

(continued)

Figure 2-27. SSLC Multiple User Example

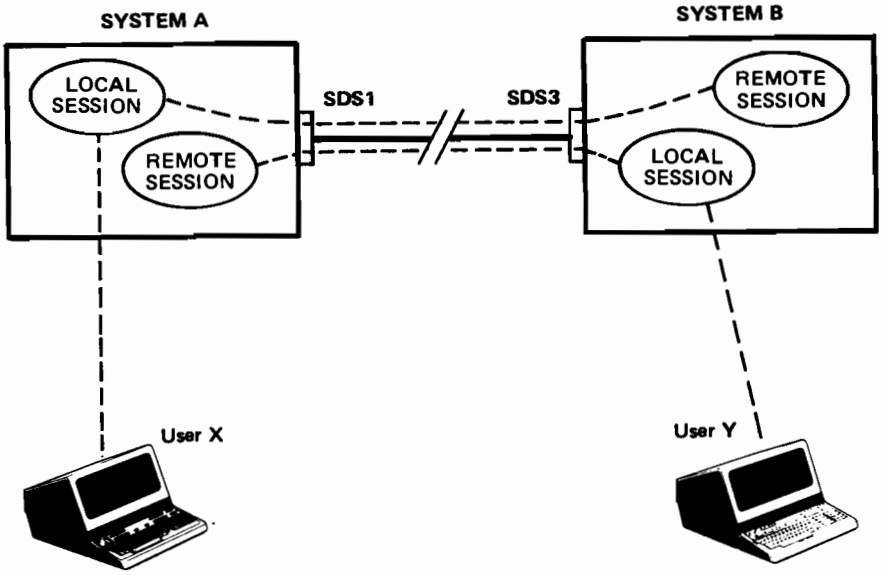
was successfully made. No ID sequences are exchanged because none were established (in either HP 3000) during system configuration and User X didn't specify any in his DSLINE command. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System A and requests access to the same SSLC (SDS1). Since that SSLC is already open, DS/3000 ignores the supplied telephone number (no message is displayed at the system console). Access to the currently opened line is granted to User Y because neither user requested exclusive access and User Y specified the currently active remote ID sequence (in this case none) in his DSLINE command.

Note that when no ID sequences are configured and the users don't supply any in their DSLINE commands, both are taking it on faith that they are connected to the proper remote computer. The total absence of configured or supplied ID sequences is safe only under very controlled circumstances. It is strongly recommended that all computers in a DS/3000 network that are capable of communicating over telephone lines have default local and remote ID sequences established during system configuration and that all SSLC users specify the ID sequence of the desired remote computer (REMID=x) in their DSLINE commands.

Figure 2-27. SSLC Multiple User Example (Continued)

```
Configured Local ID: (none)          Configured Local ID: (none)
Configured Remote IDs: (none)        Configured Remote IDs: (none)
```




```
:HELLO USER.X                        :HELLO USER.Y
:DSLINE SDS1 ;PHNUM=555-1234         :DSLINE SDS3 ;PHNUM=777-4321
:REMOTE HELLO USER.X                 :REMOTE HELLO USER.Y
```

In this example User X initiates a local session in System A and obtains access to the SSLC identified by the device class name SDS1 (he is granted access to it because at the time no one else was using that SSLC). The supplied telephone number is displayed at the system console of System A. The console operator establishes the telephone connection by dialing the number at the modem connected to the particular SSLC and then enters "YES" through the system console to let DS/3000 know that the telephone connection

(continued)

Figure 2-28. SSLC Multiple User Example

was successfully made. No ID sequences are exchanged because none were established (in either HP 3000) during system configuration and User X didn't specify any in his DSLINE command. User X then initiates a remote session in System B over the telephone line from his local log-on terminal.

User Y subsequently initiates a local session in System B and requests access to the SSLC identified by the device class name SDS3. Since that SSLC is already open, DS/3000 ignores the supplied telephone number (no message is displayed at the system console). Access to the currently opened line is granted to User Y because neither user requested exclusive access and User Y specified the currently active remote ID sequence (in this case none) in his DSLINE command.

Note that when no ID sequences are configured and the users don't supply any in their DSLINE commands, both are taking it on faith that they are connected to the proper remote computer. The total absence of configured or supplied ID sequences is safe only under very controlled circumstances. It is strongly recommended that all computers in a DS/3000 network that are capable of communicating over telephone lines have default local and remote ID sequences established during system configuration and that all SSLC users specify the ID sequence of the desired remote computer (REMID=x) in their DSLINE commands.

Figure 2-28. SSLC Multiple User Example (Continued)

The REMOTE HELLO Command

Once you have obtained access to a physical communications line using the DSLINE command, you use the REMOTE HELLO command to actually establish the communications link. The REMOTE HELLO command initiates a remote session on your behalf in the HP 3000 connected to the other end of the communications line.

The format of the REMOTE HELLO command is presented in figure 2-29 below. You will notice that, except for the three shaded items, it has exactly the same format as the standard MPE HELLO command.

Because the REMOTE HELLO command is initiating a session for you in a remote HP 3000, the parameters in that command specify information which pertains to the operating environment of the remote HP 3000 (not your local one). More specifically you must keep the following in mind:

- sessionname (if present) identifies the remote session and has no relationship to your local session.
- username, accountname, groupname, and their passwords (if any) must all be valid as defined by the accounting structure of the remote HP 3000.
- cpusecs (if present) refers to central-processor time in the remote system.
- BS, CS, DS, ES, inputpriority, and HIPRI (if present) all specify priorities for the remote session within the remote system.
- termtype (if present) has no meaning and is ignored because output from the remote session is directed to the communications line instead of to a terminal. The termtype parameter for your local session implicitly defines your log-on terminal type for any remote sessions that you initiate.

```
:REMOTE HELLO     [sessionname,]username[/userpasw].acctname[/acctpasw]
                  [,groupname[/grouppasw]]

                  [;TERM=termtype]

                  [;TIME=cpusces]

                          BS
                          CS
                  [;PRI=          ]
                          DS
                          ES

                    ;INPRI = inputpriority
                    ;HIPRI

                  [;DSLINE=dsdevice]
```

sessionname — Arbitrary name used in conjunction with *username* and *acctname* parameters to form a fully-qualified session identity. Contains from 1 to 8 alphanumeric characters, beginning with a letter. Default: null session name. (Optional parameter.)

NOTE

A *fully-qualified session identity* consists of:

[*sessionname*,]*username.acctname*

and furnishes the minimum information required for log-on. Embedded blanks are forbidden in *username.acctname* combination.

username — A user name, established by Account Manager, that allows you to log-on under this account. This name is unique within the account. Contains from 1 to 8 alphanumeric characters, beginning with letter. (Required parameter.)

Figure 2-29. The REMOTE HELLO Command

userpasw	Your user password, optionally assigned by Account Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter. Separated from *username* by slash with no surrounding blanks, as in *username/userpasw*. (Required if assigned.)
acctname	Name of your account, as established by System Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter NOTE Must be preceded by period as a delimiter. (Required parameter.)
acctpasw	Account's password, optionally assigned by System Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter. Separated from *acctname* by slash with no surrounding blanks, as in *acctname/acctpasw*. (Required if assigned.)
groupname	Name of file group to be used for local file domain and central-processor time charges, as established by Account Manager. Contains from 1 to 8 alphanumeric characters, beginning with letter. Default: Your home group if you are assigned one by Account Manager. (*Optional* if you have a home group; *Required* if you do not.)
grouppasw	Group's password, optionally assigned by Account Manager. Contains from 1 to 8 alphanumeric characters, beginning with a letter. Separated from *groupname* by slash with no surrounding blanks, as in *groupname/grouppasw*. (Not needed when you log-on under home group. Otherwise, required if assigned.)
termtype	Ignored. The TERM=termtype parameter of the HELLO command that initiated the local session also implicitly defines the log-on terminal type for any remote sessions initiated from the local session.

Figure 2-29. The REMOTE HELLO Command (Continued)

cpusecs

Maximum central-processor time that your session can use, entered in seconds. When this limit is reached, session is aborted. Must be value from 1 to 32767. To specify no limit, enter question mark or omit this parameter.

Default: no limit. (Optional parameter.)

BS
CS
DS
ES

The execution priority class that the Command Interpeter uses for your session, and also the default priority for all programs executed within the session. BS is highest priority; ES is lowest. If you specify a priority that exceeds the highest permitted for your account or user name by the system, MPE assigns the highest priority possible below BS. Default: CS.

NOTE

DS and ES are intended primarily for batch jobs; their use for sessions is generally discouraged.

(Optional parameter.)

inputpriority

Relative input priority used in checking against access restrictions imposed by the *job fence*, if one exists. Takes effect at log-on time. Must be a value from 1 (lowest priority) to 13 (highest priority). If you supply a value less than or equal to current job fence set by Console Operator, session is denied access.

Default: 8 if logging of session/job initiation is enabled, 13 otherwise. (Optional parameter.)

HIPRI

Request for maximum session-selection input priority, causing session to be scheduled regardless of current job fence or execution limit for sessions.

NOTE

You can specify this parameter only if you have System Manager or Supervisor Capability.

(Optional parameter.)

Figure 2-29. The REMOTE HELLO Command (Continued)

dsdevice	The device class name or logical device number assigned to the DS/3000 communications driver IODS0 during system configuration. This parameter, if present, specifies which hardwired line (HSI) you wish to use. (Optional parameter if a line is already open; otherwise it is required.)

Figure 2-29. The REMOTE HELLO Command (Continued)

So far we have been talking entirely about the DSLINE and REMOTE HELLO commands being used in conjunction with one another: the DSLINE command obtaining access to a physical line and the REMOTE HELLO command actually establishing the communications link by initiating a remote session over the acquired line. As you may have guessed from the above parameter definitions, the DSLINE parameter makes it possible for you to accomplish both of these tasks using just the REMOTE HELLO command.

To illustrate this, let's look at an example that uses the DSLINE command to obtain access to a telephone line (by way of the SSLC whose device class name is SDS1) and the REMOTE HELLO command to initiate a remote session over the line:

```
:DSLINE SDS1

DS LINE NUMBER = #L3

:REMOTE HELLO RUSER.RACCOUNT

 SESSION NUMBER = #S11
 FRI, MAY 9, 1976, 9:08 AM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```

< Note >

In this case the acquired line remains open when the remote session is terminated.

By including the DSLINE parameter in the REMOTE HELLO command we could perform essentially the same operations using a single command, as follows:

```
:REMOTE HELLO RUSER.RACCOUNT;DSLINE=SDS1

DS LINE NUMBER = #L3

 SESSION NUMBER = #S11
 FRI, MAY 9, 1976, 9:08 AM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```

< Note >

In this case the acquired line is closed when the remote session is terminated.

The above example will work properly for you under very limited circumstances, namely the following:

- You are satisfied to use the default DS/3000 line buffer size established during system configuration.
- The default ID sequences established in both computers during system configuration properly identify both your local HP 3000 and the desired remote HP 3000 (or no ID sequences were established during system configuration in either computer).
- You dial the remote computer yourself at the proper modem. Note that if you cannot successfully make the telephone connection you cannot abort the REMOTE HELLO command; the command will be rejected by DS/3000 if no connection is established within 15 minutes.

The likelihood of all of the above conditions existing for a particular use of DS/3000 is rather slim. In most DS/3000 application environments you will always want to explicitly define the ID sequence of the desired remote computer to guarantee that the proper connection is established and you will want to provide a phone number so that you can let DS/3000 know immediately if a telephone connection cannot be made (i.e., it will not be acceptable to tie up an SSLC and your log-on terminal for 15 minutes waiting for an unsuccessful DSLINE or REMOTE HELLO request to be rejected).

Opening Multiple Lines

Within your local session you may open more than one physical communications line and you may have remote sessions active concurrently over all of the opened lines. You are limited, however, to one remote session per physical line at any given time.

If it is able to obtain access to the specified line, DS/3000 responds to each DSLINE command by displaying a DS line number at your log-on terminal. This line number is roughly analagous to the file number returned by the MPE FOPEN intrinsic in that it is an arbitrary number that uniquely identifies (within your local session) your current access to a particular communications line. It has no relationship to the logical device number or any other configuration parameter associated with the line. DS line numbers are meaningful only if you have more than one line open concurrently within a single local session. In that case you are assigned a separate DS line number for each line you have opened and you subsequently use these numbers to specify which line you wish to use for a given remote command (or sequence of remote commands) or to close a particular line without closing the others.

Figure 2-30 illustrates a situation where a user has established two communications links concurrently from within a single local session. Let's take a closer look at that situation and examine the sequence of commands that was used to create it.

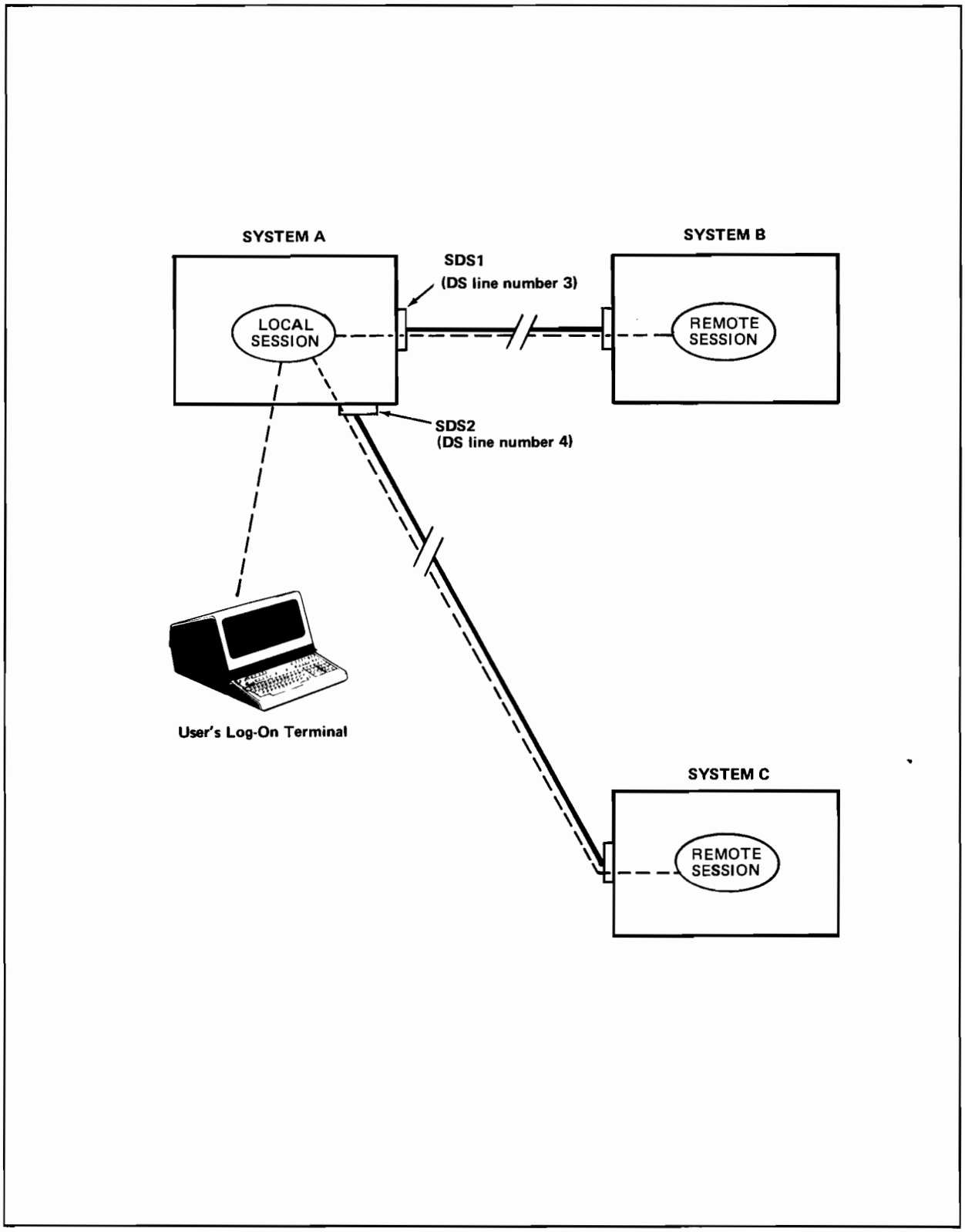


Figure 2-30. Multiple Line Example (SSLC Lines)

First the user sat down at a terminal connected to System A and initiated a local session:

```
:HELLO USER.ACCOUNT

 SESSION NUMBER = #S13
 FRI, MAY 7, 1976, 1:37 PM
 HP32002A.00.a1

WELCOME TO SYSTEM A.
:
```

USER and ACCOUNT are valid user and account names, respectively, as defined by the accounting structure of System A.

At this point we have the situation illustrated in figure 2-31. You will notice that so far no communications link exists between any of the three systems.

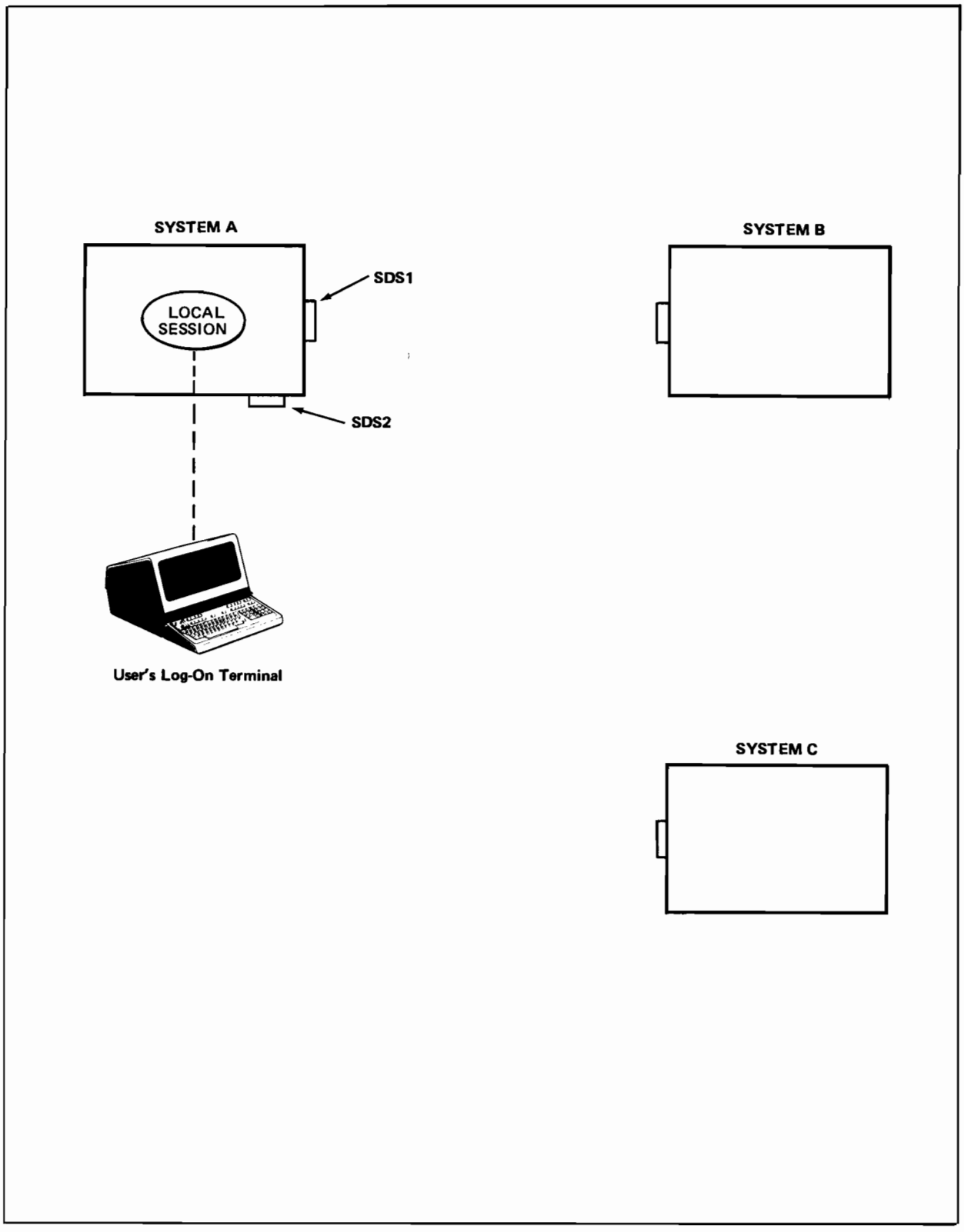


Figure 2-31. Initiating the Local Session (SSLC Example)

Next the user acquired access to a telephone connection between Systems A and B and initiated a remote session in System B:

```
:DSLINE SDS1 ;LOCID="A" ;REMID="B" ;PHNUM=257-8001

DS LINE NUMBER = #L3

:REMOTE HELLO RUSER.RACCOUNT

 SESSION NUMBER = #S35
 FRI, MAY 7, 1976, 1:38 PM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```

SDS1 is the device class name (as defined within System A) associated with the particular SSLC, A and B are the ID sequences identifying Systems A and B, respectively, and 257-8001 is the telephone number of the modem connected to an SSLC at System B. RUSER and RACCOUNT are valid user and account names, respectively, as defined by the accounting structure of System B.

Now we have the situation illustrated in figure 2-32.

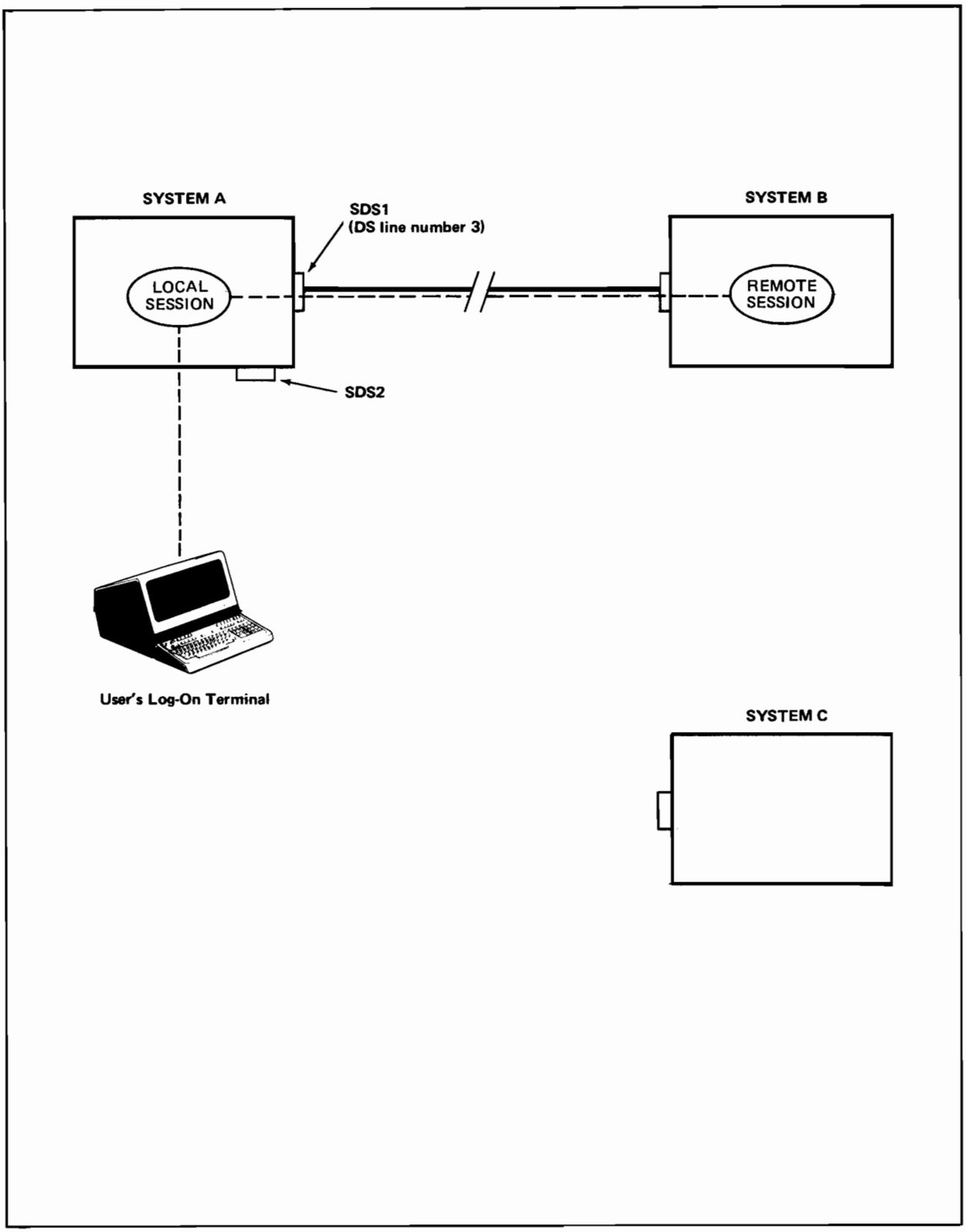


Figure 2-32. Establishing the Link With System B (SSLC Example)

Finally the user acquired access to an SSLC line between Systems A and C and initiated a remote session in System C:

```
:DSLINE SDS2 ;LOCID="A" ;REMID="C" ;PHNUM=377-2000

DS LINE NUMBER = #L4

:REMOTE HELLO RUSER.RACCOUNT

 SESSION NUMBER = #S21
 FRI, MAY 7, 1976, 1:39 PM
 HP32002A.00.A1

WELCOME TO SYSTEM C.
:
```

SDS2 is the device class name (as defined within System A) associated with the particular SSLC, A and C are the ID sequences identifying Systems A and C, respectively, and 377-2000 is the telephone number of the modem connected to an SSLC at System C. RUSER and RACCOUNT are valid user and account names, respectively, as defined by the accounting structure of System C.

We end up with the situation illustrated in figure 2-33, which you will notice is identical to figure 2-30 which started this example.

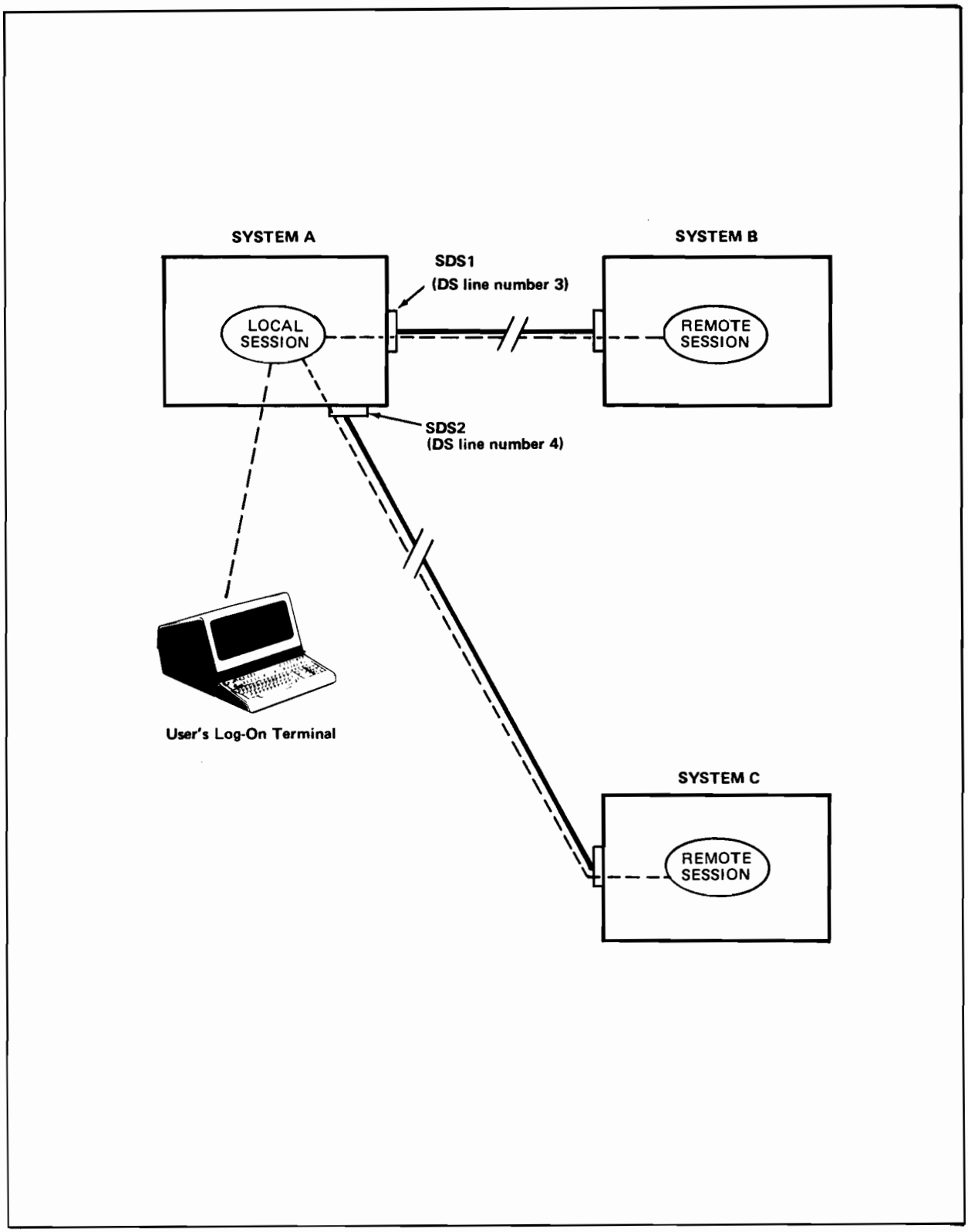


Figure 2-33. Establishing the Link With System C (SSLC Example)

Line Opening Failures

There are several reasons why a DSLINE command for opening a telephone line might be rejected by DS/3000, some of which have already been illustrated earlier in this chapter. The following list summarizes all of the likely causes of a line open failure.

- You made a syntax error in the DSLINE command.
- You gave an erroneous line specification (dsdevice) in the the DSLINE command. (There is no SSLC with the specified device class name or logical device number.)
- Someone already has exclusive access to the specified line.
- You asked for EXCLUSIVE access to a line which was already in use.
- The operator was not able to make the requested telephone connection and entered NO through the system console in response to the dial request message.
- 15 minutes elapsed and no telephone connection was established.
- The remote computer rejected your local ID sequence.
- The remote computer did not send a valid ID sequence (the received ID sequence did not match any of the remote ID sequences that you specified or, if you didn't specify any, did not match any of the configured remote ID sequences).
- The specified line is already in use and the remote ID sequence you supplied did not match the one used by the currently connected remote HP 3000.
- DS/3000 detected a hardware problem (the SSLC board is not responding correctly).

The various error numbers and messages for line opening failures are as follows:

Error Number	Message	Meaning	Corrective Action
1	UNKNOWN COMMAND	The command entered was not recognized as a legal command. You probably mistyped the command name (DSLINE or REMOTE HELLO).	Re-enter the command and be sure that the command name is entered correctly.
20,x	SYNTAX ERROR	A delimeter in the parameter list is not permitted in the command, or is not permitted at the point where it was detected. When a qualifying number is shown (x), the bad delimeter is adjacent to the indicated parameter element (usually following it).	Re-enter the command correctly.
22,x	ILLEGAL PARAMETER	A parameter in the command was not recognized as legal. The qualifying number (x) indicates which parameter was bad.	Check the parameter list, and re-enter the command correctly.
26,x	ILLEGAL KEYWORD	A keyword in the command was not recognized as legal. The qualifying number (x) indicates which keyword was bad.	Check the keywords, and re-enter the command correctly.

Error Number	Message	Meaning	Corrective Action
27,x	DUPLICATE KEYWORD	The command was rejected because the same keyword appeared twice in the parameter list. The qualifying number (x) indicates the duplicate keyword.	Re-enter the command without the duplicate keyword.
30,x	INVALID NUMBER	The buffer-size in the LINEBUF parameter is not decimal or is out of range (304 to 4096).	Check the parameter list, and re-enter the command correctly.
60,55	DS/3000 ERROR	Someone already has exclusive access to to the line. *** OR *** You requested exclusive access to the line and someone is already using it. *** OR ***	None. You must wait until the line is available. Either wait until the line is available or re-enter the command without the EXCLUSIVE parameter.

Error Number	Message	Meaning	Corrective Action
		The specified SSLC has been shut down by the console operator.	Re-enter the command specifying a different SSLC or get the console operator to bring up the particular SSLC.
60,56	DS/3000 ERROR	You gave an erroneous line specification (dsdevice). There is no SSLC with the specified device class name or logical device number.	Re-enter the command using a correct dsdevice parameter.
60,201	DS/3000 ERROR	The remote ID sequence you supplied (REMID=) was rejected by the remote computer. *** OR *** The specified SSLC is already open and you did not specify the currently active remote ID sequence (REMID=).	
60,202	DS/3000 ERROR	The telephone number you specified (PHNUM=) was either too long or contained something other than dashes and digits.	
60,204	DS/3000 ERROR	You used line opening parameters (LINEBUF=, PHNUM=,LOCID=,REMID=, or EXCLUSIVE) in a DSLINE ;CLOSE command.	

Error Number	Message	Meaning	Corrective Action
60,243	DS/3000 ERROR	The remote computer either did not respond within 16 attempts to establish the line (48 seconds) or responded each time with a negative acknowledgement (NAK).	
60,244	DS/3000 ERROR	There is a software problem in DS/3000 at either end of the line.	
60,246	DS/3000 ERROR	The remote computer terminated the connection (sent a DLE EOT) in response to your HP 3000's local ID sequence.	
60,249	(none)	The remote computer sent an EOT or a DLE EOT when DS/3000 tried to establish the line (i.e., in response to the original ENQ).	
60,250	DS/3000 ERROR	The line was momentarily established and then was lost. There may have been a brief power failure at the remote site or there may be a hardware problem with one of the MODEMs at either end of the line.	

Error Number	Message	Meaning	Corrective Action
60,251	DS/3000 ERROR	The "Data Set Ready" (CC) signal from the MODEM has changed from "set" to "clear". There are many possible causes. You may have inadvertently called a non-DS/3000 remote computer, or there may be a hardware problem with one of the MODEMs at either end of the line, or there may be a problem with the public telephone lines that you were using, or someone at either end of the line may have physically disconnected the line at one of the MODEMs.	
60,252	DS/3000 ERROR	DS/3000 detected a hardware problem with the particular SSLC interface board.	
60,253	DS/3000 ERROR	The console operator entered "NO" through the system console in response to the dial request message.	
60,254	(none)	The requested SSLC was not configured properly during system configuration.	

CLOSING A LINE

Once you have opened one or more communications lines, you can close any or all of them by using a variation of the DSLINE command. The format of the DSLINE command is presented in figure 2-34 below. For completeness all of the available parameters are shown. The shaded parameters apply to closing one or more communications lines. The others (non-shaded) apply to opening a hardwired or telephone line and are described under the appropriate topic earlier in this chapter.

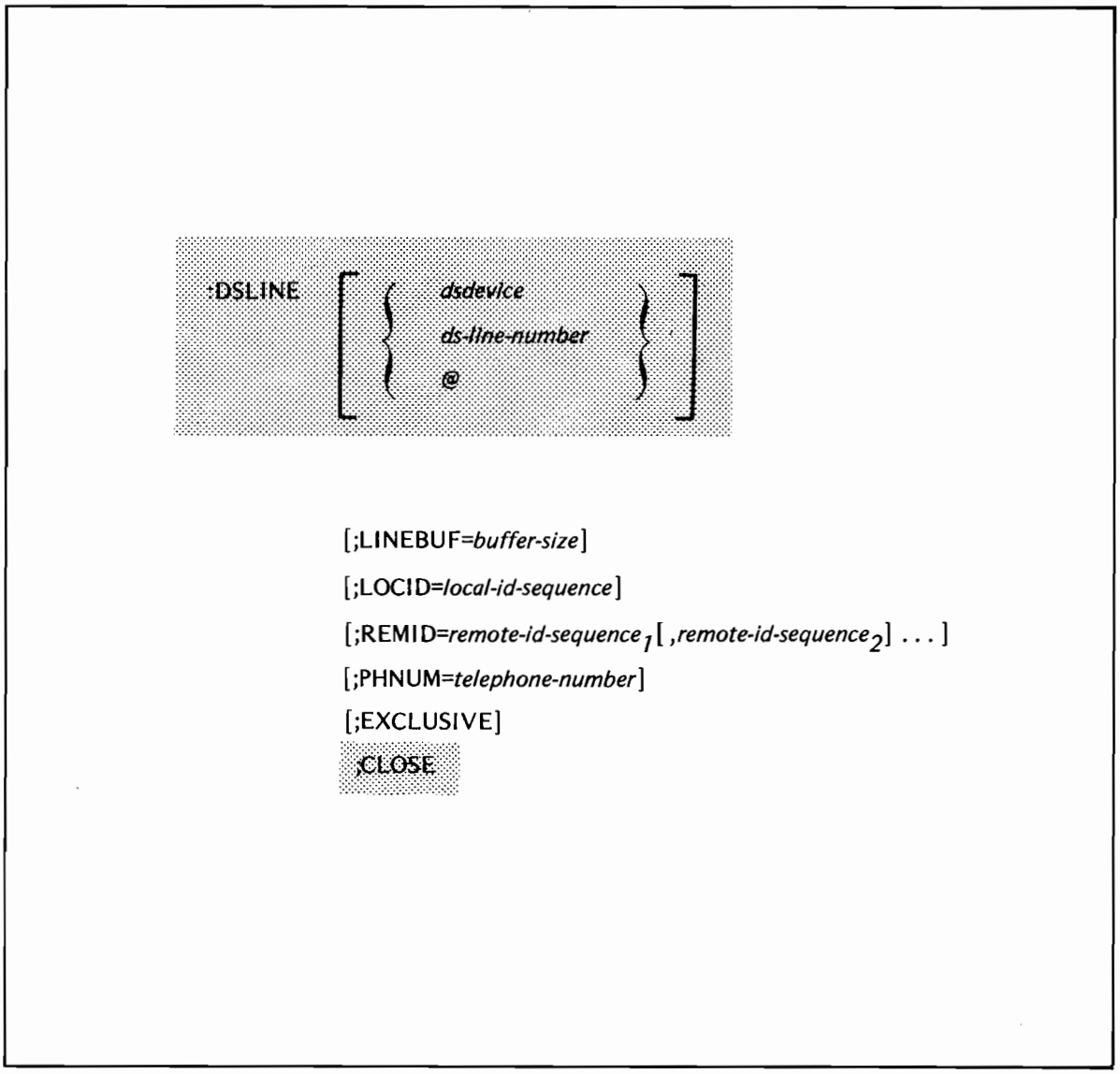


Figure 2-34. The DSLINE Command

Closing a Line

The parameters that pertain to closing one or more communications lines are as follows:

dsdevice	The device class name or logical device number specified in the DSLINE command that opened the particular HSI or SSLC line. (Optional parameter).
ds-line-number	The DS line number assigned to you by DS/3000 when the particular line was opened. (Optional parameter.)
@	This parameter specifies that you wish to close all of the lines that you currently have open. (Optional parameter.)
;CLOSE	This parameter specifies that you wish to close the specified line(s). (Required parameter.)

If no line identifier (dsdevice, ds-line-number, or @) is specified, DS/3000 closes the line that you most recently opened.

Examples

The following five examples illustrate the variations of the DSLINE command that can be used for closing one or more communications lines.


```
:DSLINE  HDS1 ;CLOSE
```

This form closes the line that is identified by the device class name HDS1.

```
:DSLINE  55 ;CLOSE
```

This form closes the line that is identified by the logical device number 55.

```
:DSLINE  0 ;CLOSE
```

This form closes all the lines that you currently have open.

Closing a Line

```
:DSLINE  ;CLOSE
```

This form closes the line that you most recently opened.

If you are sharing one or more physical communications lines with other users, the above forms of the DSLINE command close the line(s) for your application only (the other user's applications are not affected).

REMOTE SESSIONS

SECTION III

By establishing a communications link you have also initiated a session in the remote HP 3000 under the username, accountname, and groupname specified in the REMOTE HELLO command. You now have two distinct sessions in existence simultaneously from the same log-on terminal: a local session (in the HP 3000 to which you first logged on) and a remote session (in the HP 3000 at the other end of the communications line). Let's pause for a moment and see what this implies.

Within the local session you have access to all I/O devices and disc files in your local HP 3000 (subject to the usual MPE file security of course). This is a normal MPE interactive session in every respect: you enter MPE commands and use the various language and utility subsystems exactly as you would if DS/3000 were not present. This local session is running under the username, accountname, and groupname specified in the HELLO command that you used to first log-on. All user capabilities and file access available to you within the local session are determined by those log-on parameters.

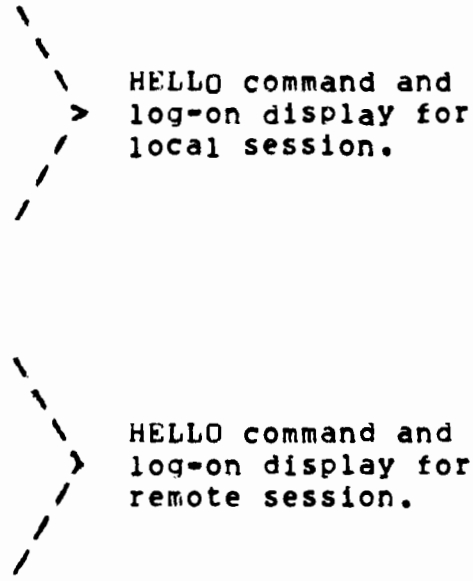
Within the remote session you have access to all I/O devices and disc files in the remote HP 3000 (again subject to the usual MPE file security). With the few minor exceptions described in the following pages this, too, is a normal MPE interactive session. All MPE commands and subsystems are, however, executed in the remote HP 3000. The output resulting from the executed commands and subsystems appears at your local log-on terminal. This remote session is running under the username, accountname, and groupname specified in the REMOTE HELLO command that you used in establishing the communications link. All user capabilities and file access available to you within the remote session are determined by those log-on parameters.

In chapter 4 of this manual ("Remote File Access") we will see how you can access the I/O devices and disc files of the remote HP 3000 from your local session and how you can access the I/O devices and disc files of the local HP 3000 from your remote session. For clarity's sake, however, the remainder of this chapter will treat local and remote sessions as separate, essentially unrelated entities that use only those resources available in the particular HP 3000 in which they are running. Let's take another look at the example presented at the end of chapter 1 and see how you execute commands within a remote session.

ISSUING REMOTE COMMANDS

If you recall, we used the following sequence of commands to establish the communications link:

```
carriage return
:HELLO USER,ACCOUNT

 SESSION NUMBER = #S49
 FRI, MAY 9, 1976, 9:05 AM
 HP32002A.00.A1

WELCOME TO SYSTEM A.
:DSLINE HDS2

DS LINE NUMBER = #L3

:REMOTE HELLO RUSER,RACCOUNT

 SESSION NUMBER = #S11
 FRI, MAY 9, 1976, 9:06 AM
 HP32002A.00.A1

WELCOME TO SYSTEM B.
:
```



At this point the remote session has been initiated but we are currently in the local session (as signified by the colon prompt character). To execute a command in the remote session we can use the following construct:

```
:REMOTE [xxx] command
```

where xxx is the DS line number returned by DS/3000 when the desired communications line was opened and command is the desired MPE command in its normal format. (The DS line number is necessary only if you have more that one communications line open simultaneously; if it is omitted then the line which you most recently opened is referenced by default). In the example at the end of chapter 1 we used this construct to execute a LISTF command, as follows:

```
:REMOTE LISTF

FILENAME

DATA1     DATA5     DATA6     FILE3     SOURCE1
:
```

Because we included the prefix REMOTE, the LISTF command is executed in the remote session (the implied account and group names are those established by the REMOTE HELLO command that initiated the remote session, in this case the public group of the SYS account). Although the LISTF command is executed in the remote HP 3000, the output generated by the command is displayed at our local log-on terminal.

In the above example you will notice that we didn't specify the DS line number associated with the particular communications line (3 in this example). This is because if no line number is specified DS/3000 uses by default the line most recently opened; we only have one communications line open from our local session so DS/3000 uses that line by default. If we had opened a second line we would need to tell DS/3000 which remote computer the remote command is to be executed in. We would do that by including the appropriate DS line number in the remote command, as follows:

```
:REMOTE  3  LISTF

FILENAME

DATA1     DATA5     DATA6     FILE3     SOURCE1
:
```

The above construct, however, only allows you to execute a single remote command. After the remote command has been executed, control returns to your local session (as signified by the colon prompt character).

But what if you want to execute a whole series of remote commands? It would obviously be a minor nuisance to have to prefix each command with the word REMOTE. DS/3000 provides a convenient solution to this situation. To execute a series of commands in the remote session, we can use the following construct:

```
:REMOTE  [xxx]
```

where xxx is again the DS line number of the desired communications line (specifying in which remote HP 3000 we want to execute commands). DS/3000 then prompts us for each command by displaying a # in column 1 of our terminal (in place of the standard MPE colon prompt). In the example at the end of chapter 1 we used this construct for entering two remote MPE commands (EDITOR and BYE), as follows:

```
:REMOTE
#EDITOR

HP32201A.5.00 EDIT/3000  FRI, MAY 9, 1976, 9:11 AM
(C) HEWLETT-PACKARD CO. 1976
/SET SHORT
/TEXT SOURCE1
/LIST ALL
    1    $CONTROL USLINIT,SOURCE
    2     IDENTIFICATION DIVISION.
    3     PROGRAM-ID. COBOL-TEST1.
```

```
     4     AUTHOR. JOE DOKES.
     5     ENVIRONMENT DIVISION.
     6     DATA DIVISION.
     7     WORKING-STORAGE SECTION.
     8     77  EDIT-FIELD      PIC $Z,ZZ9.99.
     9     77  TOTAL-COST      PIC 999V99.
    10     77  COST-OF-SALE    PIC 99V99.
    11     77  TAX             PIC 99V99.
    12     77  Y-N             PIC X.
    13
    14     PROCEDURE DIVISION.
    15     ENTER-ROUTINE.
    16         MOVE ZEROS TO TOTAL-COST.
    17         DISPLAY SPACE.
    18         DISPLAY "ENTER COST OF SALE".
    19         ACCEPT COST-OF-SALE.
    20         COMPUTE TAX = COST-OF-SALE * .06.
    21         ADD COST-OF-SALE, TAX TO TOTAL-COST.
    22         MOVE TOTAL-COST TO EDIT-FIELD.
    23         DISPLAY "TOTAL COST = " EDIT-FIELD.
    24         DISPLAY "ARE YOU FINISHED? (Y OR N)".
    25         ACCEPT Y-N.
    26         IF Y-N = "N" GO TO ENTER-ROUTINE.
    27         STOP-RUN.
/MODIFY 18
MODIFY   18
    DISPLAY "ENTER COST OF SALE".
                                  I (NO DECIMAL POINT)
    DISPLAY "ENTER COST OF SALE (NO DECIMAL POINT)".
/KEEP SOURCE1
PURGE OLD? YES
/EXIT
CLEAR? YES

 END OF SUBSYSTEM
#BYE


CPU (SEC) = 4
CONNECT (MIN) = 7
FRI, MAY 9, 1976, 9:13 AM
END OF SESSION
#:
:
```

Let's try another example that uses more than those two remote commands.

```
:REMOTE
#LISTF

FILENAME

DATA1     DATA5     DATA6     FILE3     SOURCE1
#PURGE DATA5
#PURGE DATA6
#LISTF

FILENAME

DATA1     FILE3     SOURCE1
#RUN FCOPY.PUB.SYS

HP32212A.0.03 FILE COPIER

>FROM=DATA1 ;TO=DATA2 ;NEW
EOF FOUND IN FROMFILE AFTER RECORD 679

680 RECORDS PROCESSED *** 0 ERRORS

>EXIT

END OF PROGRAM
#LISTF

FILENAME

DATA1     DATA2     FILE3     SOURCE1
#BYE


CPU (SEC) = 4
CONNECT (MIN) = 7
FRI, MAY 9, 1976, 9:13 AM
END OF SESSION
#:
:
```

You will notice that except for the # prompt (in place of the standard colon prompt) this looks exactly like a normal MPE interactive session. All of the commands shown above are entered through the local log-on terminal but the MPE and FCOPY commands are executed in the remote session within the remote HP 3000. After each remote MPE command was executed, however, control remained in the remote session (as signified by the # prompt character). When the remote session was terminated, control then returned to your local session (as signified by the colon prompt character).

Using The Remote Subsystem From a Batch Job

While in a batch job, you can establish a remote session by using the DSLINE or REMOTE HELLO command.

The job to be streamed may be similar to the following:

```
:JOB BARB,LEWIS
:DSLINE HDS2
:REMOTE HELLO RBARB,RLEWIS
:REMOTE
#FILE OUT;DEV=LP
#BUILD WORK;DISC=50
#RUN USERPROG
#PURGE WORK
#:
:REMOTE BYE
:DSLINE;CLOSE
:EOJ
```

NOTE

The remote # prompt is optional.

An important point for you to remember is that the remote session is interacting with the job in the same way as a terminal device. If the remote session detects any errors the error is printed and the next record in the job file is read in the same manner as waiting for a character or carriage return on a terminal. The record is then lost to the job.

The BREAK Key

Within a remote session you can use the BREAK key to temporarily interrupt remote processing, however, by doing so you maty return control to the MPE Command Interpreter of your local HP 3000, or you may suspend the subsystem you are executing temporarily. This is determined by how you execute the command in a remote session. There are two ways in which you can execute commands in a remote session:

- By prefixing each command with the word REMOTE.
- By entering the word REMOTE, which prompts you for each command.

PREFIXING EACH COMMAND WITH REMOTE. By prefixing eqach command with the word REMOTE, you will have a some what limited value when you interrupt remote processing with the BREAK key. You can only execute local commands. YOU CANNOT ISSUE REMOTE COMMANDS DURING A BREAK. To continue remote processing at the point where it was interrupted, you merely enter REMOTE RESUME in response to the local MPE colon prompt.

As an example, let's assume that we are in the midst of using the text editor in a remote session and we suddenly decide to start a job stream executing concurrently in our local HP 3000. The sequence of commands would be similar to the following:

```
:REMOTE EDITOR

HP32201A.5.00 EDIT/3000  FRI, MAY 9, 1976, 9:11 AM
(C) HEWLETT-PACKARD CO. 1976
/ADD
    1   DOE, JOHN              29  M  CHI
    2   BLACK, PATRICIA        23  F  SF
    3   SIMON, NEIL            43  M  NY
    4   MACK, SHIRLEY          38  F  DET
    5
```

BREAK key pressed here.

Local session prompt.

```
:STREAM COBTEST1
 #J19
:REMOTE RESUME
```

Control is now in the local session.

```
READ PENDING
MICHAELS, WILLIAM     32  M  CHI
    6   O'LEARY, TIMOTHY       49  M  DET
    7   MARTIN, MARY           34  F  LA
    8   MURIN, JOICE           42  F  CHI
```

Control is now back in the remote session

.
.
.

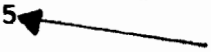
You will notice that when the BREAK key was pressed the text editor in the remote HP 3000 was waiting for us to enter the text for line 5. The BREAK key interrupted the remote session and passed control to the MPE Command Interpreter of the local HP 3000 (as signified by the colon prompt). Within the local session we issued the STREAM command which caused the file COBTEST1 to be executed in the local HP 3000. We then issued the RESUME command which passed control back to the remote session at the point where it was interrupted (that is, the text editor in the remote HP 3000 is waiting for us to enter the text for line 5). We enter the text for line 5 and the remote session proceeds as though nothing had happened.

Note that by the end of the above example we have the local job stream, the local session, and the remote session all operational simultaneously.

ENTERING REMOTE. By entering the word REMOTE, you can interrupt remote processing with the BREAK key and still continue to issue remote commands. To continue remote processing at the point where it was interrupted, you merely enter RESUME in response to the remote # prompt.

As an example, let's assume that we are in the midst of using the text editor in a remote session and we suddenly decide to start a job stream executing in our remote HP 3000. The sequence of commands would be similar to the example shown previously with a few minor differences as follows:

```
:REMOTE
#EDITOR

HP 32201A.5.00 EDIT/3000  FRI, FEB 11, 1977, 9:20 AM
(C) HEWLETT-PACKARD CO. 1976
/ADD
    1   LEWIS, LEO                51  M  SV
    2   LAGERGREN, FRED           25  M  SJ
    3   DICKINSON, MARY           21  F  SC
    4   LAGREGREN, LINDA          24  F  SJ
    5
```



```
BREAK key pressed here.

#STREAM APLTEST1          \    Control is still
  #J20                     >   in the remote
#RESUME                    /   session.

READ PENDING
MELLO, HENRY          44  M  SJ
    6   SOARES, JOE               59  M  LA
    7   LAWRENCE, ALICE           44  F  SJ
    8   LEWIS, BOB                29  M  WASH
```

You will notice that when the BREAK key was pressed the text editor in the remote HP 3000 was waiting for us to enter the text for line 5. The BREAK key interrrupted the remote session but control remained in the remote HP 3000 (as signified by the remote # prompt). The STREAM command executed the file APLTEST1 within the remote HP 3000, then we issued the RESUME command which passed control back to the point where the text editor was interrrupted (that is, the text editor is waiting for us to enter the text for line 5). We enter the text for line 5 and the remote session proceeds as though nothing had happened.

The Control Keys

Within a remote session CONTROL-H, CONTROL-X, and CONTROL-Y perform exactly the same functions as they do in a normal MPE interactive session.

For example, if you are using FCOPY or the text editor in a remote session you can use CONTROL-Y to prematurely terminate an FCOPY or text editor operation. When the operation terminates, control is still in the particular subsystem within the remote session.

Similarly, you can use CONTROL-H to delete the last character entered or CONTROL-X to delete the line of text currently being entered. In both of these cases, after the deletion occurs control remains in the remote session.

ISSUING LOCAL COMMANDS

Whenever the standard MPE colon prompt is displayed at your terminal you are in the local session. Within the local session you enter MPE commands in their normal format in response to the colon prompt. If you are in the midst of a remote session (i.e., you used the command :REMOTE and DS/3000 is issuing the # prompt character), you can return control to your local session by entering a colon, as follows:

```
#:
```

In response to the remote colon, control returns to the MPE Command Interpreter of your local HP 3000 which then prompts you for local commands with the colon prompt character. Note that the remote colon does not terminate the remote session; you can resume processing in the remote session by again using either of the constructs described under "Issuing Remote Commands", above.

TERMINATING A REMOTE SESSION

You can terminate a remote session either from within the local session or from within the remote session itself.

From The Local Session

Whenever the standard MPE colon prompt is displayed at your terminal you are in the local session. To terminate a remote session from within your local session, use the following command:

```
:REMOTE [xxx] BYE
```

where xxx is the DS line number associated with the communications line connecting the particular remote session to your local session. (The DS line number is necessary only if you have more than one communications line open simultaneously; if it is omitted then the line that you most recently opened is referenced by default.)

For instance, in the example at the end of chapter 1 we could have used either of the following sequences to terminate the remote session:

```
#:
:REMOTE BYE

CPU (SECS) = 4
CONNECT (MIN) = 7
FRI, MAY 9, 1976, 9:13 AM
END OF SESSION
:
```

*** OR ***

```
#:
:REMOTE 3 BYE

CPU (SECS) = 4
CONNECT (MIN) = 7
FRI, MAY 9, 1976, 9:13 AM
END OF SESSION
:
```

In both cases we used the remote colon to return control from the remote session to our local session. In either case the remote session is terminated.

If the communications line was opened using the DSLINE= parameter of the REMOTE HELLO command, the line is automatically closed when the remote session terminates. To initiate another remote session over the same communications line, you must once again open the line (using either the DSLINE command or the DSLINE= parameter of the REMOTE HELLO command) and then issue another REMOTE HELLO command.

If the communications line was opened using the DSLINE command, it is still open. To initiate another remote session over the same communications line, merely issue another REMOTE HELLO command (you do not need to issue another DSLINE command or DSLINE= parameter because the communications line is still open). To close the communications line, use the constructs presented in chapter 2.

From The Remote Session

Whenever the # prompt is displayed at your terminal you are in the remote session. To terminate a remote session from within the remote session itself, use the following command:

```
#BYE
```

Note that we do not need to supply a DS line number in this case because DS/3000 knows implicitly which remote session we wish to terminate (i.e., the one in which the #BYE command is executed).

If you recall, we used this command to terminate the remote session in the example at the end of chapter 1, as follows:

```
#BYE

CPU (SEC) = 4
CONNECT (MIN) = 7
FRI, MAY 9, 1976, 9:13 AM
END OF SESSION
#
```

The remote session is terminated but you will notice that DS/3000 is still issuing the # prompt character. To return control to the local session, issue a remote colon (described earlier under "Issuing Local Commands").

If the communications line was opened using the DSLINE= parameter of the REMOTE HELLO command, the line is automatically closed when the remote session terminates. To initiate another remote session over the same line, you must once again open the line (using the DSLINE command or the DSLINE= parameter of a REMOTE HELLO command) and then issue another REMOTE HELLO command.

If the communications line was opened using the DSLINE command, it is still open. To initiate another remote session over the same communications line, merely issue an appropriate remote MPE HELLO command (you do not need to use the prefix REMOTE because DS/3000 is still waiting for you to enter a remote command. Nor do you need to issue another DSLINE command or DSLINE= parameter because the communications line is still open). To close the communications line use the constructs presented in chapter 2.

REMOTE FILE ACCESS

SECTION IV

In the preceding chapters we have seen how you can establish a communications link between two HP 3000s and thereby use the computing power of the remote HP 3000. But that is only half the story! Through the use of the DS/3000 remote file access capability, programs running in your local session can:

- use any of the devices connected to the remote HP 3000 as though they were connected directly to your local HP 3000; and
- access any of the disc files of the remote HP 3000 (subject to the normal MPE file security, of course) as though they resided at your local HP 3000 site.

This capability, in conjunction with the remote session capability, suddenly puts all of the computing power and all of the hardware and software resources of a remote HP 3000 at your fingertips.

The remainder of this chapter is divided into two main parts. The first part ("Command Access") describes how you can issue local MPE FILE commands that define devices and/or files residing at the remote HP 3000 site. The second part ("Programmatic Access") describes how you can use the standard set of MPE File System intrinsics within your local programs to access devices and/or files residing at the remote HP 3000 site.

COMMAND ACCESS

Once a DS/3000 communications link has been established between your HP 3000 and a remote HP 3000, you can issue local MPE FILE commands that define devices and/or files residing at the remote HP 3000 site. To make this possible, the DEV= parameter of the MPE FILE command has been expanded to include a DS line specification in addition to the usual device specification. The format of the DEV= parameter is as follows:

```
;DEV=[dsdevice]#device
```

where dsdevice is the device class name or logical device number that you used when establishing the particular communications link (this specifies the physical line connecting the two computers) and device is the device class name or logical device number of the desired remote device as established within the remote HP 3000.

For completeness, the full syntax and parameter specifications for the MPE FILE command are presented in figures 4-1 through 4-4 below. You will notice that the only difference between this presentation and the one in the MPE Commands Reference Manual is the addition of "dsdevice#" to the DEV= parameter. This one small syntax change has enormously powerful implications. It means that from within your local session you can access any of the devices and/or disc files of a remote HP 3000 as though they resided at your local HP 3000 site. Access to remote disc files is, of course, subject to the usual MPE file security. The user, account, and group names that you specified in the REMOTE HELLO command when establishing the communications link are the ones used by MPE in the remote HP 3000 for determining your file access capabilities.

Following figures 4-1 through 4-4 are five annotated examples illustrating remote device and file access from a local session.

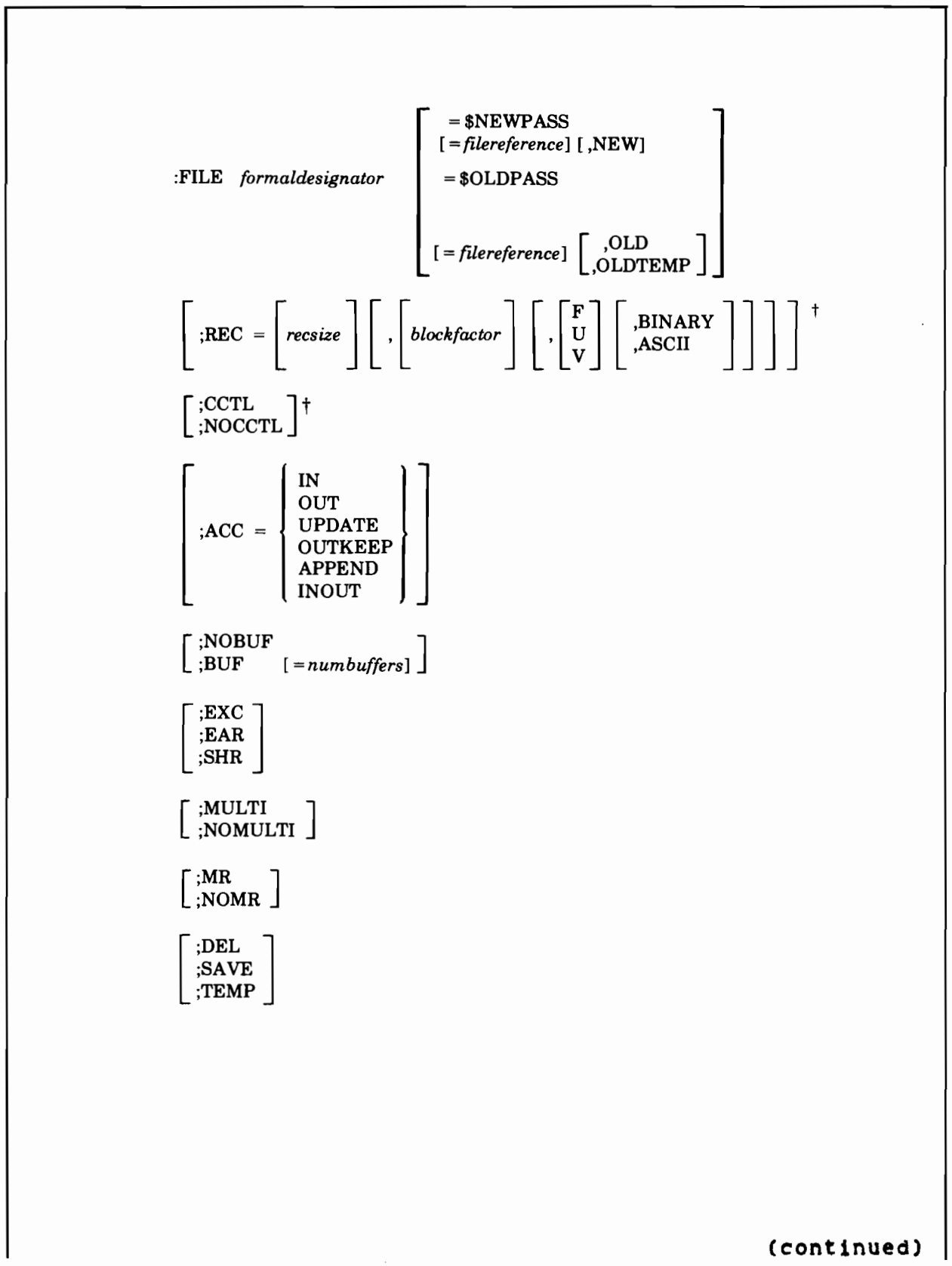


Figure 4-1. MPE FILE Command Syntax For New or Old Files

[;DEV=[[[*dsdevice*]#]*device*][,[*outputpriority*][,*numcopies*]]]††
[;CODE = *filecode*]†
[;DISC = [*numrec*] [,[*numextents*] [,*initalloc*]]]†

[;NOWAIT]
[;WAIT   ]

NOTE

The parameter group [*;DISC*=[*numrec*] [,[*numextents*] [,*initalloc*]]] cannot be included if the parameter group

[=*filereference*] [,OLD / ,OLDTEMP] is specified.

Dagger (†) indicates these parameters are not used for old disc files.

†† This parameter is used for old disc files only when defining a remote file in a DS/3000 environment.

Figure 4-1. MPE FILE Command Syntax for New or Old Files (Continued)

:FILE *formaldesignator* = **formaldesignator*

NOTE

The second parameter in this command *must* include the preceding asterisk (*) noted above.

Figure 4-2. MPE FILE Command Syntax For User Pre-Defined (Back-Referenced) Files

