64000

HP64000 Logic Development System

Computer Museum

Terminal Mode Operating Manual

HEWLETT PACKARD

CERTIFICATION

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Bureau of Standards, to the extent allowed by the Bureau's calibration facility, and to the calibration facilities of other International Standards Organization members.

WARRANTY

This Hewlett-Packard system product is warranted against defects in materials and workmanship for a period of 90 days from date of installation. During the warranty period, HP will, at its option, either repair or replace products which prove to be defective.

Warranty service of this product will be performed at Buyer's facility at no charge within HP service travel areas. Outside HP service travel areas, warranty service will be performed at Buyer's facility only upon HP's prior agreement and Buyer shall pay HP's round trip travel expenses. In all other cases, products must be returned to a service facility designated by HP.

For products returned to HP for warranty service, Buyer shall prepay shipping charges to HP and HP shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to HP from another country.

HP warrants that its software and firmware designated by HP for use with an instrument will execute its programming instructions when properly installed on that instrument. HP does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environment specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HP SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HP SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

ASSISTANCE

Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office.

FOLD HERE

HEWLETT
PACKARD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 1303 COLORADO SPRINGS, COLORADO

POSTAGE WILL BE PAID BY ADDRESSEE

HEWLETT-PACKARD
Logic Product Support Dept.
Attn: Technical Publications Manager
Centennial Annex - D2
P.O. Box 617
Colorado Springs, Colorado 80901-0617

FOLD HERE

Your cooperation in completing and returning this form will be greatly appreciated. Thank you.

READER COMMENT SHEET

Op. Manual
Terminal Mode Operating Manual
Part Number 64980-90935, April 1984

Your comments are important to us. Please answer this questionaire and return it to us. Circle the number that best describes your answer in questions 1 through 7. Thank you.

1. The information in this book is complete:

Doesn't cover enough (what more do you need?)	1	2	3	4	5	Covers everything

2. The information in this book is accurate:

Too many errors	1	2	3	4	5	Exactly right

3. The information in this book is easy to find:

I can't find things I need	1	2	3	4	5	I can find info quickly

4. The Index and Table of Contents are useful:

Helpful	1	2	3	4	5	Missing or inadequate

5. What about the "how-to" procedures and examples:

No help	1	2	3	4	5	Very helpful
Too many now	1	2	3	4	5	I'd like more

6. What about the writing style:

Confusing	1	2	3	4	5	Clear

7. What about organization of the book:

Poor order	1	2	3	4	5	Good order

8. What about the size of the book:

too big/small	1	2	3	4	5	Right size

Comments: ____

____

____

____

Particular pages with errors?

____

Name (optional): ____
Job title: ____
Company: ____
Address: ____

Note: If mailed outside U.S.A., place card in envelope. Use address shown on other side of this card.

HEWLETT
PACKARD

OPERATING MANUAL

TERMINAL MODE
OPERATING MANUAL

Manual Part No. 64980-90935

PRINTED: APRIL, 1984

Printing History

Each new edition of this manual incorporates all material updated since the previous edition. Manual change sheets are issued between editions, allowing you to correct or insert information in the current edition.

The part number changes only when each new edition is published. Minor corrections or additions may be made as the manual is reprinted between editions. Vertical bars in a page margin indicates the location of reprint corrections.

First Printing .... April 1984 (Part Number 64980-90935)

Table of Contents

Table of Contents (Cont'd)

Table of Contents (Cont'd)

List of Tables

List of Illustrations

Chapter 1

GENERAL INFORMATION

INTRODUCTION

The terminal software allows the 64000 station to function in an RS-232-C environment as an ASCII terminal. The signal interface between the 64000 station and a remote device is accomplished through the two RS-232-C ports on the rear panel of the station.

Full duplex asynchronous RS-232-C communications is initiated with the system terminal program. File transfers between the 64000 system and other devices are supported in 3 modes: first, ASCII transfer of listing and source code; second, absolute file transfer in Motorola, Tektronix (8-bit only), and Intel Hex formats; and, third, HP64000 format and protocol for highly reliable transfer of all user available 64000 file types.

The terminal supports two common protocols: XON/XOFF and ENQ/ACK. These protocols allow terminal operation up to 9600 baud. In addition, extensions to these protocols are provided to permit operation with slow, time-shared machines.

Screen editing features and escape sequences implemented for the 64000 terminal software are summarized in Appendix C in this manual.

NOTES

Chapter 2

TERMINAL OPERATION–
PHYSICAL REQUIREMENTS

Before evoking the "terminal" command it is important to identify the communication format to be used, set the 64000 Input/Output switches appropriately, and determine the appropriate RS232 interface port.

INPUT/OUTPUT CONFIGURATION

Model 64100A

The communication format for the two RS-232-C ports is controlled by five sets of switches and a jumper set located on the Model 64100A I/O board in the station card cage (see figure A-1 in appendix A). Switches on the board must be set to match the 64000 station transmission format with that of the remote system. Tables A-1 through A-3 explains briefly the different switch-setting functions. Table A-4 indicates I/O Board switch settings that may be used for connection to a specifically configured HP1000, HP3000, or HP9000 series computer 200 or 500. For a detailed description of the switches refer to the Model 64100A Service Manual.

Model 64110A

The Model 64110A functions similarly to the Model 64100A except for the Current Loop facility which is not provided. Like the 64100A, the 64110A provides asynchronous serial communications between the station and external communications devices. Communications is controlled by two switches on the RS-232/Flexible Disc Control card and a jumper on the CPU/IO card (see figure A-2 for switches and jumper locations). Table A-5 describes the switches (S1 and S2) and the jumper on the CPU/IO card.

HP64000 FORMAT I/O BOARD CONFIGURATION

If binary transfer mode is chosen, the I/O board must be set for 8 bits/character. Parity, start and stop bits are not critical for operation of binary transfers but if the installation can support 8-bit characters plus parity, this will provide some additional error checking capability.

Hex/ASCII transfers require only that the I/O board be set for at least 7 bits/character. In general, a working I/O board configuration need not be changed to support hex/ASCII transfers.

RS-232-C Interface

The signal interface to the Data Communications Equipment (DCE), usually a modem, and to the Data Terminal Equipment (DTE), usually a remote terminal or computer, is accomplished through two RS-232-C connectors located on the rear panel of the 64000 station. The "to" DTE connector is labelled TO MODEM, and the "to" DCE connector is labelled TO TERMINAL. Cables less than 50 feet (15 meters) are recommended for interconnection. Longer cables may be used provided the load capacitance conforms to EIA RS-232-C standards. Pin assignments of the ports, shown in table A-7, must be used.

The 64000 terminal hardware/software only operates in the full duplex mode; therefore, there is no hardware handshake on the RTS and DSR lines as used in half duplex operation. The terminal TO MODEM connector always asserts the RTS line and ignores the state of the DSR line. In addition, the CTS input line is monitored by the terminal hardware. If the line is false, the hardware will stop transmission until the CTS line again goes true. The DTR line is always held true by the terminal. Refer to the Installation and Configuration Reference Manual for a detailed description of RS-232-C operation.

RS-232-C TX and RX Clock Control

The 64100A RS-232-C circuits have the capability of driving (or receiving) both the TX and RX clocks independently. This capability is not available in the 64110A. Control of the clocks is provided by switch S2 and jumper terminals E1 and E2 (see figure A-1). The E1 jumper, in conjunction with the S2-RX clock switch, controls the RS-232-C pin location and direction of the receive clock. Similarly, the E2 jumper and S2-TX clock switch control the transmit clock. Valid E1 and E2 jumpers and S2 switch settings are as follows:

a. If the RXCLK switch is set to INT clock position, the corresponding E1 jumpers accomplish the following:

1). If the TO TERMINAL connector on the rear panel of the station is being used, then placing a jumper between A and B will cause the corresponding TXCLK to be driven out on pin 15 of that connector.

2). If the TO MODEM connector on the rear panel of the station is being used, then placing a jumper between B and C will cause the corresponding RXCLK to be driven out on pin 17 of that connector.

3). If NO jumper is installed, the corresponding clock will not be driven out.

b. If the RXCLK switch is set to EXT clock position, the corresponding E1 jumpers accomplish the following:

1). If the TO TERMINAL connector on the rear panel of the station is being used, then a jumper is required between A and C to receive on pin 15 of that connector.

2). If the TO MODEM connector on the rear panel of the station is being used, then NO jumper is required to receive on pin 17 of that connector.

c. If the TXCLK switch is set to INT clock position, the corresponding E2 jumpers accomplish the following:

1). If the TO TERMINAL connector on the rear panel of the station is being used, then placing a jumper between A and B will cause the corresponding RXCLK clock to be driven out on pin 17 of that connector.

2). If the TO MODEM connector on the rear panel of the station is being used, then placing a jumper between B and C will cause the corresponding TXCLK clock to be driven out on pin 15 of that connector.

3). If NO jumper is installed, the corresponding clock will not be driven out.

d. If the TXCLK switch is set to EXT clock position, the corresponding E2 jumpers accomplish the following:

1). If the TO TERMINAL connector on the rear panel of the station is being used, then a jumper from A to C is required to receive on pin 17 of that connector.

2). If the TO MODEM connector on the rear panel of the station is being used, then NO jumper is required to receive on pin 15 of that connector.

NOTES

Chapter 3

TERMINAL OPERATION– HOW TO USE

INTRODUCTION

RS-232-C communications to and from the 64000 station is initiated via the system terminal program. This program is selected with the terminal softkey, found in the Monitor mode softkey selectors.

Software Configuration

When terminal operation is selected, using the terminal softkey, a series of configuration questions will be asked by the terminal program. A command file may be established to answer these questions automatically or they may be answered individually when asked.

NOTE

> The answer to each of the configuration questions is interpreted as being the first nonblank character until the next blank character is encountered. Therefore, comments may be added to command files since the answer to a configuration question consists of only the first response on a line. Activity within terminal mode (such as file uploads or downloads) cannot be evoked through command files.

The configuration questions asked by the terminal program are as follows:

a. Auto linefeed? no (default value)

Answers: yes no

Description: Answering 'yes' results in a linefeed being transmitted after each carriage return generated by the terminal. This applies also during uploading.

b. Local echo? no (default value)

Answers: yes no

Description: If 'yes' is answered, characters typed at the keyboard will be shown on the CRT as well as being transmitted to the remote device. This function is **NOT** to support half-duplex operation. It may be used to simulate the reception of escape sequences.

c. Wait for echo during upload? no (default value)

Answers: yes no

Description: While uploading, if the answer to this question is 'yes', the station waits after sending each character for a return echo before sending the next character. If the answer is 'no', the station does not wait for an echo.

d. Download start sequence (0-6 chars)? "$^L{}_F$" (default value)

Answers: A character string of 0 to 6 characters delimited by quotation marks.

Description: At the start of a :source type download, the station must match this string before any text is written to the file. If the string is null, "", no match is required and all characters received will be written to the file.

e. Source end of file character? 04H (default value, ASCII $^E{}_T$ character)

Answer: A numerical ASCII value (see Note on page 3-6).

Description: The character assigned is used to indicate an 'end-of-file' during source file transfers. The specified character must not appear within the file being transmitted.

f. Format used for absolute file transfers? M__hex (default value)

Answers: M__hex I__hex T__hex HP64000 NONE

Description: The station can transfer absolute-type files in one of four formats. Refer to the section in Chapter 7 that covers Absolute File Formats for descriptions of M__hex, I__hex, and T__hex. Refer to the section in Chapter 6 that covers HP64000 file transfer format and protocol.

If NONE, proceed to question n.

If M_hex, proceed to question h.

If I_hex, proceed to question g(1).

If T_hex, proceed to question g.

If HP64000, proceed to question j.

g. Prompt sequence? no (default value)

Answers: yes no

Description: This question appears only if 'T_hex' format is specified in question f. If this question is asked and the answer is 'yes', the station will request:

Enter prompt sequence in quotes (1-6 chars): "GO" (default value)

Answer: Character strings up to 6 characters delimited by quotation marks.

Description: Any 7-bit ASCII characters are valid in the prompt sequence. Refer to table 3-1 for list of ASCII characters. Proceed to question h.

g(1). 8086/8088 format? no (default value)

Answers: yes no

Description: This question appears only if 'I_hex' format is specified in question f.

NOTE

Questions h through i are applicable only if M_hex, I_hex, or T_hex format is selected in question f.

h. Processor data bus width (#bits)? = 8 (default value)

Answers: 8 16

Description: This information is used by the station when uploading files. See examples of specific processors listed in question i following.

i. Smallest addressable entity (#bits)? = 8 (default value)

Answers: 8 16

Description:

Most processors address single bytes (8 bits). Therefore, eight (8) is generally the response to this question. Some examples of processor specifications are as follows:

Processor	Data Bus Width	Smallest Addressable Entity
680X	8	8
8048	8	8
8086	16	8
68000	16	8
Z800X	16	8

For all HP supported processor languages, the smallest addressable entity and data bus width are listed in the File Format Reference Manual.

Proceed to question n.

Questions j through m appear only if HP64000 format is specified in question f.

j. Use HEX/ASCII format for HP64000 type transfers? yes (default value)

Answers: yes no

Description:

This question appears only if HP64000 format is specified in question f. The default answer is yes. A no answer implies binary type transfers. If the I/O board is not set up for 8-bit characters and the user has chosen binary type transfers, an error message will be displayed on the status line.

k. Acknowledge prompt character (0=none)? 00H (default value)

Answers: Default is null, meaning this character is not used.

Description: This question appears only if HP64000 format is specified in question f. Values other than default cause the HP64000 to wait for the specified character during download before sending every acknowledgment, including the first one. This character may be the same as the upload prompt character.

l. Use assertive mode in upload? no (default value)

Answers: yes no

Description: This question appears only if HP64000 format is specified in question f. The default answer is no, meaning that if the file(s) desired to be uploaded are already existent (on another HP64000), the files (on the other HP64000) will not be overwritten.

m. Remote device input buffer size? 1024 (default value)

Answers: Default is the maximum length (# of characters) of a frame.

Description: This question appears only if HP64000 format is specified in question f. Values smaller than the default cause the HP64000 to packetize the data to fit within the specified buffer size. This buffer size should be the size of the input buffer of the remote device or smaller depending on the particular system.

n. Type of protocol? XON__XOFF (default value)

Answers: NONE XON/XOFF ENQ/ACK

Description: If NONE is selected, this will be the last configuration question. For a description of the other protocols refer to the protocol chapter. If a protocol type is selected other than NONE, the terminal will query:

o. Upload prompt character (0 = none)? 00H (default value)

Answer: A numerical ASCII value (see Note following configuration questions).

Description: See the relevant description in the protocol section in Chapter 8.

If XON__XOFF was specified in question n proceed to question p.

if ENQ_ACK was specified in question n proceed to question s.

p. XON character? 11H (default value, ASCII ^{D_1} character)

Answer: A numerical ASCII value (see Note following configuration questions).

q. XOFF character? 13H (default value, ASCII ^{D_3} character)

Answer: A numerical ASCII value (see NOTE on page 3-6).

r. Delay time(mS) after sending XOFF during download? 0 (default value)

Answer: Default is 0.

Description: Values greater than 0 and less than 32768 milliseconds cause the terminal software to delay that amount of time after sending XOFF before disabling RS-232-C interrupts and writing to the disc. After the delay has expired, only two characters can be accepted without overrun errors. This only applies during download. If download is from a device that cannot stop within two characters after receiving XOFF, refer to the pseudo program given in Chapter 8 in the protocol section titled "XON/XOFF Protocol" for use with such a device.

s. ENQ character? 05H (default value, ANSII ^{E_Q} character)

Answer: A numerical ASCII value (see Note following configuration questions).

t. ACK character? 06H (default value, ANSII ^{A_K} character)

Answer: A numerical ASCII value (see Note following configuration questions).

NOTE

The specified character required by the configuration questions must be unique, i.e., not the same as a protocol character. It may, however, be the same as the acknowledgment prompt character. The present or current value of the character in question is always displayed in hexadecimal format. Entries allowed for numerical ASCII value questions are:

Base	Suffix	Example
Decimal	None or D	26 or 26D
Hexadecimal	H	0FH
Octal	O or Q	17O or 17Q
Binary	B	1011101B

The following command file example evokes the terminal and answers the configuration questions. The configuration shown in the example is for an HP1000 (see table A-4 for switch settings) or an HP3000 using terminal type = 10.

Command file example:

terminal	this is to evoke terminal operation
no	Auto linefeed?
no	local echo?
no	Wait for echo?
"AB"	download start sequence
004H	Source EOF char
M_hex	Absolute format type used
8	Data bus width in bits
8	Addressing base in bits
ENQ/ACK	Protocol selected
011H	Upload prompt char
005H	ENQ char
006H	ACK char

CRT Display Operation

There are two modes used during terminal operation: 'terminal mode' and 'command mode'. When the cursor is in the first 18 lines of the display and the leftmost softkey label displays terminal in inverse video, the terminal mode of operation is in effect. When the cursor is in the command area of the display, the command mode of operation is in effect (see terminal Softkey Display). Line #20 will remain the STATUS line as in the monitor.

NOTE

Refer to Appendix B for a complete list of terminal status and error messages.

Except during file transfer, all characters (except protocol and control characters) received at the RS-232-C ports are displayed on the CRT. Of the possible 8 bits describing an ASCII character, only the lower 7 bits are meaningful; therefore, upon reception, the 8th bit (MSB) is masked off and ASCII characters having a value of less than 20 hex (20H) are not displayed, unless the station is in "display functions" mode (see Appendix C). The escape character is treated as the beginning of an escape sequence. Characters following an escape sequence will not be displayed until the escape sequence has been completed or no match with a valid sequence can be established.

Suppose the following escape sequence was transmitted to the terminal:

$\alpha aZ\beta$

where:

α - represents the escape character (ASCII = 1BH).
β - represents a control-B or ASCII = 02H.

The resulting line on the display would be:

Z

since both the escape and control-B character are non-printing ASCII characters and the escape sequence deviates from acceptable sequences at "a".

NOTE

All characters arriving at the 64000 station during a source download, except for protocol characters and acknowledgment prompts, will be placed in the disc file.

Softkey Display

After answering the configuration questions, the softkey configuration line will display:

terminal upload download edit_cnfg <clear> <break> <escape> end

where:

terminal - When the terminal softkey is displayed in inverse video, the cursor will be located within the first 18 lines of the display, indicating terminal mode of operation. When the terminal softkey is displayed normally, the cursor will be in the command area of the display indicating command mode of operation. To change modes of operation, press the terminal softkey.

upload - This softkey allows the user to specify a file to be uploaded to the remote device.

download - This softkey allows the user to load a file onto the 64000 disc from the remote device.

edit_cnfg - This softkey allows the user to reconfigure the terminal.

<clear> - The CRT will be cleared from the present cursor position to the end of the terminal display area. Nothing will be transmitted to the remote device.

<break> - Since the 64000 keyboard does not have a <break> key, the break function is supplied via a terminal softkey. Pressing the <break> softkey clamps the hardware serial transmit line at logic "0" for approximately 450 ms. No ASCII character is transmitted.

<escape> - ASCII character (1BH) will be transmitted.

end - Ends terminal operation.

Keyboard Operation

When the terminal is in the command mode, the keyboard functions as in the monitor. When the terminal is in the terminal mode, the keyboard functions differently due to the requirements of single character input/output, etc.

All keys in the main key group (mainly alphnumeric) send the ASCII code for the character indicated on the keycap to the remote device when pressed. If the 'local echo' question is answered 'no' during configuration, the character will not be displayed on the CRT until the remote device echoes it. If the 'local echo' question is answered 'yes', the character will be displayed immediately. The terminal will respond to any escape sequence entered in this mode as if they had come from the RS-232 line. Note that typing with local echo ON while connected to a remote device which echoes can cause intermixing of the local and remote echoes. All 127 ASCII 7-bit codes can be produced from the keyboard. Refer to table 3-1 for a complete listing of the ASCII Character Codes.

Table 3-1. ASCII Conversion Table

	Col #1 +0	Col #2 +20	Col #3 +40	Col #4 +60
0	NUL	SP	@	`
1	SOH	!	A	a
2	STX	"	B	b
3	ETX	#	C	c
4	EOT	$	D	d
5	ENQ	%	E	e
6	ACK	&	F	f
7	BEL	'	G	g
8	BS	(	H	h
9	HT	)	I	i
A	LF	*	J	j
B	VT	+	K	k
C	FF	,	L	l
D	CR	-	M	m
E	SO	.	N	n
F	SI	/	O	o
10	DLE	0	P	p
11	DC1(Xon)	1	Q	q
12	DC2(tape)	2	R	r
13	DC3(Xoff)	3	S	s
14	DC4	4	T	t
15	NAK	5	U	u
16	SYN	6	V	v
17	ETB	7	W	w
18	CAN	8	X	x
19	EM	9	Y	y
1A	SUB	:	Z	z
1B	ESC	;	[	{
1C	FS	<	\	|
1D	GS	=	]	}
1E	RS	>	^	~
1F	US	?	_	DEL

NOTE

To produce the ASCII characters in column #1 in the ASCII table, hold down the control CNTL key on the keyboard and then press the corresponding character key listed in column #3. For example, CNTL-H produces a BS or backspace (ASCII = 08H) and CNTL-[produces an ESC or escape (ASCII = 1BH).

The keypad keys (see figure 3-1) to the right of the main keyboard are functional. The (ROLL UP) and (ROLL DOWN) keys roll the screen up or down one line at a time until there are no more lines in the screen memory. (NEXT PAGE) and (PREV PAGE) keys will display the next or previous 18 lines of screen memory (or less if screen memory is not full). With memory extender card installed, the screen memory can usually buffer more than 400 lines. The **DOWN-ARROW** (↓) and **UP-ARROW** (↑) keys roll the cursor up or down in the terminal area. This is a local action and no characters are transmitted. If one attempts to move the cursor past the bottom of the screen, the screen will scroll.

When the transmit keypad is disabled and local echo is OFF, the **LEFT-ARROW** (←) key transmits a backspace character as does the (BACK SPACE) key on the keyboard. The **RIGHT-ARROW** (→) key transmits the character which it is moved under. This is useful for re-entering commands in combination with the up- and down- arrow keys.

The (INSERT CHAR) key allows insertion of characters into a line of program or text. To insert characters into a line, locate the cursor directly below the point of insertion and then press the (INSERT CHAR) key. Insert alphanumerics as required. To stop the insert mode of operation, press (INSERT CHAR) key again.

The (DELETE CHAR) key deletes from the screen the character directly above the cursor and sends nothing.

The (CLR LINE) key clears the line of the terminal display area in which the cursor is positioned. After clearing, the cursor is moved to the start of the line. Also, 80 backspaces are transmitted to clear the host buffer.

The (RECALL) key has no function in terminal mode operation.

The (CAPS LOCK) and (RESET) keys function normally.

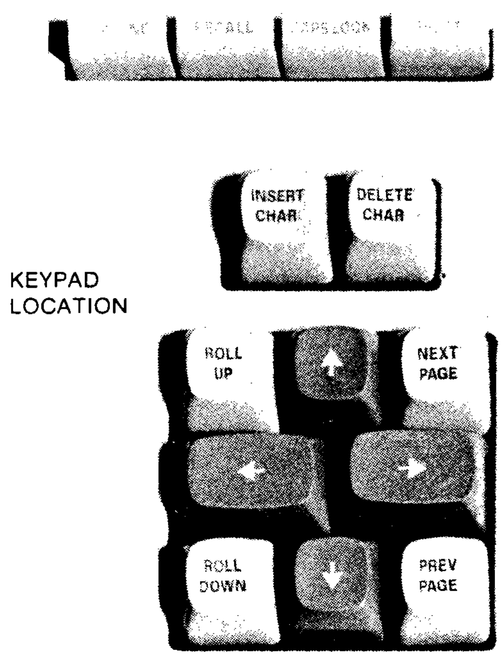


Figure 3-1. Keypad Location

The keypad is enabled and disabled using the following escape sequences:

Keypad Enable -	^{E}C & j B
	^{E}C & s 1 A
Keypad Disabled -	^{E}C & g @
	^{E}C & s 0 A

When the keypad is enabled, the individual keys in the keypad transmit the following escape sequences when pressed:

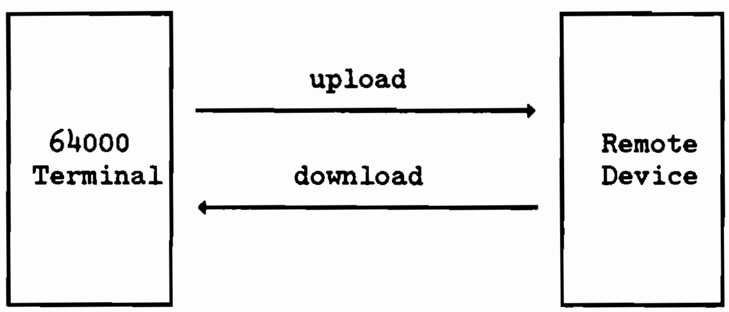
INSERT CHAR	-	^{E}C Q
DELETE CHAR	-	^{E}C P
ROLL UP	-	^{E}C S
ROLL DOWN	-	^{E}C T
NEXT PAGE	-	^{E}C U
PREV PAGE	-	^{E}C V
UP-ARROW (↑)	-	^{E}C A
DOWN-ARROW (↓)	-	^{E}C B
RIGHT-ARROW (→)	-	^{E}C C
LEFT-ARROW (←)	-	^{E}C D

Chapter 4

FILE TRANSFERS - HP64000

HP64000 FILE TRANSFER - FORMAT AND PROTOCOL

In the following discussions concerning file transfers, the terms "upload" and "download" indicate the following transfer directions:



General Information

The purpose of HP64000 file transfer format and protocol is to provide the following capabilities:

a. Totally transparent transfers of all types of HP64000 files.

b. Data integrity via error detection and re-transmission.

c. Straightforward interfacing to General Purpose Computers.

d. Flexibility in protocol used (XON/XOFF or ENQ/ACK). (An additional level of protocol is built into the software which allows application programs to control packing of data transfer in many cases.)

e. Packetizing to prevent remote-device buffer overrun caused by sending long records.

f. Transfer of multiple files per download/upload.

g. Hex/ASCII and binary modes of transfers. The hex/ASCII mode of transfer provides:

1). The ability to interface to a high-level language without terminal driver modifications or assembly coding in most cases.

2). No non-printing ASCII characters are used.

3). Extraneous characters inserted by various terminal drivers can be dealt with.

The binary mode of transfer provides for a possible factor of two improvement in throughput over hex/ASCII mode of transfer for users willing to configure their systems appropriately.

HP64000 protocol is designed to correct errors and assure data integrity. In addition, it is simple enough to be implemented as an application program on a General Purpose Computer.

OVERVIEW

HP64000 file transfer format and protocol is suitable for reliably transferring one or more files with error detection and re-transmission. Both file data and file names are transmitted over the data link (RS-232-C).

The discussion of HP64000 file transfer includes a quick sample configuration and an example transfer between HP64000 stations. This is followed by a description of the HP64000 format user interface. Information required for communication between an HP64000 station and some other device follows. A look at configuration, examples, meaning of diagnostic messages, and implementation information is given. A complete description of the format and protocol is found in Chapters 6 and 8 respectively.

Sample Configuration and Example

For transfers between HP64000 stations, the following sample configuration is one of many which will work.

Auto linefeed? yes (at user's discretion - no effect on HP64000 format)

Local echo? yes (also at user's discretion)

Wait for echo during upload? no (HP64000 does not echo)

Download start sequence? "" (not used by HP64000 format)

Source end of file character? 04H (not used by HP64000 format)

Format used for absolute file transfers? HP64000

Use HEX/ASCII format for HP64000 type transfers? yes (user's discretion)

NOTE

I/O board must be set for 8 bits/character for binary transfers.

Acknowledge prompt character (0=none)? 00H (not needed between HP64000's)

Use assertive mode during upload? no (At user's discretion - see description of record types)

Remote device input buffer size? 1024 (Default value is correct)

Type of protocol? NONE (no additional protocol needed if HP64000 protocol is used)

Sample Transfer Between HP64000 Stations

Given the above configuration and two HP64000s connected properly via RS-232-C with the matching I/O board setups, the following sequence of events is used to initiate a file transfer.

Press the download and in_HP_fmt softkeys, then the RETURN key at the receiving HP64000 station. The station will wait for the transfer to begin. At the sending station, press the upload and in_HP_fmt softkeys, type in the filename, and press the RETURN key.

NOTE

The download command may be entered at any time prior to entry of the upload command and up to 1 minute after the upload command has been issued.

HP64000 FORMAT DOWNLOAD USER INTERFACE

Once the HP64000 station is properly configured to use HP64000 format, the procedure for downloading in HP64000 format is to press the download and the in_HP_fmt softkeys, then press the RETURN key. The names of the files to be downloaded are contained in the data received.

Download Status Messages

The status of the file currently being downloaded is shown on the status line during download. After a file is downloaded, the status of the file is shown in the terminal area immediately above the status line. In addition to the file name and the number of records in that file, the file status messages contain one of the following:

** File downloaded **	file was successfully downloaded
** exists, File NOT downloaded**	file was not downloaded because of NAK on filename record. This will occur if transfer is not assertive and the file already exists.
** Abort during download **	either the user or the software caused the transfer to be terminated

If any RS-232-C errors occurred during the download, a message will be written in the terminal area after the last file status message. This message describes the RS-232-C errors and the fact that recovery occurred. Also, the STATUS line will display "one or more operations unsuccessful" if any file fails to download.