```
:FILE formaldesignator = $NULL

                or

:FILE formaldesignator = { $STDIN   }
                         { $STDINX  }
                         { $STDLIST }

[;REC = [recsize] [, [blockfactor] [, [F] [,BINARY]]]]
                                      [U] [,ASCII]
                                      [V]

[;CCTL  ]
[;NOCCTL]

[;ACC = { IN      }]
        { OUT     }
        { UPDATE  }
        { OUTKEEP }
        { APPEND  }
        { INOUT   }

[;NOBUF                    ]
[;BUF     [= numbuffers]   ]

[;EXC]
[;EAR]
[;SHR]

[;MULTI  ]
[;NOMULTI]

[;MR]
[;NOMR]

[;NOWAIT]
[;WAIT  ]
```

Figure 4-3. MPE FILE Command Syntax For System-Defined Files

formaldesignator	Formal file designator referenced in your program. This is the name by which your program recognizes the file. Contains from 1 to 8 alphanumeric characters, beginning with a letter. (Required parameter.)
filereference	Actual file designator. This is the name by which MPE recognizes the file, written in the following format: filename[/lockword][.groupname][.accountname] All four sub-parameters are names that contain from 1 to 8 alphanumeric characters, beginning with a letter. If the file has no lockword and belongs to the log-on group and account, only *filename* is necessary. If you omit the *filereference* parameter from the :FILE command, the actual designator is equated to the formal designator. (Optional parameter.)
$NEWPASS	A temporary disc file that can be automatically passed to any succeeding MPE command within the session/job, which references the file by the name $OLDPASS. When $NEWPASS is closed, its name is automatically changed to $OLDPASS, and any previous file named $OLDPASS in the session/job is deleted. (Optional parameter.)
$OLDPASS	The name of the last temporary disc file closed as $NEWPASS. (Optional parameter.)

NOTE

If you do not include *any* actual file designator (filereference, $NEWPASS, or $OLDPASS) in the :FILE command, the formal designator is used as the actual designator.

NEW	Specification that the file is a new file. (Required parameter if *filereference* is used for a new file; Optional parameter otherwise.)
OLD	Previously-existing permanent file saved in system file domain. File continues to exist after current session/job terminates. (Optional parameter).

(continued)

Figure 4-4. MPE FILE Command Parameters

OLDTEMP	Previously-existing temporary file in session/job temporary file domain. File is deleted at end of current session/job. (Optional parameter.) NOTE NEW, OLD, and OLDTEMP specify the file domain indicating where the file exists. If all are omitted, the domain specified in FOPEN intrinsic takes effect.
recsize	Size of logical records in file. If a positive number, this represents *words;* if a negative number, this represents *bytes*. For files containing fixed-length records, this is the size of each logical record. For files containing undefined-length records, this is the maximum record size. For files containing variable-length records, this is a value used to calculate maximum record size as follows: *maxrecordsize* = *recsize* x *blockfactor* Default: Determined by System Supervisor during System Configuration. The values generally specified by HP are: Disc 128 Tape 128 Printer 66 Card Reader 40 Card Punch 40 Terminal 40 (Optional parameter.)
blockfactor	Size of each buffer established for file, specified as an integer equal to number of logical records per block. For fixed-length records, *blockfactor* is actual number of records in a block. For variable-length records, *blockfactor* is a multiplier used to compute block size (maximum *recsize* x *blockfactor*). For undefined length records, *blockfactor* is always one logical record per block; any unused portion of the block is filled with ASCII blanks or binary zeros. The blockfactor value may be overridden by MPE. Default: Calculated by dividing the specified *recsize* into the configured blocksize; this value is rounded downward to an integer that is never less than 1. (Optional parameter.)

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

F	File contains fixed-length records. (Optional parameter.)
U	File contains undefined-length records, with block size equal to logical record size. (Optional parameter.)
V	File contains variable-length records, with block size equal to logical record size on unit-record devices. (Optional parameter.)
	NOTE If F, U, or V is not specified, the default value is F for disc and magnetic tape files, and U for all others.
BINARY	File contains binary-coded records. (Optional parameter.)
ASCII	File contains ASCII-coded records. (Optional parameter.)
	NOTE If neither BINARY nor ASCII is specified, the default value is BINARY.
CCTL	Indicates you are supplying carriage-control characters with your write requests. Valid for any ASCII file. (Optional parameter.)
NOCCTL	Indicates you are *not* supplying carriage-control characters with your write requests. (Optional parameter.)
	NOTE If neither CCTL nor NOCCTL are specified and carriage-control characters are pertinent to the file, the default value is NOCCTL.
IN	File permits read-access only. (FWRITE, FUPDATE, and FWRITEDIR intrinsics cannot reference this file.) (Optional parameter.)

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

OUT	File permits write-access only. This option re-sets end-of-file indicator to first record. Any data on the file prior to current FOPEN request is deleted. (FREAD, FREADSEEK, and FREADDIR intrinsics cannot reference this file.) (Optional parameter.)
OUTKEEP	File permits write-access only, allowing you to add new records both before and after current end-of-file indicator. (FREAD, FREADSEEK, and FREADDIR intrinsics cannot reference this file.) (Optional parameter.)
APPEND	File permits append-access only, allowing you to add new records after current end-of-file indicator only. (FREAD, FREADSEEK, FREADDIR, FPOINT, FSPACE, and FWRITEDIR intrinsics cannot reference this file.) (Optional parameter.)
INOUT	File permits input/output access. (Any file intrinsic except FUPDATE can be issued against this file.) (Optional parameter.)
UPDATE	Update access. (All file intrinsics can be issued for this file.) (Optional parameter.)

NOTE

IN, OUT, UPDATE, OUTKEEP, APPEND, and INOUT all belong to the ACC = keyword group. If this entire group is omitted in the :FILE command (and in the FOPEN intrinsic that opens the file), the default value assigned is IN for files on all devices except those intended for output-only such as card punches, printers, paper tape punches, and plotters. If a process attempts to violate these access restrictions on a file, an error is returned. (Optional parameter.)

NOBUF	Specification that no input/output buffering is to take place, and no buffers are allocated for the file. (Optional parameter.)

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

numbuffers	Number of buffers to be allocated for the file. This must be an integer, with a maximum value of 16. If set to 0, a value of 2 is assigned. Do not use this parameter for files representing interactive terminals, since a system-managed buffering method is used in such cases. (Optional parameter.) NOTE If *NOBUF* and *numbuffers* are both omitted, a default value of 2 buffers is assigned.
EXC	After file is opened, prohibits concurrent access (in any mode) to file through another FOPEN request, whether issued by this or another running program (process), until this process issues FCLOSE or terminates. (Optional parameter.)
EAR	After file is opened, prohibits concurrent write-access to file through another FOPEN request within this or another process, until this process issues an FCLOSE or terminates. (Optional parameter.)
SHR	After file is opened, permits concurrent access to file through another FOPEN request issued by this or another process in any session or job. (Optional parameter.) NOTE If AC= IN is specified, and EXC, EAR, and SHR are omitted, then default assigned is SHR. Otherwise, default assigned is EXC.
MULTI	Shares this file so that several concurrently-running programs (processes) *in this session/job* can transfer records to and from the file sequentially, regardless of *blockfactor* or buffering (if any) specified. All access-control information, including buffers themselves, is shared between the processes, with result that transfers occur in the order in which the processes request access. You cannot, however, share the file in this way with processes running under other sessions/jobs. (Optional parameter.)

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

NOMULTI	Prohibits sharing files in MULTI mode. (Optional parameter.) NOTE If MULTI and NOMULTI are both omitted, the default assigned is NOMULTI.
MR	Request for Multi-record Mode. Individual read or write requests are not confined to record boundaries. Thus, if the number of words or bytes to be transferred exceeds the size of the record, the remaining words or bytes are taken from subsequent records until the number requested are transferred. Restricted to NOBUF files. (Optional parameter.)
NOMR	Request for normal recording mode (with no multi-record read/write). Individual read/write requests are confined to record boundaries. (Optional parameter.) NOTE If MR and NOMR are both omitted, the default assigned is NOMR.
DEL	Request for regular temporary file disposition; the file is deleted when your program closes it. (Optional parameter.)
SAVE	Request for permanent file disposition; the file remains in the system file domain, potentially available to other users, when your program closes it. (Optional parameter.)
TEMP	Request for session/job temporary file disposition. The file remains in the session/job temporary file domain when your program closes it; but when your session/job terminates, the file is deleted. (Optional parameter.) NOTE DEL, SAVE, and TEMP denote the disposition of the file. If none of these keywords is specified in this :FILE command, or the disposition is not specified in the FCLOSE intrinsic that closes the file, the file is returned to the disposition it had when opened. For example, a new file (other than $NEWPASS) is deleted; an old file is returned to the domain in which it was found.

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

dsdevice#	The device class name or logical device number that you used when establishing the particular communications link. This specifies the physical line connecting your local HP 3000 to the remote HP 3000 or the remote HP 3000 to your local HP 3000. (Optional parameter.)
device	A *device class name* designating the type of device, or a *logical device number* indicating the specific device, on which the file resides. The device class name makes a non-specific, generic reference to a *type* of device (such as any disc drive or magnetic tape unit). This name is defined and related to a specific set of devices when the system is configured. It must contain from one to eight alphanumeric characters, begin with a letter, and terminate with any non-alphanumeric character such as a blank. Examples are CARD, LP, TTY, and TAPE2. The logical device number refers to a specific single device. This is a number assigned to each device when the system is generated. This specification is only used when assignment of a particular device is truly necessary. For example, you would specify the logical device number of a specific device when running a hardware diagnostic program for that device. The device specification is *not* used if the file is an old disc file or if the actual file designator used is $STDIN, $STDINX, $STDLIST, $NEWPASS, $OLDPASS, or $NULL (since these names are already assigned to devices by the system). If the file is a new disc file and the *device* parameter specifies a device class name, all extents are restricted to devices of this class; if the *device* parameter specifies a logical device number, all extents are restricted to the single device identified by that number. Default: Device Class Name DISC. (Optional parameter.)

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

outputpriority (Spooled output devicefiles only.) The output priority for this file, used to select next spooled devicefile (on disc) for output, among all those contending for a specific printer or card punch. Determines the order in which files are produced when several are awaiting the same device. This must be a value between 1 (lowest priority) and 13 (highest priority), inclusive. If this value is less than the current *outfence* set by the Console Operator, the file is deferred from printing/punching until the operator raises the *outputpriority* of the file or lowers the outfence. This parameter is ignored for non-spooled output devices. Default: 8 (if logging enabled) or 13 (if logging disabled). (Optional parameter.)

numcopies (Spooled output devicefiles only.) Number of copies of file to be produced by spooling facility. The copies will not appear contiguously if the Console Operator intervenes or if a file of equal or higher *outputpriority* becomes READY before the last copy is started. This parameter is ignored for non-spooled output devices. Default: 1. (Optional parameter.)

filecode Code indicating a specially-formatted file. This code is recorded in the file label and is available to programs accessing the file through the FGETINFO intrinsic. Must be a positive integer ranging from 0 to 1023, or one of the HP-defined integers or mnemonics listed below:

Mnemonic	HP-defined Integer	Defines the File as:
USL	1024	User subprogram library (USL) file.
BASD	1025	BASIC data file.
BASP	1026	BASIC program file.
BASFP	1027	BASIC fast program file.
RL	1028	Relocatable library (RL) file.
PROG	1029	Program file.
STAR	1030	STAR file.
SL	1031	Segmented library (SL) file.
(None)	1040	Cross-Loader ASCII file (SAVE).

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

(None)	1041	Cross-Loader relocated binary file.
(None)	1042	Cross-Loader ASCII file (DISPLAY).
(None)	1050	EDIT KEEPQ file (non-COBOL).
(None)	1051	EDIT KEEPQ file (COBOL).
(None)	1052	EDIT TEXT file (COBOL).
(None)	1060	RJE punch file.
(None)	1070	QUERY procedure file.

numrec

Total maximum file capacity, specified only for a NEW file, in terms of logical records (for files containing fixed-length records) and blocks (for files containing variable-length and undefined-length records). Maximum capacity allowed is 2,097,120 sectors. Default: 1023. (Optional parameter.)

numextents

Number of extents (integral number of contiguously-located disc sectors) that can be dynamically allocated to the file as logical records are written to it; specified for new files only. The size of each extent (in terms of records) is determined by the *numrecs* parameter value divided by the *numextents* parameter value. Extents can be allocated on any disc in the device class specified in the *device* parameter. If you want to ensure that all extents for a file reside on the same disc, use the logical device number of that disc or a device class name relating to a single disc device, in the *device* parameter.

If specified, *numextents* must be an integer value from 1 to 32. Default: 8. (Optional parameter.)

initalloc

Number of extents to be allocated to the file *at the time it is opened,* specified for new files only. Must be an integer from 1 to 32. If attempt to allocate requested space fails, the FOPEN intrinsic that opens the file returns an error condition code to your program. Default: 1. (Optional parameter.)

NOWAIT

Allows program that references this file to initiate an input/output request and to have control returned before completion of the request. To confirm completion of such requests, your program must call the IOWAIT intrinsic after each request, as described in *MPE Intrinsics Reference Manual.* Also, multi-record access is prohibited.

(continued)

Figure 4-4. MPE FILE Command Parameters (Continued)

	NOTE To use the NOWAIT parameter, your program must be running in Privileged Mode. If you specify NOWAIT, this implies that no buffering (NOBUF) and normal recording mode (NOMR) are used. (Optional parameter.)
WAIT	Disallows NOWAIT input/output, discussed above. (Optional parameter.)
	NOTE If NOWAIT and WAIT are both omitted, WAIT is assigned by default.
**formaldesignator*	Name of a file defined in a previous :FILE command, preceded by an asterisk. (Required parameter.)

Figure 4-4. MPE FILE Command Parameters (Continued)

Example #1

Assume that we are maintaining an ASCII file containing both uppercase and lowercase characters using the Text Editor on our HP 3000 but that we don't have an upper/lowercase line printer. Let us further assume that elsewhere in the same building there is another HP 3000 with an upper/lowercase line printer, that both HP 3000s have DS capability, and that they are connected to one another by a coaxial cable and HSIs. We can access the remote line printer as follows.

First we log-on to our HP 3000 and establish a communications link with the remote HP 3000.

```
:HELLO USER.ACCOUNT
:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2
```

where USER and ACCOUNT are valid user and account names (respectively) within the accounting structure of our local HP 3000, RUSER and RACCOUNT are valid user and account names (respectively) within the accounting structure of the remote HP 3000, and LINE2 is the device class name of the local HSI to which the coaxial cable is connected.

Next we issue a local MPE FILE command that defines the desired line printer as a remote device.

```
:FILE LIST;DEV=LINE2#SLOWLP
```

where LIST is the formaldesignator by which we will subsequently reference the line printer, LINE2 is the device class name we used when establishing the particular communications link, and SLOWLP is the device class name (as established within the remote HP 3000) of the upper/lowercase line printer.

Then we invoke the Text Editor of our local HP 3000 specifying the remote line printer as the off-line listing device:

```
:EDITOR *LIST
```

Thereafter we can direct the Text Editor off-line output to the upper/lowercase line printer as though it were connected directly to our local HP 3000. For example, we could print the content of the file TEXTFILE on the upper/lowercase line printer as follows:

```
/TEXT TEXTFILE
/LIST ALL,OFFLINE
```

The entire command sequence described above is as follows (refer to Figure 4-5):

```
:HELLO USER.ACCOUNT
 SESSION NUMBER = #S98
 TUE, AUG 3, 1976, 12:51 PM
 HP32002A.00.A1

:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2

DS LINE NUMBER = #L3

 SESSION NUMBER = #S17
 TUE, AUG 3, 1976, 12:52 PM
 HP32002A.00.A1

:FILE LIST;DEV=LINE2#SLOWLP
:EDITOR *LIST

HP32201A.5.00 EDIT/3000  TUE, AUG 3, 1976, 12:53 PM
(C) HEWLETT-PACKARD CO. 1976
/TEXT TEXTFILE
/LIST ALL,OFFLINE
*** OFF LINE LISTING BEGUN. ***

    .
    .
    .
```

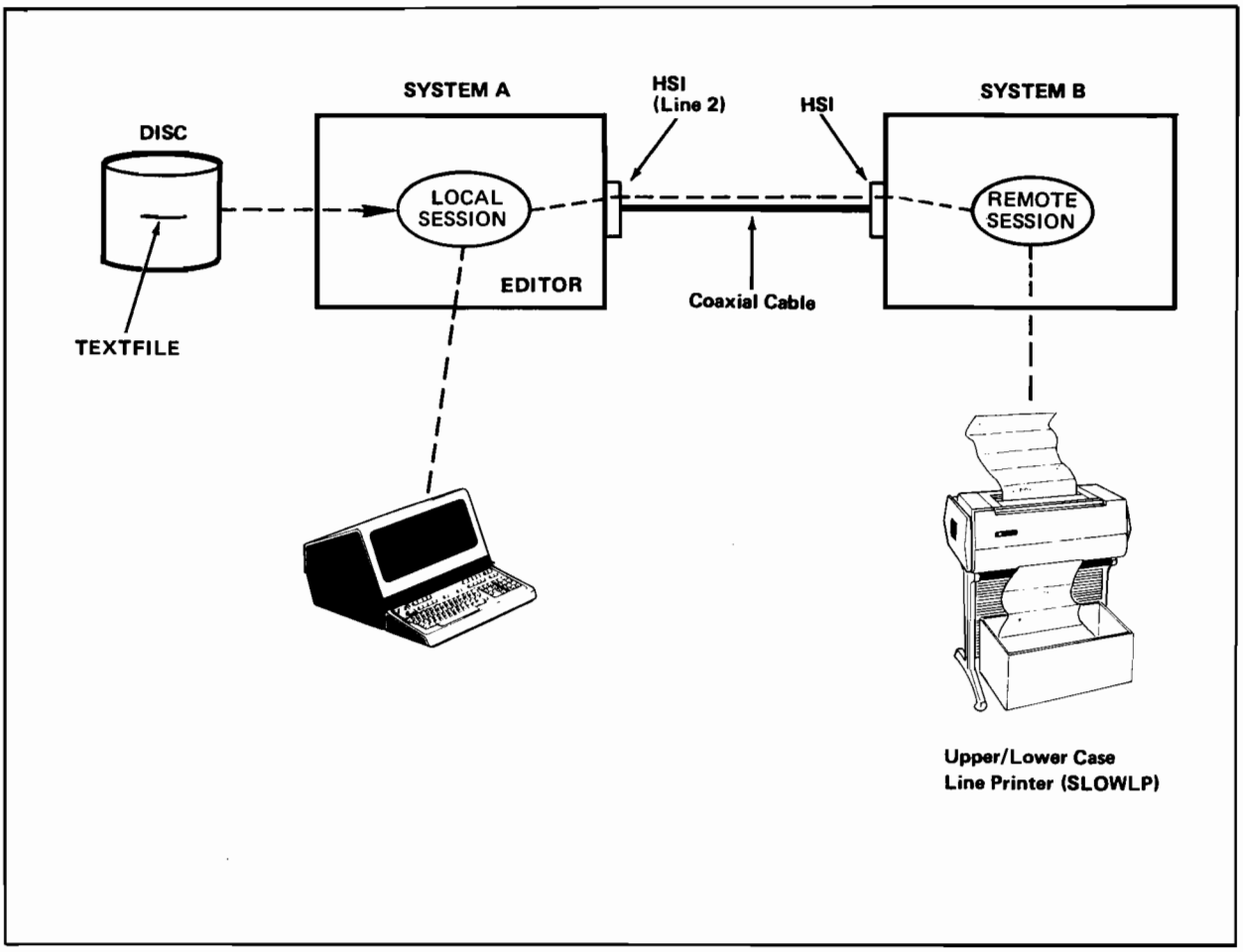


Figure 4-5. Remote Off-Line Listing Example

Example #2

Assume that there is a file named SOURCE residing on a disc connected to a remote HP 3000 and that SOURCE contains a list of clients sorted alphabetically by the clients' names. Let us assume further that the remote HP 3000 and our local HP 3000 both have DS capability and that they are connected to one another by a coaxial cable and HSIs. We wish to access the remote file SOURCE from our local HP 3000, sort its content alphabetically by tne names of the states in which the clients reside, and store the sorted version in a newly created disc file named SORTED on our local HP 3000. We can do that (without disturbing the original content of SOURCE) as follows.

First we log on to our local HP 3000 and establish a communications link with the remote HP 3000.

```
:HELLO USER.ACCOUNT
:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2
```

where USER and ACCOUNT are valid user and account names (respectively) within the accounting structure of our local HP 3000, RUSER and RACCOUNT are valid user and account names (respectively) within the accounting structure of the remote HP 3000, and LINE2 is the device class name of the local HSI to which the coaxial cable is connected.

Next we issue a local MPE BUILD command to create the local disc file SORTED that will receive the sorted output.

```
:BUILD SORTED;DISC=250,1,1;REC=-80,16,F,ASCII
```

Then we issue two local MPE FILE commands: one that defines the remote disc file SOURCE as the sort input file and one that defines the local disc file SORTED as the sort output file.

```
:FILE INPUT=SOURCE;DEV=LINE2#DISC
:FILE OUTPUT=SORTED
```

Then we invoke the Sort program, specify the sort key, and initiate the actual sort.

```
:RUN SORT.PUB.SYS
>KEY 50,9
>END
```

Note that the sort is performed in our local HP 3000 using the remote disc file SOURCE as the sort input file, the output of the sort is stored in the local disc file SORTED, and the original content of SOURCE is not altered.

The entire command sequence described above is as follows (refer to Figure 4-6):

```
:HELLO USER.ACCOUNT
 SESSION NUMBER = #S98
 TUE, AUG 3, 1976, 12:51 PM
 HP32002A.00.A1

:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2

DS LINE NUMBER = #L3

 SESSION NUMBER = #S17
 TUE,AUG 3, 1976, 12:52 PM
 HP32002A.00.A1

:BUILD SORTED;DISC=250,1,1;REC=-80,16,F,ASCII
:FILE INPUT=SOURCE;DEV=LINE2#DISC
:FILE OUTPUT=SORTED
:RUN SORT.PUB.SYS

HP32214B.00.00 SORT/3000  TUE, AUG 3, 1976,  12:53 PM

>KEY 50,9
>END

                    STATISTICS

NUMBER OF RECORDS =                          221
RECORD SIZE (IN BYTES) =                      80
NUMBER OF INTERMEDIATE PASSES =                0
```

```
SPACE AVAILABLE (IN WORDS) =                13,346
NUMBER OF COMPARES =                            45
NUMBER OF SCRATCHFILE IO'S =                    10
CPU TIME (MINUTES) =                           .01
ELAPSED TIME (MINUTES) =                       .14

 END OF PROGRAM
:
      .
      .
      .
```

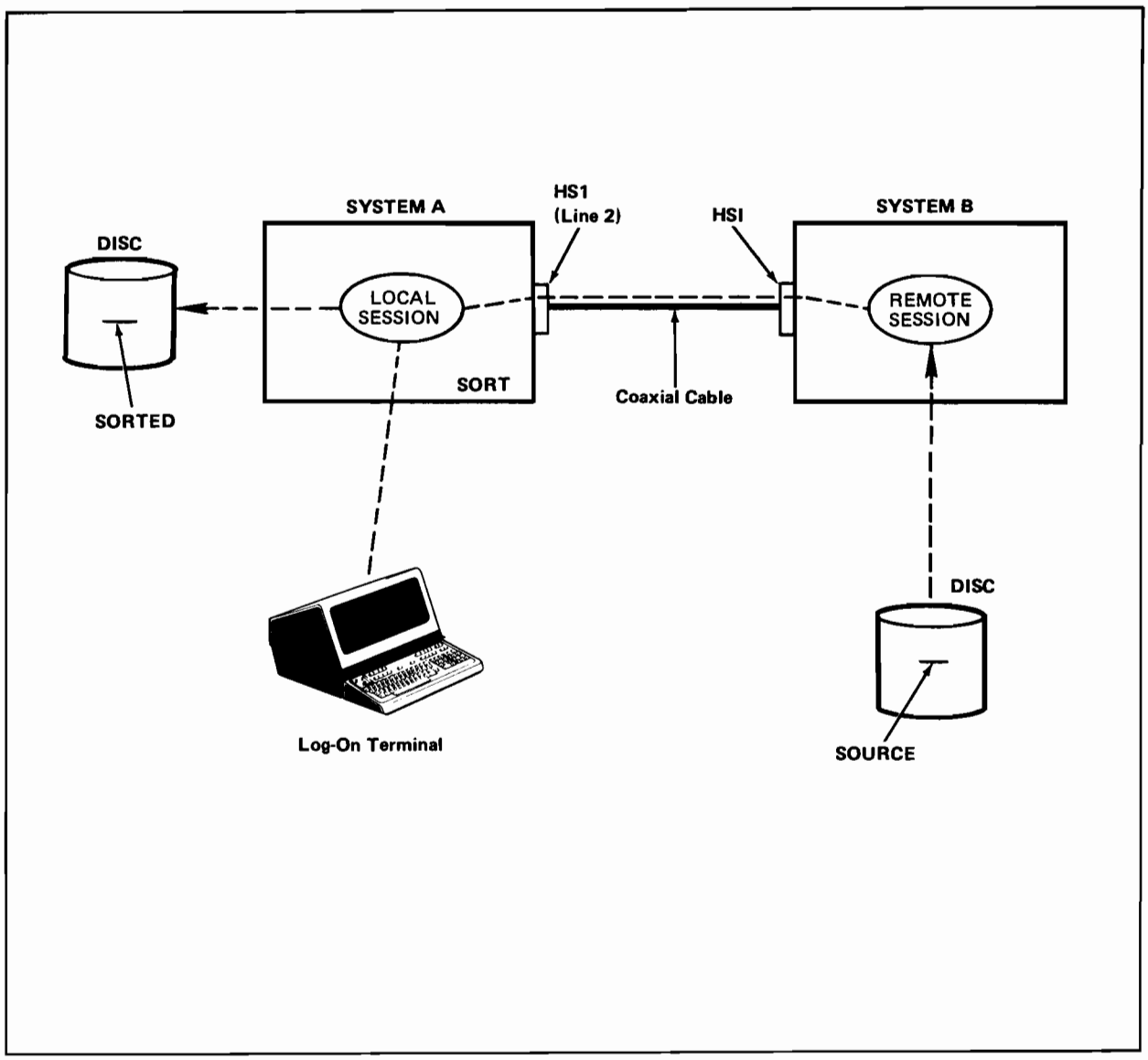


Figure 4-6. SORT Remote File Access Example

Example #3

Suppose that we wish to copy a disc file from our local HP 3000 to a remote HP 3000. Let us assume that the remote HP 3000 and our local HP 3000 both have DS capability and that they are connected to one another by a coaxial cable and HSIs. We can perform the file copy operation as follows.

First we log on to our local HP 3000 and establish a communications link with the remote HP 3000.

```
:HELLO USER.ACCOUNT
:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2
```

where USER and ACCOUNT are valid user and account names (respectively) within the accounting structure of our local HP 3000, RUSER and RACCOUNT are valid user and account names (respectively) within the accounting structure of the remote HP 3000, and LINE2 is the device class name of the local HSI to which the coaxial cable is connected.

Next we issue a local MPE FILE command defining the destination file (REMFILE) as being a remote disc file.

```
:FILE REMFILE,NEW;DEV=LINE2#DISC
```

Then we invoke the File Copier and specify the file copy parameters.

```
:RUN FCOPY.PUB.SYS
>FROM=LOCFILE;TO=*REMFILE
```

A new disc file named REMFILE is created in the public group of the RACCOUNT account in the remote HP 3000 and the content of the local disc file LOCFILE is then copied over the communications line into REMFILE.

The entire command sequence described above is as follows (refer to Figure 4-7):

```
:HELLO USER.ACCOUNT
 SESSION NUMBER = #S98
 TUE, AUG 3, 1976, 12:51 PM
 HP320022A.00.A1

:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2

DS LINE NUMBER = #L3

 SESSION NUMBER = #S17
 TUE, AUG 3, 1976, 12:52 PM
 HP32002A.00.A1

:FILE REMFILE;DEV=LINE2#DISC
:RUN FCOPY.PUB.SYS

HP32212A.00.04 FILE COPIER

>FROM=LOCFILE;TO=*REMFILE;NEW
EOF FOUND IN FROMFILE AFTER RECORD 2017

2018 RECORDS PROCESSED *** 0 ERRORS

>EXIT

END OF PROGRAM

    .
    .
    .
```

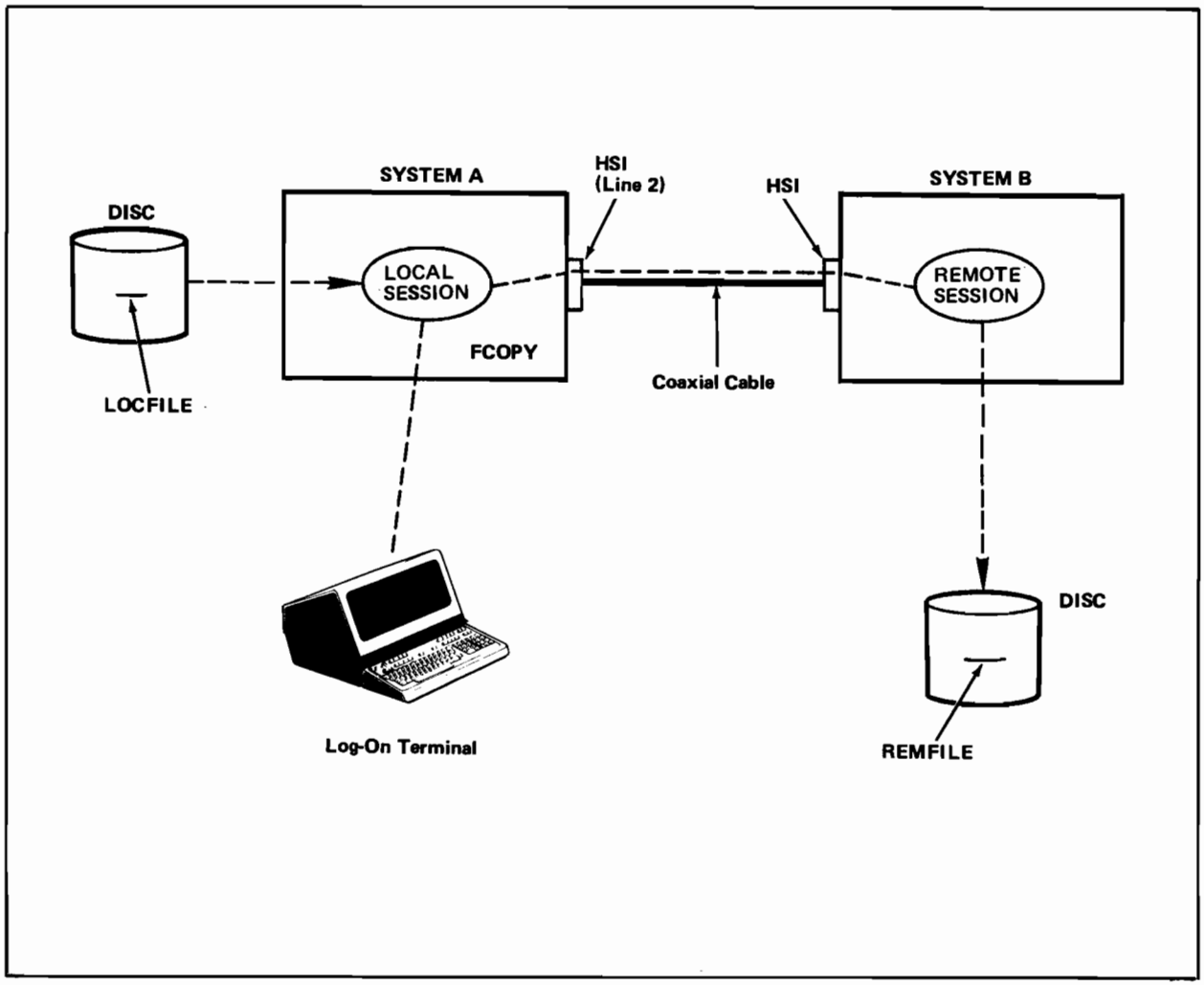


Figure 4-7. FCOPY Remote File Access Example

Example #4

Assume that there is a COBOL source file named SOURCE1 residing on a disc connected to a remote HP 3000 and that we wish to compile, prepare, and execute that program on our local HP 3000. Let us assume further that the remote HP 3000 and our local HP 3000 both have DS capability and that they are connected to one another by a coaxial cable and HSIs. We can locally perform the compile, prepare, and execute of the remote source file as follows.

First we log-on to our HP 3000 and establish a communications link with the remote HP 3000.

```
:HELLO USER.ACCOUNT
:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2
```

where USER and ACCOUNT are valid user and account names (respectively) within the accounting structure of our local HP 3000, RUSER and RACCOUNT are valid user and account names (respectively) within the accounting structure of the remote HP 3000, and LINE2 is the device class name of the local HSI to which the coaxial cable is connected.

Next we issue a local MPE FILE command defining the file SOURCE1 as being a remote disc file.

```
:FILE SOURCE1;DEV=LINE2#DISC
```

where LINE2 is the device class name we used when establishing the communications link and DISC is the device class name (as established within the remote HP 3000) of the disc on which SOURCE1 resides.

Then we invoke the COBOL compiler and the Segmenter of our local HP 3000 specifying the remote disc file SOURCE1 as the inputfile.

```
:COBOLGO *SOURCE1
```

The content of the remote disc file SOURCE1 is compiled, prepared, and executed in our local HP 3000.

The entire command sequence described above is as follows (refer to Figure 4-8):

```
:HELLO USER.ACCOUNT
 SESSION NUMBER = #S98
 TUE, AUG 3, 1976, 12:51 PM
 HP32002A.00.A1

:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2

DS LINE NUMBER = #L3

 SESSION NUMBER = #S17
 TUE, AUG 3, 1976, 12:52 PM
 HP32002A.00.A1

:FILE SOURCE1;DEV=LINE2#DISC
:COBOLGO *SOURCE1

PAGE 0001    HP322130.01.00  (C) HEWLETT-PACKARD CO. 1976

           (SOURCE1 is now being compiled.)

    DATA AREA IS  000341 WORDS.
    CPU TIME = 0:00:01.  WALL TIME  == 0:00:07.
END COBOL/3000 COMPILATION.  NO ERRORS.  NO WARNINGS.

 END OF COMPILE

  (The compiled version of SOURCE1 is now being
   prepared by the MPE Segmenter.)

 END OF PREPARE

  (The compiled and prepared version of SOURCE1 is
   now being executed.)

 END OF PROGRAM
:
```

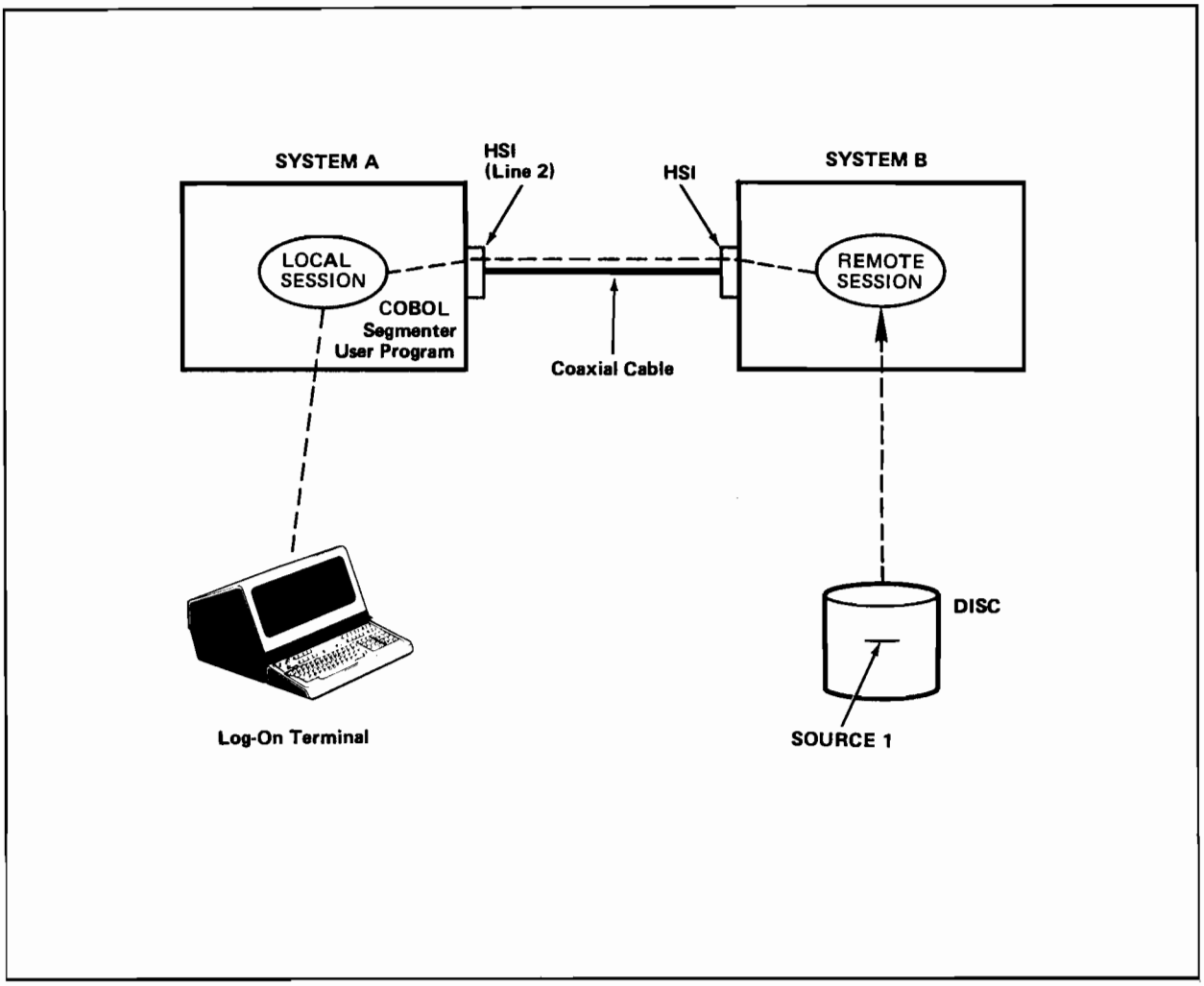


Figure 4-8. COBOLGO Remote File Access Example

Example #5

Assume that there is a COBOL source program named SOURCE1 residing on a disc connected to a remote HP 3000 and that we wish to incorporate changes into the content of that file from a local file named CHANGES, compile the updated source code on our local HP 3000, and store a copy of the updated source code in a new file named SOURCE1A on the disc connected to the remote HP 3000. Let us assume further that the remote HP 3000 and our local HP 3000 both have DS capability and that they are connected to one another by a coaxial cable and HSIs. We can perform the update and compilation as follows:

First we log-on to our HP 3000 and establish a communications link with the remote HP 3000.

```
:HELLO USER.ACCOUNT
:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2
```

where USER and ACCOUNT are valid user and account names (respectively) within the accounting structure of our local HP 3000, RUSER and RACCOUNT are valid user and account names (respectively) within the accounting structure of the remote HP 3000, and LINE2 is the device class name of the local HSI to which the coaxial cable is connected.

Next we issue two local MPE FILE commands: one that defines the source file SOURCE1 as being a remote disc file and one that defines the file SOURCE1A as a new remote disc file.

```
:FILE SOURCE1;DEV=LINE2#DISC
:FILE SOURCE1A,NEW;SAVE;DEV=LINE2#DISC
```

where LINE2 is the device class name we used when establishing the particular communications link, DISC is the device class name (as established within the remote HP 3000) of the disc on which SOURCE1 resides and SOURCE1A will reside, and NEW;SAVE specifies that SOURCE1A is to be a new permanent file.

Then we invoke the COBOL compiler of our local HP 3000 specifying the local disc file CHANGES as the update input file (textfile), the remote disc file SOURCE1 as the source input file (masterfile), and the remote disc file SOURCE1A as the updated source file (newfile).

```
:COBOL CHANGES,,,*SOURCE1,*SOURCE1A
```

The source code in the remote disc file SOURCE1 is updated by the content of the local disc file CHANGES, a new permanent disc file named SOURCE1A is created in the remote HP 3000, and the resultant source code is stored in the remote disc file SOURCE1A. Note that the updating operation is performed by the COBOL compiler in our local HP 3000.

The entire command sequence described above is as follows (refer to Figure 4-9):

```
:HELLO USER.ACCOUNT
 SESSION NUMBER = #S98
 TUE, AUG 3, 1976, 12:51 PM
 HP320022A.00.A1

:REMOTE HELLO RUSER.RACCOUNT;DSLINE=LINE2

DS LINE NUMBER = #L3

 SESSION NUMBER = #S17
 TUE, AUG 3, 1976, 12:52 PM
 HP32002A.00.A1

:FILE SOURCE1;DEV=LINE2#DISC
:FILE SOURCE1A,NEW;SAVE;DEV=LINE2#DISC
:COBOL CHANGES,,,*SOURCE1,*SOURCE1A

PAGE 0001    HP322130.01.00  (C) HEWLETT-PACKARD CO. 1976

     (SOURCE1 is now being updated and compiled.)

    DATA AREA IS  000341 WORDS.
    CPU TIME = 0:00:01.  WALL TIME = 0:00:17.
END COBOL/3000 COMPILATION.  NO ERRORS.  NO WARNINGS.

 END OF COMPILE
:
     .
     .
     .
```

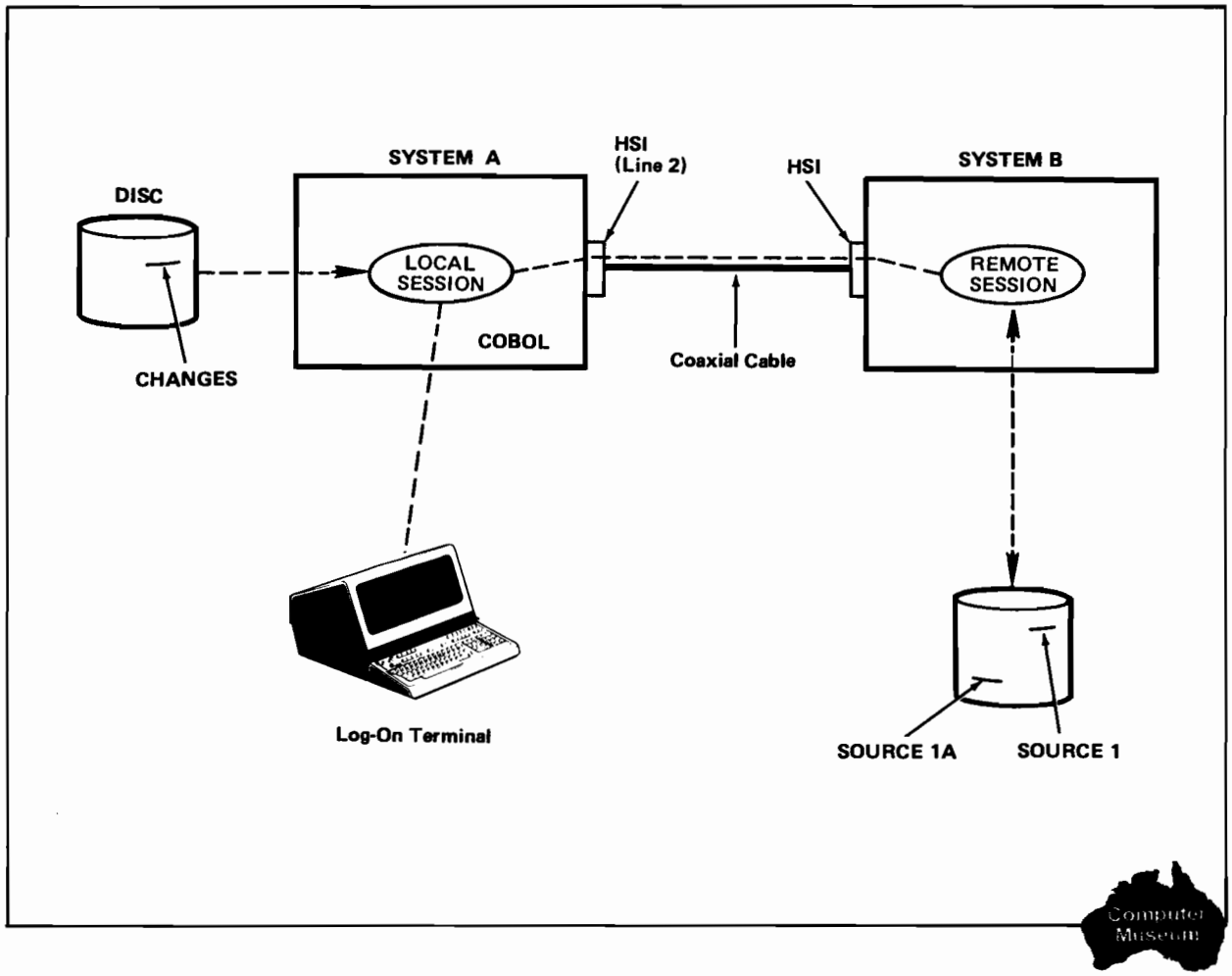


Figure 4-9. COBOL Remote File Access Example

PROGRAMMATIC ACCESS

Once a DS/3000 communications link has been established between your HP 3000 and a remote HP 3000, you can use the standard set of MPE File System intrinsics within your local programs to access devices and/or files residing at the remote HP 3000 site. To make this possible, the format of the byte array referenced by the device parameter of the MPE FOPEN intrinsic has been expanded to include a DS line specification in addition to the usual device specification. The format of that byte array is as follows:

```
dsdevice#device
```

where dsdevice is the device class name or logical device number that you used when establishing the particular communications link (this specifies the physical line connecting the two computers) and device is the device class name or logical device number of the desired remote device as established within the remote HP 3000.

For completeness, the full syntax and parameter specifications for the MPE FOPEN intrinsic are presented in figures 4-10 and 4-11 below. You will notice that the only difference between this presentation and the one in the MPE Intrinsics Reference Manual is the addition of "dsdevice#" to the format of the byte array referenced by the device parameter. As with the FILE command, this one small format change has enormously powerful implications. It means that programs executing in your local HP 3000 can easily access any of the devices and/or disc files of a remote HP 3000 as though they resided at your local HP 3000 site. Access to remote disc files is, of course, subject to the usual MPE file security. The user, account, and group names that you specified in the REMOTE HELLO command when establishing the communications link are the ones used by MPE in the remote HP 3000 for determining your file access capabilities.

Following figures 4-10 and 4-11 are two annotated examples illustrating remote device and file access from a local program running within a local session.

The Condition Codes for the various MPE File System intrinsics retain their normal meanings. Any communications line errors will return a CCL. In the event of an error, you can call the MPE FCHECK intrinsic to determine what happened. When using the MPE File System intrinsics for remote file access, the Message Block B (File System) error codes apply to the remote file. You may also use the MPE PRINTFILEINFO intrinsic to display the status of a remote file.

```
      I                          BA        LV        LV        IV
filenum:=FOPEN(formaldesignator,foptions,aoptions,recsize,

                    BA       BA          IV           IV
                device,formmsg,userlabels,blockfactor,

                         IV        DV          IV          IV
                numbuffers,filesize,numextents,initalloc,

                       IV     O-V
                filecode);
```

Figure 4-10. MPE FOPEN Intrinsic Syntax

formaldesignator *byte array (optional)*
Contains a string of ASCII characters interpreted as a formal file designator, as defined in Section III. 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 the string names a system-defined file, it can begin with a dollar sign ($); if it names a user-predefined file, it can begin with an asterisk (*).
Default: A temporary nameless file that can be read from or written to, but not saved, is assigned.

foptions *logical by value (optional)*
The *foptions* parameter allows you to specify six different file characteristics, by setting corresponding bit groupings in a 16-bit word. The correspondence is from right to left, beginning with bit 15. These characteristics are as follows, proceeding from the rightmost bit groups to the leftmost bit groups in the word. The bit settings are summarized in figure 2-1.

NOTE

Bit groups are denoted using the standard SPL notation. Thus bits (14:2) indicates bits 14 and 15; bits (10:3) indicates bits 10, 11, and 12.

Bits (14:2) — Domain *Foption.*
The file domain to be searched by MPE to locate the file, indicated by these bit settings:
00 = The file is a *new* file, created at this point. No search is necessary.
01 = The file is an old permanent file, and the system file domain should be searched.
10 = The file is an old temporary file, and the job file domain should be searched.
11 = The file is an old file that is to be located by first searching the job file domain and then, if the file is not found, by searching the system file domain.

Figure 4-11. MPE FOPEN Intrinsic Parameters

Bit (13:1) — ASCII/Binary *Foption.*
The code (ASCII or binary) in which a *new* file is to be recorded when it is written to a device that supports both codes. In the case of disc files, this also affects padding that can occur when a direct-write intrinsic call (FWRITEDIR) is issued to a record that lies beyond the current logical end-of-file indicator. In ASCII files, any dummy records between the previous end-of-file and the newly-written record are padded with blanks. In binary files, such records are padded with binary zeros. All files not on disc are treated as ASCII files.
For ASCII files, this bit is 1.
For binary files, this bit is 0.

Bits (10:3) — Default File Designator *Foption.*
The *actual* file designator is equated with the formal file designator specified in FOPEN, if

1. No explicit or implicit :FILE command equating the formal file designator to a different actual file designator occurs in the job or session; *or*
2. The Disallow File Equation *Foption* (bit 5) is specified.

The bit settings are
000 = The actual file designator is the same as the formal file designator.
001 = The actual file designator is $STDLIST.
010 = The actual file designator is $NEWPASS.
011 = The actual file designator is $OLDPASS.
100 = The actual file designator is $STDIN.
101 - The actual file designator is $STDINX.
110 - The actual file designator is $NULL.

Bits (8:2) — Record Format *Foption.*
The format in which the records in the file are recorded, indicated by these bit settings:
00 = Fixed-length records. The file is composed of logical records of uniform length.
01 = Variable-length records. The file contains logical records of varying length. This format is restricted to records that are written in sequential order. The size of each record is recorded internally. Specifically, undefined-length records are supported by all devices; fixed- and variable-length records are supported by disc and

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

magnetic tape devices only. To state this another way: disc and magnetic tape devices support *all* record formats, whereas all other devices support *only* undefined-length records. The actual record size used is determined by multiplying the *recsize* (specified or default) by the *blockfactor*, and adding two words reserved for system use. This option is not allowed when NOBUF is specified. In such a case, the record format used is undefined-length records, discussed below.

10 = Undefined-length records. The file contains records of varying length that were not written using the variable-length *foption* (01). All files not on disc or magnetic tape are treated as containing undefined-length records by default.

Bit (7:1) — Carriage Control *Foption.*
If selected, this specifies that you will supply a carriage control directive in the calling sequence of each FWRITE call that writes records onto the file.
0 = No carriage control directive expected.
1 = Carriage control directive expected.
Carriage control is defined only for character oriented, i.e., ASCII, files. This option and binary are mutually exclusive and attempts to open new files with both binary and this option results in an access violation.

This option is a physical attribute of the file and its state cannot be modified when opening an old disc file.

A carriage control character passed through the *control* parameter of FWRITE is recognized and acted upon only for files for which carriage control is specified in FOPEN. Embedded control is treated strictly as data on files for which no carriage control is specified, and does not invoke spacing for such files. You may specify spacing action on files for which carriage control has been specified, either by embedding the control in the record, indicated with a *control* parameter of one in the call to FWRITE, or by sending the control code directly via the *control* parameter of FWRITE.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

A carriage control character sent to a file on which the control cannot be executed directly, for example, line spacing to a disc or tape file, will result in having the control character embedded as the first byte of the record. Thus, the first byte of each record in a disc file having a carriage control character contains control information. Control sent to other types of files results in transmission of the control to the driver.

The control codes %400 through %403 are remapped to %100 through %103 so that they fit into one byte and thus can be embedded. Records written to the line printer with one of the above controls should not contain information other than control information.

For the purpose of computing record size, carriage control information is considered by the file system to be part of the data record. As such, specifying the carriage control option adds one byte to the record size at the time the file is created. For example, a specification of REC = -132, 1,F,ASCII;CCTL results in a *recsize* of 133 characters.

You always may read up to and including the *recsize* as returned by FGETINFO. On writes of files for which carriage control is specified, however, the data transferred is limited to *recsize* – 1 unless a control of one is passed indicating the data record is prefixed with embedded control.

Bit (6:1) — Reserved for MPE. Should be set to zero.

Bit (5:1) Disallow File Equation *Foption.*
This option ignores any corresponding :FILE command, so that the specifications in the FOPEN call take effect (unless preempted by those in the file label, for disc files).
0 = Allow :FILE.
1 = Disallow :FILE.

Bits (0:5) — Reserved for MPE. Should be set to zero.
Default: All bits are set to zero.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

aoptions — *logical by value (optional)*

The *aoptions* parameter permits you to specify up to seven different access options established by bit groupings in a 16-bit word. These access options are described below. The bit settings are summarized in figure 2-2.

Bits (12:4) — Access Type *Aoptions.*
The type of access allowed for this access of this file:

0000 = Read access only. The FWRITE, FUPDATE, and FWRITEDIR intrinsic calls cannot reference this file.

0001 = Write access only. Any data written in the file prior to the current FOPEN request is deleted. The FREAD, FREADSEEK, FUPDATE and FREADDIR intrinsic calls cannot reference this file.

0010 = Write access only, but previous data in the file is *not* deleted. The FREAD, FREADSEEK, FUPDATE, and FREADDIR intrinsic calls cannot reference this file.

0011 = Append access only. The FREAD, FREADDIR, FREADSEEK, FUPDATE, FSPACE, FPOINT, and FWRITEDIR intrinsic calls cannot reference this file. This option is not valid for files containing variable-length records.

0100 = Input/output access. Any file intrinsic except FUPDATE can be issued for this file.

0101 = Update access. All file intrinsics, including FUPDATE, can be issued for this file.

0110 = Execute access. Allows users with Privileged Mode Capability input/output access to any loaded file.

Bit (11:1) — Multirecord *Aoption.*
Signifies that individual read or write requests are not confined to record boundaries. Thus, if the number of words or bytes to be transferred (specified in the *tcount* parameter of the read or write request) exceeds the size of the physical record (i.e., block) referenced, the remaining words or bytes are taken from subsequent successive records until the number specified by *tcount* have been transferred. This option is available only if the inhibit buffering *aoption*, described below, is selected also.

0 = Non-multirecord mode.

1 = Multirecord mode.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

Bit (10:1) — Dynamic Locking *Aoption.*
Indicates that you want to use the FLOCK and FUNLOCK intrinsics to dynamically permit or restrict concurrent access to the file by other processes at certain times. The user process can continue this temporary locking/unlocking until it closes the file. Dynamic locking/unlocking is made possible through a Resource Identification Number (RIN) assigned to the file and temporarily acquired by the FOPEN intrinsic. The calling process and other processes must use the RIN in cooperation to guarantee the integrity of the file, as discussed in Section III. Non-cooperating processes are allowed concurrent access at all times, unless other provisions prohibit this.
0 = Disallow dynamic locking/unlocking.
1 = Allow dynamic locking/unlocking. A file may be multiple accessed only if all FOPEN requests for the file specify dynamic locking, or if none of them do. An FOPEN request that disagrees with the current access, if any, will fail.

Bits (8:2) — Exclusive *Aoption.*
This *aoption* specifies whether you have continuous exclusive access to this file, from the time it is opened to the time it is closed. This option often is used when performing some critical operation, such as updating the file.
01 = Exclusive access. After this file is opened, prohibits another FOPEN request, whether issued by this or another process, until this process issued the FCLOSE request or terminates. If any process already is accessing this file when this FOPEN call is issued, a CCL error code is returned to the calling process. If another FOPEN call is issued for this file while the exclusive *aoption* is in effect, an error code is returned to that calling process. The exclusive access *aoption* can be requested only by users allowed the file locking access mode by the security provisions for the file.
10 = Semi-exclusive access. After the file is opened, prohibits concurrent output access to this file through another FOPEN request, whether issued by this or another process, until this process issues the FCLOSE request or terminates. A subsequent request for the input/output or update *aoption* access type will obtain read only access. Other types of read access, however, are allowed. If any process already has output access to the file when this FOPEN call

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

is issued, a CCL error code is returned to the calling process. If another FOPEN call that violates the read-only restriction is issued while the semi-exclusive *aoption* is in effect, that call fails and an error code is returned to the calling process. The semi-exclusive access can be requested only by users allowed the file-locking access mode by the security provisions for the file.

11 = Share access. After the file is opened, permits concurrent access to this file by any process, in any access mode, subject to other basic MPE security provisions in effect.

00 = Default value. If the read access only *aoption* is selected, share access (11) takes effect. Otherwise, exclusive access (01) takes effect.

Bit (7:1) — Inhibit Buffering *Aoption.*
When selected, this *aoption* inhibits automatic buffering by MPE and allows input/output to take place directly between the user's data area and the applicable hardware device.
0 = Allow normal buffering.
1 = Inhibit buffering (NOBUF).
NOBUF access is oriented to the transfer of physical blocks rather than logical records.

With NOBUF access, you have responsibility for blocking and de-blocking of records in the file (see Section III). To be consistent with files built using buffered I/O, records should begin on word boundaries, and when the information content of the record is less than the defined record length, the record should be padded with blanks by you if the file is ASCII or with zeros if the file is binary.

The *recsize* and block size for files manipulated under NOBUF access follow the same rules as those files that are created using buffering. The default *blockfactor* for a file created under NOBUF is one.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

When a NOBUF file is opened without multirecord access, the amount of data transferred per read or write is limited to a maximum of one block.

The end-of-file, next record pointer, and record transfer count are maintained in terms of logical records for all files. The number of logical records affected by each transfer is determined from the size of the transfer.

Transfers always begin on a block boundary. Those transfers which do not transfer whole blocks leave the next record pointer set to the first record in the next block. The end-of-file pointer always points at the last record in the file.

For files opened with NOBUF access, the FREADDIR, FWRITEDIR, and FPOINT intrinsics treat the *recnum* parameter as a block number.

Bit (6:1) — Multi-Access Mode *Aoption.*
When selected, this *aoption* provides the accessor with a means of sharing access to the file, including file system buffers. Thus, the accessor can "sequentially" access records within the file in conjunction with other such accessors in the same job. This option is not allowed if the file is accessed exclusively, if NOBUF is selected, or if the multirecord option is requested.

Bit (5:1) — Reserved for MPE. Should be set to zero.

Bit (4:1) — No-Wait I/O *Aoption.*
The selection of this *aoption* allows you to initiate an I/O request and to have control returned before the completion of the I/O. The IOWAIT intrinsic must be called after each I/O request to confirm the completion of the I/O. Also, multirecord access is not available.

NOTE

You must be running in Privileged Mode to select No-Wait I/O and NOBUF.

Bits (0:3) — Reserved for MPE. Should be set to zero.
Default: All bits are set to zero.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

recsize *integer by value (optional)*
An integer indicating the size of the logical records in the file. If a positive number, this represents words; bytes are represented by a negative number. If the file is a newly-created file, this value is recorded permanently in the file label. If the records in the file are of variable length, this value indicates the maximum logical record length allowed.

Binary files are word oriented. A record size specifying an odd byte count for a binary file is rounded up by FOPEN to the next highest even number.

ASCII files may be created with logical records which are an odd number of bytes in length. Within each block, however, records begin on word boundaries.

For either ASCII or binary files with fixed or undefined length records, the record size is rounded up to the nearest word boundary. For example, a *recsize* specified as -106 for an ASCII file is 106 characters (53 words) in length. A *recsize* of -113 for a binary file is 114 characters (57 words) in length. The rounded sizes should be used in computations for *blockfactor* or block size.
Default: The default value is the configured physical record width of the associated device.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

device

byte array (optional)
Contains a string of ASCII characters terminating with any non-alphanumeric character except a slash or period, designating a local or remote device on which the file is to reside. For a local device the string may represent a device class name up to eight alphanumeric characters beginning with a letter or a logical device number consisting of a three-byte numeric string. For a remote device the string may represent a DS line identifier (the device class name or logical device number you used when establishing the particular communications link) followed by a # followed by the device class name or logical device number of the desired remote device. Device class names and logical device numbers are defined and assigned to devices and communications interfaces during system configuration.

The format of the array referenced by *device* is as follows:

dsdevice#device

where *dsdevice* is the device class name or logical device number that you used when establishing the particular communications link (this specifies the physical line connecting the two computers) and *device* is the device class name or logical device number of the desired remote device as established within the remote HP 3000.

If the file is a newly-created disc file and the device specification is a device class, then all extents of the file are restricted to members of the class. Similarly, if the device specification is a logical device number, then all extents are restricted to the specified logical device.
Default: Local DISC and remote DISC.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

formmsg — *byte array (optional)*
Contains a forms message that can be used for such purposes as telling the console operator what type of paper to use in the line printer. This message must be displayed to the operator and verified before this file can be printed on a line printer. The message itself is a string of ASCII characters terminated by a period. The maximum number of characters allowed in the array is 49, including any terminating period. Arrays with more than 49 characters are truncated by MPE.
Default: No forms message is available.

userlabels — *integer by value (optional)*
An integer specifying the number of user-label records to be written for this file.
Default: The default number of user-label records is zero.

blockfactor — *integer by value (optional)*
An integer containing the size of each buffer to be established for the file, specified as a number equal to the number of logical records per block. For fixed-length records, *blockfactor* is the actual number of records in a block. For variable-length records, *blockfactor* is interpreted as a multiplier used to compute the block size (maximum *recsize* x *blockfactor*). For undefined-length records, *blockfactor* is always one logical record per block. The *blockfactor* value specified by you may be overridden by MPE. The valid range for *blockfactor* is from 1 through 255. Specification of a negative or zero value results in the default *blockfactor* setting. Values greater than 255 are defaulted to *blockfactor* modulo 256. *Blockfactor* establishes the *physical* record size on disc and magnetic tape files.
Default: 1.

numbuffers — *integer by value (optional)*
A 16-bit word whose bits specify the following:

Bits (11:5) — Number of Buffers.
Specifies the number of buffers to be allocated to the file. This parameter is not used for files representing interactive terminals, since a system-managed buffering method is always used in such cases. If

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

omitted, set to zero, or set to a negative number, the default value of 2 is set by MPE.

Bits (4:7) — Number of Copies.
For spooled output devices, specifies the number of copies of the entire file to be produced by the spooling facility. This can be specified for a file already FOPENed (for example, $STDLIST), in which case the highest value supplied before the last FCLOSE will take effect. The copies do not appear contiguously if the console operator intervenes or if a file of higher *outputpriority* becomes READY before the last copy is complete. This parameter is ignored for non-spooled output devices. The default value is 1.

Bits (0:4) — Output Priority.
Specifies the *outputpriority* to be attached to this file. This priority is used to determine the order in which files are produced when several are waiting for the same device. This parameter must be a number between 1 (lowest priority) and 13 (highest priority), inclusive. If this value is less than the current output fence set by the console operator, file printing/punching is deferred until the operator raises the *outputpriority* of the file or lowers the output fence. This parameter can be specified for a file already FOPENed (for example, $STDLIST), in which case the highest value supplied before the last FCLOSE takes effect. This parameter is ignored for non-spooled devices. The default value is 8.
Default: The default values of all bit groupings are taken.

filesize — *double by value (optional)*
A double-word integer (as defined in SPL) specifying the maximum file capacity in terms of logical records for files containing variable-length and undefined-length records, and logical records for files containing fixed-length records. A zero or negative value results in the default *filesize* setting. The maximum capacity allowed is over two million (2^{21}) sectors. The number of sectors in a file is found by the formula shown under FILE CHARACTERISTICS in Section III of the MPE Intrinsic Reference Manual.
Default: 1023 logical records.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

numextents — *integer by value (optional)*
An integer specifying the number of extents (integral number of contiguously-located disc sectors) that can be dynamically allocated to the file as logical records are written to it. The size of each extent is determined by the *filesize* parameter value divided by the *numextents* parameter value. If specified, *numextents* must be an integer from 1 to 32. A zero or negative value results in the default setting.
Default: 8 extents.

NOTE

Extents are allocated on any disc in the device class specified in the *device* parameter when the file was created. If it is necessary to insure that all extents of a file are on a particular disc, a single disc device class or a logical device number must be used in the *device* parameter.

initialloc — *integer by value (optional)*
An integer specifying the number of extents to be allocated to the file when it is opened. This must be an integer from 1 to 32. If an attempt to allocate the requested disc space fails, the FOPEN intrinsic returns an error condition code to the calling program.
Default: 1 extent.

filecode — *integer by value (optional)*
An integer recorded in the file label and made available for general use to anyone accessing the file through the FGETINFO intrinsic. This parameter is used for new files only. For this parameter, any user can specify a positive integer ranging from 0 to 1023. If your process is running in privileged mode, you can specify a negative integer for *filecode* when initially opening a file. Then, any future accesses of the file must be requested in privileged mode and also must specify the correct *filecode*. Certain positive integers beyond 1023 have particular HP-defined meanings, as follows:

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

Integer	Meaning
-400	An IMAGE root file.
-401	An IMAGE data set.
1024	A USL file.
1025	A BASIC data file.
1026	A BASIC program file.
1027	A BASIC fast program file.
1028	A relocatable library (RL) file.
1029	A program file.
1030	A STAR file.
1031	A segmented library (SL) file.
1040	A Cross Loader ASCII file (SAVE).
1041	A Cross Loader relocated binary file.
1042	A Cross Loader ASCII file (DISPLAY).
1050	An EDIT KEEPQ file (non-COBOL).
1051	An EDIT KEEPQ file (COBOL).
1052	An EDIT TEXT file (COBOL).
1060	An RJE punch file.
1069	Reserved.
1070	A QUERY procedure file.
1071 1072	QUERY work files.
1080	Reserved.

Default: 0.

Figure 4-11. MPE FOPEN Intrinsic Parameters (Continued)

Example

The following programs will show you how remote files can be accessed by using basic remote file intrinsics to pass files between a local HP 3000 and a remote HP 3000.