HP64000 FORMAT UPLOAD USER INTERFACE

File Specifications

The user may specify the file(s) to be uploaded in several ways:

a. When only a single file is to be uploaded, press the upload and in_HP_fmt softkeys and type in the following information:

filename<:USERID><:DISC NUMBER><:filetype>
(<> indicate the field may be omitted. If the default applies refer to the System Software Reference Manual.)

Now press the RETURN key.

b. When a single-file upload is to be made and an alternate name for that file on the remote device is desired, press the upload and in_HP_fmt softkeys and type in the same information as indicated in (a) above.

Now press the into softkey, type in the ^NEWNAME^, and press the RETURN key.

Where:

^NEWNAME^ represents a literal string surrounded by delimiters which can be ^, ", or ' characters. The literal string is transmitted in the file name record as the name of the file. This string may contain any ASCII characters. If this string is null, ("") the HP64000 file name is used.

c. When multiple files are to be uploaded, press the upload, in_HP_fmt, and using softkeys and type in the following information: the file name, a colon, and the USERID. Now press the RETURN key.

Where:

The file name entered is a source file containing a list of HP64000 file names to be uploaded. This file contains lines of the form:

filename<:USERID><:DISC NUMBER><:filetype>[<^NEWNAME^>]

Where the second field is optional as indicated by the square brackets.

The ^NEWNAME^ field is defined as above.

If the ^NEWNAME^ field is absent, the HP64000 file name is transmitted in the file name record. Otherwise, the literal string is transmitted in the file name record.

Example of a File Specification

If the command:

upload in__HP64000__format TEST:HP

is issued, then the HP64000 file TEST:HP:source will be opened for uploading and the HP64000 file name will be transmitted in the file name record.

If the "using" file command is used:

upload in__HP64000__format using ABC:HP:source

and file ABC contains a line TEST:HP

then the HP64000 file TEST:HP:source will be opened for uploading and the HP64000 file name will be transmitted in the file name record.

If the command:

upload in__HP64000__format TEST:HP into "ABC.dat"

is issued, then the HP64000 file TEST:HP:source will be opened for uploading and the string:

ABC.dat

will be transmitted in the file name record.

If the "using" file command is used:

upload in__HP64000__format using XYZ:HP:source

and file XYZ:HP contains a line TEST:HP 'ABC.dat'

then the HP64000 file TEST:HP:source will be opened for uploading and the string:

ABC.dat

will be transmitted in the file name record.

Upload Status Messages

During upload, the status of the current file being transferred is shown on the status line. After a file is uploaded and/or an error occurs, the status of that file is shown in the terminal area above the status line. The status messages describe the name of the file, the number of records transferred, and one of the following:

** File uploaded **	File was successfully uploaded
** File not uploaded **	File name was rejected by remote device or a syntax error is present in the user's list of files
** Abort during upload **	User or software caused the transfer to be terminated
** Error during upload **	File on uploading system is corrupt

In addition, the STATUS line will display "one or more operations unsuccessful" if any file fails to upload.

Chapter 5

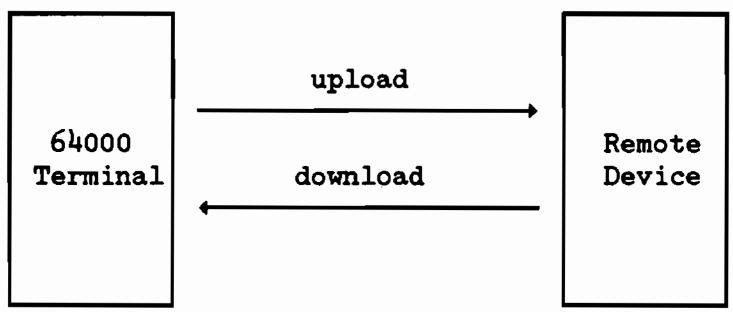
FILE TRANSFERS -

I__hex, M__hex, T__hex, AND NONE

GENERAL INFORMATION

If no file type is specified in an upload or download command, :source files are assumed. If :absolute is chosen, the absolute file format selected during configuration is used. M__hex, I__hex, and T__hex formats translate absolute data into printable ASCII characters. If NONE was specified, no absolute files can be transferred.

In the following discussions concerning file transfers, the terms "upload" and "download" indicate the following transfer directions:



Uploading Operation

The syntax for uploading a file from the terminal is as follows:

SYNTAX:

upload <FILE> [:file type]

The maximum line length used by the terminal is 240 characters. In addition, since 64000 records always contain an even number of characters, lines which initially contain an odd number of characters will be padded with spaces.

During upload, the terminal outputs a 'carriage return' (<CR>) as a record terminator (<end of line>) or, if auto linefeed is on, a <CR><LF> is used.

When uploading source files, the terminal appends the end-of-file character (selected by the user during configuration) to the end of the file being uploaded. This signals the remote device that the end of the file has occurred.

When ENQ/ACK or XON/XOFF protocol is specified and an upload prompt character is also specified, the terminal waits for the specified upload prompt character before transmitting each record (line). If no upload prompt character is specified or the protocol **NONE** is used, the terminal will begin transmitting once the upload command has been entered.

When uploading to devices which echo, overrun errors are common (and harmless), and the 64000 ignores them. Framing and parity errors, on the other hand, indicate possible problems and may result when the files being uploaded contain characters that have special meaning to the remote device. Parity and Framing errors may also indicate line noise or hardware problems.

Conventions used in the following upload examples are:

RETURN or <CR>	- carriage return.
text	- commands from 64000 to remote device.
{text}	- comments.

Example No. 1 - HP3000 Computer Upload. The HP3000 computer can be configured to use ENQ/ACK protocol and, as part of this protocol, it transmits an ASCII 11H whenever it is ready for another line. (See table A-4 and the sample command file listed previously for configuring the teminal for an HP3000.) To upload to the HP3000, perform the following:

a. Configure the terminal for ENQ/ACK protocol and an upload prompt character = 11H (DC1).

b. Log onto the HP3000 system with terminal type = 10. Then enter the editor.

:EDITOR RETURN

c. Prepare the editor to accept text:

/ADD {do not press RETURN}

d. Prepare the terminal to transmit a 64000 file named TEST:FDL -

upload TEST:FDL **[:source]** RETURN

e. If the file is found, the ABORT softkey will be the only softkey displayed. Press RETURN to initiate the ADD command.

f. The terminal STATUS line will indicate the progress of the upload. When the upload is complete, type the following:

//RETURN {'//' terminates the ADD mode of EDIT 3000}

g. The text may now be treated like any other text entered from the terminal. For example:

/KEEP TEST,UNN RETURN {save the text in a file named TEST}

Example No. 2 - Upload Without Prompt Character. Suppose that the HP3000 used in Example No. 1 is fast enough to accept text at the terminal baud rate without using a protocol. If this is the case, accomplish the following:

a. Configure the terminal for ENQ/ACK protocol but do not configure it with an upload prompt character (character = 0).

b. Log onto the HP3000 and evoke the Editor as in Example No. 1. Again enter the ADD mode:

/ADD RETURN {note the RETURN}

c. The HP3000 is now ready to accept text. Enter the upload command as described in Example No. 1. Since there is no upload prompt character, the terminal will start transmitting the file as soon as the RETURN (used to end the upload command) is transmitted.

d. Exit the ADD mode after the file has been uploaded.

Downloading Operation

The download operation is essentially the same as for upload. The syntax for downloading is as follows:

SYNTAX:

download <FILE> [:file type]

When downloading source files, the terminal exits download mode only when the character selected as the end-of-file character is received. If this character is not received, download can be terminated using the ABORT softkey. The operation of download when using a protocol (ENQ/ACK or XON/XOFF) is described in the protocol chapter of this manual.

Example - Downloading a File. Assume that the terminal is connected to a device which recognizes the following command:

TYPE FILE

which causes the indicated file, FILE, to be printed on the user's terminal followed by the end-of-file character. To download a file, perform the following:

a. Configure the terminal to recognize the end-of-file character.

b. The sequence of events for downloading would be to transmit the following from the terminal:

TYPE FILE
download FILE1 RETURN
<CR>

c. The downloaded file will be transferred to a 64000 file named FILE1 and the terminal will return to the terminal mode of operation.

NOTE

An ABORT softkey is displayed during uploading and download and may be used if problems are encountered. It is not a destructive abort.

If a line contains more than 240 characters during downloading, the terminal breaks the line into as many 240 character lines as required.

Source download expects one of the following as a record terminator (end-of-line):

carriage return (<CR>)

carriage return-line feed (<CR><LF>)

line feed (<LF>)

line feed-carriage return (<LF><CR>)

During download, the station ends the download on reception of the end-of-file character. If this character is not received, the station remains in the download mode until aborted by the user.

NOTE

Under ENQ/ACK protocol, pressing ABORT will cause the second, and all subsequent records residing in the input buffer at that time, to be written to the display and **NOT** to the disc file. To ensure that the complete file is downloaded, the remote device should always transmit an end-of-file character.

Chapter 6

DATA FORMAT - HP64000

FORMAT OVERVIEW

This section describes the data formats used in HP64000 format file transfers in a top-down fashion. At the highest level is the file system interface. The packetizing module is presented next and is followed by a description of the low-level frame formats.

This format is used on top of files which are in the formats specified in the File Format Reference Manual and applies to all user accessible file types.

Definitions of terms used in the discussions that follow are given in table 6-1.

Table 6-1. Definition of Terms

APC	Acknowledgment prompt character. The download program waits for this character before sending each acknowledgment message. This character is determined by the user during configuration.
UPC	Upload prompt character. The upload program waits for this character before sending each frame. This character is determined by the user during configuration.
ACK	A positive acknowledgment message. See ACK/NAK for a description of the contents of the message. ACK does **NOT** refer to the ASCII "ACK" character (06H).
NAK	Negative acknowledgment message. NAK does **NOT** refer to the ASCII "NAK" character (15H).
ABORT	Both the transmitter (upload) and the receiver (download) have the capability to terminate or abort the transfer. Termination can be caused by an irrecoverable software detected failure or by a user pressing the [ABORT] softkey.
Binary	A binary transfer is one in which data is transmitted over the data link (RS-232-C) in 8-bit binary bytes.

Table 6-1. Definition of Terms (Cont'd)

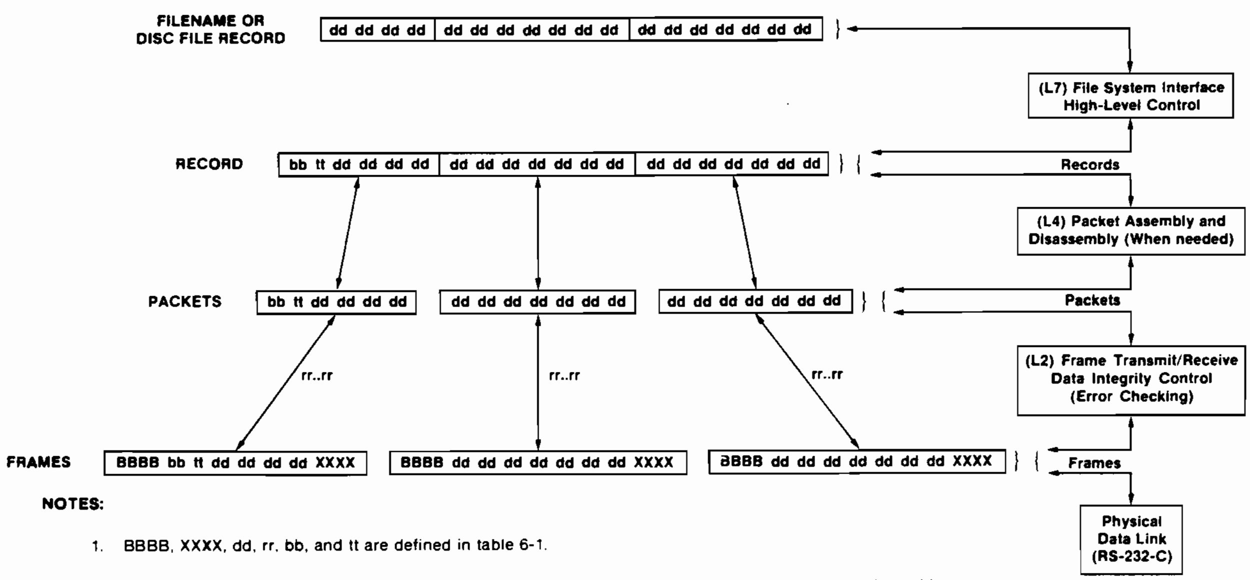
HEX/ASCII	A hex or ASCII transfer is one in which data is converted from the binary source to hex digits. For example, the data byte with a decimal value of 127 = 7FH would be transmitted as two characters, an ASCII 7 (37H) followed by an ASCII F (46H).
Upload	The direction of file transfer in this case is from the HP64000 to the remote device. The HP64000 is the source.
Download	File transfer is such that the HP64000 is the destination.

NOTE

The labels L7, L4, and L2 are described in figure 6-1.

Record	Used to refer to the data which passes between L4 and L7 (64000 record or file name record). (See figure 6-1).
Packet	Parcel of data passed between modules L4 and L2.
Frame	Data passed between module L2 and the data link.
Frame byte count	The 16-bit byte count (designated BBBB) which module L2 appends to the front of a packet of data from L4 before transmitting it (count only added in binary mode).
Frame checksum	The 16-bit checksum (designated XXXX) which module L2 appends to the end of a packet of data from L4 before transmitting it.
<CR>	Denotes the ASCII character, ODH, carriage return.
dd..dd	Record data bytes.
bb	One byte describing the number of record data bytes -1.
tt	One byte describing record type (see table 6-2).
rr..rr	Packet data bytes.
BBBB	16-bit frame byte count.
XXXX	16-bit frame checksum (sum of 8-bit data bytes).

Figure 6-1 presents the relationship of the various data structures and software modules as implemented in the HP64000 software. This figure is presented before the description for easy reference while reading the description.



NOTES:

1. BBBB, XXXX, dd, rr, bb, and tt are defined in table 6-1.

2. This diagram shows the way in which packets, frames and records are related. Note that frames are the entities which travel over the communication link to the other device. (The BBBB field is not present when using the hex/ASCII mode - see Hex/ASCII Transfer Frame Format, in this chapter.)

3. The diagram is intended to be bi-directional, that is, the receiver goes "up" the diagram - assembling frames into records, and the transmitter does the opposite.

Figure 6-1. Major Data Structures and Controls

File System Interface

The file system interface (L7) is the highest level portion of the software. During upload, it is responsible for generating the file names to be sent to the remote device, reading local disc files, and sending its records to packet assembly/disassembly (L4). The file system interface (L7) also takes appropriate action when the remote device rejects a file name, and informs the remote device when the transfer is over.