```
$CONTROL USLINIT,ADR,MAP,NOWARN,CODE


BEGIN

INTEGER
  A,
  I:=-1,
  RDISKNUM,
  RLPNUM;


BYTE ARRAY RMTLP'FILNAM(0:3):="RLP ";
BYTE ARRAY RLPDEV(0:11);
BYTE ARRAY RMTDISK'FILNAM(0:5):="RDISK ";
BYTE ARRAY MSG(0:71);
BYTE ARRAY RDISKDEV(0:11);

LOGICAL ARRAY LMSG(*)=MSG;


INTRINSIC PRINT,READ,QUIT,FOPEN,FWRITEDIR,FREADDIR,FWRITE,FCLOSE;



            <<BEGIN OUTER BLOCK>>

MOVE MSG:="INPUT REMOTE DISK DEVICE CLASS NAME";
PRINT(MSG,-35,0);
MOVE MSG:="IN THE FORM..DSDEVICE#DSKDEV";
PRINT(MSG,-28,0);
A:=READ(LMSG,-12);
MOVE RDISKDEV:=MSG,(A);


MOVE MSG:="INPUT REMOTE LP DEVICE CLASS NAME";
PRINT(MSG,-33,0);
MOVE MSG:="IN THE FORM..DSDEVICE#LPDEV";
PRINT(MSG,-28,0);
A:=READ(LMSG,-12);
MOVE RLPDEV:=MSG,(A);


MOVE MSG:="OPENING REMOTE DISK FILE";
PRINT(MSG,-24,0);
RDISKNUM:=FOPEN(RMTDISK'FILNAM,4,%104,-80,RDISKDEV);<<NEW,ASCII>>
IF <> THEN
   BEGIN
     MOVE MSG:="COULD NOT OPEN REMOTE DISK FILE";
     PRINT(MSG,-31,0);
     GO TO OUT;
   END;

MOVE MSG:="WRITING TO REMOTE DISK FILE";     <<INITIALIZE DISK FILE>>
PRINT(MSG,-27,0);
```

```
56      MOVE MSG:=" ";
57      MOVE MSG(1):=MSG(0),(70);
58
59      WHILE (I:=I +1) <10 DO
60        BEGIN
61          MOVE MSG:="REMOTE FILE ACCESS TEST";
62          FWRITEDIR(RDISKNUM,LMSG,36,DOUBLE(I));   <<RECORD TO BE WRITTEN>>
63          IF <> THEN
64            BEGIN
65              MOVE MSG:="ERROR WHEN WRITING TO REMOTE DISK";
66              PRINT(MSG,-33,0);
67              GO TO OUT;
68            END;
69        END;
70
71
72      MOVE MSG:="OPENING REMOTE LP FILE";
73      PRINT(MSG,-22,0);
74      RLPNUM:=FOPEN(RMTLP'FILNAM,4,1,,RLPDEV);
75      IF <> THEN
76        BEGIN
77          MOVE MSG:="COULD NOT OPEN REMOTE LP FILE";
78          PRINT(MSG,-29,0);
79        END;
80
81      I:=-1;                                    <<READING REMOTE DISK>>
82      WHILE (I:=I + 1)<10 DO
83        BEGIN
84          FREADDIR(RDISKNUM,LMSG,36,DOUBLE(I));
85          IF <> THEN
86            BEGIN
87              MOVE MSG:="COULD NOT READ REMOTE DISK FILE";
88              PRINT(MSG,-31,0);
89            END;
90          FWRITE(RLPNUM,LMSG,36,0);
91          IF <> THEN
92            BEGIN
93              MOVE MSG:="COULD NOT PRINT TO REMOTE LP FILE";
94              PRINT(MSG,-34,0);
95            END;
96        END;
97
98      OUT:
99      END.
```

PROGRAM-TO-PROGRAM COMMUNICATIONS

SECTION V

In the preceding chapters we have seen how you can establish communications links between several HP 3000 computers to form a telecommunications network and how you can execute programs in any of the HP 3000s from a single log-on terminal. Furthermore we have seen that programs running within any HP 3000 in the network can, under the proper circumstances, obtain access to any of the hardware or software resources available throughout the network. In a sense we already have a powerful telecommunications network capability at our disposal. There is, however, one very important feature missing, a feature that would make DS/3000 a complete teleprocessing tool:

> For most teleprocessing applications it is essential that separate user programs be able to be run simultaneously in separate computers within the network and that they be able to communicate efficiently with one another.

The DS/3000 program-to-program communications capability described in this chapter answers that need.

You might ask "Why can't the normal process handling capabilities of MPE be used for this purpose?" As you probably recall, the process handling capabilities of MPE permit a user process (referred to as the father process) to create and activate one or more son processes that then run concurrently with the father process. Father and son processes can communicate efficiently with one another through the use of the SENDMAIL and RECEIVEMAIL intrinsics, shared extra data segments, or a shared user file. You are correct, this capability would indeed do the trick. Unfortunately, however, the process handling capabilities of MPE were designed for use within a single processor. They cannot handle the intervention of a communications line between father and son processes.

It might also occur to you that the remote file access capabilities of DS/3000 provide a solution of sorts. That is true. If you look back to our definition of the need for program-to-program communication, however, you will notice that the phrase "communicate efficiently" is the key to the entire definition. The following example illustrates how the DS/3000 remote file access capabilities could be used for program-to-program communication and also drammatically points out why this approach is generally unacceptable. Refer to figure 5-1.

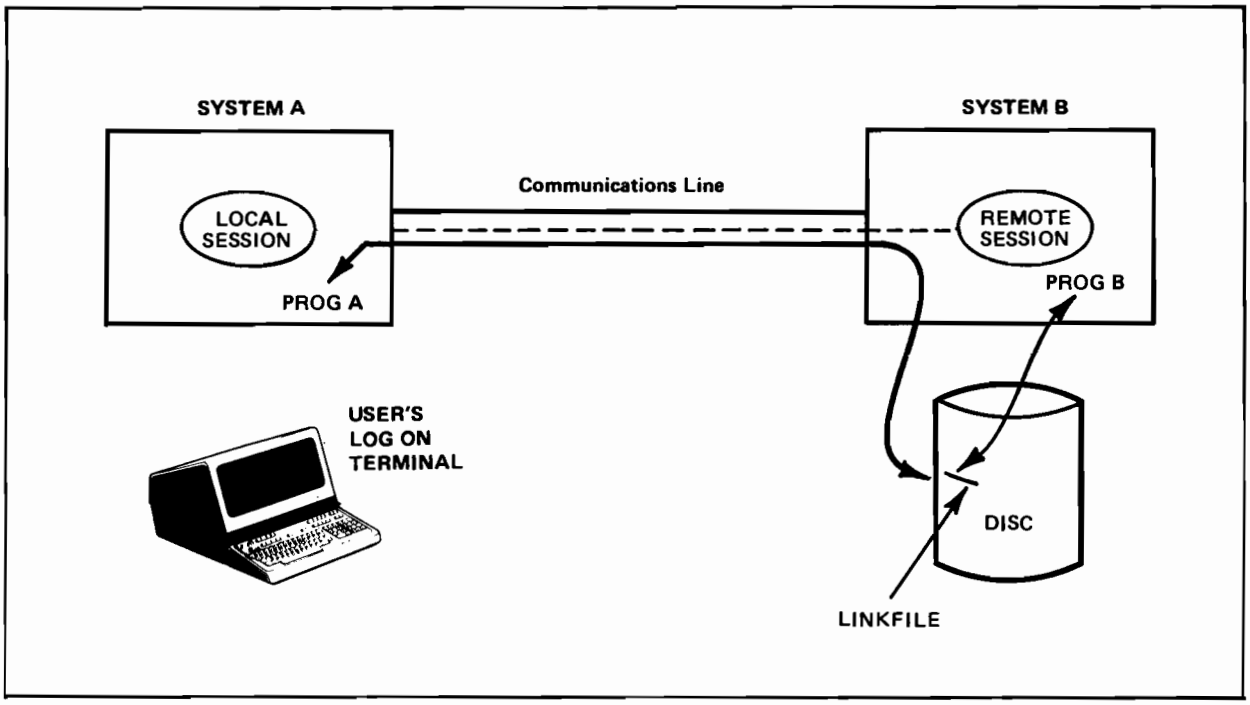


Figure 5-1. Interprogram Communication Using Remote File Access

Suppose we log-on to an HP 3000, gain access to a communications line with another HP 3000, and initiate a remote session over the line. Within the local session we use a STREAM command to initiate the execution of a program named PROGA and within the remote session we use a STREAM command to initiate execution of a program named PROGB. We now have two programs executing simultaneously, PROGA in our local HP 3000 and PROGB in the remote HP 3000.

At this point the two programs are entirely independent of one another: neither knows the other exists and they cannot exchange information back and forth across the line. If we add a shared access disc file to the situation, PROGA and PROGB can now read from and write to that file (LINKFILE in figure 5-1) and thereby communicate indirectly with one another. This arrangement works well so long as the data being deposited in LINKFILE does not have to be retrieved, processed, and responded to within a critically finite period of time.

There are teleprocessing applications where this type of arrangment is not only adequate but makes a great deal of sense. For example, consider the case where a branch office is accumulating information that must once a day be merged into a data base residing at the main office. In this case the two programs can make very effective use of the shared user file approach because the program-to-program interaction is minimal.

As soon as an application tries to be truly interactive, however, this arrangement falters and collapses under an excess of "red tape" and "housekeeping chores". Since the two programs cannot communicate directly, in a highly interactive environment each must periodically examine the status of the shared file in order to know whether or not the other program is trying to transmit data (this would most likely be done using a particular record of LINKFILE as a set of user control flags). The more dependent each program is upon receiving data from the other, the more often it must test the status of LINKFILE. Conceivably, for a very interactive application, the programs could end up spending the vast majority of their time merely keeping track of the status of LINKFILE. The more such "red tape", the less time there is for productive processing.

Thus we see that although the DS/3000 remote file access capability can be used for program-to-program communication, the user programs cannot communicate efficiently with one another.

The DS/3000 program-to-program communications capability is a set of nine new intrinsics that makes it possible for two or more user programs residing in separate HP 3000s to exchange data and control information directly (and efficiently) with one another over DS/3000 communications links.

The nature of any two programs that are communicating with one another in this manner is not symmetrical. One of them (referred to as the master program) is always in control and is the one that initiates all activity between the two programs. The other (referred to as a slave program) is always responding to requests received from the master. Those intrinsics used within a master program are summarized in table 5-1 and those used within a slave program are summarized in table 5-2.

It is true that in order to use this capability you must learn some new intrinsics, but the teleprocessing power and flexibility that they provide far outweighs this minor inconvenience.

Table 5-1. Master Program-to-Program Intrinsics

Intrinsic Name	Function
POPEN	Initiates and activates a slave process in a remote HP 3000 and initiates program-to-program communication with the slave program.
PREAD	Sends a read request to the remote slave program asking the slave to send a block of data back to the master.
PWRITE	Sends a block of data to the remote slave program.
PCONTROL	Transmits a tag field (containing user-defined control information) to the remote slave program and receives a tag field back from the slave.
PCLOSE	Terminates (kills) the remote slave program's process.
PCHECK	Returns an integer code specifying the completion status of the most recently executed master program-to-program intrinsic.

Table 5-2. Slave Program-to-Program Intrinsics

Intrinsic Name	Function
GET	Receives the next request from the remote master program.
ACCEPT	Accepts (and completes) the request received by the preceding GET intrinsic call.
REJECT	Rejects the request received by the preceding GET intrinsic call.
PCHECK	Returns an integer code specifying the completion status of the most recently executed slave program-to-program intrinsic.

Conceptually, the DS/3000 program-to-program intrinsics are very similar to the MPE process handling and file system intrinsics that are used for process-to-process communication within a single-system environment. Table 5-3 compares the intrinsics used for process-to-process communication within a single-system environment to those used for program-to-program communication within a distributed system environment.

Once a DS/3000 communications link exists between two HP 3000s, a user program (the master program) can create and activate a son process (a slave program) in the remote HP 3000. The POPEN intrinsic performs this function, in place of the standard MPE CREATE and ACTIVATE intrinsics. If you pause and think about it for a moment, you will notice that we have already accrued two major advantages over the remote file access method of program-to-program communication and over normal (single-system) process handling:

1.) With the remote file access method of proram-to-program communication the two programs had no way of knowing if the other program was actually executing. With the POPEN intrinsic the master program knows that the slave program is executing because it created and activated the slave program's process. Likewise the slave program knows that the master program is executing because without an active corresponding master program the slave itself would not be executing.

Table 5-3. Single System/Distributed System Comparison

Function	Single System (Process Handling)	Distributed System (Program-to-Program)
Initiate another process.	CREATE ACTIVATE	POPEN
Communicate with the other process.	Mail Intrinsics: SENDMAIL RECEIVEMAIL • • • User Managed Extra Data Segment: GETDSEG DMOVEIN DMOVEOUT • • • Shared User File: FOPEN FREAD FWRITE FCONTROL FCLOSE FCHECK • • •	Master (father) Requests: PREAD PWRITE PCONTROL PCHECK Slave (son) Responses: GET ACCEPT REJECT PCHECK
Terminate the other process.	Father: KILL (a son) TERMINATE (self and all sons)	Master (father): PCLOSE (a slave) TERMINATE (self and all slaves)

2.) With the standard MPE process handling capabilities the father process and any son processes that it creates and activates all compete with one another for the same hardware and operating system resources. With the DS/3000 program-to-program communications capability the master program (father process) and any slave programs (son processes) that it creates and activates are executing in separate HP 3000s and therefore do not compete with one another for the same resources.

Now that the master and slave programs are both executing, the master program can send data (PWRITE) or control information (PCONTROL) directly to the slave program or send a read request (PREAD) or control request (PCONTROL) to the slave program asking that the slave send data or control information back to the master.

You will notice the striking similarity between this method of communication and the use of the MPE File System intrinsics FREAD and FWRITE. It is as though the master program is reading from or writing to a file, a very intelligent file that is capable of making decisions, controlling input/output devices, and performing productive processing. The functional relationship between master and slave programs will be discussed at length in the next topic below ("Master/Slave Relationship"). For now it is sufficient to see the conceptual similarity between the DS/3000 program-to-program method of communicating and the normal method of program/file interaction.

The obvious advantage of the DS/3000 program-to-program communication capability over the shared user file approach is that the interaction between the two programs is now direct (we no longer must use the indirect and inefficient approach of program-to-file-to-program).

To terminate a slave program the master program uses the PCLOSE intrinsic.

Initiates program-to-program communication with a remote slave program.

POPEN

```
      I                BA        BA    IA          BA      IV

dsnum:=POPEN(dsdevice,progname,itag,entryname,param,

                     LV          IV      IV       IV       IV

               flags,stacksize,dlsize,maxdata,bufsize);
```

The POPEN intrinsic creates and activates a process in the remote HP 3000 for the specified remote slave program (progname) and optionally transmits a tag field (itag) to that remote slave program. The remote slave program must issue a GET intrinsic call followed by either an ACCEPT or REJECT call to complete the POPEN operation. The remote slave program may transmit a tag field back to the master program as part of an ACCEPT or REJECT call. If the master program transmitted a tag field, then the returned tag field (if any) is available in itag. If the master program did not transmit a tag field, then the returned tag field (if any) is not accessible.

The bufsize parameter specifies the length in words of an area to be established by the remote DS/3000 software as a communications buffer. This buffer is established implicitly as part of the GET call that receives the POPEN request.

*** Note ***

The remote slave program remains activated and both the communications link and the DS/3000 buffer remain intact even if the POPEN request is rejected by the remote slave program. The meaning of a POPEN reject by the remote slave program must be established as part of the design of the user's application.

The POPEN activity described above is illustrated in figure 5-2.

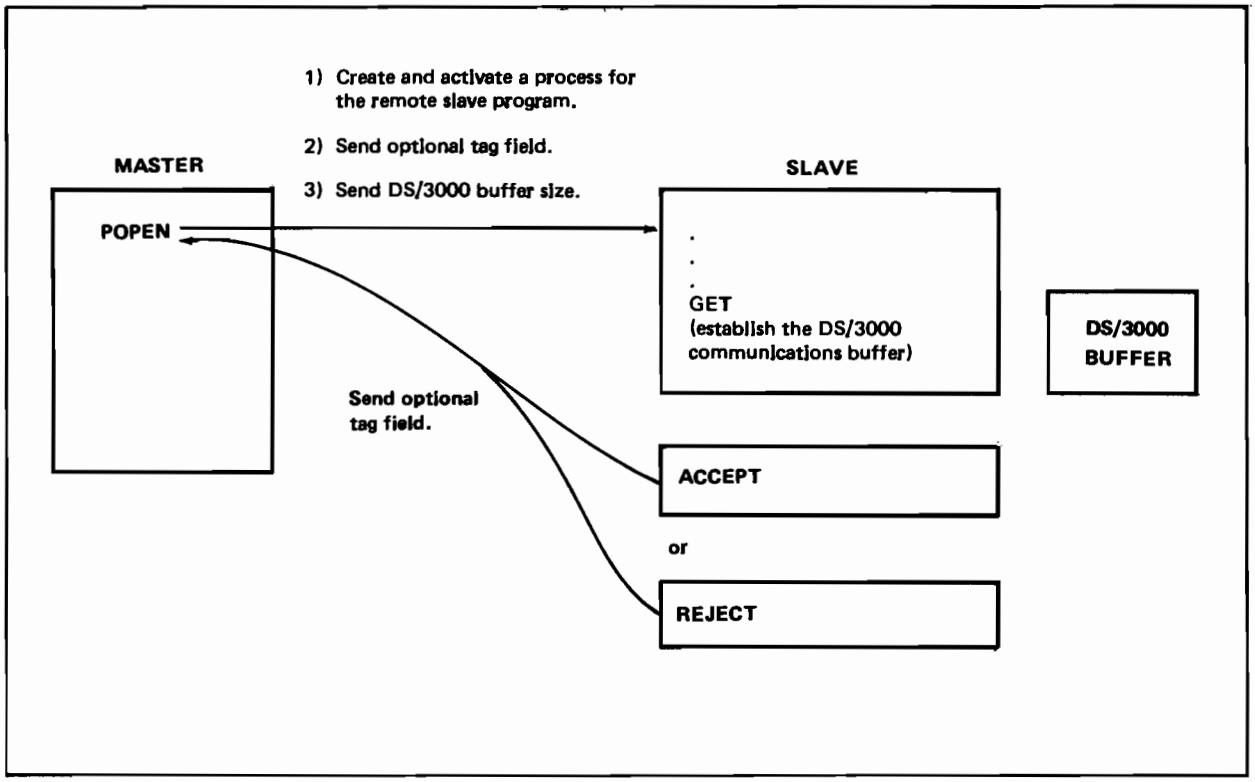


Figure 5-2. POPEN Activity

FUNCTIONAL RETURN

When the POPEN intrinsic is executed, it returns to the master program a number (dsnum) by which DS/3000 uniquely identifies the particular communications link. This number is analagous to the file number returned by the MPE FOPEN intrinsic in that it is used in all subsequent master program-to-program intrinsic calls to reference the remote slave program.

Parameters

dsdevice — byte array (required)

Contains a string of ASCII characters terminated by a space. This string must be the device class name or logical device number used in the DSLINE or REMOTE HELLO command that opened the communications line you will be using.

progname — byte array (required)

Contains a string of ASCII characters terminated by a space. This string is the name (with optional group and account names) of an MPE program file (residing on a disc connected to the remote HP 3000) containing the remote slave program.

itag — integer array (optional)

A twenty-word array that is used for transmitting and receiving tag fields. The format of the tag field is defined as part of the user's application.

Default: A tag field of all zeros is sent; the returned tag field (if any) is not available to the master program.

entryname — byte array (optional)

Contains a string of ASCII characters terminated by a space. This string is the name of the entry point (label) at which execution of the remote slave program is to begin.

Default: Primary entry point.

param — integer by value (optional)

A word used to transfer control information to the new process. Any instruction in the outer block of code in the new process can access this information in location Q-4.

Default: Word is filled with zeros.

flags — logical by value (optional)

A word whose bits, if on, specify the loading options:

NOTE

Bit groups are denoted using the standard SPL notation. Thus bit (15:1) indicates bit 15, bits (10:3) indicates bits 10,11, and 12.

Bit(15:1) - (Always set on.)

Bit(14:1) - LOADMAP bit. If on, a listing of the allocated (loaded) program is produced on the job/session list device. This map shows the Code Segment Table (CST) entries used by the new process. If off, no map is produced.

Default: Off.

Bit(13:1) - DEBUG bit. If on, a call to DEBUG is made at the first executable instruction of the new process. If off, the breakpoint is not set. This bit is ignored if the user is non-privileged and the new process requires privileged mode. It also is ignored if the user does not have read/write access to the program file of the new process.

Default: Off.

Bit(12:1) - If on, the program is loaded in non-privileged mode. If this bit is off, the program is loaded in the mode specified when the program file was prepared.

Default: Off.

Bits(10:2) - LIBSEARCH bits. These bits denote the order in which libraries are to be searched for the program:

00 - System Library.
01 - Account Public Library, followed by System Library.
10 - Group Library, followed by Account Public Library and System Library.

Default: 00.

Bit(9:1) - NOCB bit. If on, file system control blocks are established in an extra segment. If off, control blocks may by established in the Process Control Block Extension (PCBX) area.

Default: Off.

NOTE

This bit should be set on if you are using a large stack.

Bits(7:2) - Reserved for MPE. Should be set to zero.

Bits(5:2) - STACKDUMP bits. These bits control the enabling/disabling of the mechanism by which the stack is dumped in the event of an abort:

00 - Enables only if enabled at father level.
01 - Enables unconditionally.
10 - Same as 00.
11 - Disables unconditionally for new process.

Default: 00

Bit(4:1) - Reserved for MPE. Should be set to zero.

NOTE

The following bits (0:4) are used only when the bit pair (5:2) is 01. Otherwise, these bits are ignored.

Bit(3:1) - DL to QI bit. If on, the portion of the stack from DL to QI is dumped. If off, this portion of the stack is not dumped.

Default: Off.

Bit(2:1) - QI to S bit. If on, the portion of the stack from QI to S is dumped. If off, this portion of the stack is not dumped.

Default: Off.

Bit(1:1) - Q-63 to S bit. If on, the portion of the stack from Q-63 to S is dumped. If off, this portion of the stack is not dumped.

Default: Off

stacksize integer by value (optional)

An integer (Z - Q) denoting the number of words assigned to the local stack area bounded by the initial Q and Z registers.

Default: The same as that specified in the program file. dlsize integer by value (optional)

An integer (DB - DL) denoting the number of words in the user-managed stack area bounded by the DL and DB registers.

Default: The same as that specified in the program file. maxdata integer by value (optional)

The maximum size allowed for the process' stack (Z-DL) area in words. When specified, this value overrides the one established at program-preparation time.

Default: If not specified, and not specified in program file either, MPE assumes the stack will remain the same size.

bufsize integer by value (optional)

The size in words of the communications buffer (DS/3000 buffer) that is to be established by the remote DS/3000 software. Note that this parameter defines the maximum number of words of data that can be transmitted by a PWRITE or PREAD intrinsic call.

Default: Same size as the line buffer defined by the DSLINE command (LINEBUF=) for the first DSLINE issued to the dsdevice. Will never be smaller than 304 words.

If no LINEBUF= is specified then the default configuration length is used.

Condition Codes

CCE	Request accepted by remote slave program.
CCG	Request rejected by remote slave program.
CCE	Request denied; an error occurred. Issue a PCHECK intrinsic call to determine what happened.

Asks the remote slave program
to send a block of data.

PREAD