During download, the file system interface is responsible for determining whether a given file name is valid and accepting or rejecting it. It also opens/creates files and writes incoming data to files, and detects when the transfer is complete.

The file system interfaces (L7), on both sides of the data link, communicate control information with each other via record types. This information is contained in the "tt" field of each record.

Record Types

There are three general record types: file name records, data records, and the end-of-transmission records (see table 6-2). The record type is indicated by the "tt" field in the record.

Table 6-2. Record Types

Record type	Type number(tt)
Assertive file name	00
Non-assertive file name	01
Data record	02
End-of-transmission	03

DATA RECORDS. The data record (tt=02) contains in the data field (dd..dd) data to be written to, or data read from, the HP64000 file currently open.

END-OF-TRANSMISSION RECORDS. This record(type=03) indicates to the downloading (destination) station that the transmission is over. The download control module closes any open files and terminates the download. This record may be received at any time. The data field of this record may contain arbitrary data which is discarded. Upload sends a zero as the data byte in this record.

FILE NAME RECORDS. There are two types of file name records:

assertive (type=00):	If the file named already exists on the receiving system, it will be purged.
non-assertive (type=01):	If the file named already exists on the receiving system, it will not be overwritten and the file name record will be NAKed.

In downloading, the data field of a file name record must contain the file name in the normal HP64000 file name syntax (ie.. TEST:HP). All normal default values apply. In uploading, the data field of a file name record will contain the ^string^ specified in the upload command or the 64000 filename as above. (See HP64000 Format Upload User Interface in Chapter 4.)

A file name record must be received and properly acknowledged before data records are received.

A different type of response is generated to acknowledge a file name record. In short, a secondary acknowledgment is added to the frame acknowledgment normally provided. See table 8-1 in the Chapter 8 for details.

The secondary acknowledgment will be a NAK if:

a. The file name contains a syntax error.

b. The file already exists (if type=01, non-assertive mode).

c. The file cannot be opened or created for some reason.

Otherwise, the secondary acknowledgment will be an ACK.

When a file name record is ACKed, that file is opened and subsequent data records are written to that file.

When a file name record is NAKed, another may be sent.

When a file name record is received, any previously opened file is always closed.

NOTE

Abort records, or more properly, abort frames, are defined in the description of the binary and hex frame formats.

The terms ACK and NAK, as used here, are generic terms meaning "positive acknowledgment" and "negative acknowledgment", respectively. The exact contents of an acknowledgment message are described in Chapter 8 of this manual.

PACKETS

The file system interface communicates with the HP64000 file system and with the next software module responsible for packetizing. Figure 6-1 shows the data passed between the file system interface module and the packetizing module.

Packetizing is a method for splitting long records into smaller pieces for transmission and then re-assembling them into the original record upon reception. This is important to help prevent data loss when the HP64000 is transmitting (uploading) to a remote device which has a limited buffer size.

Packet Assembly During Download

The first packet of a record is assumed to be of the form:

(rr..rr) = (bb tt dd..dd)

Where:

dd..dd	are data bytes.
tt	is the record type (see table 6-2).
bb	is the 8-bit count of the data bytes in the entire record, dd..dd, MINUS ONE(1). bb can then enumerate 1 to 256 data bytes.

The packet assembler counts the number of data bytes, dd..dd, in the current packet. If there are the same number of data bytes as indicated by the record byte count, bb, then the record is complete and is passed on for further processing. If the number of data bytes is less than the number indicated by "bb", further packets (frames) will be accepted and the "rr..rr" field is interpreted as a continuation of the data byte stream, dd..dd. When the correct number of data bytes has been accumulated, as determined by "bb", the record is complete and is passed on for further processing. If more data bytes are received than indicated by "bb", the receiver will attempt to abort the download.

The three packets shown in table 6-3 would be correctly assembled into an assertive file name record with the file name: TEST:HP. File name records are explained above in the paragraph titled "Record Types".

Packet Creation During Upload

The reason the upload software may break records into packets is that many remote devices may only be able to accept one input buffer full of data at a time. By specifying the remote device input buffer size in the terminal mode configuration, the upload software is able to ensure that no frame is ever sent which exceeds this length. This buffer size is specified in bytes.

The minimum size for binary transfers is seven (7) bytes. The minimum size for hex/ASCII transfers is 11 bytes. The calculations of maximum number of data bytes sent per packet (PACKETSIZE), rr..rr, given the remote device input buffer size in bytes (RECORDSIZE), is shown in figure 6-2.

Table 6-3. Name Record Packet Assembly

Packet Byte		Meaning
(packet #1)		
06	(byte count -1)	7 data bytes in whole record
00	("tt" or record type)	assertive file name record
54	("dd" or data)	ASCII T
45	(dd)	ASCII E
(packet #2)		
53	(dd)	ASCII S
54	(dd)	ASCII T
3A	(dd)	ASCII :
48	(dd)	ASCII H
(packet #3)		
50	(dd)	ASCII P

```
IF( HEX_FORMAT ) THEN
  PACKETSIZE := (RECORDSIZE - 1 )/2 - 2

                                        checksum (XXXX) is two
                                        bytes

                                        two chars per data byte
                                        hex/ASCII format

                                        <CR> is one character

ELSE
  PACKETSIZE := RECORDSIZE - 2 - 2 ;

                                    checksum(XXXX) is two
                                     bytes

                                    frame byte count (BBBB)
                                    is two bytes

 Note that the <CR> is counted as part of the record in hex
 format.
```

Figure 6-2. Packet Size Calculation

FRAMES

Frames describe the actual bytes sent over the RS-232-C link. Basically, a frame is a data packet with additional information for error detection and synchronization.

A checksum is added to the end of a packet in both hex/ASCII and binary formats for error detection purposes. A byte count is appended to the beginning of a packet for synchronization in binary format. A <CR> is defined as a frame terminator for synchronization in hex/ASCII format. In addition, the data bytes are converted to hex characters for transmission and reception over the RS-232-C link in hex/ASCII format.

A detailed description of the two frame formats is given in the following paragraphs.

HEX/ASCII Transfer Frame Format

Note that all representation of binary data bytes and words is in hex: that is, a byte with a value of 127 decimal is shown as "7F".

Features of hex/ASCII transfer:

a. Each data byte is sent in its hex representation which is two legal, printing ASCII characters. For example, a byte with the value 7F (hex) would be transmitted as an ASCII 7 (37H) followed by an ASCII F (46H).

b. Neither the remote device nor the HP64000 need be set for 8-bit characters.

c. The remote device may echo.

d. The remote device may send extraneous characters such as fill characters.

The frame is defined to begin with the first ASCII character which is a legal hex digit (i.e.; 0-9, A-F).

All characters not belonging to the above set and which are not the ABORT_CHAR, '?' or a <CR>, are ignored.

Blank frames are ignored and are not acknowledged. A blank frame is one containing fewer than two valid hex digits and which is terminated by a <CR>. The smallest blank frame is simply a <CR>.

Hex/ASCII frame format:

rr..rr XXXX <CR>

Where:

rr..rr	are the packet data bytes
XXXX	is the 16-bit checksum of the frame data bytes, rr..rr
<CR>	is the frame terminator, a carriage return character

NOTE

If the received frame checksum(XXXX) and the calculated checksum agree, the frame is deemed to have been received correctly.

The number of packet data bytes, rr..rr, is greater than zero and less than 258.

The checksum, XXXX is calculated as the 16-bit sum of all the data bytes, rr..rr.

A correct frame is ACKed and an incorrect one is NAKed. Refer to the ACK/NAK protocol section in Chapter 8.

If the frame contains an ABORT__CHAR, '?', the frame is defined to be an abort frame and the download aborts.

Example of a Hex/ASCII Frame:

Although the packet data, rr..rr, is arbitrary, this packet is correct.

```
(rrrrrrrrXXXX)
"000102030006<CR>"
```

The quotes are not part of the frame.

The <CR> denotes an ASCII carriage return character.

Binary Transfer Frame Format

Note that all representation of binary data bytes and words is in hex; that is, a byte with a value of 127 decimal is shown as "7F".

Requirements for binary transfer:

a. The remote device and the HP64000 must both be set for 8-bit characters.

b. When the remote device is downloading, it must not echo or it must echo EXACTLY one character for each character received. Echoing <CR> as <CR><LF>, for example, is not acceptable.

c. When uploading, the remote device must not send extraneous characters such as fills or protocol characters.

Binary frame format:

BBBB rr..rr XXXX

Where:

BBBB	is a 16-bit byte count of the packet data bytes, rr..rr. BBBB must be in the range of 1 to 258.
rr..rr	are the packet data bytes
XXXX	is the 16-bit checksum of the frame data bytes, rr..rr

NOTE

If the received frame checksum (XXXX) and the calculated checksum agree, the frame is deemed to have been received correctly. A correct frame is ACKed and an incorrect one is NAKed. See the chapter on protocol.

The checksum, XXXX is calculated as the 16-bit sum of all the 8-bit data bytes, rr..rr, in that frame.

During binary download, all protocol (i.e.; XON/XOFF) is disabled. If protocol characters are received, they are interpreted as data bytes.

If the frame byte count, BBBB, is zero, the frame is defined to be an abort frame and the download aborts.

Example of a Binary Frame:

Although the packet data, rr..rr, is arbitrary, this packet is correct. Remember it is shown in hex representation.

```
(BBBBrrrrrrrrXXXX)
 0004000102030006
```

Chapter 7

ABSOLUTE FILE FORMAT DESCRIPTION - I_hex, M_hex, T_hex

GENERAL INFORMATION

General Format

The general format for an absolute file is:

<header><hex information><record terminator>

where:

<header> -	the record header character (i.e., /, S, or :).
<hex information> -	are hex characters containing data, control, or checksum information.
<record terminator> -	is the record terminator which will not be shown when describing the absolute format.

ABSOLUTE UPLOAD. The terminal outputs the record in the proper format with no extraneous characters. The record terminator output by the terminal is a 'carriage return' (<CR>), followed by a 'line feed' (<LF>) if auto linefeed is on.

ABSOLUTE DOWNLOAD. General downloading characteristics are as follows:

a. The terminal discards all characters until a header character is read.

b. After receiving the record header character, the only valid characters are HEX digits '0'-'9', 'A'-'F' (A-F must be upper case letters).

c. Upon receipt of a record terminator, the terminal attempts to translate the HEX characters into an absolute record and write it to the disc.

d. Steps a through c are repeated until an end-of-file is detected or an error occurs.

I_HEX ABSOLUTE FILE FORMAT

Intel Hexadecimal Intellec 8/MDS (I_hex) File Format for paper tape records consists of several fixed fields. An example follows:

: ll aaaa tt dddd..dd xx (spaces are inserted for clarity)

where:

:	- record header character
ll	- record length (#data bytes)
aaaa	- 16-bit load address or offset (8086)
tt	- record type number (see table 7-1)
dddd..dd	- hex data (if any)
xx	- checksum (xx=2's complement (ll+aa+aa+tt+dd+dd+ ...+dd))

The various Intel record types and their field contents are shown in table 7-1.

The older 8080 type transfers consisted of record types 00 and 01 only (see table 7-1). The 8086 and 8088 transfers use all four types of records. Question g(1) in the configuration sequence refers to these two types of transfers.

If 8086/8088 format is selected, the terminal uploads and downloads all types of records as necessary. If the old 8080 format is selected, the terminal only produces records of type 00 and 01 during uploading. During downloading, however, the terminal ignores record type 02 but does not flag it as an error. Record type 03 is interpreted by the terminal as follows:

a. The IP field is interpreted as the transfer address.

b. The CS field should be zero.

Examples of the different types of records are as follows:

Start Record:

:04 0000 03 0000 1234 B3 (spaces added for clarity)

where:

04 - byte count of data bytes
0000 - load address (meaningless)
03 - record type = start record
0000 - 8086/8088 CS register value
1234 - 8086/8088 IP register value
B3 - Checksum = two's complement of: 04+00+00+03+00+00+12+34

Table 7-1. Intel Record Types

```
Data Record:

     Record type = 00
     Record length = nn (two hex digits) #data bytes
     Load address = load address
     Data = N data bytes


End Record: (:00 0000 01 FF)

     Record type = 01
     Record length = 00
     Load address = *transfer address
     Data = none


Extended Address Record: (used with 8086 only)

     Record type = 02
     Record length = 02
     Load address = 00 (meaningless)
     Data = USBA (16-bit upper segment base address)


Start Record: (only output for 16-bit processors but
               is recognized by terminal)

     Record type = 03
     Record length = 04
     Load address = 00 (meaningless)
     Data = CS and IP of 8086 processor (Start ADDR)



    *NOTE: This field contains the transfer address
           only when 8080-type format is specified.
           This field is meaningless when using the
           8086/8088 format.
```

Extended Address Record:

:02 0000 02 1000 EC (spaces added for clarity)

where:

02 - # data bytes = 2
0000 - load address (meaningless)
02 - record type = extended address record
1000 - USBA value. SBA = USBA*10 (Hex)= 10 000
EC - checksum = two's complement of: 02+00+00+02+10+00

Data Record:

:03 0A00 00 E3 7C 47 4D (spaces added for clarity)

where:

03 - byte count
0A00 - 16-bit load address
00 - record type = data record
E3,7C,47 - data
4D - checksum = two's complement of: 03+0A+00+E3+7C+47

M_HEX ABSOLUTE FILE FORMAT

Four Motorola Hexadecimal (M_hex) record types are supported. They are distinguished by the four record headers S0, S1, S2, and S9. The record types are discussed in the following paragraphs. Other record types (S3 through S8) are not supported.

S0 Type

The S0 type is the start record and may contain miscellaneous information. The station does not output this record but will accept it from other sources. The format for an S0 type record is as follows:

S0 bb yyyyyyyy xx (spaces added for clarity)

where:

bb - is the byte count.

yy...yy - is arbitrary hex data.

xx - is the checksum (see S1 record checksum)

S1 Type

The S1 type is a data record beginning on a 16-bit load address. The format for an S1 type record is as follows:

S1 bb aaaa dddddddddddddddddddddddddddd xx (spaces added for clarity)

where:

bb	- #data bytes + #address bytes + 1(checksum)
aaaa	- 16-bit load address.
dd...dd	- Data bytes (or words) up to 24 bytes per record. (Station can receive more than 24 bytes per record.)
xx	- Checksum - one's complement of: bb+aa+aa+dd+...+dd

S2 Type

The S2 type is a data record starting on a 24-bit load address. The S2 type format is the same as an S1 type record except for a 24-bit load address (6 hex characters versus 4). The format for an S2 type record is as follows:

S2 bb aaaaaa dddddddddddddddddddddddddd xx (spaces added for clarity)

S9 Type

The S9 type is an end record and has the following format:

S9 bb yyyyyyyy xx (spaces added for clarity)

where:

bb	- is the byte count.
yy...yy	- is arbitrary hex data.
xx	- is the checksum.

Examples of the different types of records are as follows:

Data Record - Type S2:

S2 06 A00000 6C 01 EC (spaces added for clarity)

where:

06 - byte count
A00000 - 24-bit load address
6C,01 - data
EC - checksum = one's complement of:
06+A0+00+00+6C+01

End Record - Type S9:

S9 03 00 00 FC (spaces added for clarity)

where:

03 - byte count
00,00 - arbitrary data
FC - checksum = one's complement of:
03+00+00

NOTE

If checksum does not match on download, error message:

"Error in converting file"

appears on the command line, and the rest of the download goes to the screen.

T__HEX ABSOLUTE FILE FORMAT

The Tektronix Hexadecimal (T__hex) File Format is used to transfer 8-bit processor absolute information. The definition of T__hex format includes a data transfer protocol in addition to specifying the absolute data representation. This protocol provides for positive and negative acknowledgment of records received and for re-transmission of erroneous records.

T__HEX Protocol

Although this protocol permits high-speed file transfers without any additional protocol, the terminal still responds normally to the protocol selected with the exception that XON and XOFF are not transmitted by the terminal during downloading.

After the remote device downloads a record, the terminal responds with a positive acknowledgment (0 <CR>) or a negative acknowledgment (7 <CR>) as determined by the validity of the record. If the user has selected a prompt sequence in configuration, the terminal waits for this sequence before responding with an acknowledgment. The remote device is expected to wait for the acknowledgment before transmitting another record.

If the user selected a prompt sequence, the terminal waits for this sequence prior to uploading each record. Note that an upload prompt chararacter can perform this same function. If both an upload prompt character and a prompt sequence are selected, the terminal waits for the prompt sequence and then for the upload prompt character. After sending each record, the terminal waits for an acknowledgment which should be either '0 <CR>' or '7 <CR>' as described above. The terminal re-transmits a record if a '7 <CR>' is received. If there is no positive acknowledgment after five (5) transmissions of any one record, the terminal sends an abort record and terminates the upload.

The T__hex format specifies three types of records: data, terminating, and abort. The Tektronix format specifies that a maximum of 30 data bytes may be transmitted in any one data record. The record types are discussed in the following paragraphs.

DATA RECORD. The format for a data record is as follows:

/ aaaa bb cc dddd...dd xx (spaces added for clarity)

where:

/	- record header character
aaaa	- 4-hex digits specifying load address
bb	- byte count (# data bytes)
cc	- 1st checksum (cc=a+a+a+a+b+b)
dd...dd	- hex data
xx	- 2nd checksum (data checksum)

Example of a data record:

/0A00 02 0C 2A C3 1B (spaces added for clarity) where:

0A00	- load address (hex)
02	- data byte count
0C	- 1st checksum = 0+A+0+0+0+2
2A,C3	- two data bytes
1B	- 2nd checksum = 2+A+C+3

TERMINATING RECORD. The format for a terminating record is as follows:

/ aaaa bb cc (spaces added for clarity)

where:

/	- record header character
aaaa	- transfer address
bb	- byte count (always 0)
cc	- checksum (same as 1st checksum in data record)

ABORT RECORD. The format for an abort record is as follows:

// text

An abort record is identified as having two header characters. The text is optional and may be used to describe the condition which caused the abort. The terminal does not send any text with an abort block. Only the device transmitting a file may send an abort. An acknowledgment is not expected after an abort record has been sent.

Chapter 8

PROTOCOLS

GENERAL INFORMATION

Terminal mode supports three protocols: XON/XOFF, ENQ/ACK, and HP64000 protocol. The first two protocols are used solely for flow control, to keep from overrunning the computer during a file transfer.

HP64000 protocol operates with or without one of the other protocols. In addition, the flow control, error detection, and retransmission of bad records are handled by this protocol.

XON/XOFF Protocol

The mnemonics, XON and XOFF, usually refer to the ASCII characters DC1 and DC3 (or ASCII 11H and 13H respectively) which translate to 'transmit on' and 'transmit off'. When a remote device receives an XOFF character, for example, it should stop transmitting immediately. Upon receiving an XON character, the device should start (or continue) transmitting.

In addition, for slow time-sharing devices which do not function well using the XON/XOFF protocol, an upload prompt character can be specified during configuration; however, the upload prompt character cannot be the same character used for XON or XOFF. During uploading, the terminal waits for the 'upload prompt character' before sending each record. The remote device must send the upload prompt character every time that it is ready to read another record.

Upon successful completion of the upload command, the terminal waits for an upload prompt character (if not a null) as defined during configuration. Upon receipt of the prompt character, the terminal transmits a record of data which for source-type transfers consists of one line terminated by a carriage return (<CR>) and optionally a linefeed (<LF>). The terminal sends no protocol characters at any time during uploading. It still accepts XON and XOFF signals at any time and will send no more than two characters after receipt of an XOFF. One simple method of using upload prompt characters is illustrated by the following pseudo program which is intended to run on the remote device (i.e. computer):

```
While NOT <End-of-file> do
 begin
  Send terminal an upload prompt character
  Read a record from the terminal
  Process the record just read
 end
```

For situations where no upload prompt characters are specified, the following pseudo program may be used by the remote device:

```
While NOT <End-of-file> do
 begin
  Read a line from the terminal
  Send terminal an XOFF
  Process the line
  Send terminal an XON
 end
```

The method described above for uploading is essentially the same method used by the terminal when downloading. Once a download command is successfully entered, the following pseudo program describes the action of the terminal:

```
While NOT <End-of-file> do
 begin
  Read a line from the remote device {line must be terminated
                                      with a record terminator}
  Send an XOFF to the remote device
  Write the line just read to disc
  Send an XON to the remote device
 end
```

If the remote device (computer) cannot stop within two characters after receiving XOFF, the following pseudo program can be used on the remote device to interface with the terminal:

```
While NOT <End-of-file> do
 begin
  Read a line from the file
  Send the line to the terminal
  Receive XOFF
  Wait for XON
 end
 Send an end-of-file character to terminal
```

The user can define which ASCII characters are interpreted by the terminal as XON, upload prompt, and XOFF. It is suggested that ASCII 11H, 12H, and 13H be used, respectively, for these characters in an attempt to adhere to the standard protocol.

ENQ/ACK Protocol

The ENQ/ACK protocol is only implemented in one direction by the HP1000 and HP3000 computers. The computers send ENQ and expect to receive an ACK. Sending ENQ to these computers is meaningless. The terminal operates as if it were an HP terminal, therefore, it never ENQuires - only ACKnowledges.

The ENQ/ACK protocol enables the terminal to control the remote device to prevent it from overrunning the terminal. At least once in every 80 characters transmitted, the sender should send an ENQ character to the receiver. The receiver sends an ACK character when it can handle 80 more characters.

Like the XON/XOFF protocol, the ENQ/ACK protocol is not sufficient to prevent the terminal from overrunning some remote devices. Again, the use of the upload prompt character is recommended when uploading.

During the uploading of a file, the terminal will wait for an upload prompt character before sending each record. It is the responsibility of the remote device using ENQ/ACK protocol to send an upload prompt character to the terminal to initiate the transmission of each record. The HP1000 and HP3000 computers use a DC1 character (ASCII = 11H) as an upload prompt character. This character is sent automatically, therefore, standard file listing and entry utilities (such as editors) may be used for file transfers with the HP1000 and HP3000.

If ENQ/ACK protocol is desired but not directly supported by the device being used, the following pseudo-programs, intended to run on the remote device, will interface with the terminal for uploading and downloading:

```
{UPLOADING PROGRAM}

     While NOT<End-of-file> do
      begin
         Send terminal an upload prompt character
         Read a record from terminal
         Process the record
      end

{DOWNLOADING PROGRAM}

     While NOT <End-of-file> do
      begin
        Read a line from the file to download
        Send the line to terminal
        Send terminal an ENQ character
        Wait for terminal to respond with an ACK character
      end
      Send an end-of-file character to terminal
```

NOTE

Even though the sample download program may send more than 80 characters between ENQ queries, the terminal will operate properly with lines up to 240 characters long.

HP64000 PROTOCOL OVERVIEW

This section describes the protocol defined by HP64000 file transfer format/protocol. Flow control and re-transmission control are handled by the acknowledgment messages. Prompt characters are solely for flow control. The protocol also defines the method by which file transfers start, stop, and abort. This section ends with a discussion of error recovery and timeouts.

Acknowledgment and Abort Messages

The acknowledgment messages are the same in both binary and hex/ASCII modes of transfer. These messages are listed below.

ACK

ACK is used to refer to a positive acknowledge message. The content of that message is "ACK#<CR>" in both binary and hex modes where "#" can be a "0" or a "1".

An ACK0/ACK1 scheme is used to maintain acknowledgment synchronization. Frames are acknowledged with alternating ACK0 and ACK1's. The first frame of a transfer is acknowledged with an ACK0. The next with an ACK1, and so on.

The character, ACK, referred to in reference to HP64000 file transfers is a capital U (ASCII = 055H).

A positive acknowledgment message is then "U#<CR>". Where "#" is a "0" or a "1".

NAK

When NAK refers to a negative acknowledgment message, that message contains "NAK#<CR>". The character NAK is chosen to be an asterisk (ASCII = 2AH). Every time a frame is received, the "#" is toggled. The "#" is the same as the one used in positive acknowledgment responses. An example NAK message is "*0<CR>".

ABORT

In addition to positive and negative acknowledgments, an ABORT acknowledgment is defined. This is the method which the receiver (download) uses to abort a transfer. The ABORT message is "ABORT<CR>" where the character ABORT is a question mark (ASCII = 3FH). Please note that the ABORT message is an acknowledgment and is not the method the transmitter (upload) uses to abort a transfer.

Secondary Acknowledgment/File Name Acknowledgment

The acknowledgment to a file name record will contain a secondary acknowledgment if the record was received correctly. Table 8-1 shows the possible responses to a file name record.

Table 8-1. File Name Record Responses

Text	Mnemonic	Meaning
"*#<CR>"	NAK#<CR>	Received record has a checksum error.
"U#*<CR>"	ACK#NAK <CR>	Record was received but file name is not correct.
"U#U<CR>"	ACK#ACK <CR>	Record and file name are correct.

Where "#" is either a "0" or a "1".

Upload Prompt Character

The upload prompt character (UPC), which can be defined by the user during configuration, is also used during HP64000 format uploads. It is used to pace the transmission of frames to the remote device. The HP64000 waits for the UPC before sending each frame. The typical sequence when an UPC is used is as follows:

a. The remote device sends the UPC character to the HP64000.

b. The HP64000 sends a frame to the remote device.

c. The remote device acknowledges the frame.

d. The HP64000 waits for the UPC character before sending the next frame.

Acknowledgment Prompt Character

The acknowledgment prompt character (APC) is defined by the user during configuration, if HP64000 format is chosen. This character may be the same as the upload prompt character if desired.

The APC is used during download to pace the transmission of the acknowledgment message for each frame. The HP64000 will not acknowledge a frame until the APC is received.

The typical sequence when an APC is used is as follows:

a. The remote device sends a frame to the HP64000.

b. The HP64000 waits for the APC character.

c. Upon receipt of the APC, the HP64000 acknowledges the frame.

STARTING A DOWNLOAD SESSION

When not using an acknowledgment prompt character (APC) to download, the following events occur:

As soon as the "download in__HP64000__format" command is entered, and the RETURN key is pressed, the HP64000 sends an ACK to indicate its readiness to accept data. This ACK is always an ACK0 ("U0<CR>"). This ACK message will be re-transmitted every 10 seconds until the download begins. There is no limit to the number of re-transmissions at this point.

The sequence of events used in this case is:

a. Invoke the program on the remote device which will be waiting for an ACK message.

b. Press the download and the in__HP__fmt softkeys, then the RETURN key. The download will then begin.

When using an Acknowledgment Prompt Character (APC) to download, the following events occur:

The HP64000 will wait for the APC before sending each acknowledge message. This includes the first ACK as described above. Assuming the program on the remote device is invoked by typing the sequence HPD RETURN and that it transmits an APC to the HP64000 and then waits for the ACK message, the sequence of events for downloading is as follows:

a. Type in HPD but do not press the RETURN key.

b. Press the download and the in__HP__fmt softkeys and then the RETURN key to prepare the HP64000.

c. Press the RETURN key again (which finishes the "HPD<CR>" sequence and starts the download).

STARTING AN UPLOAD SESSION

When not using an Upload Prompt Character (UPC) to initiate upload, the following should be accomplished:

After the HP64000 upload command (refer to the upload discussion, above) is entered, the software will wait for a positive acknowledgment (see ACK/NAK) before sending the first data frame.

Assuming a program called UPL executes on the remote device to interface with the HP64000 upload, the sequence of events in this case is as follows:

a. Type in UPL from the keyboard to prepare to invoke that program on the remote device. Do not press the RETURN key yet.

b. Enter the HP64000 upload command by pressing the upload and in_HP_fmt softkeys, typing in the file name, and pressing the RETURN key.

or

Press the upload, in_HP_fmt, and using softkeys. Type in the listfile and press the RETURN key. Now press the RETURN key again to invoke the UPL program on the remote device. The UPL program must send an acknowledgment which will start the upload.

If the remote device is another HP64000 or the program on that device, UPL, sends the acknowledgment every 10 seconds as an HP64000 in download mode will, the following sequence may be used to start the upload:

a. Invoke the UPL program on the remote device (type in UPL and press the RETURN key), or, enter the download command on the remote HP64000.

b. Enter the HP64000 upload command and press the RETURN key.

When the acknowledgment is received, the upload will start. Note that an acknowledgment may already be present in the input buffer.

When using an Upload Prompt Character (UPC) to upload, the following events occur.

The upload software will wait for a positive acknowlegement message and the UPC before sending the first record. The first upload sequence above is the correct one for this case.

NOTE

The HP64000 sends neither APC's or UPC's. They are not needed for communication between HP64000 stations.

When communicating HP64000 ⟷ HP64000 in this format/protocol, the order of events (i.e., the times at which the upload and download command are entered on the stations) is very flexible. With respect to the entry of the upload command, the download command may have been entered at any time prior to the upload command and it may be entered up to one minute after the upload command.

ENDING THE TRANSFER

The normal termination of the file transfer session (upload/download) is as follows:

a. The transmitter (HP64000 uploading for example) detects that no more files are to be transmitted and sends an end-of-transmission record to the receiver (HP64000 downloading for example).

b. The receiver acknowledges the record (frame). If the data was received correctly, the receiver terminates.

c. Upon reception of a positive acknowledgment from the receiver, the transmitter terminates.

Download Aborts

The download can be aborted by the remote device by sending an abort frame (see discussion of frame formats) to the HP64000 in download mode. The HP64000 does not acknowledge the abort frame and exits download mode.

The user of the HP64000 can abort the download by pressing the ABORT softkey. This causes the HP64000 to acknowledge the next record with an ABORT message and then terminate download mode.