```
                I              IV      IA      IV    IA

lgth:=PREAD(dsnum,target,tcount,itag);
```

The PREAD intrinsic transmits a read request to the remote slave program and optionally transmits a tag field from itag to the remote slave program. The remote slave program must issue a GET intrinsic call followed by either an ACCEPT or REJECT call to complete the PREAD operation. The ACCEPT call moves the requested block of data from the user's buffer in the remote HP 3000 to target in the master program. The REJECT call transmits no data (other than an optional tag field). The remote slave program may transmit a tag field back to the master program as part of the ACCEPT or REJECT call. If the master program transmitted a tag field, then the returned tag field (if any) is available in itag. If the master program did not transmit a tag field, then the returned tag field (if any) is not accessible.

The PREAD activity described above is illustrated in figure 5-3.

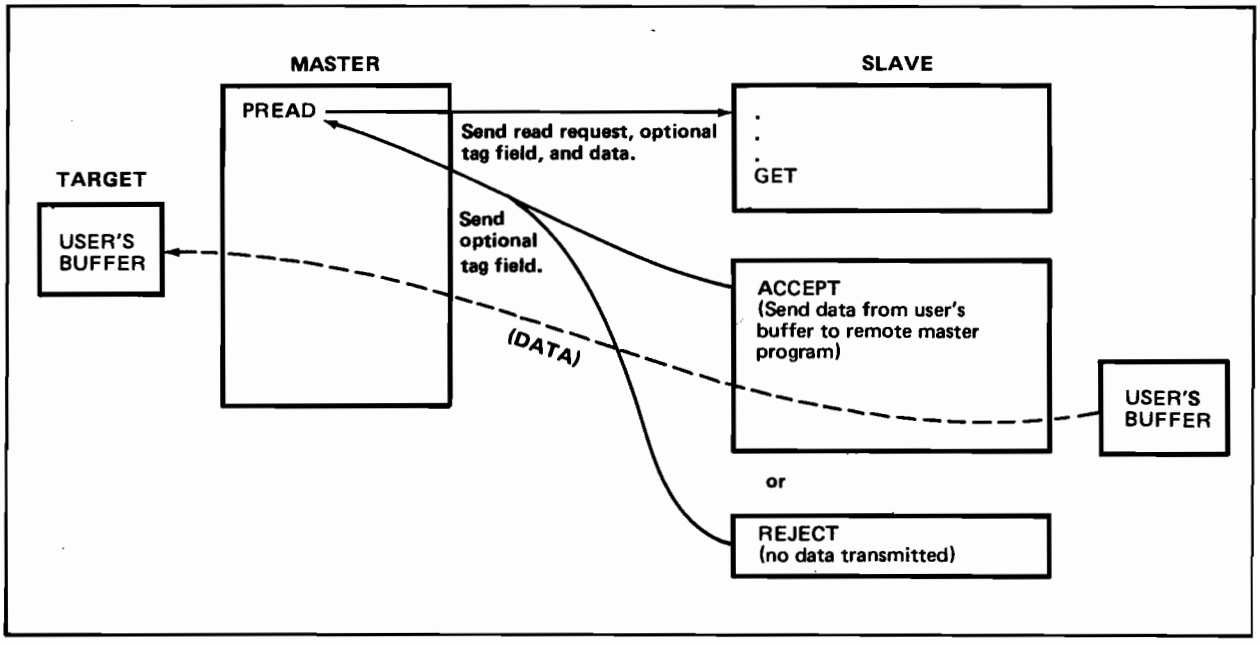


Figure 5-3. PREAD Activity

Functional Return

When the PREAD intrinsic is executed, it returns to the master program a number (lgth) specifying how many words or bytes of data were actually received into target.

Parameters

dsnum	integer by value (required) The link identifier returned by the particular POPEN intrinsic call which initiated communication with the remote slave program.
target	integer array (required) The array into which data received from the remote slave program will be deposited.
tcount	integer by value (required) The requested number of words (if positive) or bytes (if negative) of data.
itag	integer array (optional) A twenty-word array used for transmitting and receiving a tag field. The format of the tag field is defined by the user's master and slave programs.

Condition Codes

CCE	Request accepted by remote slave program.
CCG	Request denied by remote slave program.
CCL	Request denied; an error occurred. Issue a PCHECK intrinsic call to determine what happened.

Sends a block of data to the remote slave program.

PWRITE

```
                    IV      IA      IV    IA

PWRITE(dsnum,target,tcount,itag);
```

The PWRITE intrinsic transmits a block of data (number of words = tcount) from target to the DS/3000 buffer in the remote HP 3000, transmits a write request to the remote slave program, and optionally transmits a tag field from itag to the remote slave program. The remote slave program must issue a GET intrinsic call followed by either an ACCEPT or REJECT call to complete the PWRITE operation. The ACCEPT call moves the block of data from the DS/3000 buffer to the user's buffer in the remote HP 3000. The REJECT call refuses the write request (the data in the DS/3000 buffer is no longer accessible to the remote slave program). The remote slave program may transmit a tag field back to the master program as part of the ACCEPT or REJECT call. If the master program transmitted a tag field, then the returned tag field (if any) is available in itag. If the master program did not transmit a tag field, then the returned tag field (if any) is not accessible.

The PWRITE activity described above is illustrated in figure 5-4.

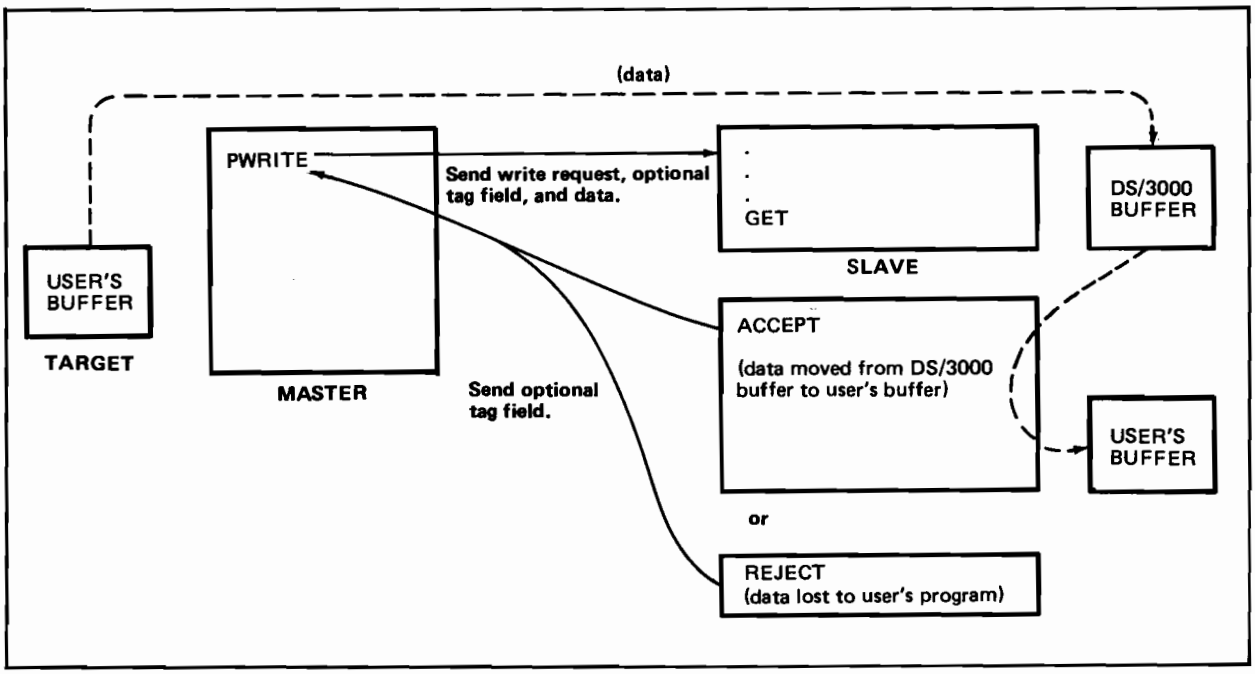


Figure 5-4. PWRITE Activity

Parameters

dsnum	integer by value (required) The link identifier returned by the particular POPEN intrinsic call which initiated communication with the remote slave program.
target	integer array (required) The array from which data will be transmitted to a remote slave program.
tcount	integer by value (required) The requested number of words (if positive) or bytes (if negative) of data.
itag	integer array (optional) A twenty-word array used for transmitting and receiving a tag field. The format of the tag field is defined by the user's master and slave programs.

Condition Codes

CCE	Request accepted by remote slave program.
CCG	Request denied by remote slave program.
CCL	Request denied; an error occurred. Issue a PCHECK intrinsic call to determine what happened.

Exchanges tag fields with the remote slave program.

PCONTROL

```
                    IV   IA

PCONTROL(dsnum,itag);
```

The PCONTROL intrinsic transmits a tag field to the remote slave program and accepts one in return. The remote slave program must issue a GET intrinsic call followed by either an ACCEPT or REJECT call to complete the PCONTROL operation. Both the ACCEPT and REJECT calls transmit a tag field back to the master program (available in itag).

Although this intrinsic was designed specifically for the exchanging of tag fields, you will notice that itag is an optional parameter (it is also optional for the ACCEPT and REJECT slave program-to-program calls). If the master program did not transmit a tag field, then the returned tag field (if any) is not accessible.

The PCONTROL activity described above is illustrated in figure 5-5.

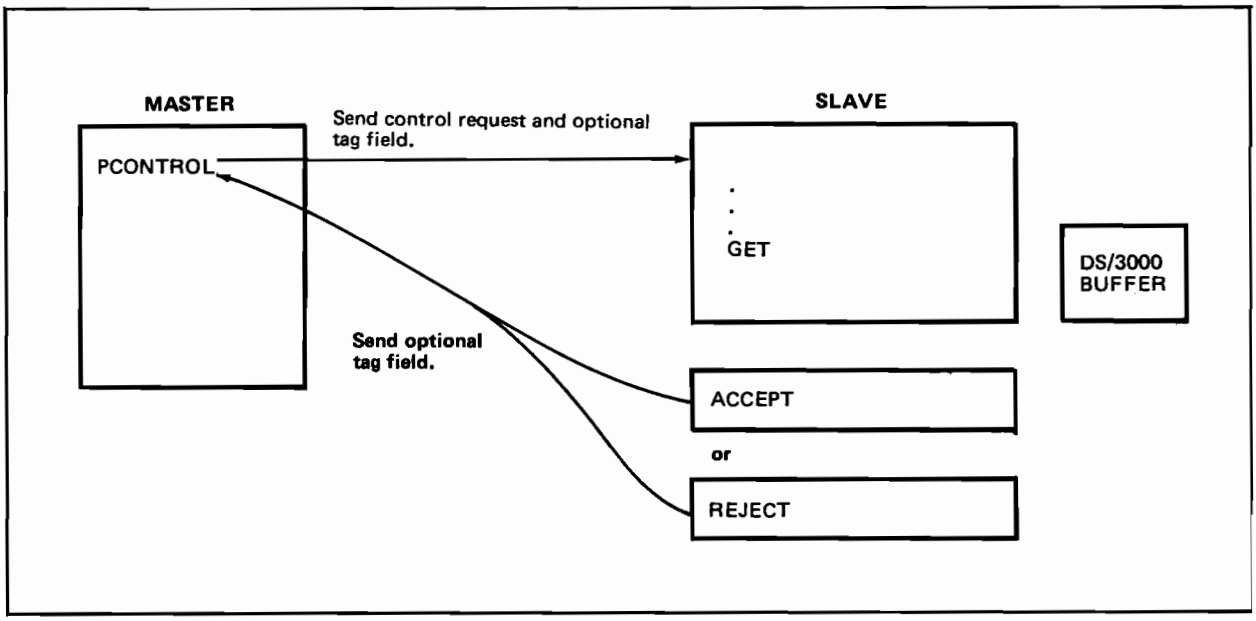


Figure 5-5. PCONTROL Activity

Parameters

dsnum	integer by value (required) The link identifier returned by the particular POPEN intrinsic call which initiated communication with the remote slave program.
itag	integer array (optional) A twenty-word array used for transmitting and receiving a tag field. The format of the tag field is defined by the user's master and slave programs.

Condition Codes

CCE	Request accepted by remote slave program.
CCG	Request denied by remote slave program.
CCL	Request denied; an error occurred. Issue a PCHECK intrinsic call to determine what happened.

Terminates program-to-program communication with a remote slave program.

PCLOSE

```
IV

PCLOSE(dsnum);
```

The PCLOSE intrinsic deletes the remote slave program associated with dsnum. The particular communications line remains open.

Parameters

dsnum	integer by value (required) The line number returned by the particular POPEN intrinsic call which initiated communication with the remote slave program.

Condition Codes

CCE	Successful completion.
CCG	(Not returned.)
CCL	Request denied; an error occurred. Issue a PCHECK intrinsic call to determine what happened.

Receives the next request from the remote master program.

GET

```
                I          IA  I          I

ifun:=GET(itag,il,ionumber);
```

The GET intrinsic receives the next request from the remote master program and accepts an optional tag field (available in itag).

Functional Return

When the GET intrinsic is executed, it returns to the slave program a number (ifun) specifying what type of request was received from the remote master program, as follows:

0	An error occurred. This value is returned only when the condition code CCL is also returned. Issue a PCHECK intrinsic call (with a dsnum parameter of zero) to determine what happened.
1	POPEN request received.
2	PREAD request received.
3	PWRITE request received.
4	PCONTROL request received.
5	This value is returned only when the condition code CCG is also returned. It indicates that a pending MPE-II File System I/O without wait request was completed (instead of a DS/3000 remote I/O request). ionumber contains the file number associated with the completed I/O request.

The GET intrinsic call implicitly issues an IOWAIT(0) intrinsic call. An ifun of 0 indicates that an IOWAIT error occurred. An ifun of 5 will occur only if you are executinbg MPE-II File System intrinsic calls without wait in your program and the implicit IOWAIT(0) call completes a pending File System I/O request instead of the expected DS/3000 remote I/O request (in this case you will have to issue another GET call after processing the completed File System I/O request in order to receive the expected DS/3000 remote I/O request).

*** Note ***

You must not use IOWAIT(0) calls witnin a program containing DS/3000 GET calls. If you were to use an IOWAIT(0) call and it responded to a DS/3000 remote I/O request, your program would not be able to make any sense out of the information returned by the IOWAIT call.

Parameters

itag — integer array (optional)

A twenty-word array used for receiving a tag field. The format of the tag field is defined by the user's master and slave programs.

il — integer (optional)

A word that has meaning only when a PREAD or PWRITE request is received from the remote master program.

For a PREAD request, il contains an integer specifying the number of words requested by the remote master program.

For a PWRITE request, il contains an integer specifying the number of words transmitted from the remote master program to the DS/3000 buffer.

ionumber — integer (optional)

A word that has meaning only when the condition code CCG and an ifun of 5 are returned. In that case ionumber contains the MPE-II File System file number associated with the completed I/O without wait request.

Default: No file number is returned.

Condition Codes

CCE — Request received successfully.

CCG — The implicit IOWAIT(0) call issued by the GET intrinsic completed a pending MPE-II File System I/O without wait request instead of a DS/3000 remote I/O request. ionumber contains the file number associated with the completed File System request.

CCL — An error occurred. Issue a PCHECK intrinsic call to determine what happened.

ACCEPT

Accepts (and completes) the request received by the preceding GET intrinsic call and returns an optional tag field back to the remote master program.

```
                    IA      IA      IV

ACCEPT(itag,target,tcount);
```

The ACCEPT intrinsic accepts the request received by the most recent GET intrinsic call, completes the requested operation, and transmits an optional tag field back to the remote master program.

In the case of a POPEN request, the ACCEPT call transmits an optional tag field (itag) to the remote master program.

In the case of a PREAD request, the ACCEPT call transmits the specified number of words (tcount) from target to the remote master program and transmits an optional tag field (itag) to the remote master program.

In the case of a PWRITE request, the ACCEPT call moves the specified number of words (tcount) from the DS/3000 buffer to target and transmits an optional tag field (itag) to the remote master program.

In the case of a PCONTROL request, the ACCEPT call transmits an optional tag field (itag) to the remote master program.

Parameters

itag — integer array (optional)

A twenty-word array used for transmitting a tag field. The format of the tag field is defined by the user's master and slave programs.

target — integer array (optional)

An array used for transmitting or receiving blocks of data.

For PREAD requests, this array contains the block of data to be transmitted to the remote master program.

For PWRITE requests, this array receives the block of data from the DS/3000 buffer.

For POPEN and PCONTROL requests, this parameter has no meaning and should be omitted.

tcount — integer by value (optional)

An integer specifying the number of words to be transmitted or received.

For PREAD requests, this parameter specifies how many words of data are to be transmitted from target to the remote master program.

For PWRITE requests, this parameter specifies how many words of data are to be moved from the DS/3000 buffer to target.

For POPEN and PCONTROL requests, this parameter has no meaning and should be omitted.

Condition Codes

CCE	Request completed successfully.
CCG	(Not returned.)
CCL	An error occurred. Issue a PCHECK intrinsic call to determine what happened.

Rejects the request received by the preceding GET intrinsic call and returns an optional tag field back to the remote master program. program.

REJECT

```
                    IA

REJECT(itag);
```

The REJECT intrinsic rejects the request received by the most recent GET intrinsic call and transmits an optional tag field (itag) back to the remote master program.

Parameters

itag	integer array (optional) A twenty-word array used for transmitting a tag field. The format of the tag field is defined by the user's master and slave programs.

Condition Codes

CCE	Response transmitted successfully to the remote master program.
CCG	(Not returned.)
CCL	An error occurred. Issue a PCHECK intrinsic call to determine what happened.

PCHECK

Returns an integer code specifying the completion status of the most recently executed DS/3000 program-to-program intrinsic.

```
                    I              IV

          icode:=PCHECK(dsnum);
```

The PCHECK intrinsic returns an integer value that specifies the completion status of the most recently executed DS/3000 program-to-program intrinsic.

Functional Return

When the PCHECK intrinsic is executed, it returns to the calling program a number (icode) that specifies the completion status of the most recently executed DS/3000 program-to-program intrinsic. The various values of icode are shown in Appendix B under Communication Link Errors and Program-to-Program Errors.

Parameters

dsnum	integer by value (required)	
	MASTER PROGRAM:	The link identifier returned by the particular POPEN intrinsic that initiated communication with the remote slave program.
	SLAVE PROGRAM:	0 (zero); identifies an error which caused GET, ACCEPT, or REJECT to fail in a slave program.

Condition Codes

CCE	PCHECK request successfully completed.
CCG	(Not returned.)
CCL	PCHECK request denied because dsnum was invalid.

EXAMPLE

The following programs will show you how two programs can communicate with one another by using the master and slave program-to-program intrinsics.

```
1     $CONTROL USLINIT,ADR,MAP,CODE,NOWARN
2     BEGIN
3
4
5     COMMENT
6        NAME OF PROGRAM IS MASTERP(S).
7        THIS PROGRAM IS TO BE RUN ON THE MASTER CPU.IT WILL
8        START THE "SLAVE" PROGRAM ON THE SLAVE CPU.THE PROGRAM
9        WILL THEN RECIEVE A KNOWN TEST PATTERN FROM THE USER
10       TERMINAL,WRITE IT TO THE REMOTE DISK FILE,READ IT
11       BACK AND PRINT IT ON THE LOCAL LP.
12       THE TRANSFER OD DATA IS DONE THRU PTUPC.;
13
14
15    INTEGER
16      LINE'NUM,
17      I,
18      J,
19      LPDEV'NUM;
20
21
22    BYTE ARRAY DS'DEVICE(0:6):=7(%020040);
23    BYTE ARRAY LPDEV(0:2):="LP ";
24    BYTE ARRAY LPFILE(0:6):="LPFILE ";
25    BYTE ARRAY MSG(0:79);
26    BYTE ARRAY PROG'NAME(0:19):="SLAVEP.PUB.SUPPORT ";
27
28
29    LOGICAL ARRAY IOBUF(0:39);
30    LOGICAL ARRAY ITAG(0:19):=20(%020040);
31
32
33
34
35    INTRINSIC PRINT,QUIT,READ,FOPEN,FWRITE,POPEN,PCONTROL,PCLOSE,PREAD;
36    INTRINSIC PWRITE,FCLOSE;
37
38    MOVE MSG:="      INPUT NAME OF DSDEVICE";
39    PRINT(MSG,-28,0);
40    READ(DS'DEVICE,-7);
41
42    MOVE MSG:="      POPEN ISSUED";
43    PRINT(MSG,-18,0);
44
45    LINE'NUM:=POPEN(DS'DEVICE,PROG'NAME,ITAG);
46      IF <> THEN
47
48        BEGIN
49          PRINT(ITAG,20,0);
50          QUIT(1);
51        END
52        ELSE
53          PRINT(ITAG,20,0);
54
55
```

```
56      MOVE MSG:="       POPEN COMPLETED SUCESSFULLY";
57      PRINT(MSG,-33,0);
58
59      LPDEV'NUM:=FOPEN(LPFILE,4,1,40,LPDEV);
60        IF <> THEN QUIT(2);
61
62      MOVE MSG:="IN PUT TEST RECORD MAX. 80 CHAR";
63      PRINT(MSG,-30,0);
64
65      MOVE IOBUF:="  ";                <<CLEAR OUT BUFFER AREA>>
66      MOVE IOBUF(1):=IOBUF,(39);
67
68      READ(IOBUF,-80);                 <<GET RECORD TO WRITE >>
69
70      FWRITE(LINE'NUM,IOBUF,40);       <<SEND RECORD TO REMOTE>>
71      IF <> THEN QUIT(3);
72
73      MOVE MSG:="       DISK FILES BEING XFERED FROM REMOTE";
74      PRINT(MSG,-41,0);
75      J:=-1;                                    <<START READING FROM REMOTE>>
76      WHILE (J:=J+1)<5 DO
77        BEGIN
78          MOVE MSG:="        PREAD ISSUED";
79          PRINT(MSG,-19,0);
80
81          MOVE IOBUF:="  ";
82          MOVE IOBUF(1):=IOBUF,(39);
83
84          I:=PREAD(LINE'NUM,IOBUF,40,ITAG);
85          IF = THEN
86            BEGIN
87              IF J=4 THEN
88                BEGIN
89                  MOVE MSG:="       ALL DISK RECORDS XFERED";
90                  PRINT(MSG,-29,0);
91                END;
92            END
93              ELSE
94            QUIT(4);
95          FWRITE(LPDEV'NUM,IOBUF,1,0);
96            IF <> THEN QUIT(4);
97        END;
98
99      FCLOSE(LPDEV'NUM,0,0);
100     FCLOSE(LINE'NUM);
101     IF <> THEN QUIT(15);
102     MOVE MSG:="END OF MASTER PROGRAM";
103     PRINT(MSG,-21,0);
104     END.
```

```
$CONTROL USLINIT,ADR,MAP,CODE,NOWARN

BEGIN
COMMENT
   THE NAME OF THIS PROGRAM IS SLAVEP(S).
   THIS PROGRAM IS TO BE COMPILED AND PREP'ED ON THE
   SLAVE HP3000 SYSTEM.IT WILL BE INITIATED FOR RUN
   BY THE MASTER.THE FUNCTION OF THIS PROGRAM IS TO
   TO LOAD A DISK FILE WITH KNOWN TEST PATTERNS THAT WILL
   BE TRANSFERRED TO THE MASTER AND PRINTED ON THE
   MASTER'S LINE PRINTER;


INTEGER
  DISK'FILENUM,
  I,
  IL,
  IONUMBER,
  J;




LOGICAL ARRAY DISK'BUF(0:39);
LOGICAL ARRAY ITAG(0:19):=7(020040);

BYTE ARRAY MSG(0:79);
BYTE ARRAY TEST(0:4):="TEST ";

INTRINSIC FOPEN,DEBUG,QUIT,PRINT,READ,FWRITEDIR,FREADDIR,FCLOSE;
INTRINSIC GET,ACCEPT,REJECT;




MOVE MSG:="ISSUING A GET (REMOTE)";
PRINT(MSG,-22,0);
I:=GET(ITAG);                                  <<GET FOR POPEN>>
IF < THEN
  BEGIN
    MOVE ITAG:="ERROR ON GET;POPEN";
    GO TO ERR'OPEN;
  END;

IF I=1 THEN
  BEGIN
    MOVE MSG:="POPEN RCVD...ISSUING AN ACCEPT (REMOTE)";
    PRINT(MSG,-39,0);
  END;

MOVE ITAG:="POPEN ACCEPT SUCCFSSFUL (REMOTE)";
ERR'OPEN:
ACCEPT(ITAG);                                  <<ACCEPT FOR POPEN>>

DISK'FILENUM:=FOPEN(TEST,4,%104,-80,,,,1,1,10D);
  IF <> THEN QUIT(1);

I:=GET;                                        <<TEST REC FROM MASTER>>
IF <> THEN QUIT(2);
IF I=3 THEN                                    <<PWRITE RECEIVED>>
  BEGIN
    ACCEPT( ,DISK'BUF);
    IF <> THEN QUIT(3);
  END;


I:=-1;                                         <<START WRITING TEST FILE>>
WHILE(I:=I+1) < 5 DO
```

```
70        BEGIN
71          FWRITEDIR(DISK'FILENUM,DISK'BUF,40,DOUBLE(I));<<WRITE REC TO DISK>>
72            IF <> THEN QUIT(4);
73
74        END;                                          <<END WRITING TEST FILE>>
75
76      J:=-1;                                           <<SEND DISK FILE TO MASTER>>
77      WHILE(J:=J+1)<5 DO
78        BEGIN
79          MOVE MSG:="ISSUING A GET (REMOTE)";
80          PRINT(MSG,-22,0);
81          I:=GET(ITAG,40,IONUMBER);
82          IF < THEN QUIT(5);
83          IF I=2 THEN
84            BEGIN
85              MOVE MSG:="PREAD RCVD...ISSUING AN ACCEPT (REMOTE)";
86              PRINT(MSG,-39,0);
87            END
88            ELSE
89              QUIT(6);
90          MOVE DISK'BUF:=%020040;
91          MOVE DISK'BUF(1):=DISK'BUF(0),(39);
92          FREADDIR(DISK'FILENUM,DISK'BUF,40,DOUBLE(J));
93          IF <> THEN QUIT(7);
94          ACCEPT(ITAG,DISK'BUF,40);
95          IF <> THEN QUIT(8);
96        END;
97
98      FCLOSE(DISK'FILENUM,0,0);
99      END.
```

CONFIGURATION DIALOGUE

APPENDIX A

The configuration dialogue is accomplished through an interactive dialogue between you and the computer system. As the questions or prompts appear on your console, enter the appropriate replies through the console keyboard for your desired system configuration.

NOTE

In all responses, Y or N can be used for YES and NO. A carriage return is equivalent to NO.

Prior to entering the dialogue, log onto the system and input at least a file reference to a magnetic tape, as follows:

```
:FILE name;DEV=TAPE
:SYSDUMP*name
```

The dialogue then commences as follows:

Step No.	Dialogue
1	ANY CHANGES? To prepare for changes, enter YES To omit changes, and skip to Step 12, enter NO.
2	SYSTEM ID = HP 32002 V.<UU>.>FF>? In this message V is the current MPE version, UU is the present update-level number and FF is the fix-level number To prepare for updating software for a new fix level, enter the new fix-level digits (FF). (These digits indicate the latest system fix provided by Hewlett-Packard.) Otherwise, enter a carriage return.
3	MEMORY SIZE = <XXX>.? In this message, XXX denotes the present size of main memory. To indicate the size of main-memory for the system for which MPE is being configured, enter one of the following values: 64, 96, 128, 160, 192, 224, or 256. This denotes memory size in a multiple of 1024 words. To retain the present memory size, enter a carriage return.

Step No.	Dialogue

3.1 I/O CONFIGURATION CHANGES?

To prepare for addition or deletion of input/output devices, enter YES.

To maintain the same input/output device configuration, and proceed to Step 4, enter NO.

3.2 LIST I/O DEVICES?

To print a list of input/output devices currently assigned to the system, enter YES. The format of the output is:

```
LOG DRT U C T SUB      TERM     REC    OUTPUT MODE DRIVER DEVICE
DEV  #  N H Y TYPE TYPE SPEED WIDTH   DEV          NAME   CLASSES
        I A P
        T N E
```

To suppress this listing, enter NO.

NOTE

The prompt in Step 3.3, below, appears only if a communications subsystem (CS) device was previously configured into the system.

3.3 LIST CS DEVICES

To print a list of the characteristics of all CS devices currently assigned to the system, enter YES. The format of the output is:

```
LDN PM PRT LCL TC  RCV    LCL    CON  MODE  TRANSMIT  TM  BUFFER D DRIVER
           MOD   TMOUT  TMOUT  TMOUT        SPEED       SIZE  C OPTIONS
```

If you have a switched device, such as those that are connected through a dial up telephone line, then you will receive the additional output:

```
LDN CTRL PHONE NUMBER LIST   LOCAL ID SEQUENCE
    LEN                      REMOTE ID SEQUENCE
```

In this format, the headings denote the following information:

HEADING	MEANING
LDN	Logical device number.
PM	Reserved for future use; presently always 0.
PRT	Protocol.
LCL MOD	Local mode.
TC	Transmission code.
RCV TMOUT	Receive timeout.
LCL TMOUT	Local timeout.
CON TMOUT	Connect timeout.
MODE	O = Dial out. I = Manual answer. A = Automatic answer. D = Dual speed. H = Half speed. C = Speed changeable.
TRANSMIT SPEED	Transmission speed. (characters per second).
TM	Transmission mode.
BUFFER SIZE	Default buffer capacity, in words.
DC	Driver changeable or not changeable.
DRIVER OPTION	Driver options.

To suppress this listing, enter NO.

3.4 HIGHEST DRT=<XX>.?

In the output, XX is a value denoting the present highest DRT entry number that can be assigned to a device.

To change XX, enter the new value desired. If the highest-numbered device in the configuration is a device that uses more than one DRT entry (such as a terminal controller with one or two data set controllers), be sure to enter the *highest* of the DRT numbers.

To maintain the current XX, enter a carriage return.

3.5 LOGICAL DEVICE #?

To specify a device to be added or removed, enter the logical device number of that device.

To skip to Step 3.80 enter zero or a carriage return.

Step No.	Dialogue
3.6	DRT #? To add a device, enter its DRT entry number. To remove a device and return to Step 3.3, enter zero.
3.7	UNIT #? Enter the physical hardware unit number of the device, if the device shares its controller with other devices. Otherwise, enter zero to continue.
3.8	CHANNEL #? If the device is to be on a multicontroller channel, enter channel number; if not, enter zero.
3.9	TYPE? Enter the device type, where

Octal	Decimal
0	0 = Moving-Head Disc
1	1 = Fixed-Head Disc
10	8 = Card Reader
11	9 = Paper Tape Reader
20	16 = Terminal
22	18 = Synchronous Single-Line Controller
23	19 = Hardwired Serial Interface
24	20 = Printing Reader/Punch
27	23 = Programmable Controller
30	24 = Magnetic Tape
40	32 = Line Printer
41	33 = Card Punch
42	34 = Paper Tape Punch
43	35 = Plotter
51	41 = DS/3000 Communications Driver

3.10	SUBTYPE? Enter the device sub-type in the range 0 to 15. (For DS devices refer to tables A-1 through A-4; for non-DS devices refer to Appendix A and Appendix B in the System Manager/System Supervisor Manual.)

Step No.	Dialogue

NOTE

If you are configuring a terminal (type 16), the dialogue continues to step 3.11. If you are configuring an HSI (type 19), the dialogue skips to step 3.13. If you are configuring an SSLC (type 18), the dialogue skips to step 3.14. For all other device types, the dialogue skips to step 3.40.

3.11 TERM TYPE?

This question is asked only if type is 16. To specify a default terminal type to be used at log-on, enter a number as follows:

0	HP 30124A (HP 2749B), ASR33 or ASR35
1	ASR37
2	(Reserved)
3	Execuport 300
4	HP 30123 (HP 2600A), or Datapoint 3300
5	Memorex 1240
6	HP 30120A (HP 2762A), GE Terminet 300 or 1200
7	(Reserved)
8	(Reserved)
9	HP 30122A (HP 2615A) MINI BEE
10	HP 2640A, HP 2644A
11	HP 2640A, HP 2644A (If you switch between character and block/line mode)
13	TELENET message switching network

3.12 SPEED IN CHARACTERS PER SECOND?

This question is asked only if device type is 16. To specify the terminal speed in characters per second, enter 10, 14, 15, 30, 60, 120, or 240.

Otherwise, enter 0 or a carriage return.

NOTE

The dialogue skips to step 3.40.

Step No.	Dialogue

3.13 PORTMASK?

This question is asked only if device type is 19. The values allowable are shown below and must be entered in decimal. This forms a mask indicating which HSI channel will be used. Only one of the four channels may be designated:

Dec 8 = HSI Channel 0.
4 = HSI Channel 1.
2 = HSI Channel 2.
1 = HSI CHannel 3.

3.14 PROTOCOL?

To define the data communication protocol, enter 1.

3.15 LOCAL MODE?

To define the appropriate mode number for the local station, enter *1* (if local is a primary-contention station) or *2* (if local is a secondary-contention station).

NOTE

To resolve the problem of contention in point-to-point operation, each station is assigned a priority—primary or secondary. Thus the secondary station can gain control of the line for a transmission only when the line is left free by the primary station. The HP 2780/3780 Emulator is usually a primary station.

3.16 TRANSMISSION CODE?

Enter the appropriate number for the transmission code in use. The code numbers are:

1 = Automatic code sensing of ASCII and EBCDIC if initially receiving ASCII if initially sending; or for Hardwired Serial Interface.
2 = ASCII
3 = EBCDIC

3.17 RECEIVE TIMEOUT?

Enter the positive number of seconds the Emulator will wait to receive text before terminating the read mode. Entering a carriage return provides a 20-second timeout.

Step No.	Dialogue

NOTE

For all timeout responses: Entering 0 disables the timeout; maximum timeout is 32000 seconds; the Emulator displays an error when the communications software (CS) disconnects because of a timeout.

3.18 LOCAL TIMEOUT?

Enter the positive number of seconds a connected local station will wait to transmit or receive before disconnecting. Entering a carriage return provides a 60-second timeout.

3.19 CONNECT TIMEOUT?

Enter the positive number of seconds the local station will wait after one attempt to make a connection to a remote station. Entering a carriage return provides a 900-second timeout.

NOTE

Steps 3.20 through 3.22 apply only to CS devices with switched lines connected through a modem (dial telephones subtype 0). For CS devices with non-switched lines connected through a modem (private lines, subtype 1), the dialogue skips to Step 3.23. If the CS device is hardwired (subtype 3), the dialogue skips to Step 3.25.

3.20 DIAL FACILITY?

Enter YES when calls can be dialed from the local station. Enter NO when they cannot.

3.21 ANSWER FACILITY?

Enter YES if the local modem can answer calls, either manually or automatically. Enter NO if it cannot. A NO response causes the next step to be skipped.

3.22 AUTOMATIC ANSWER?

Enter YES if the local modem can automatically answer calls. Enter NO if manual answering is required.

3.23 DUAL SPEED?

Enter YES if the local modem is dual speed (European models). Enter NO if it is single speed. A NO response causes the next step to be skipped.

Step No.	Dialogue

3.24 HALF SPEED?

Enter YES if the local modem is to operate at half speed. Enter NO if it is to operate at full speed. The dialogue skips to Step 3.26.

3.25 SPEED CHANGEABLE?

Enter YES if the speed of the line is changeable. Enter NO if the line speed is fixed.

3.26 TRANSMISSION SPEED?

For SSLC (type 18) devices, enter the transmission speed of the line in characters per second (Bit Rate/8). For HSI (type 19) devices, enter 250 000 for cable lenghts up to 1000 feet, or enter 125 000 for cable lengths above 1000 feet.

3.27 TRANSMISSION MODE?

Enter the appropriate number for the transmission mode in use. The CS device may be either half of full duplex, depending upon the type of line and modem. The mode numbers are:

0 = Full duplex
1 = Half duplex

HSI devices are full duplex.

NOTE

A full duplex line cannot be configured in a half duplex mode.

3.28 PREFERRED BUFFER SIZE?

Enter the desired buffer-size in words, up to a maximum of 4096 words. 576 is generally adequate. Large buffer-sizes increase transmission efficiency, but use up memory space. Match buffer-sizes whenever possible, since the effective buffer-size that can be utilized is the smaller of the two buffer-sizes between sender and receiver.

NOTE

The dialogue skips to Step 3.50. (Dialogue Steps 3.31 to 3.39 are reserved for future use.)

3.29 DRIVER CHANGEABLE?

Enter NO.

3.30 DRIVER OPTIONS?

Enter 0.

Step No.	Dialogue

3.40 RECORD WIDTH?

Enter the record width for the device. Default widths are referenced in tables A-1 through A-4 for DS devices or Appendix A of the System Manager/System Supervisor Manual for non-DS devices. Disc device defaults should be used. However, for other devices, any record width up to the maximum may be specified for your configuration.

3.41 OUTPUT DEVICE?

If the device is ever used as a job or session input device, enter the class name or logical device number to be used for the corresponding job/session listing device.

There are advantages in using class names under certain circumstances. Suppose there are two line printers and two card readers in the system. Both line printers are in class LP. You may configure the output device for both card readers as LP. In this way, either card reader can acquire *either* one of the line printers dynamically (provided at least one line printer is unallocated) at run time.

If this device is not a job/session input device, enter zero.

3.42 ACCEPT JOBS/SESSIONS?

To specify that this device can accept a job or session input stream (J), enter YES.

Otherwise, enter NO.

Note: Disc devices should *not* be job accepting.

3.43 ACCEPT DATA?

To specify that this device can accept data external to a job or session input stream (A), enter YES.

Otherwise, enter NO.

Note: Disc devices should *not* be data accepting.

3.44 INTERACTIVE?

To specify that this is an interactive device (I), enter YES.

Otherwise, enter NO.

Note: Disc devices should *not* be interactive.

Step No.	Dialogue
3.45	DUPLICATIVE?

To specify that this is a duplicative device (D), enter YES.

Otherwise, enter NO.

Note: Disc devices should *not* be duplicative.

3.46 INITIALLY SPOOLED?

To designate this device as being spooled at cold-load (S), enter YES.

Otherwise, enter NO and skip to Step 3.50.

3.47 INPUT OR OUTPUT?

This question is asked only if the device is initially spooled and the device is an input/output device.

Enter IN or OUT.

NOTE

Dialogue Steps 3.48 and 3.49 are reserved for future use.

3.50 DRIVER NAME?

Enter the name of the program file containing the driver for this device (Refer to tables A-1 through A-4 for DS devices or Appendix A in the System Manager/System Supervisor Manual for non-DS devices. For drivers written and supplied by the user, this name must contain from one to eight alphanumeric characters, beginning with a letter. (If the driver name is preceded by an asterisk, the driver will reside permanently in main-memory).

NOTE

Steps 3.51 through 3.55 apply to CS devices with switched lines (type 18, subtype 0). The dialogue for all other devices skips to Step 3.70.

3.51 CONTROL LENGTH?

Enter 0.

Step No.	Dialogue

3.52 PHONELIST?

Enter YES to provide a default phone number list. Enter NO if none provided. A NO response causes the next step to be skipped.

NOTE

The Emulator will sequentially step through the default phone number list if there is no specified number. The default phone number list is overridden at run time by specifying a phone number for the ;CONNECT= parameter in the #RJLINE command.

3.53 PHONE NUMBER?

Enter a string of numbers and hyphens, but not more than 20 characters. This question is repeated until a carriage return is entered.

3.54 LOCAL ID SEQUENCE?

The default local ID sequence can be specified in terms of code or number system. Enter a carriage return for a null local ID sequence. Enter one of the letters below, followed by the ID sequence in quotes, if code, or parentheses, if number system:

A	=	ASCII	Example: A "JOE"
E	=	EBCDIC	Example: E "STRING"
O	=	Octal	Example: O (7, 35, 5)
H	=	Hexadecimal	Example: H (A1, 1F, BB)

NOTE

Do not enter more than 16 characters for the local or remote ID sequence.

3.55 REMOTE ID SEQUENCE?

Enter the default remote ID sequence in the same format as the local ID sequence (above). This can be repeated until a carriage return is entered.

NOTE

Dialogue Steps 3.56 to 3.69 are reserved for future use.

Step No.	Dialogue

3.70 DEVICE CLASSES?

Enter a list containing a device class name (up to eight alphanumeric characters, beginning with a letter). Class names are separated from each other by commas. These names are left to the discretion of the System Supervisor. They will be used in certain file commands or intrinsics when any member of a group of devices (such as any disc drive) can be referenced. *No name need be entered.*

A device can belong to more than one class, such as DISC and FHDISC. Only the classes DISC and SPOOL (if spooling is desired) are specifically required by MPE. DISC is the default device class for building files. SPOOL is the device class for designating "spooling discs." Spoolfiles will only be allocated on discs which are included in the special device class SPOOL. User files also may reside on spooling discs.

When the input is complete, enter a carriage return to return to Step 3.5.

NOTE

Dialogue Steps 3.71 to 3.79 are reserved for future use.

3.80 MAX # OF OPENED SPOOLFILES = <XXX>?

To change the maximum number of input and output spoolfiles which can be FOPENed at one time, enter the new limit. To retain the current value, enter a carriage return. This can be used to control the generation of output spoolfiles.

NOTE

Each concurrent batch job to be executed requires two spoolfiles — one for standard input and one for standard output. Thus, if you specify a maximum of 20 spoolfiles in response to the above prompt, the system is restricted to running ten concurrent jobs.

3.81 LIST I/O DEVICES?

To print a listing of the new input/output device configuration, enter YES. This list appears in the format described in Step 3.2.

To suppress the list, enter NO.

NOTE

The prompt in Step 3.82, below, appears only if you have configured a CS device into the system.

Step No.	Dialogue

3.82 LIST CS DEVICES?

Enter YES to list the characteristics of the new CS device configuration. Enter NO to suppress the listing.

3.83 CLASS CHANGES?

To add a class whose preferred order of device allocation can be specified or to add devices to previously defined classes, enter YES.

To avoid class changes and skip to Step 3.93, enter NO.

3.84 LIST CLASSES?

To list the device classes and the logical devices contained therein, enter YES.

To suppress the listing, enter NO.

The format of the listing is:

CLASS NAME	ACCESS TYPE	LOGICAL DEVICES

Where:

CLASS NAME shows the classes specified in the I/O configuration.

ACCESS TYPE is in the form:

IN - serial input (device types 8-31)
OUT - serial output (device types 16-39)
DA - Direct access (device types 0-7)
I/O,C - Input/output, concurrent devices (device types 16-23)
I/O,NC - Input/output, non-concurrent devices (device types 16-31)

LOGICAL DEVICES are the logical device numbers of all devices specified for this class in the I/O configuration.

3.85 DELETE CLASSES?

To delete previously defined classes, enter YES.

Otherwise, enter NO to skip to Step 3.87.

3.86 CLASSES?

Enter names, separated by commas, of classes to be deleted.

3.87 ADD CLASSES?

To define new classes or to add devices to previously defined classes, enter YES. To skip to Step 3.92, enter NO.

3.88 CLASS NAME?

To define a new class or to add devices to a previously defined class, enter class name.

To skip to Step 3.92, enter a carriage return.

3.89 LOGICAL DEVICE #S?

Enter logical device numbers, separated by commas, in the preferred order of allocation.

If it is not necessary to ask question 3.90, then a return is made to Step 3.88.

3.90 IN, OUT, OR IN/OUT?

Depending upon the types of the devices within the class, this question is asked to determine the desired device class access type.

Enter: IN
OUT
IN/OUT

If your response to this question was IN/OUT, then you are asked an additional question:

3.91 CONCURRENT OR NON-CONCURRENT?

Are all the devices in the class capable of concurrent IN/OUT?

Enter: NC
C

Return to Step 3.88.

3.92 LIST CLASSES?

To list the device classes and logical devices contained therein, enter YES.

To suppress the listing, enter NO.

Step No.	Dialogue
3.93	LIST I/O DEVICES? To print a listing of the new I/O configuration, enter YES. To suppress the listing, enter NO.

NOTE

The prompts in Steps 3.94 through 3.100, below, appear only if a CS device is configured or if additional drives exist (for the CS driver-changeable option). If neither case exists, the dialogue skips to Step 4.

3.94	ADDITIONAL DRIVER CHANGES? To prepare for additional driver changes, enter YES. To skip to Step 4, enter NO.

NOTE

The prompts in Steps 3.95 through 3.97, below, appear only if additional drivers are already configured; otherwise, the dialogue skips to Step 3.98.

3.95	LIST ADDITIONAL DRIVERS? To print a listing showing the presently-configured additional drivers, enter YES. To suppress this listing, enter NO.
3.96	DELETE DRIVERS? To delete an existing additional driver, enter YES. To skip to Step 3.98, enter NO.
3.97	DRIVER NAME? Enter the name of the driver to be deleted. This prompt is repeated until you enter a carriage return, or until all drivers are deleted.

Step No.	Dialogue
3.98	ADD DRIVERS? To prepare for adding drivers, enter YES. To skip to Step 3.100, enter NO.
3.99	DRIVER NAME? Enter the name of the driver to be added. This prompt is repeated until you enter a carriage return, or until the maximum of 32 drivers have been added.
3.100	LIST ADDITIONAL DRIVERS? To print a listing showing the presently-configured additional drivers, enter YES. To suppress this listing, enter NO.
4	SYSTEM TABLE CHANGES? To prepare for changing the CST, DST, PCB, IOQ, or MTAB, or other parameters relating to memory usage, enter YES. To bypass these changes, and proceed to Step 5, enter NO.
4.1	CST = <XXX>? To change the size of the shareable portion of the CST from XXX entries to another value, enter the new value. To retain the current value, enter a carriage return.
4.2	EXTENDED CST= <XXXX>? To change the size of the program portion of the CST from XXXX entries to another value, enter the new value. To retain the current value, enter a carriage return.
4.3	DST = <XXXX>? To change the size of the DST from XXXX entries to another value, enter the new value. To retain the current value, enter a carriage return.

Step No.	Dialogue
4.4	PCB = <XXX>.? To change the size of the PCB table from XXX entries to another value, enter the new value. To retain the current value, enter a carriage return.
4.5	I/O QUEUE = <XXX>.? To change the number of the input/output queue entries permitted from XXX entries to another value, enter the new value. To retain the current value, enter a carriage return.
4.6	TERMINAL BUFFERS = <XXX>? To change the number of terminal buffers in the system from XXX, enter the new value. To retain the current value, enter a carriage return.
4.7	SYSTEM BUFFERS = <XXX>? To change the number of system buffers in the system, enter the new value. To retain the current value, enter a carriage return.
4.8	MEMORY MANAGEMENT TABLE = <XXXX>? To change the size of the table from XXXX to another value, enter the new value. To retain the current value, enter a carriage return.
4.9	ICS = <XXXX>? To change the number of words in the interrupt control stack (ICS), enter the new value. To retain the current value, enter a carriage return.
4.10	UCOP REQUEST QUEUE = <XXX>.? To change the number of entries allowed in the user controller process request queue to another value, enter the new value. To retain the current value, enter a carriage return.

Step No.	Dialogue
4.11	TIMER REQUEST LIST = <XXX>.?
	To change the maximum number of concurrent time-out requests for the system clock allowed, enter the new value.
	To retain the current value, enter a carriage return.
4.12	BREAKPOINT TABLE = <XXX>.?
	To change the size of the breakpoint table from XXX entries, enter new value <256. To retain the current value, enter a carriage return.
5	MISC CONFIGURATION CHANGES?
	To prepare for the following miscellaneous configuration changes, enter YES:
	• Listing and (optionally) deleting global resource identification numbers (RIN's) assigned to users.
	• Number of RIN's available in the RIN pool.
	• Maximum number of global RIN's available.
	• Number of seconds allowed for logging-on.
	• Maximum number of jobs allowed on the system.
	• Maximum number of concurrent sessions allowed in execution.
	• Default central-processor time-limit for jobs.
	• Message catalog changes.
	To bypass these changes and proceed to Step 6, enter NO.
5.1	LIST GLOBAL RINS?
	To list the currently-assigned global resource identification numbers (RIN's), enter YES.
	To suppress this listing, enter NO.
	The listing consists of the RIN number and the name of the user and account to which it is assigned (for each RIN).

Step No.	Dialogue
5.2	DELETE GOBAL RIN? (RELOAD option only.) To prepare for deleting any of the currently-assigned global RIN's, enter YES. To bypass deletion and skip to Step 5.3, enter NO.
5.2.1	ENTER RIN NUMBER? To delete a currently-assigned global RIN, enter the RIN number. This step is repeated until a cariage return is entered.

NOTE

Since global RIN's are premanently assigned to users and the RIN numbers will be hard-coded into their programs, RIN's should be deleted with caution.

For this same reason the most up-to-date RIN table (which resides on disc) is used when the system is cold-loaded, except in the case of a RELOAD. This implies that any changes to the RIN table occurring during a :SYSDUMP operation, including changes to the size of the table, only take effect when the tape produced by :SYSDUMP is cold-loaded using the RELOAD option.

5.2.2	LIST GLOBAL RINS? To list the updated global RIN's (as in Step 5.1), enter YES. To suppress the listing, enter NO.
5.3	# OF RINS MIN = <YYY>, MAX = (XXXX).? To change the number of RIN's available in the RIN pool, enter a new value for XXXX. This value must be at least as great as YYY. (YYY is the maximum of 5 and the highest currently-assigned global RIN number.) To maintain the current maximum, enter a carriage return.
5.4	# OF GLOBAL RINS USED = <YYY>.MAX = <XXXX>.? To change the maximum number of global RIN's available, enter a new value for XXXX. Because of the current assignment of global RIN numbers, this must be at least as great as YYY. To maintain the current value, enter a carriage return.

Step No.	Dialogue
5.5	# OF SECONDS TO LOG ON = <XXX>.?
	To change the number of seconds allowed for logging-on, enter the new value.
	To retain the current value, enter a carriage return.
5.6	MAX # OF CONCURRENT RUNNING SESSIONS = <XXXX>,?
	To change the maximum number of sessions allowed in execution at one time, enter the new value.
	To retain the current value, enter a carriage return.
5.7	MAXIMUM # OF CONCURRENT RUNNING JOBS = <XXX>.?
	To change the maximum number of jobs allowed in execution at one time, enter the new value.
	To retain the current value, enter a carriage return.
5.8	DEFAULT JOB CPU TIME LIMIT = <XXXXX>.?
	To change the value, enter the new value. A zero implies that jobs are not limited; sessions are limited only if the user supplies a limit on the :HELLO command.
	To retain the current value, enter a carriage return.
5.9	LIST MESSAGE CATALOG?
	To list the current message catalog, enter YES.
	To suppress the listing, enter NO.
5.10	MESSAGE CATALOG CHANGES?
	To create a new message catalog from supplied file, enter YES.
	To skip to Step 6, enter a carriage return.
5.10.1	CATALOG INPUT FILE NAME?
	Name of disc file or formal designator to define the input file from which new catalog is to be built.

Step No.	Dialogue
5.10.2	LIST MESSAGE CATALOG? To list new catalog, enter YES. To suppress the listing, enter NO.
6	LOGGING CHANGES? To prepare for changes to the logging characteristics of the system, enter YES. To bypass such changes and proceed to Step 7, enter NO.
6.1	LIST LOGGING STATUS? To print a list of the events that can be logged and whether or not they are currently being logged, enter YES. To suppress the listing, enter NO.
6.2	STATUS CHANGES? To prepare for changes to the logging status, enter YES. If no changes are desired, enter NO to skip to Step 6.3.
6.2.1	ENTER TYPE, ON/OFF? You should enter the type number of the event (defined below), a comma, and ON to signify that it is to be logged or OFF to signify that it is not. The following Events may be logged:

Type No.	Event
1	Logging enabled
2	Job initiation
3	Job termination
4	Process termination
5	File close
6	System shutdown
7	Power failure
8	Spooling log record
9	Line disconnection
10	Line close
11	I/O error

Step No.	Dialogue

NOTE

Event 1 must be ON for any logging to take place. If event 2 is on, the default input priority for jobs and sessions is 8; if event 8 is on, this default output priority is 8. Otherwise, the normal default is 13 for both input and output priorities.

Step 6.2.1 is repeated until a carriage return is entered.

6.2.2 LIST LOGGING STATUS?

To list the updated logging status, respond with YES. To suppress the listing, enter NO.

6.3 LOG FILE RECORD SIZE (SECTORS) = <XX>?

To change the value of the log file physical record size, enter the number of sectors desired. This number determines the size of the buffer for entries in the log file. (A sector is equal to 128 words.)

To retain the current value, respond with a carriage return.

6.4 LOG FILE SIZE (RECORDS) = <XXXXX>.?

To change the maximum number of physical records permitted in the log file, enter a new value. The log file has 16 extents, so each extent will contain:

$$\left(\frac{\text{(log file size)}}{16} \times \text{(log file record size)} \right) \text{ sectors of disc space.}$$

To retain the present value, enter a carriage return.

7 DISC ALLOCATION CHANGES?

To prepare for disc allocation changes, enter YES.

To bypass such changes and proceed to Step 8, enter NO.

7.1 VIRTUAL MEMORY = <XXXXX>.? (RELOAD option only.)

To change the size of the area on disc used for virtual memory from XXXXX sectors to another value, enter the new value.

To retain the current value, enter a carriage return.

Step No.	Dialogue
7.2	DIRECTORY USED =<YYYY>,MIN = <ZZZZ>,MAX = <XXXX>.?

To change the maximum size of the directory from XXXX sectors, enter the new value; YYYY specifies the amount of directory currently used; ZZZZ specifies the minimum value to which XXXX can be set. (ZZZZ will often be greater than YYYY due to unused areas that are not at the end of the space allotted to the directory.) Maximum size cannot exceed 6000 sectors.

To retain the present maximum size, enter a carriage return.

7.3 LIST VOLUME TABLE?

To list the disc volumes and their currently-assigned logical device numbers, enter YES. The listing is printed in the following format:

VOLUME	*LOG DEV #*
volname	*ldn*
.	.
.	.
.	.

In this listing, *volname* is a name of up to eight alphanumeric characters, beginning with a letter, identifying the volume; *ldn* is the logical device number assigned to that volume.

To suppress this listing, enter NO.

7.4 DELETE VOLUME? (RELOAD option only.)

To prepare to delete a volume, enter YES.

To bypass deletion and skip to Step 7.5, enter NO.

7.4.1 ENTER VOLUME NAME? (RELOAD option only.).

To delete a volume, enter the volume name. (When the name is entered, the question is repeated.)

Otherwise, enter a carriage return.

Step No.	Dialogue
7.5	ADD VOLUME?
	To prepare to add a volume, enter YES.
	To bypass addition and skip to Step 7.6, enter NO.
7.5.1	ENTER VOLUME NAME
	To add a volume, enter the volume name. (When the name is entered, the question is repeated.)
	Otherwise, enter a carriage return.
7.6	LIST VOLUME TABLE?
	To list the disc volumes and their currently assigned logical device numbers (as in Step 7.3), enter YES. In this listing, volumes just added (in Step 7.5) will have logical device numbers of zero.
	To suppress this listing, enter NO.
7.7	MAX # OF SPOOLFILES KILOSECTORS = <XXXXXX>.?
	To change the maximum number of sectors which can be allocated to spoolfiles (expressed in thousands of sectors), enter the new value.
	To retain the current limit, enter a carriage return.
7.8	# OF SECTORS PER SPOOLFILE EXTENT?
	Enter the size, in sectors, for each spoolfile extent. This must be a value between 128 and 1024.
8	SCHEDULING CHANGES?
	To prepare for changes to the scheduling queue, enter YES.
	To bypass these changes and proceed to Step 9, enter NO.
8.1	TIME QUANTUM = <XXXXX>.?
	To change the time quantum in milliseconds, enter the new value. To keep the old value, enter a carriage return.
8.2	TERMINAL PRIORITY = <XXX> ?
	To change the priority assigned to a time-shared process when it completes a terminal read operation, enter the new value. To keep the old value, enter a carriage return.

Step No.	Dialogue
8.3	CS PRIORITY LIMIT = <XXX> ? To change to the lowest priority to which a process in the CS subclass can fall, enter the new value. To retain the old value, enter a carriage return.
8.4	DS PRIORITY LIMIT = <XXX> 0 To change to the lowest priority to which a process in the DS subclass can fall, enter the new value. To retain the old value, enter a carriage return.
9	SEGMENT LIMIT CHANGES? To prepare for changing the limits on code and data segments, enter YES. To retain the current limits and skip to Step 10, enter NO.
9.1	MAX # OF CONCURRENT RUNNING PROGRAMS = <XXX>.? To change the maximum number of concurrent running programs from XXX, enter the new value. To retain the current value, enter a carriage return.
9.2	MAX CODE SEG SIZE = <XXXXX>.? To change the maximum number of words allowed in any code segment from XXXXX, enter the new value. To retain the current value, enter a carriage return.
9.3	MAX # OF CODE SEGMENTS/PROCESS = <XX>.? To change the maximum number of code segments allowed any user process, enter the new value. To retain the current value, enter NO.
9.4	MAX STACK SIZE = <XXXXX>.? To change the maximum number of words allowed in any user stack from XXXXX, enter the new value. (A maximum value of 31232 is permitted.) To retain the current value, enter a carriage return.

Step No.	Dialogue
9.5	MAX EXTRA DATA SEG SIZE = <XXXXX>.?
	To change the maximum number of words allowed in any extra data segment from XXXXX, enter the new value.
	To retain the current value, enter a carriage return.
9.6	MAX # OF EXTRA DATA SEGMENTS/PROCESS = <XXX>.?
	To change the maximum number of extra data segments that a process can have, enter the new value.
	To retain the current value, enter a carriage return.
9.7	STD STACK SIZE = <XXXX>.?
	To change the number of words initially assigned for a user stack (Z-Q area) by default (when the user specifies no value) at preparation time from XXXX, enter the new value.
	To retain the current value, enter a carriage return.
10	SYSTEM PROGRAM CHANGES?
	To prepare to replace a program belonging to the system, enter YES.
	To proceed directly to Step 11, enter NO.
10.1	ENTER PROGRAM NAME, REPLACEMENT FILE NAME?
	To replace a program belonging to the system, enter the name of the program, a delimiting comma, and the name of the program file which is to replace the program. The replacement program need not be in the public group of the system account, PUB.SYS. However, a fully qualified program file name will be required if the program file is not in the logon account/group structure.
	The question is repeated until a carriage return is entered.
11	SYSTEM SL CHANGES?
	To prepare for changes to the System Library (SL.PUB.SYS) enter YES. Otherwise, enter NO to skip to Step 12.
11.1	LIST LIBRARY?
	To list the names of the code segments in the System Library and their entry-points and external procedures, enter YES.
	To suppress this listing, enter NO.

Step No.	Dialogue
11.2	DELETE SEGMENT?

To prepare for deleting a code segment from the System Library (SL.PUB.SYS), enter YES.

To proceed directly to Step 11.3, enter NO.

11.2.1 ENTER SEGMENT NAME?

To delete a code segment from the System Library, enter the name of that segment. (When the segment name is entered, the question is repeated.)

Otherwise, enter a carriage return.

11.3 REPLACE SEGMENT?

To prepare for replacing a code segment in the System Library, enter YES.

To proceed directly to Step 11.4, enter NO.

11.3.1 ENTER SEGMENT NAME,USLFILE NAME [,S/C/P]?

To replace a code segment in the System Library, enter the name of the segment; a delimiting comma; and the name of the USL file where the replacement segment can be found. Also, optionally, enter a delimiting comma followed by one of these three characters:

S To declare the segment to be a permanently-allocated system intrinsic segment (in virtual memory).

C To declare the segment to be a main-memory resident system intrinsic segment.

P To declare the segment to be a permanently-allocated user segment (in virtual memory). (This option requests the same function as the :ALLOCATE command, defined in Section II.)

The question then is repeated.

Otherwise, enter a carriage return.

NOTE

If you enter a USLLFILE name which is in error (typographic input error), an error message results and the Configurator proceeds back to Step 11.2. In this regard the segment which was to have been replaced has, in fact, been *deleted*. You may proceed to Step 11.4 and attempt to add the USL file.

Step No.	Dialogue
11.4	ADD SEGMENT?

To prepare for adding a code segment to the System Library, enter YES.

Otherwise, enter NO to skip to Step 11.5.

11.4.1 ENTER SEGMENT NAME, USLFILE NAME [,S/C/P]?

To add a code segment to the System Library, enter the name of the segment; a delimiting comma; the name of the USL file where the segment can be found. Also, optionally, enter a delimiting comma followed by one of these three characters:

S To declare the segment to be a permanently-allocated system intrinsic segment (in virtual memory).

C To declare the segment to be a main-memory resident system intrinsic segment.

P To declare the segment to be a permanently-allocated user segment (in virtual memory). (This option requests the same function as the :ALLOCATE command, defined in Section II.)

The question then is repeated.

Otherwise, enter a carriage return.

NOTE

If you enter a USL file name which is in error (any error), an error message results and the Configurator proceeds back to Step 11.2.

11.5 LIST LIBRARY?

To list the updated System Library, enter YES.

To suppress this listing, enter NO.

12 ENTER DUMP DATE?

To copy only the modified operating system to tape, enter a carriage return; the dialogue skips to Step 13.

To copy the MPE system, the current accounting structure, and all files to tape, enter 0. This tape can then be used to RELOAD the system (Console Operator function). The tape can also be used with the :RESTORE command (described in Section II) to retrieve a file.

Step No.	Dialogue

To copy the MPE system, the current account structure, and any files that were changed on or after a particular date, enter that date in the format *mm/dd/yy*. (In this format, *mm, dd,* and *yy* are one or two decimal digits representing the month, day, and year, respectively.) This tape can be used in conjunction with other tapes to RELOAD the system, or to retrieve one or more files by using the :RESTORE command.

NOTE

Because files in use with write, append, update, or read/write access will not be copied, system back-up should be performed only when no users are logged onto the system.

12.1 LIST FILES DUMPED?

To obtain a list showing the name of each file copied, enter YES. To suppress this list, enter NO. The optional listfile parameter to the :SYSDUMP command is useful to direct the list to a high speed printing device when it is known that a great many files exist in the system. A list showing the number (count) of files copied, the number of files not copied, the names of the files not copied, and the reasons why they were not copied is always provided. The formats for these listings are:

```
FILES DUMPED = XXX

 FILE    .GROUP    .ACCOUNT      LDN   ADDRESS
                      .
                      .
                      .
FILES NOT DUMPED = XX