If the HP64000 detects an error during download from which it cannot recover, it aborts the download in the same manner as when the user presses the ABORT softkey.

An acknowledgment message which is an ABORT message is composed of the ABORT_CHAR ('?') followed by a <CR> (carriage return).

Abort frames are:

In binary mode:	Two zero bytes (a zero length frame) (No checksum is required on a binary abort frame)
In hex/ASCII mode:	A sequence of characters containing the ABORT_CHAR('?') followed by a <CR>

When an APC character is specified, pressing the ABORT softkey once will cause the abort acknowledgment message to be sent after an APC is received. If the APC is not received, the user may have to press the ABORT softkey again to cause immediate transmission of the abort acknowledgment message and termination of the download.

Upload Aborts

When the upload software aborts, it notifies the remote device by sending an abort frame as described above. If an Upload Prompt Character is specified by the user, Upload waits first for that character before sending the abort frame. If the user desires an immediate abort, the ABORT softkey will have to be pressed twice as described in the above paragraph.

ERROR RECOVERY/TIMEOUTS

This section describes the error recovery capabilities and procedures of HP64000 file transfer format and protocol.

Detection of data corruption is implemented with a 16-bit additive checksum on every frame and normal one-bit parity if chosen by the user. Framing, overrun, and parity errors detected by the USART in the HP64000 cause frames to be flagged as incorrect.

Timeouts

Timeouts are used to detect certain errors and correct them or cause the transfer to be properly terminated. All the timeouts are described in the following paragraphs.

Next Record Timeout:

Download:	After sending a frame acknowledgement (other than an abort), download will wait 10 seconds for the start of the next frame. If the frame is not detected within this time, the acknowledgment (positive or negative) is re-transmitted. After 5 re-transmissions, download will send the ABORT acknowledgment and exit download mode. The reason the acknowledgments are numbered is to combat a problem which can occur when a remote device is very slow and could buffer several of the re-transmitted acknowledgment messages. Note that the re-transmissions continue, when download is first initiated, until the first frame is received.

Inter-character Timeout:

Download:	After the beginning of a frame has been detected, if no characters are received for 15 seconds, download assumes that some type of dropout has occurred and sends a NAK message.

Acknowledgment Wait Timeout:

Upload:	After sending a frame, if no acknowledgment has been received within 60 seconds, upload sends an abort frame and exits upload mode.

APC or UPC Timeout:

Upload: The UPC timeout is in effect during HP64000 format uploads.

Download: The APC timeout is in effect during HP64000 format downloads. When waiting for an APC or UPC character, the HP64000 will display the message:

"STATUS: Waiting for prompt character"

after 30 seconds has elapsed. The wait for the character continues until it is received or the user aborts the transfer.

Recoverable Errors

Definitions:

Zero-length frame in hex/ASCII format;
A sequence of characters followed by a <CR>, which contains no more than one valid hex digit. The echo of an acknowledgment, "U0<CR>" is interpreted as a zero-length frame. Zero-length frames are completely ignored by download except that the next frame timeout counter is reset to zero.

Zero-length binary frame;
A zero-length binary frame is an abort frame. The zero-length frames discussed in this error section refer only to hex/ASCII zero-length frames.

Error: Checksum error - download found a bad checksum on a frame.

Recovery : A negative acknowledgment is sent to illicit the re-transmission of the frame.

Error: Character dropout during transmission of a hex/ASCII record. Character lost is not the frame terminator <CR>.

Recovery : Download will flag a bad checksum and send a NAK as above.

Error: Character dropout during transmission of binary frame or the <CR> is lost when transmitting a hex/ASCII frame. Some data has been received.

Recovery : Download will catch this with the inter-character timeout. If no character is received within 15 seconds, download assumes that the noted error has occurred and sends a NAK to cause the transmitter to re-transmit the frame.

Error: Acknowledgment lost.

Recovery : Download will timeout waiting for the beginning of the next frame and re-transmit the lost acknowledgment.

Irrecoverable Errors

Error: Entire frame lost. Download receives either a zero-length record or nothing.

Recovery : None. Download will keep re-transmitting the last acknowledgment but the transmitter will discard these ACKs because they are numbered incorrectly. After 5 re-tries, an abort is sent and both parties should terminate the transfer.

Error: All acknowledgments are lost.

Recovery : None. Upload times out waiting for an acknowledgment and sends an abort frame and terminates upload mode.

NOTE

These are the recovery mechanisms as implemented by the HP64000 and exemplified by HP64000-to-HP64000 file transfers. Depending on reliability needed, data link error rate, and other factors, some or all of these techniques may be needed for remote device implementations.

NOTES

Chapter 9

GENERIC TRANSFER EXAMPLES

INTRODUCTION

An acknowledge prompt character (APC) is used in the example shown in table 9-1. Note that since the HP64000 does not send APC's but only waits for them, the transmitter in this example is not an HP64000.

In the example shown in table 9-2, the Upload Prompt Character (UPC) character is used. Note that since an HP64000 never generates the UPC character, the receiver in this example is not an HP64000.

The example shown in table 9-3 is a sample "conversation" between two HP64000's during a HP64000 file transfer. No acknowledge prompt character (APC) is used in this example.

Table 9-1. Transfer Example Using an APC

Transmitter(Remote device)		Receiver(HP64000)
		(Rx ready) (waiting for APC)
APC →		
(waiting for ACK)	←	ACK0<CR>
File name record Assertive "TEST:HP:5" →		
APC →		(Bad disc #)
	←	ACK1NAK <CR>
(Go on to next file) Assertive "TEST1:HP:source" →		
APC →		(Name is OK!)
	←	ACK0ACK <CR>
(Now send data) Data, frame#1 →		
APC →		(Bad checksum)
	←	NAK1<CR>
(retransmit) Data, frame#1 →		
APC →		(Frame OK!)
	←	ACK0<CR>
Data, frame#2 →		
APC →		(Frame received OK!)
	←	ACK1<CR>
(File is over) (New file name record) Assertive "TEST3:HP:asmb_sym" →		
APC →		(OK! Previous file is closed)
	←	ACK0ACK <CR>
Data, frame#1 →		
APC →		
	←	ACK1<CR>
DATA, frame#2 →		
APC →		
	←	ACK0<CR>
• •		
Data, frame #N →		
APC →		
	←	ACK#<CR>
End of transmission record →		
APC →		(Close file, etc..)
	←	ACK#<CR>

Table 9-2. Transfer Example Using a UPC

Transmitter(HP64000)		Receiver(Remote Device)
(waiting for ACK)		
		(Rx ready)
	←	ACK0<CR>
	←	UPC
File name record		
Assertive		
"TEST:HP:5"	→	(Bad disc #)
	←	ACK1NAK <CR>
	←	UPC
(Go on to next file)		
Assertive		
"TEST1:HP:source"	→	(Name is OK!)
	←	ACK0ACK <CR>
	←	UPC
(Now send data)		
Data, frame#1	→	(Bad checksum)
	←	NAK1<CR>
	←	UPC
(retransmit)		
Data, frame#1	→	(Frame OK!)
	←	ACK0<CR>
	←	UPC
Data, frame#2	→	(Frame received OK!)
	←	ACK1<CR>
	←	UPC
(File is over)		
(New file name record)		
Assertive		
"TEST3:HP:asmb_sym"	→	
	←	ACK0ACK <CR>
	←	UPC
Data, frame#1	→	
	←	ACK1<CR>
	←	UPC
DATA, frame#2	→	
	←	ACK0<CR>
	←	UPC
.		
.		
Data, frame #N	→	
	←	ACK#<CR>
	←	UPC
End of transmission		
record	→	(Close file, etc...)
	←	ACK#<CR>

Table 9-3. Example Transfer Between Two HP64000s

Transmitter(HP64000)		Receiver(HP64000)
(waiting for ACK)		
		(Rx ready)
	←	ACK0<CR>
File name record Assertive "TEST:HP:5"	→	(Bad disc #)
	←	ACK1NAK <CR>
(Go on to next file) Assertive "TEST1:HP:source"	→	(Name is OK!)
	←	ACK0ACK <CR>
(Now send data) Data, frame#1	→	(Bad checksum)
	←	NAK1<CR>
(retransmit) Data, frame#1	→	(Frame OK!)
	←	ACK0<CR>
Data, frame#2	→	(Frame received OK!)
	←	ACK1<CR>
(File is over) (New file name record) Assertive "TEST3:HP:asmb_sym"	→	
	←	ACK0ACK <CR>
Data, frame#1	→	
	←	ACK1<CR>
DATA, frame#2	→	
	←	ACK0<CR>
. .		
Data, frame #N	→	
	←	ACK#<CR>
End of transmission record	→	(Close file, etc...)
	←	ACK#<CR>

HEX/ASCII FORMAT FILE TRANSFER EXAMPLE

Figure 9-1 shows an actual example of a hex/ASCII format file transfer. Note that the remote device buffer size was configured to be 80 characters both for convienence in formatting the example and to illustrate packetizing. The example closely follows the generic examples. The file, TEST1:HP:source, contains the following two lines/records (available with the "copy TEST1:HP:source to display" command):

This is a test file (Line #1)
(Line #2)

The contents of the file, TEST3:HP:asmb__sym is shown in table 9-4. This file contains two records as shown in the figure. The file is available with the "copy TEST3:HP:asmb__sym to display" command. The actual numerical contents of TEST3:HP:asmb__sym is shown in table 9-5.

Tx/ Rx	Hex Format File Transfer Data	Comments
Tx:	U0	Initial ACK0 sent by download
Rx:	0800544553543A48503A350289	TEST:HP 5, assertive file name record.
Tx:	U1*	Bad file name: frame is ACKed but name is NAKed.
Rx:	100054455354313A48503A736F757263652020055E	TEST1:HP:source
Tx:	U0U	
Rx:	1D025468697320697320612074657374206669 6C6520284C696E65202331292000 00	Frame has a bad checksum = 0000. This frame contains record 1 of TEST1
Tx:	*1	Frame is NAKed because of bad checksum
Rx:	1D0254686973206973206120746573742066696C6520284C696E6520233129200961	Frame is re-transmitted by upload
Tx:	U0	Checksum is correct this time.
Rx:	0902284C696E6520233229200279	This frame contains record 2 of TEST1:
Tx:	U1	
Rx:	100054455354333A48503A61736D625F73796D05EA	TEST3:HP:asmb_sym
Tx:	U0U	The first record of the file, TEST3:HP:asmb_sym, requires 7 frames f transmitting because of the small (80-character) packet size
Rx:	EB0200064143484152200008804 54C4153544C494E45001241434F4E54200039814F544845092C	
Tx:	U1	
Rx:	5257495345007A6143415345313020008A6143415345313120008D61434153453133200082099B	
Tx:	U0	
Rx:	6143415345313420007D6143415345313520008481 4B42445F4D41534B2000426147455443 09C9	
Tx:	U1	
Rx:	484152004761454E445F4745540099214348520 02F41494E445350000A41535441525400 5C0953	
Tx:	U0	
Rx:	21435254001C614B42445F4F4E2000448154494D45525F4F4646000981 4B42445F4F4846000969	
Tx:	U1	
Rx:	3E414352543120001D414454454D500007A1535441525453435245454E000161454E4450550965	
Tx:	U0	
Rx:	5420003B6143414E5F495420004F75BA047C	
Tx:	U1	
Rx:	93020006A14E4F5F42494E4B5F454E4400294143415345390080614341534532332000 87A109FB	This frame begins the transmission of record 2 of TEST3:HP:asmb_sym.
Tx:	U0	
Rx:	43415345494E5345525420009 1A1434153454445 4C4554452000948145 4E4453435245454E0A0E	
Tx:	U1	
Rx:	0002814C4153544C494E45200000214649580097214453500003415445 4D502000042150540809	
Tx:	U0	
Rx:	52003021505554002B614E4F5F42494E4B001E614745545F4C3120005941425445 4D50000608BB	
Tx:	U1	
Rx:	79A2011B	
Tx:	U0	
Rx:	0003000003	End of transmission frame/record.
Tx:	U1	The end of transmission frame/record must be acknowledged.

Figure 9-1. Hex/ASCII Format File Transfer

Table 9-4. TEST3:HP:asmb_sym File Format

```
                    Page #  1

File = TEST3:HP:asmb_sym

Record #    1    size = 118
Asmb_sym record:
CHAR             0008H Program      ELASTLINE     0012H Absolute
CONT             0039H Program      OTHERWISE     007AH Program
CASE10           008AH Program      CASE11        008DH Program
CASE13           0082H Program      CASE14        007DH Program
CASE15           0084H Program      KBD_MASK      0042H Program
GETCHAR          0047H Program      END_GET       0099H Program
CHR              002FH Program      INDSP         000AH Program
START            005CH Program      CRT           001CH Program
KBD_ON           0044H Program      TIMER_OFF     0009H Program
KBD_OFF          003EH Program      CRT1          001DH Program
DTEMP            0007H Program      STARTSCREEN   0001H Program
ENDPUT           003BH Program      CAN_IT        004FH Program

Checksum = 75BAH

Record #    2    size =   74
Asmb_sym record:
NO_BINK_END      0029H Program      CASE9         0080H Program
CASE23           0087H Program      CASEINSERT    0091H Program
CASEDELETE       0094H Program      ENDSCREEN     0002H Program
LASTLINE         0000H Program      FIX           0097H Program
DSP              0003H Program      TEMP          0004H Program
PTR              0030H Program      PUT           002BH Program
NO_BINK          001EH Program      GET_L1        0059H Program
BTEMP            0006H Program

Checksum = 79A2H

End of file after record #   2
```

Table 9-5. Numerical Contents of File TEST3:HP:asmb__sym

```
                    Page #  1

File = TEST3:HP:data

Record #    1     size = 118
0006 4143 4841 5220 0008 8045 4C41 5354   ACHAR    ELAST
4C49 4E45 0012 4143 4F4E 5420 0039 814F  LINE  ACONT  9 O
5448 4552 5749 5345 007A 6143 4153 4531  THERWISE zaCASE1
3020 008A 6143 4153 4531 3120 008D 6143  0   aCASE11   aC
4153 4531 3320 0082 6143 4153 4531 3420  ASE13   aCASE14
007D 6143 4153 4531 3520 0084 814B 4244   }aCASE15    KBD
5F4D 4153 4B20 0042 6147 4554 4348 4152  _MASK  BaGETCHAR
0047 6145 4E44 5F47 4554 0099 2143 4852   GaEND_GET  !CHR
002F 4149 4E44 5350 000A 4153 5441 5254   /AINDSP  ASTART
005C 2143 5254 001C 614B 4244 5F4F 4E20   \!CRT  aKBD_ON
0044 8154 494D 4552 5F4F 4646 0009 614B   D TIMER_OFF  aK
4244 5F4F 4646 003E 4143 5254 3120 001D  BD_OFF >ACRT1
4144 5445 4D50 0007 A153 5441 5254 5343  ADTEMP   STARTSC
5245 454E 0001 6145 4E44 5055 5420 003B  REEN  aENDPUT  ;
6143 414E 5F49 5420 004F 75BA            aCAN_IT  Ou

Record #   2     size =  74
0006 A14E 4F5F 4249 4E4B 5F45 4E44 0029     NO_BINK_END )
4143 4153 4539 0080 6143 4153 4532 3320  ACASE9  aCASE23
0087 A143 4153 4549 4E53 4552 5420 0091     CASEINSERT
A143 4153 4544 454C 4554 4520 0094 8145   CASEDELETE    E
4E44 5343 5245 454E 0002 814C 4153 544C  NDSCREEN   LASTL
494E 4520 0000 2146 4958 0097 2144 5350  INE    !FIX  !DSP
0003 4154 454D 5020 0004 2150 5452 0030    ATEMP    !PTR 0
2150 5554 002B 614E 4F5F 4249 4E4B 001E  !PUT +aNO_BINK
6147 4554 5F4C 3120 0059 4142 5445 4D50  aGET_L1   YABTEMP
0006 79A2                                   y

End of file after record #   2
```

Binary Format File Transfer Example

The binary format file transfer is similar to the hex/ASCII format file transfer example shown in figure 9-1. The binary format file transfer requires a byte count at the start of each frame. Figure 9-2 shows an example of a binary format file transfer. The two frames shown in the figure correspond exactly to the first two frames shown in the hex/ASCII example. Remember that the hex digits shown are now representing data bytes as opposed to literal characters (7F represents the data byte 127 decimal) All carriage returns <CR> implicit in the hex example are indicated literally in this example.

Tx/ Rx	Binary Format File Transfer Data	Comments
Tx:	U0<CR>	Initial ACK0 sent by download.
Rx:	000D0800544553543A48503A350289	TEST:HP 5, assertive file name record
Tx:	U1*<CR>	Bad file name: frame is ACKed but name is NAKed
Rx:	0015100054455354313A48503A736F757263652020055E	TEST:HP source.

Figure 9-2. Binary Format File Transfer

Chapter 10

SOFTWARE IMPLEMENTATION ON A REMOTE DEVICE

DESIGN HINTS

The remote device HP64000 format interface code may be written as a normal application program on many computers. A modular structure is highly recommended as shown in the figure describing the software structure (figure 6-1). Each "high-level" module shown in the figure will likely be composed of several subprograms. For example, HP64000 format upload is designed somewhat as shown in the following example:

```
Procedure Upload_file_sys ;
  begin
    if( uploading a list of files ) then
      Upload_file_list
    else
      Upload_a_file ;
  end ; {Upload_file_sys}

Procedure Upload_file_list ;
  begin
    open( file_containing_list_of_files ) ;
    while( not(end_of_file) ) do
      begin
        read_next_line_from_file ;
        parse_file_name_from_line ;
        Upload_a_file ;
      end ;
    close( file_with_list_of_files ) ;
  end ; {Upload_file_list}

Procedure Upload_a_file ;
  begin
    open( file_to_be_uploaded ) ;
    Upload_record( filename_record ) ;
    if( filename_record_accepted ) then
      while( not(end_of_file) ) do
        begin
          read_record_from_file ;
          Upload_record ;
        end ;
    close( file_just_uploaded ) ;
  end ; {Upload_a_file}

Procedure Upload_record ;
  begin {packetizing subprogram}
    repeat
      make_packet_from_record ;
      Upload_packet ;
    until( whole_record_is_sent ) ;
  end ; {Upload_record}

Procedure Upload_packet ;
  begin
    repeat
      if( using_binary_format ) then
        Upload_binary_frame
      else
        Upload_hexascii_frame ;
    until( frame_accepted or retry_count_exceeded or
    timeout ) ;
  end ; {Upload_packet}
```

One of the many issues not addressed in the above pseudo program is how errors are handled. It has been useful to incorporate a global flag, say abort__flag, which is tested in every loop condition. For example, the file-reading loop might be:

while(not(end__of__file) and not(abort__flag)) do ...

Though the pseudo program shows procedures, functions which return a value describing possible errors or file name acknowledgment conditions are more useful.

Timeouts are alluded to in the pseudo program. If a remote device is transmitting files to an HP64000, the HP64000 contains most of the timeouts needed to recover from errors so the application program on the remote device could quite likely not include timeouts.

The interface to the remote device's file system was not explicitly described. Recommendations are:

a. If the remote device supports a variable-length record file structure, use it. The HP64000 also uses a variable-length record file structure.

b. If the remote device supports a "stream of bytes" file system, two methods of storing files on the remote device need to be considered. A 'text' mode of storage implies that each HP64000 record is stored as a stream of bytes terminated with the correct line termination character which must be programmatically supplied. For storage of binary data files (ie.. symbol tables), a variable-length record structure should be simulated. One which seems to work well is to store a byte preceding each 'record' which contains the length-1 (in bytes) of the record.

c. One of the predicted uses of HP64000 format file transfer is to develop software on a remote device with cross-software and then to transfer the code and symbols to the HP64000 to take advantage of its powerful emulation capabilities. The recommended approach is to make files on the remote device which are images of the desired files on the HP64000. Separating the steps of making the file images from the file transfer results in a simpler and more flexible system.

d. Mapping of file names from HP64000-style names to remote device names is an issue to be considered. Three basic strategies are:

   1). Implement, on the remote device, a scheme of auxiliary file names like HP64000 upload uses. Keep a cross-reference table (a file list) of HP64000 names and host names.

   2). Develop a one-to-one mapping algorithm for mapping remote device names to HP64000 names and incorporate this algorithm into the application program(s) which do the transfer.

   3). Put the HP64000 file names into the first line/record of each file. Let the remote device make up file names and keep a cross-reference table.

NOTES

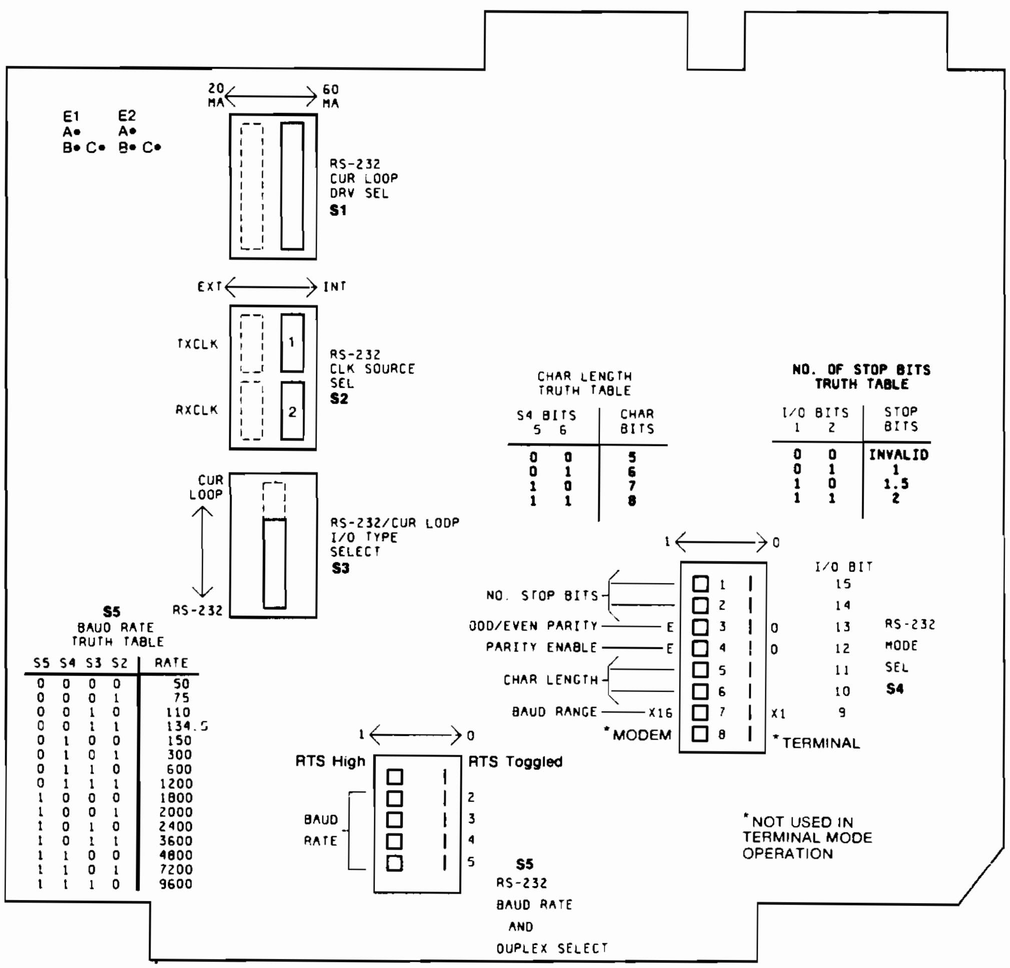
Appendix A

I/O HARDWARE CONFIGURATION

HARDWARE CONFIGURATION

Model 64100A

The communication format for the two RS-232-C ports is controlled by five sets of switches and a jumper set located on the Model 64100A I/O board in the station card cage (see figure A-1). Switches on the board must be set to match the 64000 station transmission format with that of the remote system. Tables A-1 through A-3 explains briefly the different switch-setting functions. Table A-4 indicates I/O Board switch settings that may be used for connection to a specifically configured HP3000 or HP1000. For a detailed description of the switches refer to the Model 64100A Service Manual.



S4 BITS 5	S4 BITS 6	CHAR BITS
0	0	5
0	1	6
1	0	7
1	1	8

I/O BITS 1	I/O BITS 2	STOP BITS
0	0	INVALID
0	1	1
1	0	1.5
1	1	2

S5	S4	S3	S2	RATE
0	0	0	0	50
0	0	0	1	75
0	0	1	0	110
0	0	1	1	134.5
0	1	0	0	150
0	1	0	1	300
0	1	1	0	600
0	1	1	1	1200
1	0	0	0	1800
1	0	0	1	2000
1	0	1	0	2400
1	0	1	1	3600
1	1	0	0	4800
1	1	0	1	7200
1	1	1	0	9600

Figure A-1. Model 64100A I/O Board Switch Identification

Table A-1. I/O Switch Functions

Switch	Position	Name	Functions
S1	 left right	Current Loop Drive Select 20 mA 60 mA	Selects appropriate current operation for TTY if current loop selected.
S2	left right	Ext clock select Int clock select	Provides clock selection.
S3	up down	Current Loop operation RS-232-C operation	Selects current loop or RS-232-C ASYNC operation.
S4	See table A-2	Mode Select	Selects TERM/MOD, Stop Bit Qualifier Parity and Character length.
S5	See table A-3	Baud Rate	Selects baud rate (bit 2 to 5). Selects RTS function (bit 1).

Table A-2. Switch S4 Bit Function

Bit	Function	Comments
1 (msb) 2	# of stop bits	00 = invalid 01 = 1 bit 10 = 1 1/2 bits 11 = 2 bits
3	Odd/even Parity	1 = even, 0 = odd
4	Parity enable/ disable	1 = enable, 0 = disable
5 (msb) 6	Word (Char) Length	00 = 5 01 = 6 10 = 7 11 = 8
7	Clock Mode	1 or divide by 16
8	Modem/Terminal	MODEM/TERMINAL select (Not used by terminal mode software)

Table A-3. Switch S5 Baud Rate Selection

Baud Rate	Switch Position			
	S5 Bits			
	MSB 5	4	3	LSB 2
50	0	0	0	0
75	0	0	0	1
110	0	0	1	0
134.5	0	0	1	1
150	0	1	0	0
300	0	1	0	1
600	0	1	1	0
1200	0	1	1	1
1800	1	0	0	0
2000	1	0	0	1
2400	1	0	1	0
3600	1	0	1	1
4800	1	1	0	0
7200	1	1	0	1
9600	1	1	1	0

Table A-4. I/O Board Switch Settings - HP1000/HP3000/HP9000

Switch	Setting
S2	TXCLK - Internal RXCLK - Internal
S3	RS-232
S4	No. Stop Bits - 2 Parity - even Parity - enabled Char Length - 7 bits Baud Range - x16
S5	RTS - high Baud Rate - 2400

NOTE: A special cable must be used and a jumper connected between A and C on E2 of the I/O Board if it is desirable to drive the external clock of the HP1000. If not, a modem cable may be used. The above discussion assumes the use of an HP BACI interface board.

Model 64110A

The Model 64110A functions similarly to the Model 64100A except for the Current Loop facility which is not provided. Like the 64100A, the 64110A provides asynchronous serial communications between the station and external communications devices. Communications is controlled by two switches on the RS-232/Flexible Disc Control card and a jumper on the CPU/IO card (see figure A-2 for switches and jumper locations). Table A-5 describes the switches (S1 and S2) and the jumper on the CPU/IO card.

Table A-5. Model 64110A RS-232-C Switch Identification

Switch	Position	Name	Function
S2	See Table A-3	Baud Rate	Selects baud rate (bits 2 to 5). Selects RTS function (bit 1). (Same function as S5 on Model 64100A I/O Card.)
S1	See Table A-6	Mode Select	Selects Stop Bit Qualifier, Parity, and Character length.
E2 Jumper	MOD or TERM	Modem/Terminal	When jumper installed in E2 MOD position, "To Peripheral" interconnection effective. When jumper installed in E2 TRM position, "To Modem" interconnection effective.