 FILE    .GROUP    .ACCOUNT FILESET        REASON
                      .
                      .
                      .
```

13 The operator is now requested to assign the magnetic tape device on which you have arranged for a fresh magnetic tape to be mounted. After operator assignment, the system is copied to tape (multi-reel files). It then can be loaded and initialized as directed under the heading *System Start-Up and Modification.*

Step No.	Dialogue

Check the list of files not dumped for user files you want to save. Frequently the files; LOADLIST, MEMLOG, SL, and LOGXXXX are open when SYSDUMP is running, so their names often appear on the list of files not dumped. By a special process, SYSDUMP records the system file, SL, on tape, so it is saved. (See Appendix F for the list of system files dumped.) The other three files mentioned above are recreated from scratch by the Initiator each time the system is started.

If any file belonging to the system is not copied, the message:

WARNING FOLLOWING SYSTEM FILES NOT DUMPED

is issued and the file name and the reason it was not copied is listed. If the file was to replace a system program, the program name follows in parentheses.

If a response other than a carriage return was entered in answer to the ENTER DUMP DATE? question in Step 12, the list and count of files will be provided as described in Step 12.1.

To denote termination of the Configurator/User Dialogue, the following message is printed:

END OF SUBSYSTEM

Table A-1. Synchronous Single-Line Controller (CSSBSC0)

CONFIGURATOR STEP NO.	CONFIGURATOR OUTPUT	USER RESPONSE
3.9	TYPE?	18
3.10	SUBTYPE?	0 or 1
3.14	PROTOCOL?	1
3.15	LOCAL MODE?	1 or 2
3.16	TRANSMISSION CODE?	1, 2, or 3
3.17	RECEIVE TIMEOUT?	CARRIAGE RETURN
3.18	LOCAL TIMEOUT?	CARRIAGE RETURN
3.19	CONNECT TIMEOUT?	CARRIAGE RETURN
3.20*	DIAL FACILITY?	YES or NO
3.21*	ANSWER FACILITY?	YES or NO
3.22*	AUTOMATIC ANSWER?	YES or NO
3.23	DUAL SPEED?	YES or NO
3.25	SPEED CHANGEABLE?	NO
3.26	TRANSMISSION SPEED?	(ENTER LINE SPEED)
3.27	TRANSMISSION MODE?	0 or 1
3.28	PREFERRED BUFFER SIZE	576
3.29	DRIVER CHANGEABLE?	NO
3.30	DRIVER OPTIONS?	0
3.50	DRIVER NAME?	CSSBSC0
3.51*	CONTROL LENGTH	0
.	.	
.	.	
.	.	
10.1	ENTER PROGRAM NAME, REPLACEMENT FILE NAME?	CSSBSC0, <prepared file name>

* This question is asked only if subtype is 0.

Table A-2. Hardwired Serial Interface (CSHBSC0)

CONFIGURATOR STEP NO.	CONFIGURATOR OUTPUT	USER RESPONSE
3.9	TYPE?	19
3.10	SUBTYPE?	3
3.13	PORTMASK?	1, 2, 4 or 8
3.14	PROTOCOL?	1
3.15	LOCAL MODE?	1
3.16	TRANSMISSION CODE?	2
3.17	RECEIVE TIMEOUT?	CARRIAGE RETURN
3.18	LOCAL TIMEOUT?	CARRIAGE RETURN
3.19	CONNECT TIMEOUT?	CARRIAGE RETURN
3.25	SPEED CHANGEABLE?	YES
3.26	TRANSMISSION SPEED?	250000 or 125000
3.27	TRANSMISSION MODE?	0
3.28	PREFERRED BUFFER SIZE?	576
3.29	DRIVER CHANGEABLE	NO
3.30	DRIVER OPTIONS?	0
3.50	DRIVER NAME?	CSHBSC0
10.1	ENTER PROGRAME NAME, REPLACEMENT FILE NAME?	CSHBSC0, <prepared file name>

Table A-3. DS/3000 Communications Driver (IODS0)

CONFIGURATOR STEP NO.	CONFIGURATOR OUTPUT	USER RESPONSE
3.9	TYPE?	41
3.10	SUBTYPE	0
3.40	RECORD WIDTH	128
3.41	OUTPUT DEVICE	0
3.42	ACCEPT JOBS/SESSIONS	NO
3.43	ACCEPT DATA	NO
3.44	INTERACTIVE	NO
3.45	DUPLICATIVE	NO
3.46	INITIALLY SPOOLED	NO
3.50	DRIVER NAME	IODS0
10.1	ENTER PROGRAM NAME, REPLACEMENT FILE NAME	IODS0, <prepared file name>

Table A-4. Pseudo Terminal (IODSTRM0)

CONFIGURATOR STEP NO.	CONFIGURATOR OUTPUT	USER RESPONSE
3.9	TYPE?	16
3.10	SUBTYPE?	0
3.11	TERMTYPE?	CARRIAGE RETURN
3.12	SPEED IN CHARACTERS PER SECOND?	CARRIAGE RETURN
3.40	RECORD WIDTH?	36
3.41	OUTPUT DEVICE?	(No. of corresponding listing device)
3.42	ACCEPT JOBS/SESSIONS?	YES
3.43	ACCEPT DATA?	NO
3.44	INTERACTIVE?	YES
3.45	DUPLICATIVE?	YES
3.46	INITIALLY SPOOLED?	NO
3.50	DRIVER NAME?	IODSTRM0
10.1	ENTER PROGRAM NAME, REPLACEMENT FILE NAME?	IODSTRM0, <prepared file name>

ERROR CODES AND MESSAGES

APPENDIX B

The following is a summary of the error code messages that may be encountered together with their meanings. The messages, as listed here, have been grouped according to their associated activities. For this reason, some messages are listed under more than one heading.

COMMUNICATION LINK ERRORS

These errors may appear while doing the DSLINE, Remote File Access, or Program-to-Program calls. The method of reporting depends on the operation in progress when the error occurs.

If doing commands, the error will appear as a

60, nnn

If doing Remote File Access, it will be returned by the FCHECK intrinsic.

If doing Program-to-Program, it will be retuned by the PCHECK intrinsic.

Error Code	Meaning	Corrective Action
242	You have an internal DS error. (Probably a software malfunction.)	Contact your HP Customer Engineer.
243	The remote computer did not respond within 16 attempts to establish the line (48 seconds).	

Error Code	Meaning	Corrective Action
244	There is a software problem in DS at either end of the line.	
245	Receive timeout.	
246	Remote disconnect.	
247	Local timeout.	
248	Connect timeout.	
249	Remote rejected connection.	
250	Carrier lost.	
251	Data set not ready.	
252	DS detected a hardware problem with the particular HSI interface board.	Contact your HP Customer Engineer.
253	No operator response to dial.	
254	The requested HSI was not configured properly during system configuration.	Re-configure HSI.
255	Unanticipated error condition.	

LINE OPENING FAILURES

The various error numbers and messages for line opening (DSLINE) failures are as follows:

Error Number	Message	Meaning	Corrective Action
1	UNKNOWN COMMAND	The command entered was not recognized as a legal command. You probably mistyped the command name (DSLINE or REMOTE HELLO).	Re-enter the command and be sure that the command name is entered correctly.
20,x	SYNTAX ERROR	A delimeter in the parameter list is not permitted in the command, or is not permitted at the point where it was detected. When a qualifying number is shown (x), the bad delimeter is adjacent to the indicated parameter element (usually following it).	Re-enter the command correctly.
22,x	ILLEGAL PARAMETER	A parameter in the command was not recognized as legal. The qualifying number (x) indicates which parameter was bad.	Check the parameter list, and re-enter the command correctly.
26,x	ILLEGAL KEYWORD	A keyword in the command was not recognized as legal. The qualifing number (x) indicates which keyword was bad.	Check the keywords, and re-enter the command correctly

Error Number	Message	Meaning	Corrective Action
27,x	DUPLICATE KEYWORD	The command was rejected because the same keyword appeared twice in the parameter list. The qualifying number (x) indicates the duplicate keyword.	Re-enter the command without the duplicate keyword.
30,x	INVALID NUMBER	The buffer-size in the LINEBUF parameter is not decimal or is out of range (304 to 4096).	Check the parameter list, and re-enter the command correctly.
60,55	DS/3000 ERROR	Someone already has exclusive access to the line. ***OR*** You requested exclusive access to the line and someone is already using it. ***OR*** The specified HSI has been shut down by the console operator.	Re-enter the command specifying a different HSI or get the console operator to bring up the particular HSI.
60,56	DS/3000 ERROR	You gave an erroneous line specification (dsdevice). There is no HSI with the specified device class name or logical device number.	Re-enter the command using a correct dsdevice parameter.

Error Number	Message	Meaning	Corrective Action
60,201	DS/3000 ERROR	You supplied a remote ID sequence (REMID=) that was rejected by the remote system. ***OR*** The specified SSLC is already open and you did not specify the currently active remote ID sequence (REMID=).	Check the remote ID (REMID=) and re-enter the ID correctly.
60,202	DS/3000 ERROR	You specified a telephone number (PHNUM=) that was either too long or contained something other than dashes and digits.	Check the telephone number (PHNUM=) and re-enter the number correctly.
60,204	DS/3000 ERROR	You used line opening parameters (PHNUM=, LOCID=, REMID=, or EXCLUSIVE) in a a DSLINE;CLOSE command.	Re-enter the DSLINE;CLOSE command omitting PHNUM=, LOCID=, REMID= or EXCLUSIVE parameters.
60,216	DS/3000 ERROR	REMOTE HELLO rejected the connection because the remote system does not have anymore virtual terminals available.	

Error Number	Message	Meaning	Corrective Action
60,243	DS/3000 ERROR	The remote computer did not respond within 16 attempts to establish the line (48 seconds).	
60,244	DS/3000 ERROR	There is a software problem in DS at either end of the line.	
60,252	DS/3000 ERROR	DS detected a hardware problem with the particular HSI interface board.	Contact your HP Customer Engineer.
60,254	DS/3000 ERROR	The requested HSI was not configured properly during system configuration.	Re-configure HSI.

ERRORS ON REMOTE HELLO

Error Code	Meaning	Corrective Action
60,55	The device is not available. The operator has not executed a =DSLINE console command; the line is opened exclusively; or the line is in use and you asked for EXCLUSIVE access.	Ask the console operator to enter the =DSLINE command.
60,56	You specified an invalid dsdevice. You probably mistyped the device name.	Re-enter the dsdevice and be sure the device class name is typed right.
60,214	You issued a REMOTE HELLO command before the DSLINE command has been issued.	Enter the DSLINE command before issuing the REMOTE HELLO command.
60,216	REMOTE HELLO rejected the connection because the remote system does not have anymore virtual terminals available.	
60,243	Remote is not responding. You did not open a DS line at the remote site or the hardware connection is faulty.	Open the DS line at the remote site. If this fails contact your HP Customer Engineer; it may be a faulty hardware connection.

REMOTE FILE ACCESS ERRORS

These error messages are returned by the DSCHECK intrinsic (in addition to any of the communication link errors).

Error Code	Meaning	Corrective Action
55	The device is not available. The operator has not executed a =DSLINE console command.	Ask the console operator to enter the =DSLINE command.
56	You specified an invalid dsdevice. You probably mistyped the dsdevice name.	Re-enter the dsdevice and be sure the device class name is typed right.
214	You issued an FOPEN to a remote device prior to executing a REMOTE HELLO.	Enter a REMOTE HELLO prior to issuing any instruction to a remote device.
224	You entered an invalid combination of file equations. (FOPEN cannot execute without recursing.)	Check your file equations and re-enter the equation correctly.

PROGRAM-TO-PROGRAM ERRORS

These error messages are returned by the PCHECK intrinsic (in addition to any of the communication link errors).

Error Code	Meaning	Corrective Action
206	You specified a slave PTOP function from a Master program on the same line.	Check your program.
207	A slave function was out of sequence (must be GET followed by ACCEPT or REJECT).	Recode the slave program.
208	You specified a master PTOP function from a Master program on the same line as the slave funtions.	Check your program.
209	You specified a program by POPEN that does not exist on the slave.	Check your program.
211	The slave has issued a reject.	

Error Code	Meaning	Corrective Action
212	You specified an ID number that is not for DS. (This only happens if the slave is doing a No-wait I/O.)	Check your program.
213	An error on the first GET intrinsic was issued by the slave (POPEN).	
219	Too many POPENs issued by the master (>4).	
222	You entered a PTOP request prior to doing a POPEN (sequence error).	Check your program.
223	You specified an invalid PTOP buffer-size. PREAD or PWRITE requested a larger buffer than was specified in the POPEN.	Specify a larger buffer-size in the POPEN intrinsic.

ERRORS REPORTED TO CONSOLE OPERATOR

These errors are reported to the console operator when a user attempts to establish a communication link.

Error Code	Meaning	Corrective Action
101	Non-responding device.	
102	Transfer error.	
103	Data set not ready.	
104	Carrier loss.	
105	Data overrun.	

DSLINE CONSOLE OPERATORS COMMAND

APPENDIX C

In order to establish communications links between HP 3000 and HP 1000 Computers, the console operator first must initiate the =DSLINE command before you can access a specified communcations line. The =DSLINE command allows you to enable or disable the DS subsystem on the communication link.

For easy reference this command is shown in the following format:

- SYNTAX — Shows the format of the command.
- PARAMETERS — Describes the variables in the command.
- NOTES — Describes in detail the command.
- EXAMPLES — Shows the command in use.

Enables or disables the DS subsystem on the communication link.

=DSLINE

```
                  { {OPEN} ,[[MASTER] [,speed]]                              }
                  { {SHUT}   [[SLAVE ]        ]]                             }
=DSLINE   ldn     { ,                                                        }
                  {  TRACE { , ON [,[ALL] [,[mask] [,[numentries]            }
                  {        {       [,[wrap][,filename]]]]]                   }
                  {        {  OFF                                            }
```

Parameters

ldn	The logical device number of the DS line device. (Required parameter).
OPEN	Allows the DS line to OPEN a corresponding CS line. (Required parameter.)
SHUT	Allows the DS line to CLOSE a corresponding CS line. (Required parameter.)
MASTER	Allows the local HP 3000 to process only local master requests. (Optional parameter.)
SLAVE	Allows the local HP 3000 to process only remote slave requests.

Note: OPEN,MASTER = SHUT,SLAVE
OPEN,SLAVE = SHUT,MASTER
SHUT,MASTER = OPEN,SLAVE
SHUT,SLAVE = OPEN,MASTER

If neither MASTER or SLAVE is specified the default is both MASTER and SLAVE processing allowed.

(Optional parameter.)

speed — The transmission speed in characters per second. This parameter will be effective if system generation selected SPEED is changeable. (Optional parameter.)

TRACE — Allows CS TRACE to be activated or deactivated. (Required parameter.)

ON — Allows TRACE to be activated. (Required parameter.)

OFF — Allows TRACE to be deactivated. (Required parameter.)

ALL — Generates trace records for all line activity. If ALL is not specified, the trace record will be written only when a transmission error occurs. (Optional parameter.)

mask — An octal integer preceded by a sign (nn). (Optional parameter.)

numentries — A decimal integer for the maximum number of trace entries in a trace record (not greater than 512). (Optional parameter.)

wrap — Causes trace entries that overflow the trace area (greater than numentries) to over lay the prior trace entries. (Optional parameter.)

filename — A trace filename (default file CSTRACE in PUB. SYS). (Optional parameter).

Notes

By initiating this console operator command, it will allow other subsystems to use the communication line or allow the use of anothe subchannel on the HSI. It also allows the local HP 3000 to process Master/Slave requests.

Examples

To allow DS line number 55, enter:

```
=DSLINE 55,OPEN
```

To allow the local HP 3000 to process only local master requests from DS line number 55, enter:

```
=DSLINE 55,MASTER
```

To activate TRACE from DS line number 55, enter:

```
=DSLINE 55,TRACE,ON
```

To activate TRACE with a maximum of 250 entries in a trace record, enter:

```
=DSLINE 55,TRACE,ON,,,250
```

DS TRACE FACILITY

APPENDIX D

The DS Trace Facility can provide a record of the line actions, states and events that occur during DS operations. When problems occur during operation, the trace facility provides the means to pinpoint the problem area.

The trace facility is invoked at the user's request. Tracing can be invoked for any communication line. Once invoked for a particular communications line, the trace facility continues to record line activity until either the line is terminated or the operator issues a new trace command. The trace facility keeps track of actions, states and events in a user-defined trace file. The contents of this file may be dumped once the trace is completed. The trace file is of particular importance for logical trouble shooting during program debug, as well as the isolation and troubleshooting of system malfunctions.

INVOKING THE TRACE FACILITY

To invoke the trace facility, use the following format of the =DSLINE console operator command:

```
=DSLINE  ldn,TRACE { , ON  [,[ALL] [,[mask] [,[numentries]
                           [,[wrap] [,filename]]]]]
                   { OFF
```

where:

ldn — The logical device number of the DS line device.

TRACE — Allows TRACE to be activated or deactivated.

ON	Allows TRACE to be activated.
OFF	Allows TRACE to be deactivated.
ALL	Generates tarce records for all line activity. If ALL is not specified, then the trace record will be written only when a transmission error occurs.
mask	An octal integer preceded by a sign (nn).
numentries	A decimal integer for the maximum number of trace entries in a trace record. May not be greater than 512.
wrap	Causes trace entries that overflow the trace area to over lay the prior trace entires.
filename	A TRACE filename. Default is CSTRACE in PUB.SYS.

DUMPING THE TRACE FILE

To dump the contents of the trace file, enter the following commands:

```
:FILE CSTRACE=filename
:FILE LIST=LP
:RUN CSDUMP.PUB.SYS
```

These commands cause a listing of the contents of the trace file to be printed on the line printer for later examination.

EXAMPLE

To activate trace from logical device 55 and generate trace records with a maximum of 250 entries in a record, enter:

```
=DSLINE 55,TRACE,ON,ALL,,128
```

SYSTEM VERIFICATION TEST

APPENDIX E

Both the system software and the physical link connecting the computers can be tested with a diagnostic program called DSTEST. DSTEST conducts a simple, yet effective, test of the system, including Remote File Access (RFA) and Program-to-Program Communications (PTOP).

DSTEST can be run in either of two modes: the Diagnostic mode or the Normal mode. In the Diagnostic mode of DSTEST, you can select the number of passes, the word pattern to be transmitted, the mode of transmission, and the block size. In the Normal mode, DSTEST automatically assigns typical values for each option.

NOTE

To perform DSTEST you must have a remote session. Also, remote command processing can be used indendently of DSTEST for checkout of the various system configurations.

DIAGNOSTIC MODE

To run the diagnostic mode perform the following steps:

1. Enter the following line to initiate the linetest:

```
:RUN DSTEST,DIAG
```

If you are testing RFA, then a :FILE command is required to direct the file accessed to the proper DS line before initiating the DSTEST:

```
:FILE REMOTE;DEV=dsdevice#DISC
```

2. Answer the following questions:

.RFA AND PTOP?

Enter RFA for Remote File Access or enter PTOP for Program-to-Program.

.DSLINE?

Enter the device class or logical device number that was assigned to IODS0 during system configuration.

.NUMBER OF PASSES?

Enter the number of passes desired, up to a maximum of 32797 (decimal). Zero or a carriage return causes a default value of one pass to be used.

.PATTERN?

Enter an octal word to be transfered.
Note: Illegal input causes the message

INPUT ERROR

to be printed. Enter a correct value or enter a carriage return to the default value 177777 (the sign must be entered).

.BLOCKSIZE?

Enter the desired blocksize of the transfer (<4096). If a value greater than 4096 is entered, an error message will be printed.

.CONTINUE(Y/N)?

Enter an affirmative response (Y) to return to the beginning of the option selection phase if you wish to repeat the cycle, or enter a negative response (N) to terminate the test.

NORMAL MODE

To run the normal mode perform the following steps:

1. Enter the following line to initiate the DSTEST:

```
:RUN DSTEST,DIAG
```

   In the normal mode, you are not required to select options; the default values are automatically used.

2. Answer the following question:

   .DSLINE?

   Enter the device or class logical device number (decimal) that was assigned to IODS0 during system configuration.

NOTE

The normal mode default is a 512 word program-to-program transfer or all 177777.

EXAMPLE

```
:HELLO BARB.LEWIS

 SESSION NUMBER = #S13
 MON, FEB 222, 1977, 10:50 AM
 HP32002A.00.A1

WELCOME TO SYSTEM A.

:DSLINE HDS2
DS LINE NUMBER = #L3


:REMOTE HELLO RBARB.RLEWIS
 SESSION NUMBER = #S35
 MON, FEB 22, 1977, 1051 AM
 HP32002A.00.A1

WELCOME TO SYSTEM B.

:REMOTE

#FILE REMOTE;DEV=CPU3#DISC
#RUN DSTEST.DIAG

HEWLETT PACKARD 32195A.00.0 DSTEST/3000 MON, FEB 22, 1977, 10:54 AM
.RFA OR PTOP? RFA
.NUMBER OF PASSES? 22
.PATTERN? %11
.BLOCK SIZE? 256
256 WORD REMOTE RECS WRITTEN/READ: 22 ,SECS: 42.012 ,AVE; 1.909
.CONTINUE(Y/N)? Y
.RFA OR PTOP? PTOP
.NUMBER OF PASSES? 22
.PATTERN? %22
.BLOCK SIZE? 512
256 WORD PROG TO PROG WRITES DONE: 12 ,SECS: 9.524 ,AVE: 1.626
.CONTINUE(Y/N)? N

 END OF PROGRAM
:REMOTE BYE


CPU (SEC) = 5
CONNECT (MIN) = 4
MON, FEB 22, 1977, 10:54 AM
END OF SESSION
```

MODEM OPTIONS

APPENDIX F

Tables F-1 through F-6 present the Bell System options for six MODEMs which can be used with the HP DS Network. Whenever possible, recommendations for which option are also shown.

Type of MODEM: Bell System Type 201A3 Data Set.

Type of Line: Public Telephone Network (Switched).

Transmission Rate: 2000 bits-per-second.

Table F-1. 201A3 Options and Recommendations

OPTION NUMBER	DESCRIPTION	RECOMMENDATION
A1 A2	EIA interface. Contact interface.	A1 (required)
B3 B4	With alternate voice. Without alternate voice.	B3 *
C5 C6	With new sync. Without new sync.	C6 (required)
D7 D8	Half duplex (2-wire). Full duplex (4-wire).	D7
E9 E10	4-wire continuous carrier. 4-wire carrier controlled by REQUEST TO SEND.	**

* If option B3 is selected and automatic answering is to be used, the automatic answering capability is normally provided as a key-control function.

** If option D7 is selected, the E options have no meaning and should be ignored.

Type of MODEM: Bell System Type 201B3 Data Set.

Type of Line: Private Leased Line.

Transmission Rate: 2400 bits-per-second.

Table F-2. 201B3 Options and Recommendations

OPTION NUMBER	DESCRIPTION	RECOMMENDATION
A1 A2	EIA interface. Contact interace.	A1 (required)
B3 B4	With alternate voice. Without alternate voice.	B3 *
C5 C6	With new sync. Without new sync.	C6 (required)
D7 D8	Half duplex (2-wire). Full duplex (4-wire).	D8
E9 E10	4-wire continuous carrier. 4-wire carrier controlled by REQUEST TO SEND.	E9 **

* If option B3 is selected and automatic answering is to be used, the automatic answering capability is normally provided as key-controlled function.

** If Option D7 is selected, the E options have no meaning and should be ignored.

Type of MODEM: Bell System Type 201C Data Set (also called DATAPHONE 2400).

Type of Line: Public Telephone Network (Switched) or Private Leased Line.

Transmission Rate: 2400 bits-per-second.

Table F-3. 201C Options and Recommendations

OPTION NUMBER	DESCRIPTION	RECOMMENDATION
A1	Transmitter internally timed.	A1 (required)
A2	Transmitter externally timed.	
B3	Without 801 Automatic Calling Unit.	B3
B4	With 801 Automatic Calling Unit.	
C5	EIA interface.	C5 (required)
C6	Contact interface.	
D7	Without automatic answer.	D8
D8	With automatic answer.	
E9	Automatic answer permanently wired.	Either *
E10	Automatic answer key-controlled.	

* If Option D7 is selected, the E options have no meaning and should be ignored.

Type of MODEM: Bell System Type 208A Data Set (also called DATAPHONE 4800).

Type of Line: Private Leased Line.

Transmission Rate: 4800 bits-per-second.

Table F-4. 208A Options and Recommendations

OPTION NUMBER	DESCRIPTION	RECOMMENDATIONS
A1	Transmitter internally timed.	A1 (required)
A2	Transmitter externally timed.	
B3	Continuous carrier.	B3
B4	Switched carrier.	
C5	Switched REQUEST TO SEND.	C6
C6	Continuous REQUEST TO SEND.	
D7	One second holdover used.	D7
D8	One second holdover not used.	
E9	With new sync.	E10 (required)
E10	Without new sync.	
F11	CC ON when analog loop is present.	F11
F12	CC OFF when analog loop is present.	

Type of MODEM: Bell System Type 208B Data Set (also called DATAPHONE 4800).

Type of Line: Public Telephone Network (Switched).

Transmission Rate: 4800 bits-per-second.

Table F-5. 208B Options and Recommendations

OPTION NUMBER	DESCRIPTION	RECOMMENDATION
A1	Transmitter internally timed.	A1 (equired)
A2	Transmitter externally timed.	
B3	Without 801 Automatic Calling Unit.	B3
B4	With 801 Automatic Calling Unit.	
C5	CC OFF when analog loop is present.	C6
C6	CC ON when analog loop is present.	
D7	Without automatic answer.	D8
D8	With automatic answer.	
E9	Desk mounting.	Either
E10	Rack or cabinet mounting.	

Type of MODEM: Bell System Type 209A Data Set (also called DATAPHONE 9600).

Type of Line: Private Leased Line.

Transmission Rate: 9600 bits-per-second.

Table F-6. 209A Options and Recommendations

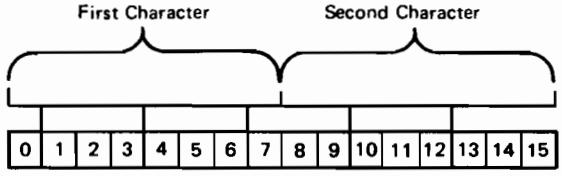
OPTION NUMBER	DESCRIPTION	RECOMMENDATION
A1	Transmitter internally timed.	A1 (required)
A2	Transmitter externally timed.	
E9	Continuous carrier.	E9
E10	Switched carrier.	
F11	Switched REQUEST TO SEND.	F12
F12	Continuous REQUEST TO SEND.	
D7	Elastic store in.	D8
D8	Elastic store out.	
	Slaved Transmitter timing by receiver.	out
	Data set ready circuit.	cc off
	Grounding	AA to AB
	Alternate - voice service Without alternate - voice service.	Either

ASCII CHARACTER SET

APPENDIX G

ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
A	040400	000101
B	041000	000102
C	041400	000103
D	042000	000104
E	042400	000105
F	043000	000106
G	043400	000107
H	044000	000110
I	044400	000111
J	045000	000112
K	045400	000113
L	046000	000114
M	046400	000115
N	047000	000116
O	047400	000117
P	050000	000120
Q	050400	000121
R	051000	000122
S	051400	000123
T	052000	000124
U	052400	000125
V	053000	000126
W	053400	000127
X	054000	000130
Y	054400	000131
Z	055000	000132
a	060400	000141
b	061000	000142
c	061400	000143
d	062000	000144
e	062400	000145
f	063000	000146
g	063400	000147
h	064000	000150
i	064400	000151
j	065000	000152
k	065400	000153
l	066000	000154
m	066400	000155
n	067000	000156
o	067400	000157
p	070000	000160
q	070400	000161
r	071000	000162
s	071400	000163
t	072000	000164
u	072400	000165
v	073000	000166
w	073400	000167
x	074000	000170
y	074400	000171
z	075000	000172
0	030000	000060
1	030400	000061
2	031000	000062
3	031400	000063
4	032000	000064
5	032400	000065
6	033000	000066
7	033400	000067
8	034000	000070
9	034400	000071
NUL	000000	000000
SOH	000400	000001
STX	001000	000002
ETX	001400	000003
EOT	002000	000004
ENQ	002400	000005

ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
ACK	003000	000006
BEL	003400	000007
BS	004000	000010
HT	004400	000011
LF	005000	000012
VT	005400	000013
FF	006000	000014
CR	006400	000015
SO	007000	000016
SI	007400	000017
DLE	010000	000020
DC1	010400	000021
DC2	011000	000022
DC3	011400	000023
DC4	012000	000024
NAK	012400	000025
SYN	013000	000026
ETB	013400	000027
CAN	014000	000030
EM	014400	000031
SUB	015000	000032
ESC	015400	000033
FS	016000	000034
GS	016400	000035
RS	017000	000036
US	017400	000037
SPACE	020000	000040
!	020400	000041
"	021000	000042
#	021400	000043
$	022000	000044
%	022400	000045
&	023000	000046
'	023400	000047
(	024000	000050
)	024400	000051
*	025000	000052
+	025400	000053
,	026000	000054
-	026400	000055
.	027000	000056
/	027400	000057
:	035000	000072
;	035400	000073
<	036000	000074
=	036400	000075
>	037000	000076
?	037400	000077
@	040000	000100
[	055400	000133
\	056000	000134
]	056400	000135
Δ	057000	000136
←	057400	000137
`	060000	000140
{	075400	000173
|	076000	000174
}	076400	000175
~	077000	000176
DEL	077400	000177



INDEX

A

ASCII Character Set, G-1

B

Batch Job
  establishing a remote session,3-7,3-8
BREAK key, The, 3-8 - 3-11

C

Communications Link
  definition, 2-1
  Errors, B-1,B-2
  established, 2-1
Character Set, ASCII, G-1
Closing a Line, 2-87 - 2-90
Command Access, Remote File, 4-2 - 4-31
Configuration Dialogue, A-1 - A-
Control keys, The, 3-11
Console Operator Command
  DSLINE, C-1 - C-4
CSHBSC0 (HSI driver), 2-5,2-6
CSSBSC0 (SSLC driver), 2-37, 2-38

D

Dialing the Remote Computer, 2-43
Distributed systems Network, 1-1
DSLINE Console Operator Command, C-1 - C-4
DSLINE Command, The, 2-7,2-8,2-39 - 2-42
DS/3000
  communications driver (IODS0), 2-5,2-6,2-37,2-38
  definition, 1-1
DS Trace Facility, D-1, D-2

E

Error Codes and Messages, B-1 - B-10
  Communication Link Errors, B-1, B-2
  Errors on Remote Hello, B-7
  Line Opening Failures, B-3 - B-6
  Program-to-Program Errors, B-9,B-10
  Remote File Access Errors, B-8

H

Hardwired Line
  access, 2-3 - 2-28
  definition, 2-3
Hardwired Serial Interface (HSI), 2-3
HSI, 2-3
HSI driver (CSHBSC0), 2-5,2-6

I

ID Sequences, 2-44
Intrinsics
  Master, 5-4,5-9 - 5-24,5-29
  Program-to-Program, 5-3 - 5-33
  Slave, 5-5,5-25 - 5-29

L

Line Opening Failures, 2-29 - 2-32,2-81 - 2-86,B-3 - B-6
Local commands, Issuing, 3-12

M

Multiple Users
  Hardwired Line, 2-8 - 2-10
  telephone line, 2-45 - 2-65
Modem Options, F-1 - F-6
  201A3, F-1
  201B3, F-2
  201C, F-3
  208A, F-4
  208B, F-5
  209A, F-6

O

Opening A Hardwired Line, 2-3 - 2-32
Opening A Telephone Line, 2-33 - 2-86
Opening Multiple Lines, 2-21 - 2-28,2-73 - 2-80

P

Program-to-Program
  Communications, 5-1 - 5-33
  Errors, B-9,B-10
  Intrinsics, 5-3 - 5-33

Remote commands, Issuing, 3-2 - 3-7
Remote File Access (RFA), 4-1 - 4-49
  Command Access, 4-2 - 4-31
  Errors, B-8
  Programmatic Access, 4-32 - 4-47

S

Specifying a Line
  hardwired, 2-5,2-6
  telephone, 2-36 - 2-38
SSLC driver (CSSBSC0), 2-37,2-38
System Verification Test, E-1 - E-4
  Diagnostic Mode, E-1,E-2
  Example, E-4
  Normal Mode, E-3

T

Telephone Line, 2-33 - 2-86
Terminating a Remote Session
  From the Local Session, 3-13,3-14
  From the Remote Session, 3-15,3-16
Trace
  DS Facility, D-1,D-2
  Dumping, D-2
  Example, D-2
  Invoking, D-1,D-2

U

Using the Remote Subsystem From a Batch Job, 3-7,3-8

W

What is a Communications Link, 2-1,2-2