Table A-6. Model 64110A S1 Bit Functions

Bit	Function	Comments
1 (msb) 2	# of stop bits	00 = invalid 01 = 1 bit 10 = 1 1/2 bits 11 = 2 bits
3	Odd/even Parity	1 = even, 0 = odd
4	Parity enable/ disable	1 = enable, 0 = disable
5 (msb) 6	Word (Character) length	00 = 5 01 = 6 10 = 7 11 = 8
7		no connection
8		no connection

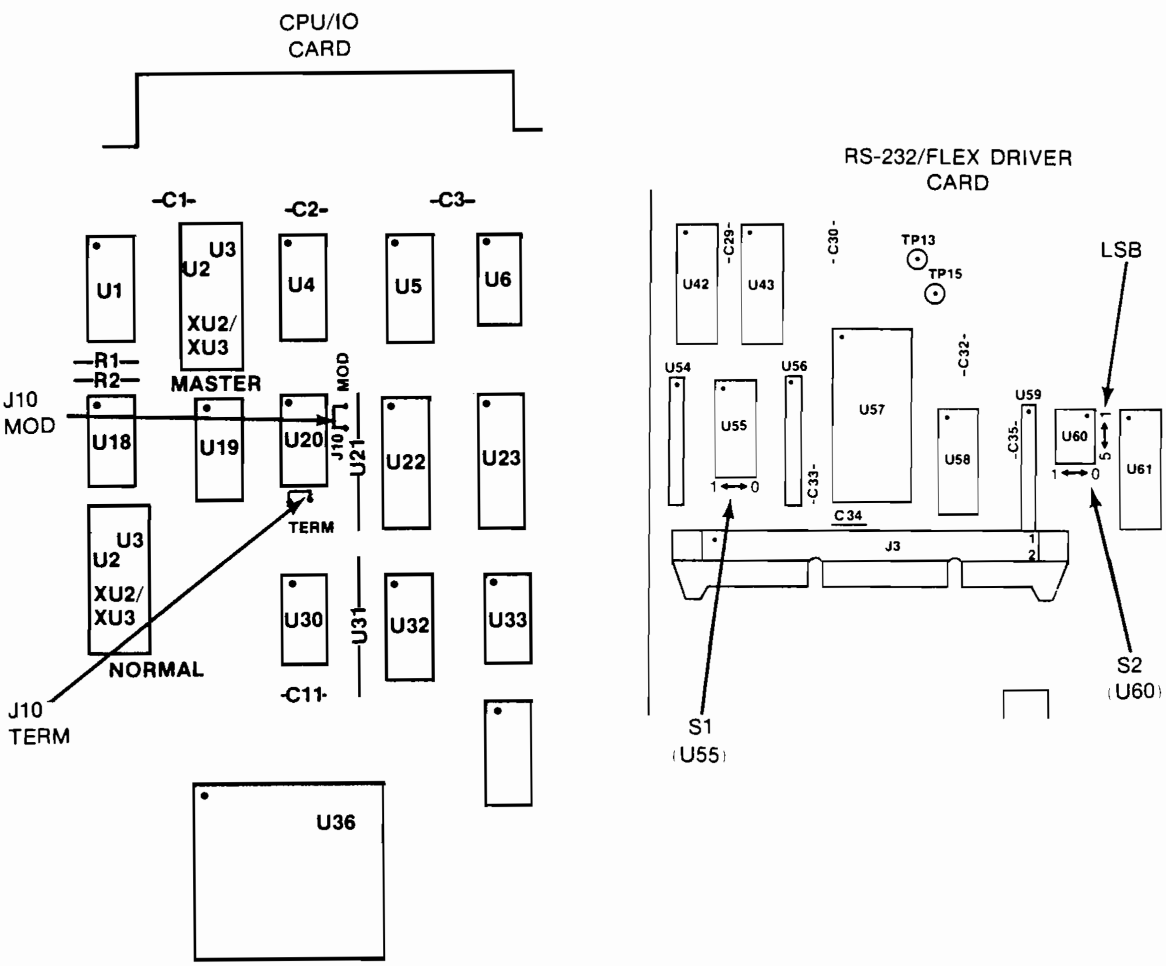


Figure A-2. Model 64110A RS-232-C Switch Locations

Table A-7. RS-232-C Interface Connector Pin Assignments

RS-232-C Connector Pin	Mnemonic	Description
1	GRD	Protective ground.
2	TXD	Information sent by DTE and received by DCE.
3	RXD	Information sent by DCE and received by DTE.
4	RTS	Request to send.
5	CTS	Clear to send.
6	DSR	Data set ready.
7	SGD	Signal ground.
8	CARDET	Carrier detect.
9 - 14		unassigned.
15	TXCLK	Transmit clock (refer to the following RS-232-C TX and RX Clk Control paragraph).
16		unassigned.
17	RXCLK	Receive clock (refer to the following RS-232-C TX and RX Clk Control paragraph).
18 - 19		unassigned.
20	DTR	Data terminal ready.
21 - 25		unassigned.

NOTES

Appendix B

STATUS AND ERROR MESSAGES

This appendix may be used as a quick reference guide for general status/error messages that may be generated when the station is configured for terminal mode operation.

Table B-1. Terminal Mode Status Messages

Status Message	Meaning
WARNING: x1 setting requires external clock	Warning indicates baud rate clock set to x1 instead of x16.
Absolute file transfer aborted	When the sender receives too many negative acknowledgements during a T__hex file transfer, an abort block is sent to the receiver.
RS232: char bits = ______, ______ parity, stop bit = ______	These status messages describe the RS-232-C switch settings on the I/O Board.
Sending <break>	This message will be displayed during the 450 ms duration of a break function.
Upload or download aborted by the user	This message is displayed when user presses the ABORT softkey.
Waiting for prompt character	This message indicates that the 64000 is waiting for an Acknowledge Prompt Character (APC) or an Upload Prompt Character (UPC).

Table B-2. Terminal Mode Error Messages

Error Messages	Meaning
ERROR: nn Parity nn Overrun nn Framing, where: nn = no. of errors	These errors indicate the RS-232-C hardware status. Check I/O Board switch settings for compatibility with the communication characteristics of the remote device. Overrun errors usually occur during downloading at baud rates greater than 1200 baud when the software protocol is not functioning properly. The baud rate may be lowered or software protocol used to prevent overrun errors.

NOTE

RS-232-C errors are accumulated during each upload and download so that the total errors can be noted at the end of the file transfer. The characters which are received with framing or parity errors are turned into DEL characters so that they are easily found in text.

Error Messages	Meaning
*Abort received	This message is displayed when either an abort acknowledgement is received (upload) or an abort frame is received (download).
Absolute format conversion error	This message is displayed if, during downloading, there is a format error; for example, checksum.
*Byte count wrong in packet	This message is displayed when download cannot recover from certain byte count errors.
Corrupt file	This is a file manager error.
*Data record received with no file open	This message indicates that a file must be open (download) before data records are received.
End of file	This is a file manager error.
File already exists. Delete old?	This is a file manager error.
File not found	This is a file manager error.

Appendix B

STATUS AND ERROR MESSAGES

This appendix may be used as a quick reference guide for general status/error messages that may be generated when the station is configured for terminal mode operation.

Table B-1. Terminal Mode Status Messages

Status Message	Meaning
WARNING: x1 setting requires external clock	Warning indicates baud rate clock set to x1 instead of x16.
Absolute file transfer aborted	When the sender receives too many negative acknowledgements during a T__hex file transfer, an abort block is sent to the receiver.
RS232: char bits = _______, _______ parity, stop bit = _______	These status messages describe the RS-232-C switch settings on the I/O Board.
Sending <break>	This message will be displayed during the 450 ms duration of a break function.
Upload or download aborted by the user	This message is displayed when user presses the ABORT softkey.
Waiting for prompt character	This message indicates that the 64000 is waiting for an Acknowledge Prompt Character (APC) or an Upload Prompt Character (UPC).

Table B-2. Terminal Mode Error Messages

Error Messages	Meaning
ERROR: nn Parity nn Overrun nn Framing, where: nn = no. of errors	These errors indicate the RS-232-C hardware status. Check I/O Board switch settings for compatibility with the communication characteristics of the remote device. Overrun errors usually occur during downloading at baud rates greater than 1200 baud when the software protocol is not functioning properly. The baud rate may be lowered or software protocol used to prevent overrun errors.

NOTE

RS-232-C errors are accumulated during each upload and download so that the total errors can be noted at the end of the file transfer. The characters which are received with framing or parity errors are turned into DEL characters so that they are easily found in text.

*Abort received	This message is displayed when either an abort acknowledgement is received (upload) or an abort frame is received (download).
Absolute format conversion error	This message is displayed if, during downloading, there is a format error; for example, checksum.
*Byte count wrong in packet	This message is displayed when download cannot recover from certain byte count errors.
Corrupt file	This is a file manager error.
*Data record received with no file open	This message indicates that a file must be open (download) before data records are received.
End of file	This is a file manager error.
File already exists. Delete old?	This is a file manager error.
File not found	This is a file manager error.

Table B-2. Terminal Mode Error Messages (Cont'd)

Error Messages	Meaning
INPUT DATA LOST	This message is displayed when the input buffer is overrun. This usually occurs when the remote device continues to transmit characters while the user is entering a command. The buffer is not serviced while the terminal is in the command mode.
Invalid disc	This is a file manager error.
I/O board must be set for 8-bit char for binary transfers	If the user chooses binary HP64000 transfers in configuration, this error will be displayed if the I/O Board is not set for 8 bits per character (this is required for binary transfers).
No free disc pages	This is a file manager error.
No prompt sequence found within 256 characters	When using T_hex absolute file format, the terminal will wait for up to 256 characters before issuing this error message.
No space for directory	This is a file manager error.
Not configured for this feature - edit configuration	This message is displayed when the user attempts to use a feature which is not properly configured. For example, trying to transfer a relocatable file with other than 64000 format. This error can only be generated by the user entering commands which are not explicitly shown on the softkeys.
OUTPUT DATA LOST	This message is displayed if the transmitter is not sending (due to an XOFF when using XON/XOFF protocol) and the user continues typing until the output buffer overflows.
*Retry limit exceeded	This message is generated when either the current acknowledgement has been retransmitted 5 times (download) or the current frame has been retransmitted 5 times (upload).
Syntax error	This message is displayed whenever the user enters an invalid command.

Table B-2. Terminal Mode Error Messages (Cont'd)

Error Messages	Meaning
Target Processors disagree	This message is displayed if the configuration specification (of data width or addressing base) does not match the processor specification in the absolute file being uploaded.
*Timeout error	There are a number of timeouts in HP format up-load/download. Please refer to the approriate section of the manual for details.
*Unknown record type	This message is displayed when a record type is detected by download which is not in the range of 0 through 3.

***NOTE**

Errors indicated by an asterisk (*) cause termination of the HP64000 format file transfer.

Appendix C

ESCAPE CODES

This appendix may be used as a quick reference guide for control functions implemented for the 64000 terminal software and the character sequence to envoke each feature. These features may be evoked from the keyboard when in the local echo mode or via the RS232 line.

The escape sequences required to invoke each feature along with their commonality with other HP terminals are listed in Table C-1.

NOTE

The escape function, denoted by ^{E}C in Table C-1, can be invoked from the 64000 keyboard by pressing the following keyboard keys simultaneously:

CNTL and [

or

An ^{E}C can be sent out on the RS-232-C line by pressing the <escape> softkey.

Table C-1. Escape Codes and Functions

Function	Code	Term. 262X	Term. 264X
Insert Line	^{E}C L	X	X
Delete Line	^{E}C M	X	X
Start Insert Mode	^{E}C Q	X	X
Stop Insert Mode	^{E}C R	X	X
Delete Char	^{E}C P	X	X
Number of columns	80	X	X
Number of rows	18		

Table C-1. Escape Codes and Functions (Cont'd)

Function	Code	Term. 262X	Term. 264X
Set Tab	^{E}C 1	X	X
Clear Tab	^{E}C 2	X	X
Clear ALL tabs	^{E}C 3	X	X
Horz Tab	^{E}C I	X	X
Back Tab	^{E}C i	X	X
Clear Display	^{E}C J	X	X
Clear to End of line	^{E}C K	X	X
Roll up	^{E}C S	X	X
Roll down	^{E}C T	X	X
Next Page	^{E}C U	X	X
Prev Page	^{E}C V	X	X
Disp Functions ON	^{E}C Y	X	X
Disp Functions OFF	^{E}C Z	X	X
Up arrow	^{E}C A	X	X
Down arrow	^{E}C B	X	X
Left arrow	^{E}C D	X	X
Right arrow	^{E}C C	X	X
Back Space	Back Space key CNTL h CNTL H	X	X

Table C-1. Escape Codes and Functions (Cont'd)

Function	Code	Term. 262X	Term. 264X
Home cursor Up	Ec h Ec H	X	X
Home cursor Down	Ec F	X	X
Home Screen	Ec G	X	X
Transmit keypad Enable	Ec & s 1 A	X	X
	Ec & j B		
Transmit keypad Disable	Ec & s 0 A	X	X
	Ec & j @		
Primary Status request - causes 64000 to identify itself by outputting HP64000	Ec * S ^	X	X
Col Addressing - where col addressed is nn (1 or 2 digits)	Ec & a nn C (see NOTE #2)		X
Row Addressing - where row addressed is nn (1 or 2 digits)	Ec & a nn Y (see NOTE #2)		X
	Ec & a nn R (see NOTE #2)		X
(rolls display up nn rows)	Ec & r nn U (see NOTE #1)	X	

Table C-1. Escape Codes and Functions (Cont'd)

Function	Code	Term. 262X	Term. 264X
(rolls display down nn rows)	^{E}C & r nn D (see NOTE #1)	X	
RxC Addressing - screen rel	^{E}C & a mm c nn Y (see NOTE #2)	X	X
(where nn = row; mm = column)	^{E}C & a nn y mm C (see NOTE #2)		X
	^{E}C & a mm x nn Y (see NOTE #2)		X
RxC Addressing - Absolute	^{E}C & a nn r mm C (see NOTE #3)		X
(where nn = row; mm = column)	^{E}C & a mm c nn R (see NOTE #3)	X	X
RxC Addressing - Cursor rel	^{E}C & a+/-nn r+/-mm C (see NOTE #1)		X
(where nn = deltarow; mm = deltacol)	^{E}C & a+/-mm c+/-nn R (see NOTE #1)	X	X
	^{E}C & a+/-mm x+/-nn Y (see NOTE #1)	X	
Start Inverse Video	^{E}C & d char	X	X
Start Underline Mode	^{E}C & d char	X	X
Blink Char	^{E}C & d char	X	X
End all enhancements	^{E}C & d @	X	X

NOTES for table C-1:

#1. mm and nn must be 2 digits each; +/- notation indicates either + or - is valid.

#2. nn and mm may be any combination of 1 or 2 digits.

#3. nn may be 1, 2, or 3 digit numbers; mm may be 1 or 2 digit numbers.

NOTE

The following chart describes the escape sequence characters (char) required for video enhancements. These enhancements, half-bright, underline, inverse video, blinking, and end enhancements, are for terminals HP2626A, HP2647A, and the enhanced terminal software. It should be noted that the half-bright enhancement is not applicable to the HP64000 terminal mode Software.

	"char"															
	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Half-bright									x	x	x	x	x	x	x	x
Under-line					x	x	x	x					x	x	x	x
Inverse Video			x	x			x	x			x	x			x	x
Blinking		x		x		x		x		x		x		x		x
End Enhance-ment	x															

NOTES

Index

The following index lists important terms and concepts of this manual along with the location(s) in which they can be found. The numbers to the right of the listings indicate the following manual areas:

- Chapters – references to chapters appear as "Chapter X", where "X" represents the chapter number.
- Appendices – references to appendices appear as "Appendix Y", where "Y" represents the letter designator of the appendix.
- Figures – references to figures are represented by the capital letter "F" followed by the section figure number.
- Other entries in the Index – references to other entries in the index are preceded by the word "See" followed by the reference entry.

a

b

c

c (Cont'd)

d

e

e (Cont'd)

f

g

h

i

k

l

m

n

p

r

s

t

u

u (Cont'd)

X

64980-90935, APRIL 1984

HEWLETT PACKARD

PRINTED IN U.S.A.